Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

ESTUDIO DE UN SISTEMA DE POSICIONAMIENTO PARA INTERIORES

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: JOHN ENRIQUE CARRANZA ROSERO

Tutor: FRANCISO JOSÉ ABAD CERDÁ

Curso 2017/2018

Resumen

La tecnología de posicionamiento más usada en el mundo es el GPS. Consiste en una constelación de satélites que emiten una señal sincronizada y un receptor que, a partir de la señal que le llega desde varios satélites, es capaz de calcular su posición en la Tierra. El problema del GPS es que no funciona en interiores. La tecnología de radio de banda ultra ancha (UWB) se está utilizando desde hace poco para implementar sistemas de posicionamiento en interior. El objetivo de este proyecto es probar un sistema de posicionamiento por UWB estudiando sus características y limitaciones. También se implementará una demostración del sistema.

Palabras clave: UWB, GPS, sistema de posicionamiento.

Abstract

The positioning technology most used in the world is GPS. It consists of a constellation of satellites that emit a synchronized signal and a receiver that, from the signal that comes from several satellites, is able to calculate its position on Earth. The problem with GPS is that it does not work indoors. Ultra-wideband radio (UWB) technology has recently been used to implement indoor positioning systems. The objective of this project is to test a positioning system by UWB studying its characteristics and limitations. A demonstration of the system will also be implemented.

Keywords: UWB, GPS, positioning system.

Resum

La tecnologia de posicionament més usada en el món és el GPS. Consisteix en una constel·lació de satèl·lits que emeten un senyal sincronitzada i un receptor que, a partir del senyal que li arriba des de diversos satèl·lits, és capaç de calcular la seva posició a la Terra. El problema del GPS és que no funciona en interiors. La tecnologia de ràdio de banda ultra ampla (UWB) s'està utilitzant des de fa poc per implementar sistemes de posicionament en interior. L'objectiu d'aquest projecte és provar un sistema de posicionament per UWB estudiant les seves característiques i limitacions. També s'implementarà una demostració del sistema.

Paraules clau: UWB, GPS, sistema de posicionament.

Índice

Índice de ilustraciones

Índice de tablas

1. Introducción

En este primer apartado se hace una breve presentación sobre las ideas básicas que componen este trabajo, explicando brevemente algunas de las tecnologías que se usan actualmente para el posicionamiento. También se aclaran los objetivos del proyecto y finalmente se ofrece una breve explicación de los distintos capítulos que lo componen.

1.1. Tecnologías actuales de posicionamiento

Hoy en día, todos estamos familiarizados con los sistemas de localización o de posicionamiento debido a que están presentes en el día a día de las personas, un buen ejemplo es el sistema de posicionamiento global (GPS) que está incorporado en todos los teléfonos inteligentes. Esta tecnología nos ofrece un gran abanico de posibilidades, desde saber dónde está un pedido o saber cuál es la ruta más rápida o más corta entre dos puntos hasta localizar a una persona en caso de emergencia.

Una gran parte de los usos esta tecnología se centra en la localización exterior, debido a que el GPS no es lo suficientemente preciso para utilizarlo en interiores. En estos casos se emplea un sistema de posicionamiento en interiores (IPS), el cual está compuesto por una red de dispositivos orientados a localizar de forma inalámbrica objetos o personas.

Este sistema, en lugar de utilizar satélites, se basa en dispositivos fijos que están en una posición conocida, consiguiendo de esta forma una localización con una exactitud mayor que la de los GPS, aunque no por ello libres de error.

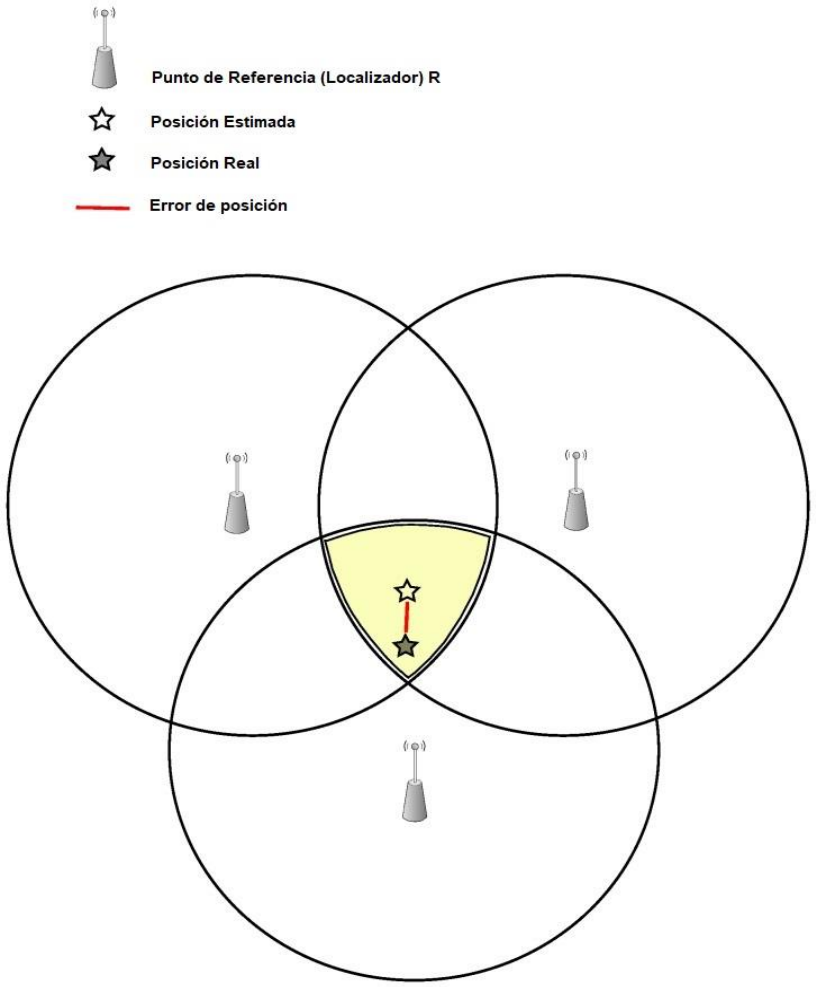

Ilustración 1: posicionamiento con IPS[16]

Para los sistemas IPS hay que tener en cuenta el lugar donde se realiza la medición. Por ejemplo, si hay obstáculos que dispersen las señales, pueden aparecer errores en los resultados. Por otro lado hay que considerar que estos resultados pueden variar en términos de precisión, coste, escalabilidad, tecnología y seguridad.

Actualmente hay múltiples opciones de tecnologías IPS, de las cuales, las más conocidas son las que utilizan las señales *WiFi* que usan los puntos de acceso que emiten continuamente una señal con un identificador único. Otra tecnología conocida es el *Bluetooth* pero esta está más enfocada hacia conocer la proximidad de un objeto en lugar de su localización, este es el caso del iBeacon desarrollado por Apple [1]. Pero fijándonos en que un aspecto crítico en la medición de posiciones es la precisión, hay que destacar las tecnologías que utilizan una banda de señal ultra ancha (UWB).

Este trabajo se centra en el estudio de la tecnología Pozyx, que basándose en la tecnología UWB y con la ayuda de distintos dispositivos consiguen una precisión de hasta 10 centímetros de error [2]. Dicha precisión es notoriamente superior a la que ofrecen el resto de tecnologías IPS que rondan los 1.5 metros de error y muy superior a la precisión que ofrece el GPS que es de 8 a 10 metros de error [14].

1.2. Objetivos

Como se ha comentado antes, las tecnologías IPS más comunes (*WiFi* y *Bluetooth*) son las que más problemas conllevan a la hora de intentar localizar un objeto. Su señal puede verse obstaculizada, dispersada o incluso interrumpida por distintos obstáculos o personas, provocando errores en la localización.

Debido a estos problemas surge la motivación de este proyecto, estudiar la tecnología UWB y analizar los resultados que nos ofrece, aplicando algoritmos para la minimización de problemas en la localización.

La finalidad del trabajo consiste en comprobar la viabilidad de utilizar estos sistemas en la actualidad. Para ello se considerará un caso de estudio de un plató de televisión en el que se automatiza la cámara para que enfoque al presentador. Para ello, se construirá una simulación utilizando la tecnología Pozyx y Arduino, haciendo frente a sus limitaciones, así como a los problemas que surgen al intentar realizar el seguimiento de un objeto.

1.3. Organización del Trabajo Fin de Grado

En el primer capítulo se ha introducido el objetivo básico del proyecto, repasando las tecnologías IPS más conocidas, y explicando de manera muy básica cómo se obtiene el posicionamiento con estas y se presenta nuestro caso de estudio.

El segundo capítulo explica más a fondo la tecnología UWB, su funcionamiento, algunos ejemplos de uso de la misma y se da a conocer las principales formas de calcular la posición que tiene esta tecnología.

En el tercer capítulo se presenta la tecnología Pozyx, que es con la que se ha trabajado durante la realización del proyecto. Se aclara cómo realiza la localización y se detallan las distintas partes que lo componen. Además, se define brevemente qué es Arduino.

En el cuarto capítulo vamos a hablar sobre el entorno en el cual se desarrollan las pruebas, se expondrán los resultados de dichas pruebas para distintos algoritmos y opciones de filtrados de posición, se explicará brevemente el *software Processing* y el uso que se le ha dado, y finalmente se aclarará qué algoritmo y qué filtrado se han elegido para realizar la simulación.

