UAV

La idea de nuestro proyecto es montar un dron al cual definamos un trayecto, escoja el camino más apropiado, realice el recorrido y una vez hecho el recorrido aterrice de manera precisa donde se le especifique. Además en situaciones concretas el dron será capaz de esquivar obstáculos a partir de la información de sus sensores.

PROJECT SPRINT #4
DATE: 26th May 2021

1529314 - Pau Riba Escobar
1459024 - Christian Espinosa Reboredo
1529184 - Arnau Balaguer Albareda
1528948 - Alberto Vallespín Herranz

Table of Contents

Project description 1

Electronic components 2

Hardware Scheme 3

Software Architecture 4

Amazing contributions 8

3D pieces 8

Simulation Strategy 12

Foreseen risks and contingency plan 25

References 26

Sprint #4
Date: 19 May 2021

UAV

Project description

La idea de nuestro proyecto es montar un dron de 4 brushless motors, con cuatro sensores ultrasonido para evitar que se choque con elementos próximos a las hélices. Además el dron llevará equipado un acelerómetro y un GPS para poder ubicar y controlar apropiadamente el dron. Para poder aterrizar con precisión se le va a implementar visión por computador que va a reconocer el terreno y distancias del dron.

El objetivo principal del dron es que vuele, se pueda ubicar y mover según se le haya programado. La idea es dejarlo volando y según algoritmos de reconocimiento de patrones sepa donde y como tiene que aterrizar. El patrón estará en el suelo, bien visible para que el dron pueda reconocerlo a una altura decente. El dron se pondrá en vuelo programado y aterrizará automáticamente una vez llegue a cierta altitud o a las coordenadas cercanas a la marca (patrón) del destino.

Electronic components

This is the list of the used components:

- Motores y ESCs RS2205 2205 2300KV
- Hélices
- Arduino nano
- Batería
- MPU6050
- RaspberriPi
- RP Cam
- GPS
- Sensores Ultrasonido
- Impresiones 3d

Hardware Scheme

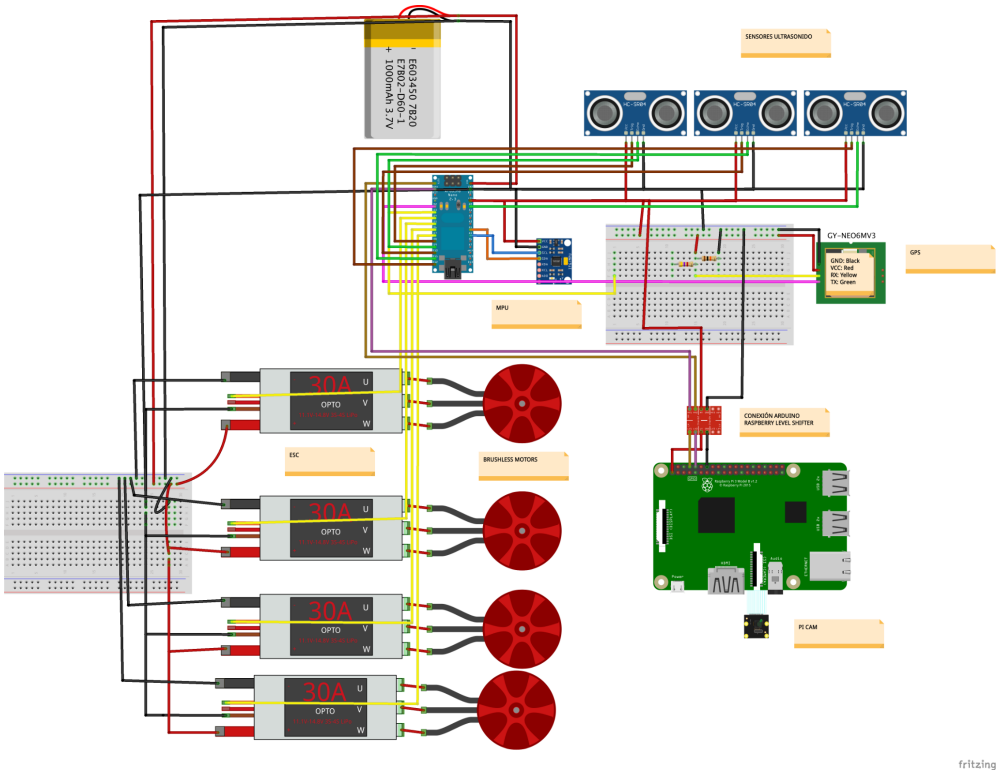

Fig 1: Esquema hardware

Hemos decidido reducir el número de sensores de ultrasonido de 4 a 3 ya que después de estar pensando con 3 nos serán suficientes para detectar posibles obstáculos, uno delantero y dos laterales. Como no iremos hacia atrás con el dron no necesitaremos uno trasero.

Software Architecture

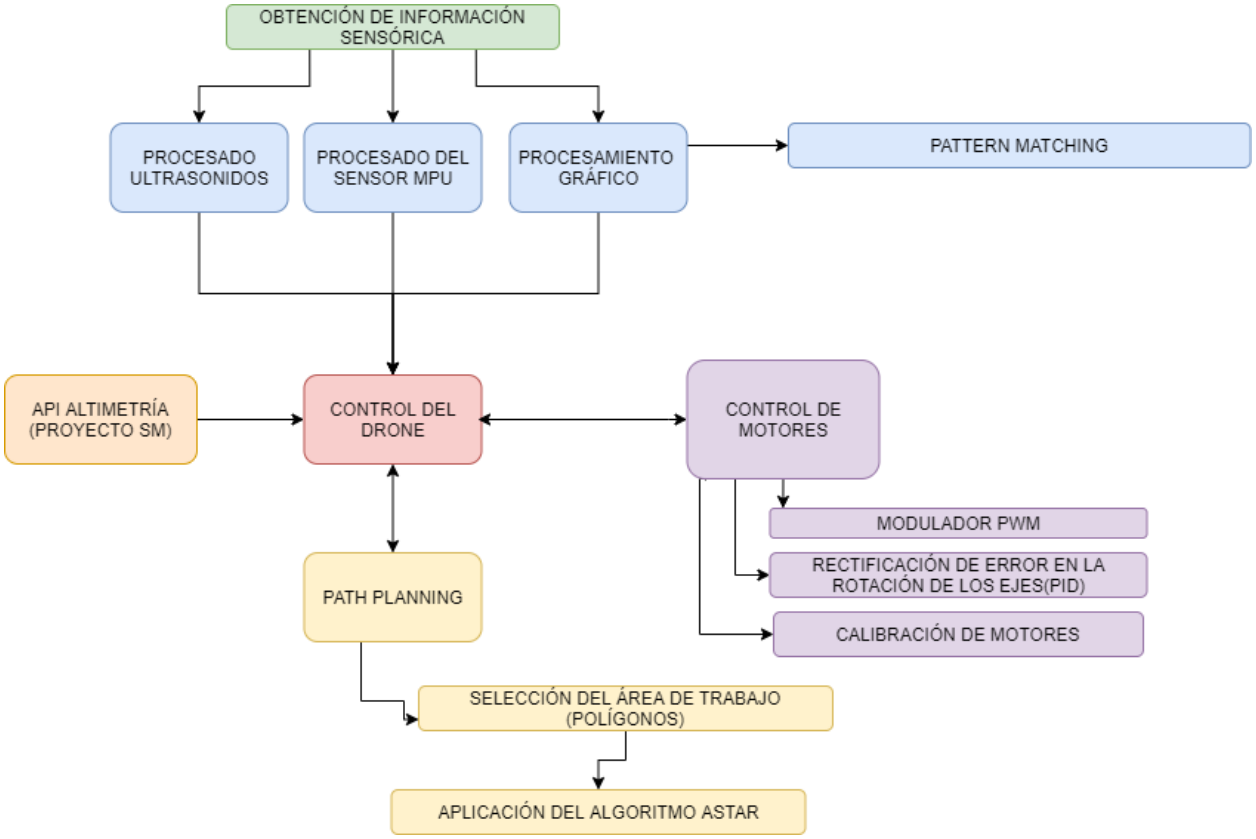

Fig 2: Arquitectura software

CONTROL DEL DRON*:*

Es nuestra unidad central de procesamiento y se encarga de comunicarse con el resto de módulos para garantizar las funcionalidades del dispositivo. Este consta de software que dependiendo de las necesidades de cada módulo adjunto, enviará las señales pertinentes al controlador de motores para que el dron pueda ejecutar las maniobras pertinentes.

Este código es crucial para que se pueda tener en cuenta todos los comandos de los diferentes módulos y dar prioridades dependiendo de las necesidades.

OBTENCIÓN DE INFORMACIÓN SENSÓRICA: a partir de los sensores de los que dispone nuestro dron obtendremos información para luego procesarla y corregir la ruta establecida por el *Coverage Path Planning* si es necesario.

Tendremos 3 módulos que se encargaran cada uno de ellos de preparar y procesar la información recibida para actuar en consecuencia. Estos tres módulos son:

- **PROCESADO DE ULTRASONIDOS:** Este aspecto de procesado de datos es el más importante y el que le vamos a dar más preferencia. Lo que detecten los sensores ultrasonido se tiene que tener en cuenta por delante de cualquier algoritmo para evitar colisiones de nuestro robot con objetos en el espacio. Si observamos con los sensores que está muy cerca de un objeto definido por un threshold, procederemos a hacer las maniobras del dron pertinentes para evitar la colisión. Esta decisión se tendrá en cuenta con los 3 sensores ultrasonidos distribuidos en el dron.