En el quinto capítulo se explica cómo se ha realizado el montaje de la simulación. Se describirán los elementos utilizados y su disposición, y se explicará los detalles más relevantes del código utilizado.

Para el sexto y último capítulo se presentarán las conclusiones finales del proyecto, así como unas posibles líneas futuras que se utilizarían en caso de proseguir con el trabajo presentado.

2. Tecnología UWB

El UWB es una de las tecnologías más precisas y prometedoras de la actualidad, que se basa en la trasmisión de pulsos cortos y, utilizando diversas técnicas, provocan la dispersión de la energía de radio, en una frecuencia de banda ultra amplia. [3]

La transmisión de ráfagas de señales más corta hace que sea más fácil de medir el principio y el final de los pulsos, facilitando el cálculo del tiempo de desplazamiento de estas ondas y por ende el cálculo de la distancia entre dos dispositivos UWB.

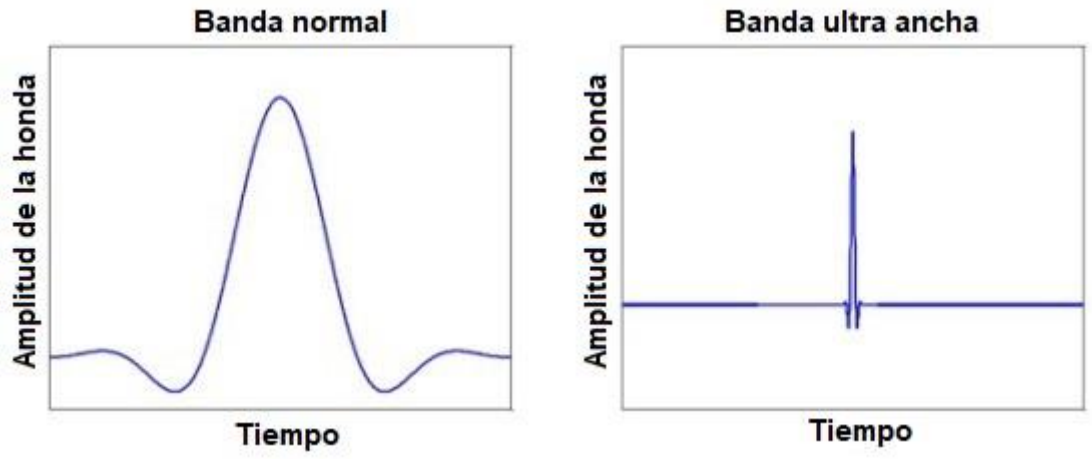

Ilustración 2: señal estándar y señal UWB [15]

La baja frecuencia de pulsos UWB hace posible que la señal pase a través de obstáculos de manera efectiva.

Las principales áreas donde se aplica el uso de UWB son: comunicación y sensores, posicionamiento y seguimiento y radar [4]. Este trabajo se centra en el segundo, dado que las tecnologías UWB pueden dar seguimiento de precisión en interiores en tiempo real.

Actualmente existen varios IPS disponibles comercialmente que utilizan UWB. Un ejemplo es el sistema Ubisense. En un sistema Ubisense, un usuario lleva etiquetas que transmiten señales UWB a sensores fijos que usan las señales para determinar la posición del usuario usando el método de tiempo de llegada (TOA) [5], el cual se explica más adelante.

Otro uso de UWB, en este caso militar, es Alereon que se ha diseñado para contratistas de defensa y agencias gubernamentales para permitir la integración inalámbrica de equipos y objetos de posicionamiento. El sistema de posicionamiento UWB de Alereon proporciona información sobre dispositivos, armas y teléfonos inteligentes, y facilita la detección de soldados. [6]

La tecnología UWB tiene distintas características, entre ellas, la velocidad de datos que puede llegar hasta los 100 megabits por segundo (Mbps), convirtiéndola en una buena solución para la transferencia de datos. Su ancho de banda junto a los pulsos extremadamente cortos, facilita reducir el efecto de interferencia multitrayecto y facilita el cálculo de los TOA (tiempo de llegada) cuando se trata de transmisiones de ráfagas entre receptor y transmisor correspondiente. Además a diferencia de otras tecnologías, como infrarrojos o ultrasonido, no requiere una visibilidad directa. Estas características hacen que UWB sea una mejor opción para el posicionamiento en interiores cuando lo que se requiere es un resultado lo más preciso posible. [7]

Hay que tener en cuenta que aunque UWB tiene grandes ventajas no es perfecto. Algunas de sus debilidades más claras es la distancia de transmisión, que es más corta respecto a otras tecnologías. Otro inconveniente es las interferencias que se pueden producir con sistemas cercanos.

2.1. Algoritmos de posicionamiento UWB

Para utilizar esta tecnología se han desarrollado distintos algoritmos de posicionamiento, los cuales se puede clasificar en cinco categorías: hora de llegada (TOA), ángulo de llegada (AOA), intensidad de la señal recibida (RSS), diferencia de tiempo de llegada (TDOA) e híbrido.

2.1.1. Algoritmo TOA

Esta técnica se basa en la intersección de los círculos que se crean entre el dispositivo que se desea posicionar y los dispositivos de referencia, siendo el radio de estos círculos la distancia entre cada par de elementos. Dicha distancia se obtiene calculando el tiempo de propagación de la señal de forma unidireccional [8]. Este algoritmo necesita la sincronización del tiempo de todos los transmisores.

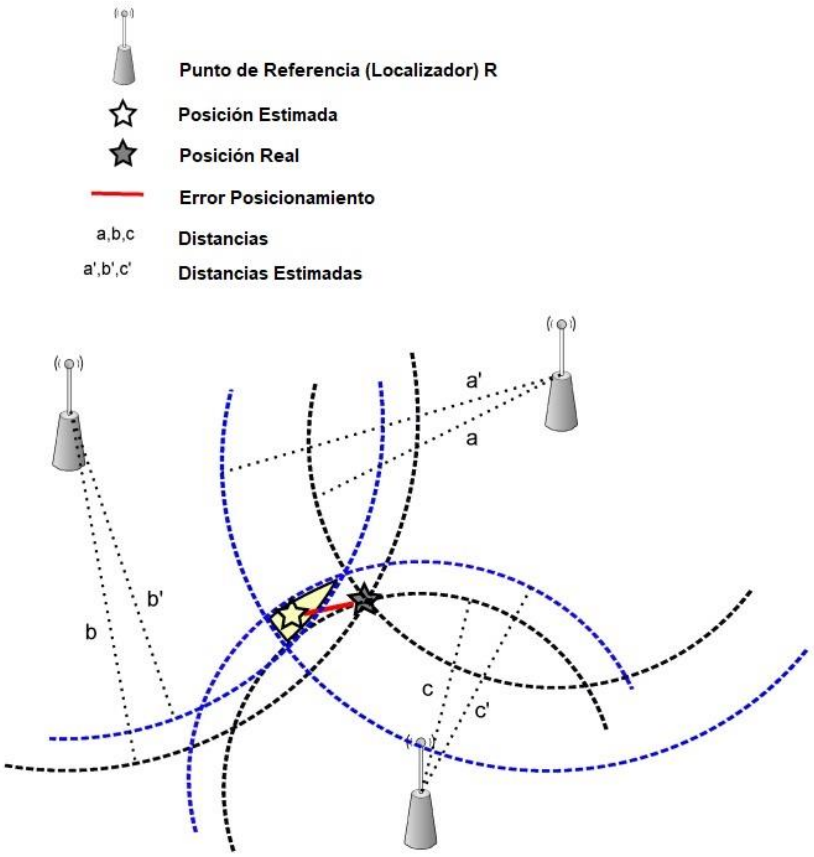

Ilustración 3: algoritmo TOA [16]

2.1.2. Algoritmo AOA

La técnica AOA, se basa en la estimación de los ángulos de recepción de la señal, se compara la amplitud de la señal con dicha estimación y la ubicación se calcula con la intersección de las líneas de ángulo de cada una de las fuentes utilizadas, de las cuales necesitamos al menos dos.

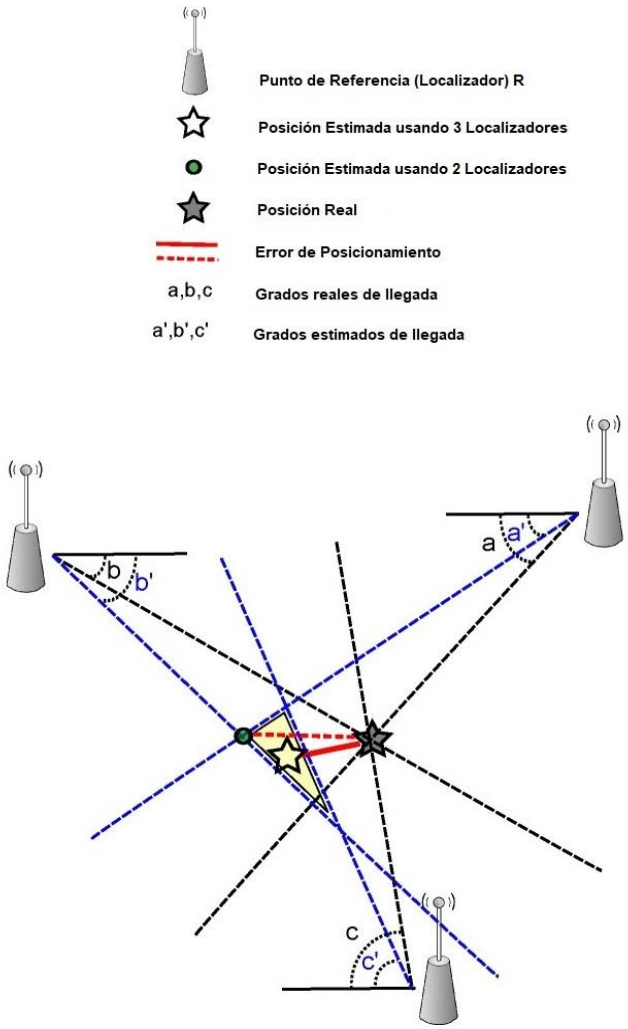

Ilustración 4: algoritmo AOA [16]

Estos algoritmos son muy sensibles a varios factores, como por ejemplo la distancia entre el emisor y receptor. Al aumentar la distancia puede disminuir la precisión del cálculo de la posición, por lo que no suelen usarse de forma aislada si no que se utilizan junto a otras técnicas para mejorar su precisión. Además estos algoritmos tienen una mayor complejidad en comparación a otros métodos.