- **PROCESADO DEL SENSOR MPU:** Con el módulo MPU nos permite saber las coordenadas del dron. Este módulo es un poco impreciso, así que para saber la ubicación del dron en el espacio utilizaremos este módulo, pero cuando tengamos que hacer cosas que merezcan precisión tendremos que utilizar otros métodos. Por ejemplo en el sistema de aterrizaje, es importante implementar el algoritmo de reconocimiento de patrones para poder aterrizar exactamente donde se propone. Para este aspecto, como hemos descrito anteriormente se implementará en el proyecto de Visión por computadores, que describimos a continuación:

- **PROCESADO DE IMÁGENES:** en cuanto a este módulo vemos que se divide en otros dos módulos:

- **PATTERN MATCHING:** este módulo nos permitirá determinar si hemos o no encontrado la marca en la cual deberemos aterrizar.

- **RECTIFICACIÓN DE MEDIDAS DE DISTANCIA:** en primer lugar nos permitirá rectificar las distancias a objetos obtenidas mediante el sensor de ultrasonidos, y en segundo lugar nos permitirá determinar a qué distancia se halla la marca de aterrizaje si es que hemos hecho contacto visual con ella.

CONTROL DE MOTORES: en último lugar tenemos el módulo encargado de mover al dron. Utilizará toda la información recabada mediante la trayectoria y las medidas en tiempo real del dron para modular la velocidad y aceleración de los motores y así permitirnos movernos y acelerar. Está formado por 2 módulos que son los siguientes:

- **MODULACIÓN DE PWM:** será el módulo que nos permitirá mediante pulsos PWM controlar a los motores.
- **ESTABILIZACIÓN DE MOTORES:** Mediante PID este módulo nos permitirá calcular el error que hay en los parámetros de los motores y poder rectificar posteriormente.
- **CALIBRACIÓN DE MOTORES:** antes de poner el dron en funcionamiento debemos calibrar los motores para evitar que las pequeñas vibraciones que estos producen puedan suponer errores en la orientación y velocidad del dron.

En el caso de implementar el dron a nivel físico, se deberían hacer este tipo de tests. En nuestro caso hemos simulado el comportamiento del dron en un entorno controlado. Para cumplir estos tests hemos realizado simulaciones.

TESTS:

- **Prueba de ESTABILIDAD:**

  El dron será capaz de mantenerse en un sitio quieto sin hacer ningún movimiento.

  Es necesario destacar que todos los motores irán a la misma velocidad para conseguir aguantar el equilibrio.

- **Prueba de ATERRIZAJE:**

  El dron será capaz de aterrizar en el suelo sin ningún problema.

- **Prueba de ESTABILIDAD EN MOVIMIENTO:**

  El dron será capaz de mantener la estabilidad mientras haga movimientos , es decir, mientra rota, avanza etc.

- **Correcta detección de la maraca del suelo**

  Esta parte se está implementando en visión por computación

- **Prueba de EVITAR OBJETOS:**

  El dron será capaz de esquivar objetos sólidos que se le pongan por delante.

Amazing contributions

El dron puede actuar por sí mismo. Se le puede aplicar una ruta y a partir de entonces, ser totalmente autónomo con los objetivos que se le marquen hasta que aterrice.

Tendrá un sistema aterrizaje inteligente y sensores para poderse mover con seguridad y libertad en cualquier eje.

3D pieces

Utilizaremos la impresora 3D para poder hacer aquellas piezas de la estructura del dron.

En este caso vamos a crear una caja donde guardaremos algunos componentes del dron, como las distintas placas base etc...

Por otra parte, hemos creamos los distintos brazos del dron, donde colocaremos tanto las hélices como los distintos motores.

Finalmente, un conjunto de acoples para cada servomotor.

- Base del dron con componentes
- Brazos de los motores

Brazo (Soporte para motores y ESC)

Se han diseñado los brazos donde se pondrán tanto los motores brushless como los controladores ESC de los motores. Los 4 brazos del dron son exactamente iguales. Los agujeros donde irán atornillados los motores se han medido según la ficha técnica de los motores elegidos. Ver en la siguiente figura:

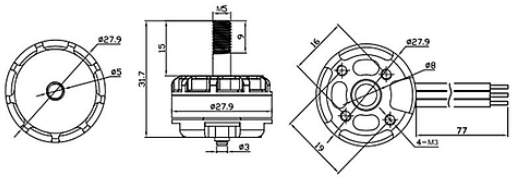

Fig 3: Ficha técnica motores RS2205 2205 2300KV

Podéis ver el proceso de creación del brazo:

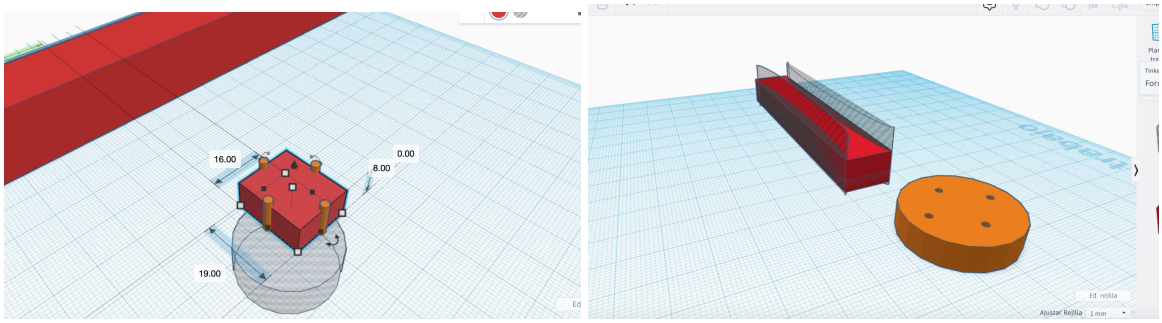

Fig 4: Proceso de creación

Se ha añadido una pata en el brazo para que el dron no caiga directamente al suelo y tenga cierta distancia con el suelo para proteger las hélices de posibles rocas u obstáculos. Como mejora en un futuro se podría añadir agujeros en el brazo tanto para aligerar el peso como para reducir el rozamiento con el aire y reducir el tiempo de impresión 3d.

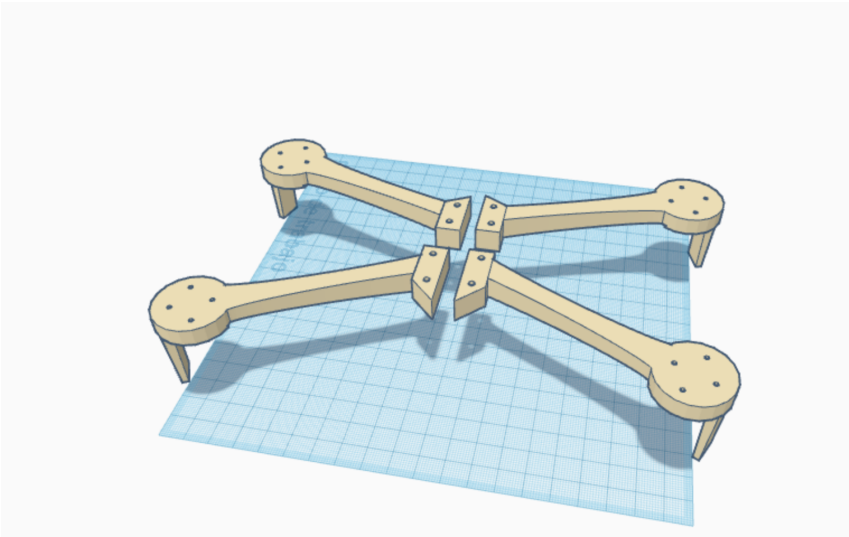

Fig 5: Diseño final de los cuatro brazos

Como mejora de los brazos hemos agregado un soporte en la parte final del brazo para poder atornillarlo a la base del dron. Con ello esperamos que ganemos cierta rigidez estructural y sea más fácil de imprimir ya que nuestra primera idea era hacer todo de una sola pieza.

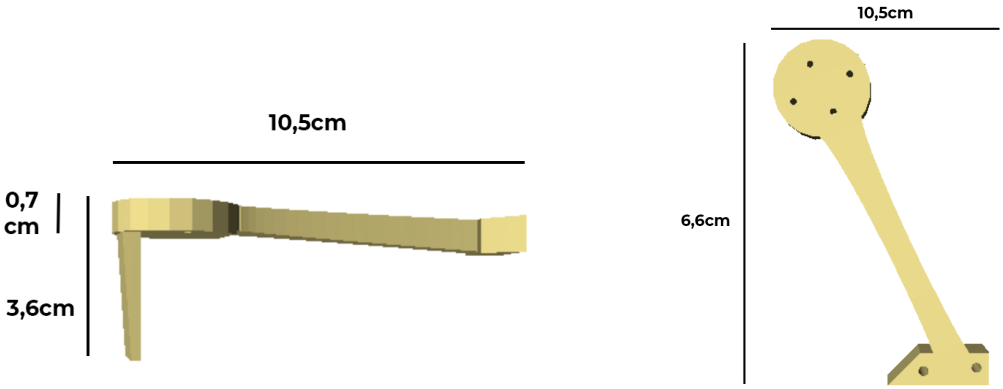

Fig 6: Alzado y planta archivo stl brazo

Base:

Se ha diseñado la base del dron, nexo de los 4 brazos y soporte para todos los componentes (arduino, raspberry, PiCam, batería Lipo, etc). Se han creado unas hendiduras dónde irán acopladas las patas y posteriormente atornilladas con dos tornillos cada una.