2.1.3. Algoritmo TDOA

En el algoritmo TDOA, el objeto que se quiere localizar envía una señal a cada uno de los receptores, la posición se obtiene en base a la diferencia de tiempo de llegada de la señal a cada uno de ellos, asumiendo que las posiciones de los receptores son conocidas.

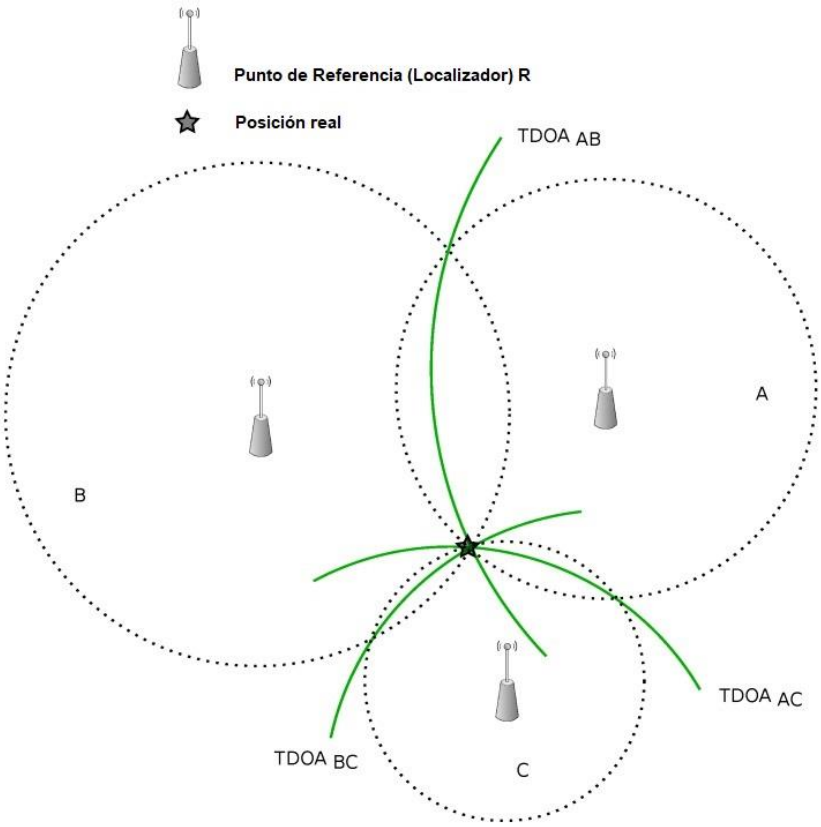

Ilustración 5: algoritmo TDOA [16]

Para ello es necesario una cooperación entre los receptores, los cuales tienen que compartir los datos para determinar la ubicación del objeto, dicha cooperación requiere un ancho de banda significativamente mayor que el resto de algoritmos.

2.1.4. Algoritmo RSS

En los algoritmos basados en RSS, el objeto del cual queremos saber la posición, mide la intensidad de la señal que recibe de distintos emisores. De esta forma, el objeto es capaz de aproximar la distancia que hay entre los emisores y el receptor basándose en la intensidad de la señal.

Este algoritmo no es el más adecuado para el posicionamiento en interiores, ya que la precisión de RSS en entornos sin línea de visión (NLOS), es baja. Sin embargo, este algoritmo puede resultar atractivo en algunos casos, ya que las etiquetas móviles, que son las que queremos localizar, actúan únicamente como receptores. De esta forma RSS tiene menos tráfico de comunicación, lo que ayuda a superar la limitación en el número de etiquetas en uso.

2.1.5. Algoritmos híbridos

Estos algoritmos combinan los antes mencionados, para aliviar algunos de los puntos débiles que tienen por separado. Hay que tener en cuenta que aunque la precisión aumenta, también lo hace la complejidad y el coste de estos algoritmos.

3. Pozyx y Arduino

En este capítulo se detalla el funcionamiento de Pozyx, dando a conocer el *hardware* que lo compone y se presenta la plataforma Arduino.

3.1. Sistema Pozyx

Pozyx es una solución hardware que proporciona un posicionamiento preciso e información de movimiento utilizando tecnología UWB. Los elementos que componen a Pozyx son dos: por un lado los *tags* o etiquetas, que son los dispositivos que se desea localizar y por otro lado están los *anchors* o anclas, que son los dispositivos que se utilizan como punto de referencia y que permanecen inmóviles en una posición conocida.

Para poder realizar el posicionamiento se necesita:

- Mediciones: Pozyx calcula la distancia entre los *anchors* y los *tags* midiendo el pulso de la señal.
- Puntos de referencia: Para poder saber la posición relativa de un objeto se utilizan puntos de referencia. Como se ha mencionado, los puntos de referencia son los *anchors*.

Para el cálculo de la posición se utiliza la trilateración [2], que mide la distancia desde el *tag* hasta, al menos, tres *anchors*. En base a la distancia medida, se obtienen varios círculos, uno por cada *anchor* como centro. El objeto se encontraría en la intersección de los tres círculos.

Esta técnica tiene como dificultad que las mediciones no son perfectas, ya que siempre habrá ruido en las mediciones, provocando que los círculos no tengan un solo punto de intersección. Para solventar este problema, en lugar de utilizar un único punto de intersección, se trata de encontrar el punto que está más cerca a todos los círculos. Este proceso es equivalente al algoritmo TOA antes mencionado.

Pozyx recomienda áreas de 35 metros cuadrados como máximo para obtener los mejores resultados, siempre teniendo en cuenta que cualquier obstáculo o material que interfiera en las mediciones puede reducir el alcance de las mismas.

En términos de precisión Pozyx ofrece dos posibles algoritmos de posicionamiento. El primero de ellos utiliza únicamente las medidas de UWB, el cual está diseñado específicamente para funcionar en los casos en que la medición se haga sin línea de visión al objetivo o casos donde la posición se obtenga solo una vez o se tarde varios segundos entre cada medición. El segundo algoritmo, denominado algoritmo de rastreo usa, además, datos de la Unidad de Medición Inercial (IMU) junto

a información de mediciones UWB. Esta opción se recomienda para cuando los movimientos del objeto que se vaya a localizar sean continuos.

Para validar la precisión del sistema, Pozyx realiza pruebas exhaustivas en un espacio de oficina de 7 por 9 metros con una persona sujetando el *tag*, se obtiene que la media de error horizontal en condiciones en las que si tenemos línea de visión (LOS) son de 92 milímetros (mm) y en condiciones de NLOS, la que se consigue bloqueando la visión de dos *anchors* con metal, son de 140 milímetros (mm) [2].

A pesar de los resultados similares de ambos algoritmos, se resuelve que el algoritmo de rastreo proporciona una trayectoria más suave. Esto se debe a que este algoritmo utiliza estimaciones de posiciones anteriores.

3.1.1. Anchor

Los *anchors* son los elementos que proporcionan a la etiqueta la información necesaria para localizar el objeto. Este elemento está formado por una placa que posee un emisor-receptor de señal UWB montado sobre una cubierta que facilita su montaje.

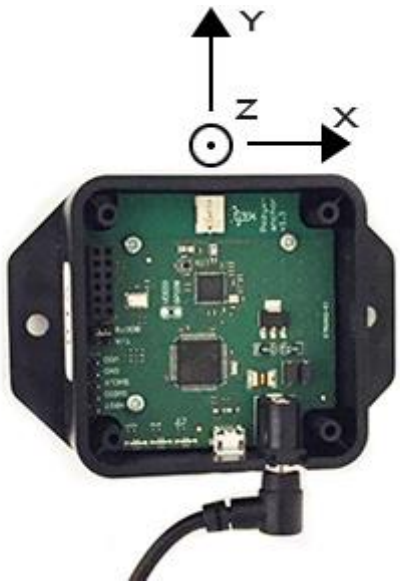

Ilustración 6: Anchor de Pozyx [2]

En su página, Pozyx ofrece unas recomendaciones sobre el montaje de estos elementos para obtener un óptimo funcionamiento.

- Posicionarlos lo más alto posible en la habitación y en LOS para incrementar la posibilidad de recibir una buena señal.

- Distribuirlos alrededor del usuario de tal forma que cubra el mayor número de ángulos, evitando a toda costa fijarlos en línea recta, ya que esto provocaría un aumento en el error de posición.