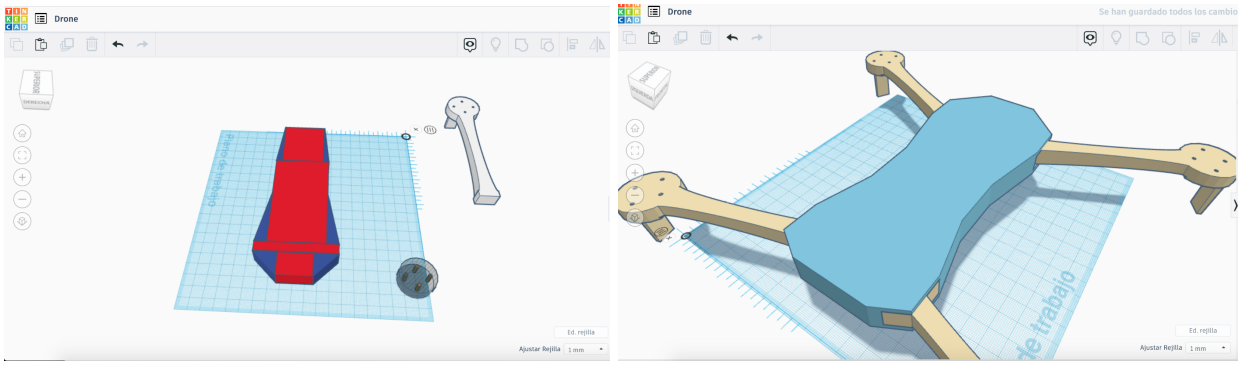

Fig 7: Proceso de creación de la base en tinkercad

La parte frontal de la base tiene un acabado más puntiagudo para favorecer la aerodinámica.

En la figura a continuación se puede observar el diseño de la base del dron, tiene una longitud de 16,7cm y una anchura de 7cm, suficiente para sostener tanto una raspberry pi como un arduino y la batería.

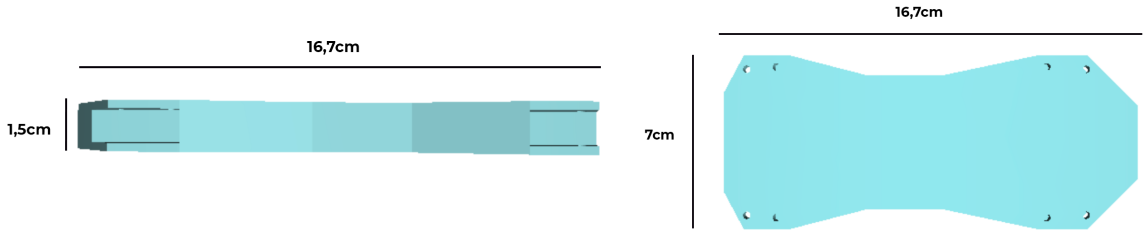

Fig 8: Vista alzada y planta de la base

Simulation Strategy

En nuestro caso vamos a utilizar un simulador llamado coppelia, que es con el que estamos trabajando en clase. Una de las primeras pruebas que queremos realizar en el simulador es conseguir que el dron se levante del suelo en una altura concreta y que sea capaz de desplazarse hasta un lugar indicado, de este aspecto se habla más adelante en el documento.

Para poder realizar las simulaciones ya tenemos implementado en el propio coppelia el objeto dron que lo vamos a modificar según las especificaciones de piezas hardware, especificadas anteriormente. De entrada hemos añadido 3 sensores para medir las distancias del dron a objetos/obstáculos cercanos.

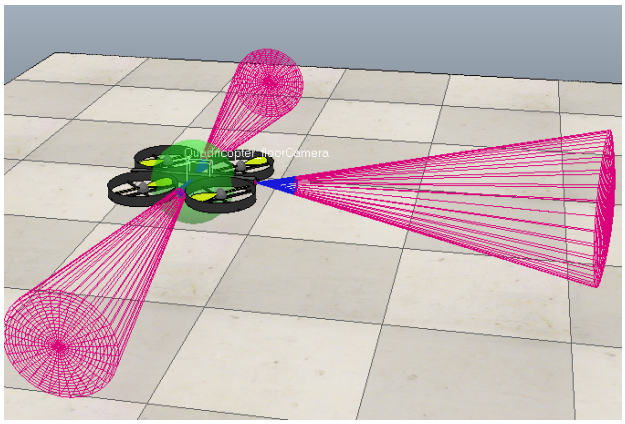

Fig 9: Visualización de los sensores en el dron

Este dron está compuesto por los distintos componentes:

- El dron consta de una cámara colocada en la parte inferior del dron para poder ver el suelo. Esta cámara nos devolverá las imágenes necesarias para poder controlar el descenso del dron en el aterrizaje. Las imágenes serán procesadas con los algoritmos del proyecto de VC.
- 3 sensores para captar objetos delante y a los lados del dron. Lo que se capte con estos sensores, como descrito previamente se le da la máxima prioridad, para evitar colisiones.