- Colocar los *anchors* verticalmente con la antena en la parte superior. Dicha antena emite señales en todas las direcciones del plano horizontal, pero funciona peor en el plano vertical, por ello se recomienda la posición vertical de la misma. Esta regla sirve también para los *tag.*

- Para realizar el posicionamiento en tres dimensiones (3D) se recomienda colocar los *anchors* a distintas alturas entre ellos y en una posición más elevada que la de los *tags*, pero esta regla es menos relevante, debido a que las placas están equipadas con un altímetro.

Para el posicionamiento 3D hacen falta cuatro *anchors*, en posiciones fijas. Pozyx nos ofrece la posibilidad de calcular de forma automática la posición de estos, pero se aconseja medir a mano las posiciones, ya que de esta forma los cálculos son más precisos.

3.1.2. Tag

El *tag* o etiqueta es el segundo dispositivo del conjunto Pozyx. Esta placa es la que utiliza los *anchors* para su localización. También se puede utilizar para averiguar la posición de un segundo *tag,* usando para ello su transmisor-receptor UWB.

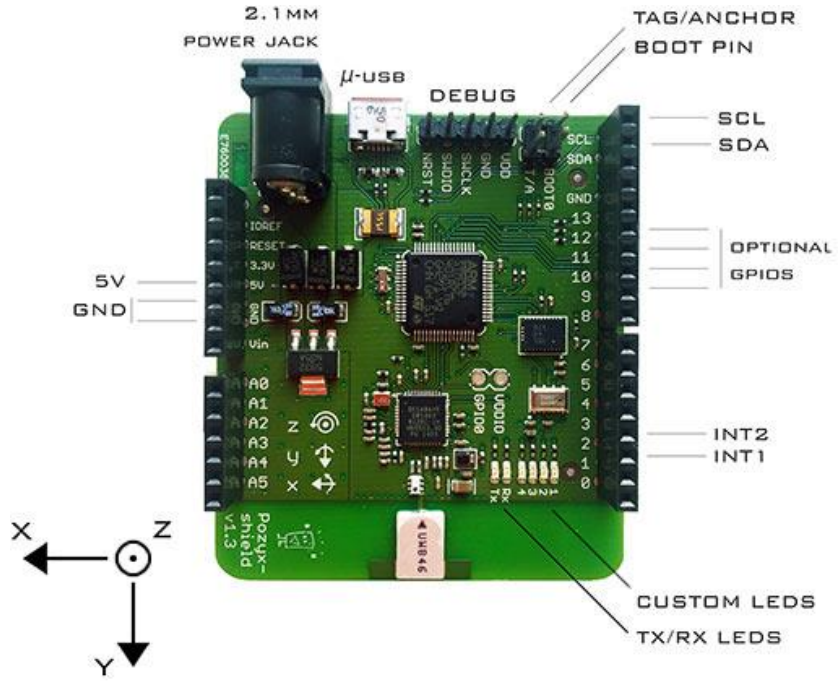

Ilustración 7: *Tag* de Pozyx [2]

3.2. Plataforma Arduino

Para trabajar con Pozyx hay dos opciones: utilizar el lenguaje *Python* desde un PC o la plataforma Arduino. Se ha optado por Arduino por la sencillez que ofrece y porque es una solución de *software* y *hardware* abierto.

Arduino es una plataforma sencilla que ayuda a interactuar con el entorno del usuario. Una de las ventajas de este software es que es multiplataforma, la mayoría de sistemas microcontroladores están limitados a *Windows*. Entendemos por microcontrolador como un circuito integrado programable, capaz de ejecutar órdenes grabadas en su memoria con el objetivo de automatizar o controlar algún proceso [9].

Para desplegar programas en Arduino se utilizar su propio entorno de desarrollo (IDE). Los programas Arduino están separados en dos secciones: la sección *setup* y la sección *loop.* La primera es un bloque de código que se ejecutará solo una vez cuando arranca el programa en el Arduino, y está pensado para la inicialización de variables. Por otro lado la sección *loop* desarrolla la principal funcionalidad de los programas. Como su propio nombre indica, está pensada para ejecutarse en bucle mientras que el programa está en ejecución.

Dicho código se carga en las placas Arduino, que están capacitadas para leer señales de entrada de sensores, botones y otros elementos y transformarlos en salidas, como activar motores, encender luces, etc.

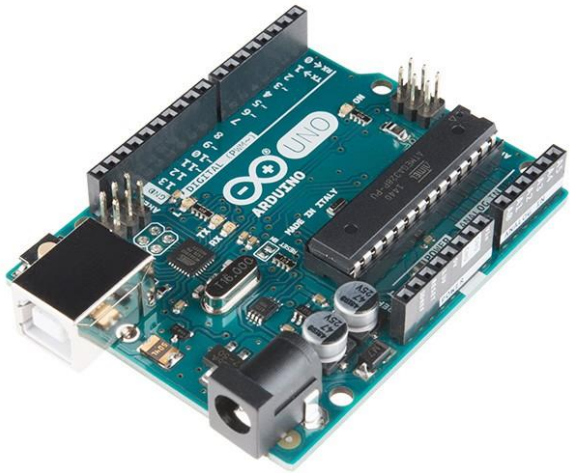

Ilustración 8: placa Arduino [10]

Estas placas se diseñaron en el Instituto de Diseño e Interacción Ivrea [10] como una herramienta para el prototipado, orientado a estudiantes sin amplios conocimientos sobre electrónica o programación.

4. Entorno de pruebas

Las pruebas se han desarrollado en dos entornos: el primero es una habitación de 12 metros cuadrados, pero todo este espacio no era utilizable por lo que el tamaño con el que trabajan las mediciones es de 2 metros en el eje X y de tan solo 1.24 metros en el eje Y. Siguiendo las recomendaciones que nos ofrece Pozyx, los *anchors* se han dejado fijos en la pared y se ha medido a mano las distancias entre ellos, posicionándolos en el punto más alto posible del cuarto, que es de 1.85 metros. El segundo es un laboratorio de la Universidad Politécnica de Valencia (UPV), pero como la gran cantidad de medidas se han tomado en el primer entorno, utilizaremos los resultados obtenidos en este.

Se ha utilizado el paquete *Ready to Localize* que proporciona un *tag* y cuatro *anchors,* los cuales tienen identificadores únicos y que tienen la siguiente disposición en la habitación.

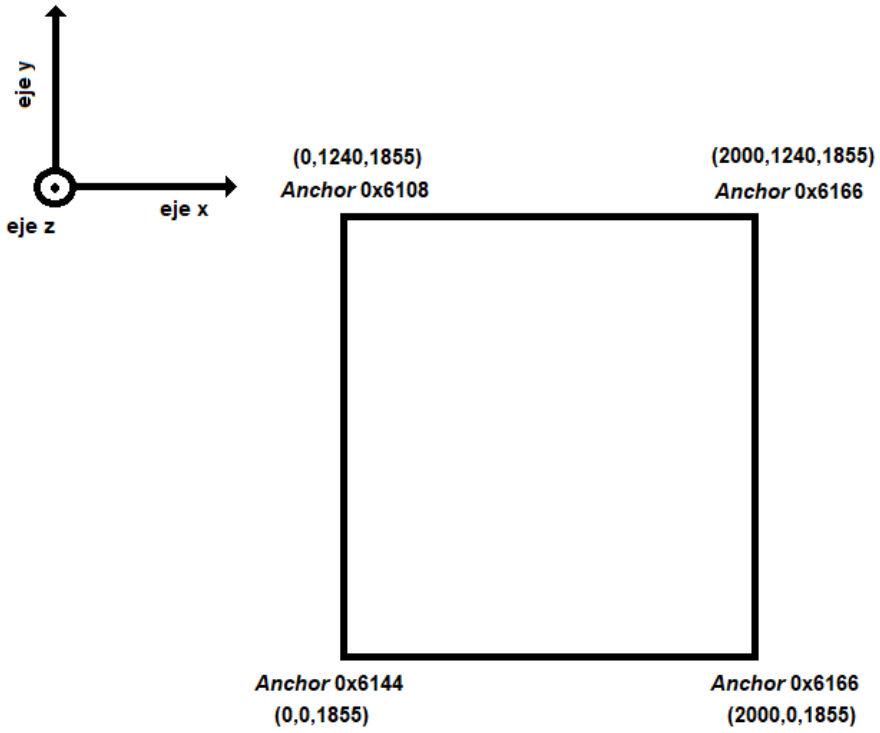

Ilustración 9: disposición de los *anchors* para las pruebas

Estas medidas son en mm, y las medidas se obtiene en referencia al *anchor* cuyo identificador es 0x6144 ya que se establece que está en la posición cero de ambos ejes horizontales.

Las pruebas se han diseñado con tres puntos de localización distintos: el punto central de la habitación que aproximadamente tiene los valores 1000 y 620 en los ejes X e Y respectivamente, y teniendo LOS con solo dos *anchors*. La esquina de la habitación, que está en 2000 en el eje X y en 1240 en el eje Y, es decir se intenta observar el comportamiento del sistema cuando la localización se realiza en un extremo de la habitación. Y por último un punto intermedio entre ambos, es decir un punto que esta entre el centro de la habitación y un extremo, en este caso es el punto que tiene como valores, 430 en el eje X y 500 en el eje Y, y teniendo LOS con todos los *anchors*. Hay que tener en cuenta que estas medidas están tomadas a mano, por lo que no son del todo precisas.