- La estructura del dron consta de bordes para que las hélices no colisionan con objetos muy próximos. Esta base se colocada en el medio del dron donde también encontramos el target (bola de color verde que se encuentra en el medio del dron).

La idea de simulación más simple para empezar, tenemos pensado que el dron sea capaz de levantarse hasta una distancia de 3m del suelo, a partir de allí, que avance en línea recta varios metros hasta llegar al destino indicado y finalmente mediante visión por computador detectaremos la marca de aterrizaje y el dron aterrizará según lo establecido.

PROYECTO VC

La idea de este proyecto de Visión por Computador es diseñar y crear el módulo de aterrizaje autónomo del dron UAV.

El objetivo principal es conseguir que el dron de manera autónoma consiga aterrizar encima de una plataforma dotada de una marca de identificación única. El dron se situará en las proximidades de la plataforma de aterrizaje gracias a las coordenadas gps anteriormente definidas, el problema en la vida real es que el gps tiene una precisión de unos cuantos metros.

Para ello se va a implementar un algoritmo de reconocimiento de patrones, específicamente patrones conocidos en el campo de la robótica llamados ArUco. El cual va a obtener imágenes en tiempo real de una cámara situada en la parte inferior del dron apuntando siempre al suelo.

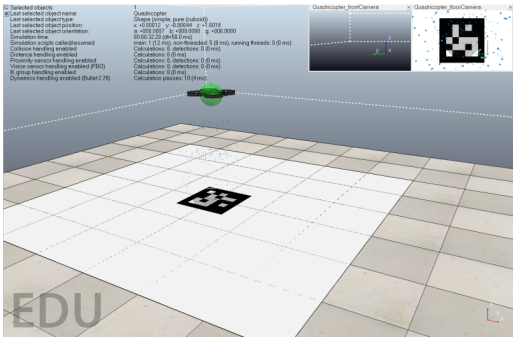

Fig 10: Dron coppelia con marcador ArUco

Con este tipo de patrones podremos definir tanto la orientación del dron como la posición y la altura del dron respecto al patrón. La idea es, que el dron pueda aterrizar de manera autónoma a pesar de ventiscas o cualquier tipo de error con odometria, el algoritmo devolverá la corrección del error que detecte tanto en un plano bidimensional X-Y como Z, que podrá definir la velocidad de aterrizaje del dron aproximadamente para evitar que se estrelle si va muy rápido.

Una vez obtenida toda la información extraíble del marcador se procederá a enviar las instrucciones pertinentes a los módulos de control del dron para mover los motores y con ello centrar el dron.

Una vez tengamos esta idea implementada, la idea es que al dron se le den dos puntos y busque la manera de llegar evitando los obstáculos utilizando los sensores de proximidad que lleva integrado.

Simulaciones Prácticas

Las simulaciones simples las hemos podido implementar:

- La primera consta de la creación de un PATH que hemos creado en coppelia. Coppelia da la opción de definir unos puntos en el espacio tridimensional que se unen por unas rectas automáticamente formando el path. La

simulación es compleja en cuanto a variedad de movimientos pero no consta de un código modular, ya que el path es definido en el simulador. No obstante, esta manera de programar la ruta para empezar es muy correcta.

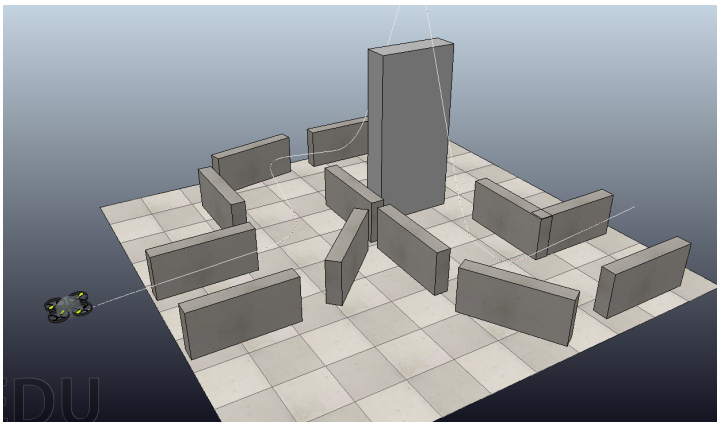

Fig 11: Path planning de Coppelia

Para realizar esta simulación sólo es necesario crear un script en el target del dron.En el target hemos creado una función para que siga el PATH mencionado anteriormente.Esta está formada por una subfunción del coppelia *sim.followPath*,donde se pasan como parámetros el path, el target, la posición del target y las velocidades.