4.1. Posicionamiento sin filtrado de posición

Como se ha comentado anteriormente, para realizar el posicionamiento en *Pozyx* disponemos de dos posibilidades: el algoritmo que usa solamente mediciones UWB y el algoritmo de rastreo que usa mediciones de la IMU e información de mediciones previas. A continuación se realizaran distintas pruebas para observar y comparar los resultados con cada opción de posicionamiento y con cada uno de los tres puntos mencionados antes.

4.1.1. Posicionamiento con algoritmo UWB

El objetivo de las pruebas realizadas es encontrar que algoritmo obtiene los mejores resultados, para ello se establecen tres criterios clave:

- La estabilidad del algoritmo, es decir, cuan fuerte es ante el ruido.
- La precisión, cuanto se llega a acercar al punto real de posicionamiento.
- La velocidad con la que obtiene los resultados.

Con este primer algoritmo, el cual se recomienda usar cuando las medidas se obtienen cada varios segundos, se han registrado los siguientes resultados:

Posiciones	Valor medio		Valor mínimo		Valor máximo	
	Eje X	Eje Y	Eje X	Eje Y	Eje X	Eje Y
Centro (1000,620,0)	1126	630	1038	503	1255	712
Esquina (2000,1240,0)	1871	1330	1632	950	2010	1949
Intermedio (430,500,0)	347	511	340	493	407	583

Tabla 1: valores obtenidos con algoritmo UWB para una muestra de 19 valores

Observando la tabla se puede deducir que la precisión de este algoritmo, en el mejor de los casos obtiene un error de posicionamiento aproximado de ocho centímetros, si se trata de una ubicación que no está en los extremos del entorno de pruebas y con línea de visión clara entre los *anchors* y el *tag.*

Sin embargo si se trata de realizar una localización sin línea de visión entre alguno de estos o intentado calcular el posicionamiento en algún extremo del entorno, el error aumenta hasta trece centímetros aproximadamente.

También hay que tener en cuenta los valores máximos y mínimos de las mediciones, ya que como podemos notar, en algunos casos los valores están muy alejados de la posición real. Esto se puede deber al ruido que hay en la habitación, a la posición en la que se coloca el *tag,* al margen de error de las mediciones reales, ya que puede ser que no sean del todo exactas, etc.

Como conclusión de este algoritmo podemos decir que aunque su precisión sea bastante adecuada cuando hay LOS y la velocidad de obtención de resultados es casi insignificante, pocos milisegundos, su estabilidad no es adecuada, hay muchas variaciones en las medidas debido al ruido. Por lo que si queremos usar este algoritmo para nuestro caso de estudio necesitaríamos un filtrado de las posiciones para mejorar su estabilidad.

4.1.2. Posicionamiento con algoritmo de rastreo

Con este segundo algoritmo, recomendado para mediciones de movimientos continuos, se registran los siguientes resultados en las mismas condiciones que la prueba anterior:

Posiciones	Valor medio		Valor mínimo		Valor máximo	
	Eje X	Eje Y	Eje X	Eje Y	Eje X	Eje Y
Centro (1000,620,0)	1001	619	1000	617	1002	621
Esquina (2000,1240,0)	2056	1264	2054	1262	2058	1268
Intermedio (430,500,0)	429	501	428	500	430	503

Tabla 2: valores para algoritmo de rastreo con una muestra de 19 valores

Como se puede ver a simple vista en este caso se obtiene unos valores mucho mejores que antes. Esto se debe en gran parte al uso que hace el algoritmo de las mediciones previas.

Sigue siendo notorio que el error aumenta según nos acercamos a los bordes del entorno de pruebas, pero en el resto de los casos el posicionamiento es casi perfecto.

4.1.3. Resumen

Comparando los resultados obtenidos se podría concluir que se debería utilizar el algoritmo de rastreo para el caso de estudio, debido a que se esperarían menos errores en el posicionamiento. Sin embargo, hay otro factor que no se ha tenido en cuenta en estas mediciones: el desplazamiento del objeto localizado.

El caso de estudio considera un plató de televisión, por lo que se deduce que el objeto que queremos localizar es una persona que aunque no se mueva de forma continua, sí se desplazará por el plató.

Los resultados de todas las pruebas se han obtenido dejando el *tag* fijo en las posiciones, es decir que no se ha tenido en cuenta el posicionamiento en movimiento.

Para medir este factor se ha dejado el tag en una posición inicial de la habitación y se ha empezado a obtener la localización, pasado unos segundos movemos el tag hasta otra posición final que queramos localizar y se ha observado cuanto tiempo tarda en algoritmo en obtener la posición final.

Aquí es donde surge el problema del algoritmo de rastreo, ya que este tiene un retraso en el tiempo de obtención de la nueva posición final A pesar de que sigue obteniendo mediciones a la misma velocidad que antes, son mediciones que se acercan a la posición final poco a poco. Es decir no es una medición directa como la que hace el algoritmo UWB.

Este retraso del tiempo se debe al uso que hace de las posiciones anteriores, y es exponencial, es decir, cuanto más tiempo permanezca el tag en una posición más le costará obtener la siguiente haciendo que este algoritmo sea inservible para el caso de estudio.

Por ello la solución más adecuada es utilizar el primer algoritmo pero con la ayuda de algún filtrado de las posiciones, para así evitar los errores provocados por el ruido mejorando su precisión.

4.2. Posicionamiento con algoritmo UWB y filtrado de posiciones

Para la mejora de la estabilidad del algoritmo de posicionamiento se han probado tres soluciones de filtrado, que permiten combinar varias muestras de la señal obtenida por Pozyx para obtener así un valor con mayor precisión.

4.2.1. Filtro de media móvil

Este filtro es muy utilizado porque su implementación es sencilla, es intuitivo y rápido de calcular. Para su cálculo se utilizan N valores, este valor N se le denomina "ventana", y se calcula su media, obteniendo como resultado una señal que elimina parte del ruido. Dicho tamaño de ventana tiene gran influencia en el filtro.

$$\overline{X} = \frac{\sum_{i=1}^{n} X_i}{n}$$

Ilustración 10: fórmula del filtro de media móvil [18]

Para mejorar la eficiencia de este filtro, se usa un *buffer* circular. Dicho *buffer* se utiliza para eliminar la necesidad de recorrer todos los elementos para calcular el nuevo valor filtrado. Solo es necesario restar el primer valor de la ventana y sumar el nuevo.

La importancia del tamaño de la ventana reside en que cuanto mayor sea la misma, más grande será el suavizado de la señal. Sin embargo, aumentar este tamaño puede tener consecuencias negativas.

Una de estos aspectos negativos es que podemos perder datos reales, por ejemplo un movimiento rápido del objeto localizado podría interpretarse en este filtrado como una señal de ruido. Otro aspecto negativo es que a medida que la ventana crece, también lo hace el desfase entre la señal original y la filtrada, ya que la introducción de un nuevo dato correspondiente a la nueva posición no podrá contrarrestar inmediatamente a los N-1 valores anteriores.

Utilizando un tamaño de ventana de cinco, se han obtenido estos resultados:

Posiciones	Valor medio		Valor mínimo		Valor máximo	
	Eje X	Eje Y	Eje X	Eje Y	Eje X	Eje Y
Centro (1000,620,0)	1027	632	1017	501	1174	704
Esquina (2000,1240,0)	2030	1242	2012	1235	2042	1256
Intermedio (430,500,0)	426	525	403	500	478	548

Tabla 3: valores para ventana N=5 con filtro de media móvil para 19 valores

Posiciones	Valor medio		Valor mínimo		Valor máximo	
	Eje X	Eje Y	Eje X	Eje Y	Eje X	Eje Y
Centro (1000,620,0)	1016	618	983	601	1215	689
Esquina (2000,1240,0)	2012	1237	1998	1227	2027	1250
Intermedio (430,500,0)	423	518	412	501	457	532

Tabla 4: valores para ventana N=10 con filtro de media móvil para 19 valores

Como era de esperar, todos los valores mejoran su precisión respecto a usar UWB sin filtrado. Los valores medios para ambas ventanas no varían demasiado, así que para decantarse por una de estas ventanas hay que prestar mayor atención a los valores máximos y mínimos. Se puede observar que, con la ventana de diez valores, las mediciones se acercan más al valor idóneo.

También se debe considerar el tiempo que tarda el filtrado en ofrecer los valores cuando se realiza un posicionamiento, para la ventana de diez valores el tiempo se retrasa hasta en veinte décimas de segundo para cada valor filtrado, mientras que para la ventana de cinco valores, el tiempo apenas se ve afectado, tarda entre cinco y quince décimas de segundo.

4.2.2. Filtro de paso bajo exponencial

El filtro exponencial EMA consiste en obtener un valor filtrado a partir de una medición de valores aplicando esta fórmula:

$$A_n = \alpha M + (1 - \alpha) A_{n-1}$$

Ilustración 11: fórmula de cálculo para filtro EMA [18]

Siendo A_n el valor que obtenemos con el filtrado, $A_{n\prime 1}$ el valor filtrado anterior, M el valor de la muestra que se desea filtrar y α un factor entre cero y uno.

Como se puede observar este filtro aporta una solución parecida al algoritmo de rastreo ya que usa información del valor filtrado anterior suavizando la señal filtrada. Este suavizado depende del factor alpha que utilicemos.

Este filtro se puede emplear como un filtro de paso bajo, es decir, un filtro que marque un límite de frecuencias que pueden pasar, pudiendo eliminar el ruido de alta frecuencia, lo cual es idóneo para mejorar la medición de nuestra señal.