- Hemos implementado un módulo software mediante la herramienta jupyter notebook que interactúa con el dron con la API . Este permite al dron evitar los obstáculos que se le pongan por delante hasta llegar a su destino. (Pre código antes de empezar a pensar en todas las situaciones del dron).

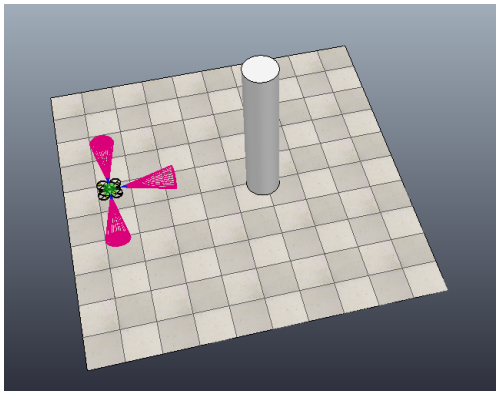

Fig 12: Simulación más simple

Algoritmo de Movimiento Autónomo por Sensores Ultrasonido

Cabe destacar que este algoritmo no ha sido reaprovechado de ningún sitio, ha sido formulado desde cero por el grupo para dotar al dron de movimiento mediante la sensórica de este, que consta de 3 sensores de ultrasonidos: uno frontal y dos a los laterales.

El módulo tiene como entradas 2 puntos: origen = [xo,yo,zo] y destino =[xd,yd,zd], el lugar de donde va empezar el dron el desplazamiento y la posición exacta donde va a aterrizar.

En primer lugar, a partir de la posición de aterrizaje o destino creamos el área de aterrizaje donde el dron tiene que llegar.

Una primera aproximación fue hacer que el dron llegue exactamente a la posición destino, pero eso tiene un problema debido a la forma en que movemos el dron. El dron se mueve en el coppelia siguiendo su target, así pues lo que estaremos moviendo será realmente el target, al cual no le aplicamos una velocidad para que se mueva sino que cambiamos su posición continuamente con pequeños cambios en las coordenadas x, y y z para que este se desplace . Esto provoca que el dron no esté pasando por todos los puntos, y puede ocurrir que no pasemos por el punto de destino, con lo cual nunca llegaríamos ahí.

El área de aterrizaje nos permite solucionar este problema.

Con el área de aterrizaje creada el dron se moverá en la línea recta que existe entre los puntos de origen y destino y **solo** si detecta uno o varios obstáculos los rodeará, y una vez rodeados volverá a la recta que une al origen con el destino.

Respecto a esto existe un problema, y es que los sensores tienen un rango limitado, en nuestro caso los sensores de ultrasonido tienen un rango en forma de cono. Este rango limitado supone que si un objeto queda fuera del rango del cono el dron pueda colisionar con él si es detectado.

Dejando este problema de lado vamos a ver qué maniobras va a hacer el dron en las diferentes situaciones que presenta la sensórica. Para mostrarlo de forma más gráfica vamos a ver una tabla donde tendremos como columnas los distintos sensores , en cada casilla tendremos **0 si no están activados, 1 si lo están**. Por último tendremos la columna de qué movimiento se debe realizar según la activación de los sensores.

Sensor Izquierdo	Sensor Derecho	Sensor Frontal	Maniobra
0	0	0	Seguir hacia delante
0	0	1	Girar a la izquierda
0	1	0	Seguir hacia delante
0	1	1	Girar a la izquierda
1	0	0	Seguir hacia delante
1	0	1	Girar a la derecha
1	1	0	Seguir hacia delante
1	1	1	Elevarse por encima del objeto

Tabla 1. Movimento del dron según la activación de sus sensores

Así pues observamos que:

- siempre que la parte frontal del dron esté despejada avanzaremos hacia delante,
- si tenemos un objeto en frente y ninguno a los lados tiraremos hacia la izquierda,
- si tenemos un objeto en frente y en alguno, y sólo uno, de los lados tiraremos hacia el lado donde no hay nada,
- y por último si tenemos objetos en todas direcciones nos elevaremos para sortearlos.

De esta forma ha quedado determinado el movimiento del dron de un punto inicial a un punto final mediante la sensórica de la que dispone.

Ahora para poder demostrar que este algoritmo funciona correctamente, hemos realizado distintas simulaciones:

Simulaciones

- 1: El dron es capaz de aumentar la altura en el caso de que no pueda ni avanzar hacia delante ni hacia los dos lados.