Hay que considerar que el factor α condiciona el filtro. Con un valor de α igual a uno, la señal filtrada es igual a la señal medida, es decir, no se filtra el valor, ya que se prescinde de la segunda parte de la ecuación. Por otro lado para valores de α cercanos a cero, el valor resultante dependerá más de la historia previa que de la muestra. Por tanto, a medida que el valor de α se acerca a cero, el suavizado de la señal aumenta. Sin embargo hay dos aspectos negativos en la disminución de α, por una parte podríamos eliminar frecuencias que fueran de interés y por otra se aumenta el tiempo de respuesta del sistema. Los valores habituales de α utilizados varían entre 0.2 y 0.6.

En nuestro caso, después de probar distintos valores de α, se ha obtenido los mejores resultados tanto en precisión como en tiempo de respuesta con un valor de 0.1, estos resultados son:

Posiciones	Valor medio		Valor mínimo		Valor máximo	
	Eje X	Eje Y	Eje X	Eje Y	Eje X	Eje Y
Centro (1000,620,0)	1002	624	1001	6019	1004	626
Esquina (2000,1240,0)	2001	1243	1997	1240	2004	1246
Intermedio (430,500,0)	429	500	426	498	430	503

Tabla 5: valores de para alpha = 0.1 con filtro de paso bajo para 19 valores

Como podemos observar este filtro obtiene mejores valores que el filtro anterior, tanto en los valores medios como en los máximos y mínimos, y el tiempo que tarda el filtrado es similar al que tarda el filtro de media móvil para N igual a 5.

4.2.3. Filtro de mediana móvil

Tanto el filtro media móvil como el filtro EMA utilizan como estimador de tendencia la media, la cual es poco robusta. Por ello aparecen filtros que utilizan la mediana en su lugar.

Este filtro se asemeja al primero que se ha descrito, ya que utiliza también una ventana N de valores y el valor filtrado es la mediana de estas mediciones. Los tamaños de ventana utilizados suelen ser de entre 5 y 11 valores, teniendo en cuenta dos aspectos: que cuanto mayor es la ventana más suave es la señal y más tarda en reaccionar a los cambios y que computacionalmente es más costoso, debido a que hay que realizar una ordenación de los valores de la ventana.

El tamaño de las ventanas suelen ser número impares para evitar hacer una media entre los dos valores centrales, ya que se perdería parte de la información.

También hay que considerar que este filtro solo puede devolver valores muestreados, provocando que la señal resultante sea menos suave comparada con otros filtros.

Los siguientes resultados se obtuvieron con una ventana de 11 valores, que obtiene los mejores resultados para este filtro sin incrementar demasiado el tiempo de filtrado.

Posiciones	Valor medio		Valor mínimo		Valor máximo	
	Eje X	Eje Y	Eje X	Eje Y	Eje X	Eje Y
Centro (1000,620,0)	1004	620	998	615	1008	622
Esquina (2000,1240,0)	2004	1244	1996	1238	2014	1250
Intermedio (430,500,0)	437	496	427	489	450	503

Tabla 6: valores para ventana N=10 con filtro de mediana móvil para 19 valores

4.3. Resumen

De todas estas pruebas se puede deducir que principalmente podríamos usar dos filtros para nuestra simulación, el filtro exponencial con valor de α de 0.1 y el filtro de mediana móvil con un valor para la ventana de 10.

Por tanto, la pregunta clave es cómo decidir cuál de estos dos filtrados utilizar, ya que ambos obtienen valores similares en precisión y en tiempo de obtención de los valores. Aquí es donde entra en juego el uso de *Processing.*

Processing es un entorno de programación que se utiliza para facilitar el desarrollo de aplicaciones gráficas, el cual está basado en Java y que tiene un entorno de desarrollo integrado [11].

En nuestro caso solo se ha utilizado esta herramienta para mostrar las posiciones de los *anchors* y el *tag y* para poder visualizar de una manera más intuitiva los valores que se obtienen con los filtros.

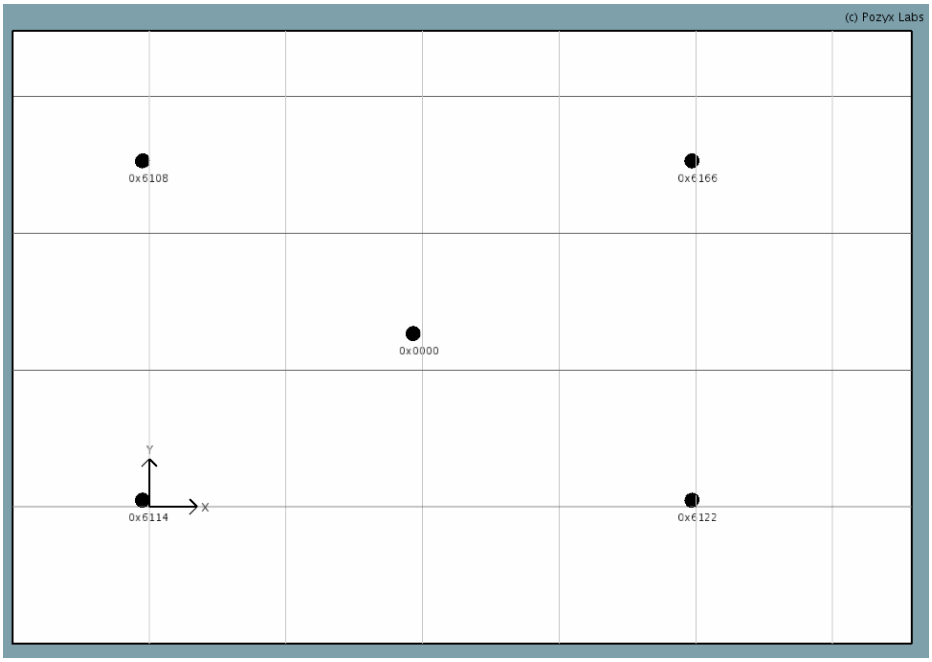

Ilustración 12: aspecto del entorno de pruebas en *Processing*

Volviendo a la pregunta de qué filtro utilizar, se utilizó *Processing* para observar cómo se comportaban ambos filtros cuando se está realizando un movimiento continuo.

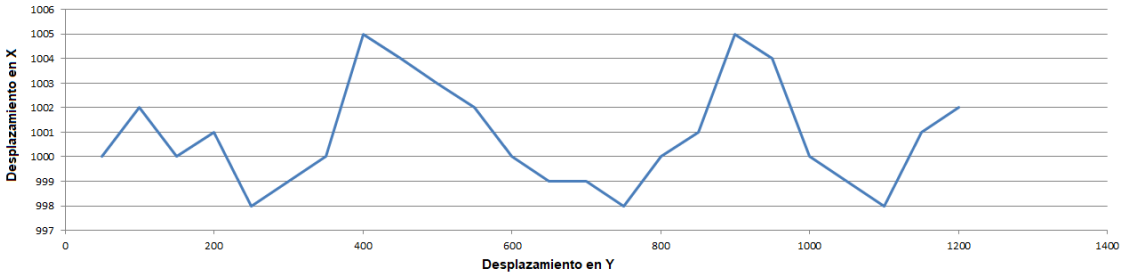

Ilustración 13: línea de desplazamiento para filtro de mediana móvil

Esta ilustración es el resultado de medir un desplazamiento en línea recta en el eje Y, como se observa, los cambios de posición con este filtro producen variaciones bruscas, por lo que el movimiento de la cámara será poco fluido.

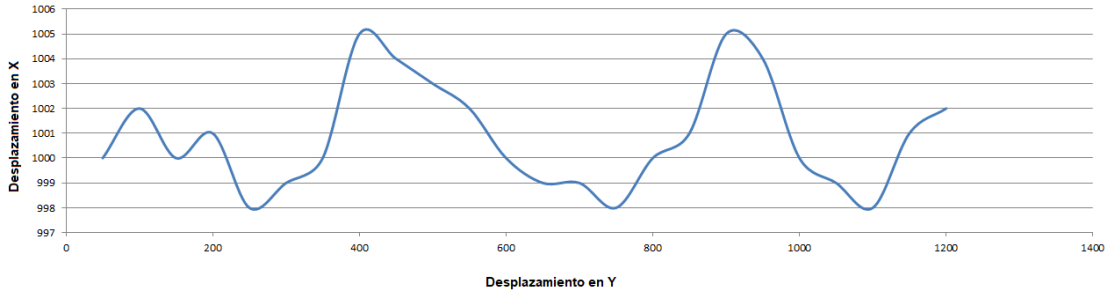

Ilustración 14: línea de desplazamiento para filtro EMA

Con este filtro, los movimientos conseguidos son más suaves gracias a que, el valor que resultado del filtrado es una media.

Se llega a la conclusión de que el filtro de mediana se ve afectado negativamente por el hecho de devolver solo puntos registrados de las mediciones como su valor de filtrado.

Este punto es clave para nuestra simulación ya que necesitamos un filtrado que además de ser preciso sea también lo más suave posible, porque en la realidad es improbable que se realicen movimientos rectilíneos. Esto nos lo ofrece el filtrado EMA, por lo que se ha decidido que es la mejor opción.

5. Simulación del caso de estudio

Para realizar una simulación del caso de estudio propuesto en un entorno real, se ha utilizado el mismo sistema descrito anteriormente y algunos elementos hardware extra. Se han utilizado 4 *anchor* y 1 *tag,* 1 placa Arduino Uno, como la de la ilustración 8, dos servomotores y una pequeña cámara, además de usar un segundo *tag*.

Los servomotores son motores eléctricos con dos características especiales. La primera es que permiten mantener una posición indicada por el usuario, en el rango de giro del servo que va desde -90º hasta 90º, ofreciendo un total de 180º de giro. La segunda característica es la posibilidad de controlar la velocidad de giro, es decir, se puede ajustar el tiempo que tarda el servo para moverse entre una posición y la siguiente [12]. Estos servos requieren, para controlarlos desde una placa Arduino, una batería extra.

Ilustración 15: Servomotor MG995 [19]

En esta ocasión se han realizado las pruebas en una habitación que permite establecer un espacio para realizar el posicionamiento más grande. Los anchors están ahora posicionados a una altura de 2.5 metros y como antes forman una caja.

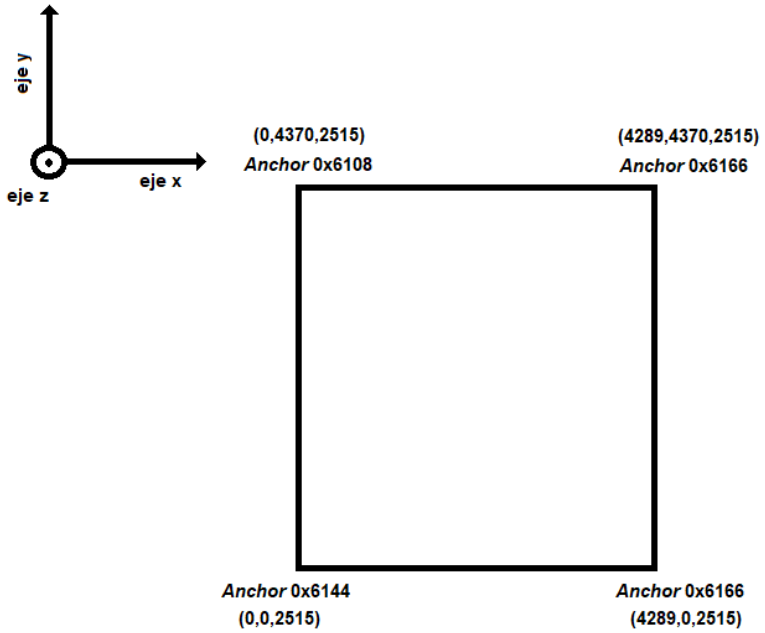

Ilustración 16: disposición de los *anchors* para la simulación

Gracias a como está fabricado el *tag* de Pozyx, el montaje de este se hace directamente sobre la placa Arduino, este *tag* será el que proporcione la información que utilizará la placa para controlar la posición de los servomotores.

Ilustración 17: *tag* montado sobre placa Arduino Uno [2]

Para completar el montaje hardware se han utilizado dos servomotores, uno que realiza el giro en horizontal y otro que realiza el giro en vertical. El servo que gira en vertical se monta sobre el que gira en horizontal y junto a este se coloca la cámara. De esta forma se consigue que la cámara pueda enfocar en un rango de 180º en horizontal, y 180º en vertical con respecto a su posición.

Ilustración 18: montaje de servomotores y camara

Los servos y la cámara están junto a la placa Arduino y tag, y se pueden colocar en cualquier lugar de la escena, pero siempre teniendo en cuenta que su orientación debe ser la que tiene la dirección del eje Y positivo. El segundo tag, que es al que apunta la cámara, se deberá colocar delante de la cámara. El espacio por el que puede moverse el segundo tag se muestra con un cuadrado naranja en el siguiente esquema:

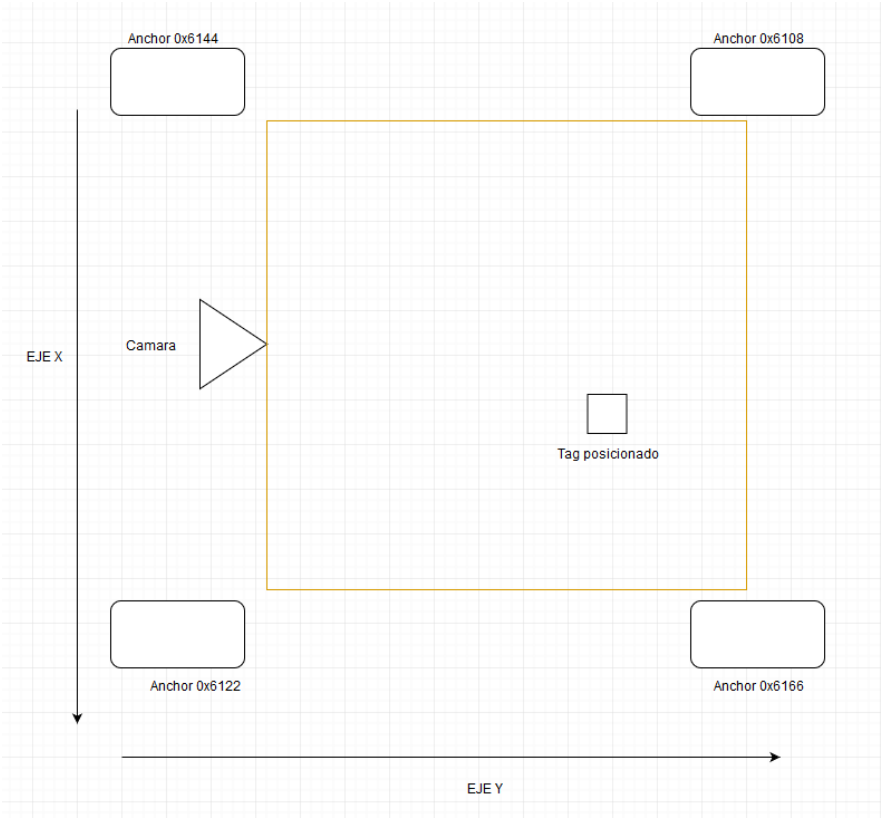

Ilustración 19: esquema de la simulación

En cuanto a la parte de software, se ha utilizado como base el código *Multitag positioning* que proporciona Pozyx. El cambio básico que hay que realizar es el de la posición de los *anchors* y el de los identificadores de los *tags*. Una vez hecha esta modificación, ya se puede realizar la localización. Para realizar el filtrado de posicionamiento, en este caso se aplica el filtro de paso bajo para cada una de las variables que nos devuelve Pozyx: X, Z e Y, tanto para las posiciones del tag que vamos a posicionar como el que esta fijo junto a la cámara. Este último se filtra por si deseamos cambiar la posición de la cámara.

Estos valores, indicados en milímetros, hay que tratarlos para usarlos en los servos. Para ello se ha aplicado el teorema del coseno o ley de cosenos, que es una generalización del teorema de Pitágoras en los triángulos rectángulos en trigonometría. [13]

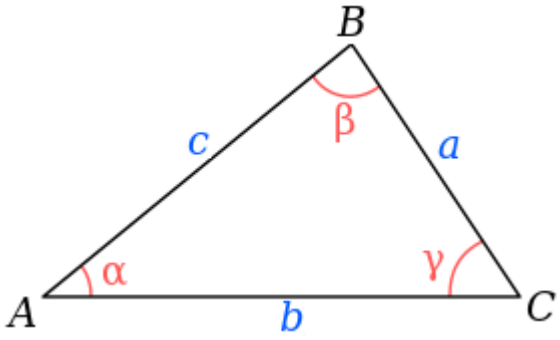

Ilustración 20: notación habitual de un triángulo [20]

El teorema del coseno dice que dado un triángulo ABC, como el de la ilustración 19, siendo α, β, γ, los ángulos, y a, b, c los lados opuestos a estos ángulos, se cumple que:

$$c^2 = a^2 + b^2 - 2ab\cos\gamma$$

Ilustración 21: fórmula del teorema del coseno [21]

A partir de esta fórmula se obtiene que:

$$\alpha = arc\cos\frac{b^2 + c^2 - a^2}{2bc}$$

Siendo b el resultado de restar la coordenada X del *tag* fijo y la coordenada X del *tag* en movimiento. Realizar la misma resta con las coordenadas Y nos da el valor para la variable a, y por último obtenemos que c es el valor de la hipotenusa que se forman entre el eje de coordenadas, que será los valores de X e Y del tag que está junto a la cámara y los valores a y b antes calculados.

Por último, el valor α obtenido en esta fórmula, que es el valor que queremos que se desplacen los servos, hay que transformarlo en grados para poder ser usado por los servos, para ello simplemente multiplicamos este valor por 180 y lo dividimos entre 3.14. Para el cálculo de los grados para el giro vertical, solo hay que considerar que el valor de b ahora será el que nos proporciona la coordenada Z, el resto de los cálculos son similares.

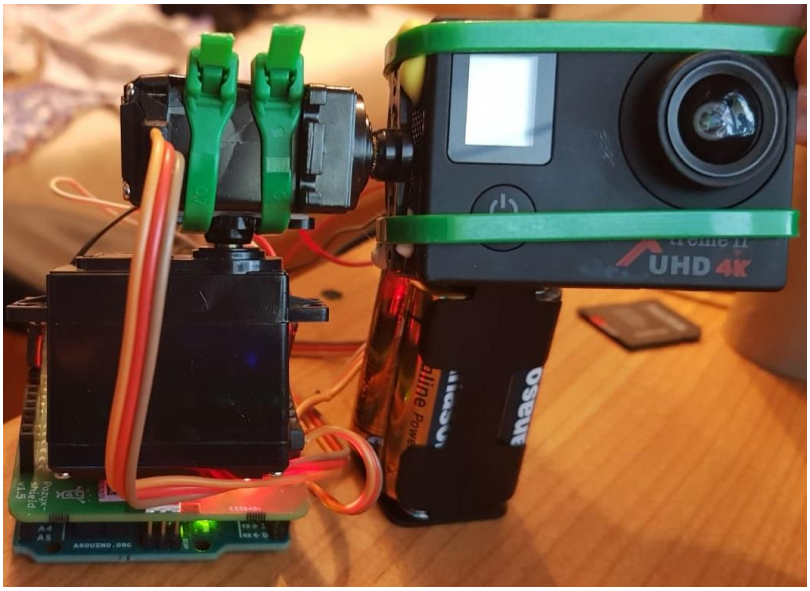

Ilustración 22: montaje final de servos y cámara junto al *tag*

Hay que tener dos consideraciones en este montaje, el primero es que la resta entre las coordenadas Y siempre será positiva, ya que suponemos que el tag móvil va a estar siempre enfrente del fijo, el cual estará siempre junto a la cámara y a la placa Arduino, como se ve en la ilustración 22. Esto se hace porque el servo no puede girar más de 180 grados por lo que el espacio que abarca será solo lo que este delante de él. El segundo aspecto es que hay que diferenciar cuando la resta entre las coordenadas X sea positiva o negativa, debido a que se tendrá que calcular de forma similar pero en distintos cuadrantes. Con esto conseguimos que la cámara apunte siempre a las coordenadas que nos ofrece Pozyx, tanto verticales como horizontales.

Ilustración 23: posición inicial de la simulación

Al principio de la simulación hacemos que la cámara apunte hacia delante en línea recta, para comprobar que los servos responden de manera adecuada antes de seguir con la simulación. En este caso se ha posicionado la cámara cerca de uno de los bordes, esto es simplemente por comodidad.

En cuanto el tag recibe los datos de la primera localización, la cámara se mueve inmediatamente hacia el objetivo, en este caso el objetivo soy yo mismo.

Ilustración 24: posición de la cámara apuntando al objetivo

Tal y como se ha dicho, el seguimiento no se hace solo en horizontal sino que también lo hace en vertical, como se ve en la ilustración 24. Aunque a esto se le dará menos uso, ya que la distancia desplazada en vertical no será tan grande como para que se note.

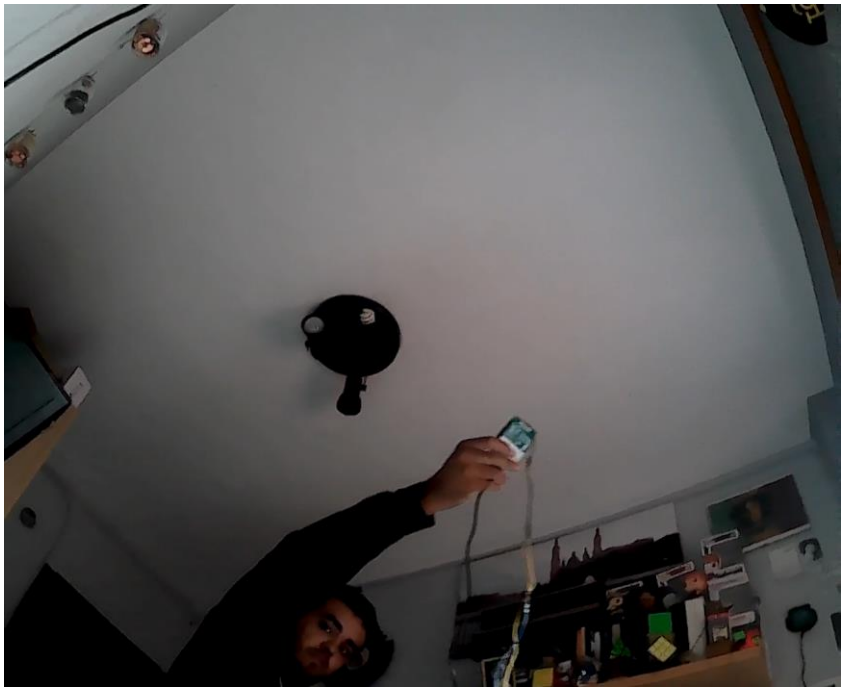

Ilustración 25: desplazamiento del objetivo en vertical

Hay que tener en cuenta que al ser una simulación no va a ser perfecto, las posibles mejoras se comentan en el siguiente y último capítulo.

6. Conclusiones y líneas futuras

Como se ha podido observar a lo largo del trabajo, la tecnología UWB es una buena opción para realizar el posicionamiento en interiores, pero no es perfecta. Hay que considerar que tiene un gran problema y ese es el ruido en las mediciones, que puede alterar en gran medida los resultados. Por esta razón, es altamente recomendable el uso de un algoritmo de filtrado para estas posiciones y evitar siempre que se pueda obstáculos metálicos en la línea de visión.

Gracias al sistema montado, también se llega a la conclusión que la distancia es un aspecto a tener en cuenta, ya que cuanta mayor distancia exista entre los *anchors* y el objeto localizado, o entre los dos *tags,* el alcance de la señal disminuye y por tanto se debilitará, incluso llegando a perderse por momentos.

En cuanto al uso práctico de esta tecnología, considero que es totalmente factible su uso en la actualidad. Con tan solo el uso de los 4 *anchors* y 2 *tags* se ha conseguido hacer un seguimiento considerablemente bueno de una persona y, por tanto con el uso de más *anchors* y con el uso de otros sistemas para evitar el ruido se podría escalar este trabajo a un plató de televisión.

Si se llegase a continuar con este trabajo, el primer punto a mejorar sería el ruido. Con el filtrado utilizado se evita en gran medida, pero no es perfecto por lo que habría que buscar otras medidas de filtrado u otros algoritmos que ofrecieran mejores resultados.

Otro punto a mejorar seria la agilidad de los servos, es decir, poder introducir un parámetro de forma manual en vivo para que el filtrado de la posición varié según si queremos que sea más o menos rápido, teniendo en cuenta siempre que esto afectará a la precisión del mismo. También se debería poder incluir un parámetro para modificar la precisión de los servos, ya que los servos tienen una precisión de un grado, es decir, varían su posición para el grado 90, 91, 92, etc. Esto en ocasiones puede ser un comportamiento no deseado, ya que estas variaciones en la cámara se pueden ver como movimientos erráticos. El parámetro comentado serviría para decirle a los servos que en lugar de moverse con variaciones de un grado, lo haga con variaciones del número de grados indicado. Por ejemplo darle un valor de 2 haría que si estamos en el grado 50 no tome en cuenta los valores 49 y 51 haciendo que estos tres grados enfoquen al mismo punto. Hay que tener cuidado con esto ya que si estamos en movimiento y tenemos un valor de 2, 3 o mayor el movimiento dejaría de ser fluido.

Bibliografía

[1] https://developer.apple.com/ibeacon/ (15 de julio de 2018)

[2] https://www.pozyx.io/ (23 de julio de 2018)

[3] Ghavami M., Michael LB, Kohno R. Ultra-Wideband Signals and Systems in Communication Engineering. John Wiley & Sons, Ltd; Newark, NJ, EE.UU, 2006.

[4] Siwiak K., McKeown D. Ultra-Wideband Radio Technology. John Wiley & Sons, Ltd; Newark, NJ, EE.UU, 2005.

[5] http://www.ubisense.net/ (20 de julio de 2018)

[6] http://www.alereon.com/?page_id=2992 (25 de julio de 2018)

[7] Svalastog MS Cand Scient Thesis. Universidad de Oslo; Oslo, Noruega: 2007. Posicionamiento en interiores: tecnologías, servicios y arquitecturas.

[8] Reddy N., Sujatha B. Computación TDOA usando modulación multiportadora para redes de sensores. En t. J. Comput. Sci. Commun. Netw. 2011.

[9] https://es.wikipedia.org/wiki/Microcontrolador (14 de agosto de 2018)

[10] https://www.arduino.cc/ (14 de agosto de 2018)

[11] https://processing.org/ (18 de agosto de 2018)

[12] https://programarfacil.com/tutoriales/fragmentos/servomotor-con-arduino/ (22 de agosto de 2018)

[13] https://es.wikipedia.org/wiki/Teorema_del_coseno (3 de julio de 2018)

[14] http://www.elmundo.es/economia/2015/01/22/54bff268268e3efa718b4583.html (8 de septiembre de 2018)

[15] https://www.sewio.net/technology/ (8 de septiembre de 2018)

[16] Lyudmila Mihaylova, Byung-Gyu Kim, Debi Prosad Dogra, (2016). Ultra-Wideband Indoor Positioning Technologies: Analysis and Recent Advances. Sensors (Basel).

[17] https://www.luisllamas.es/arduino-filtro-media-movil/ (1 de agosto de 2018)

[18] https://www.luisllamas.es/arduino-paso-bajo-exponencial/ (1 de agosto de 2018)

[19] https://electrocrea.com/products/servo-motor-mg995 (8 de septiembre de 2018)

[20] https://es.slideshare.net/Licarcres/el-triangulo-elementos-notacion (8 de septiembre de 2018)

[20] https://www.universoformulas.com/matematicas/trigonometria/teorema-coseno/ (8 de septiembre de 2018)