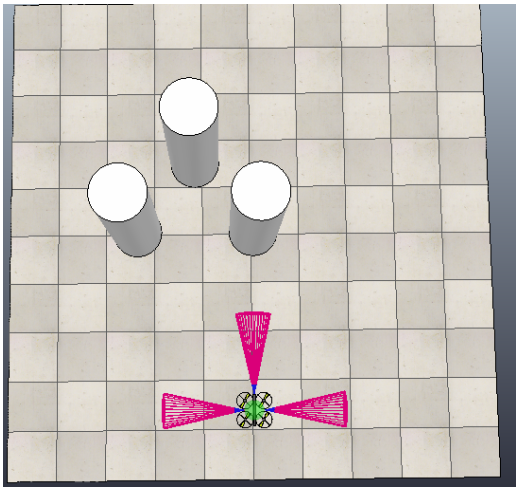

Fig 13: Simulación 1

- 2: El dron es capaz de evitar los dos obstaculas rodeando-los

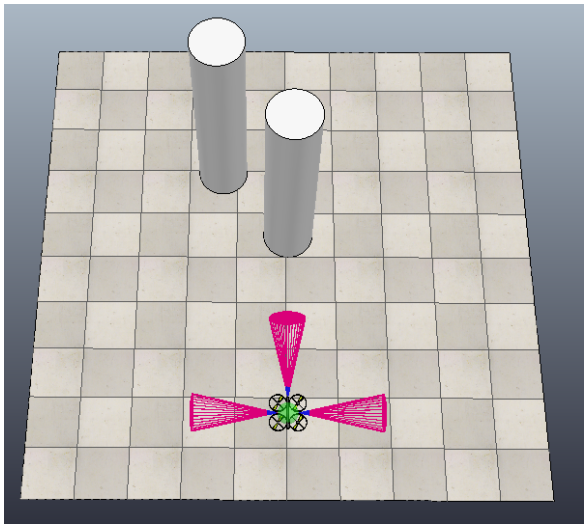

Fig 14: Simulación 2

- 3: El dron es capaz de rodear el cilindro que se encuentra en el medio de la escena.

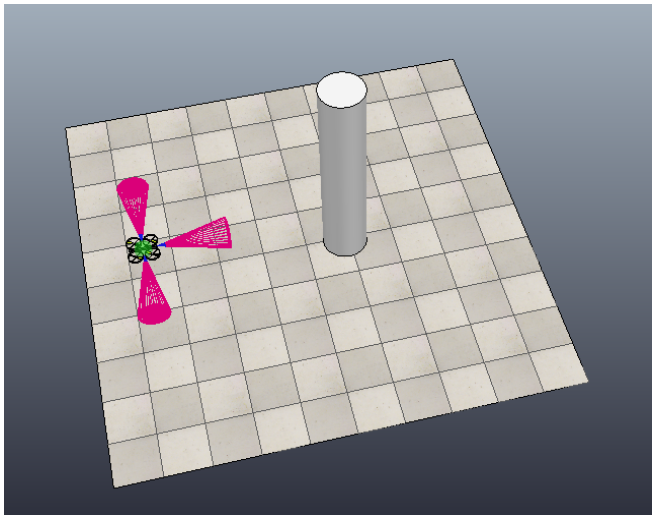

Fig 15: Simulación 3

- 4: El dron es capaz de pasar por el medio de los dos primeros cilindros y finalmente rodear el último.

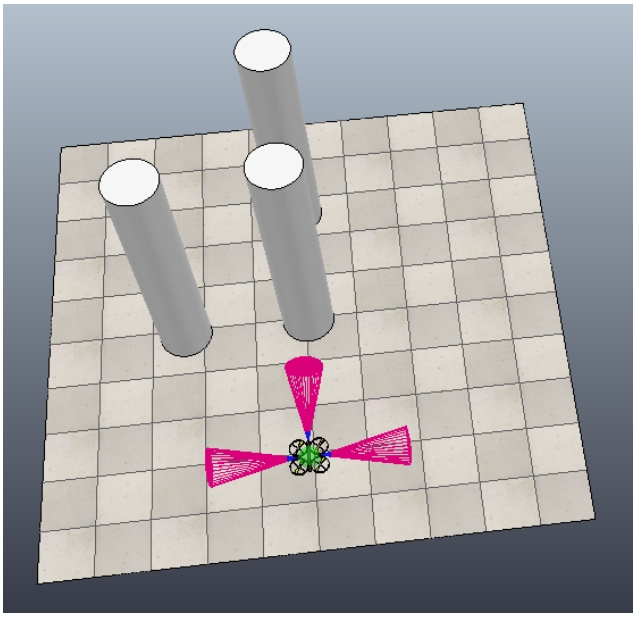

Fig 16: Simulación 4

- 5: El dron es capaz de rodear la línea de cilindros.

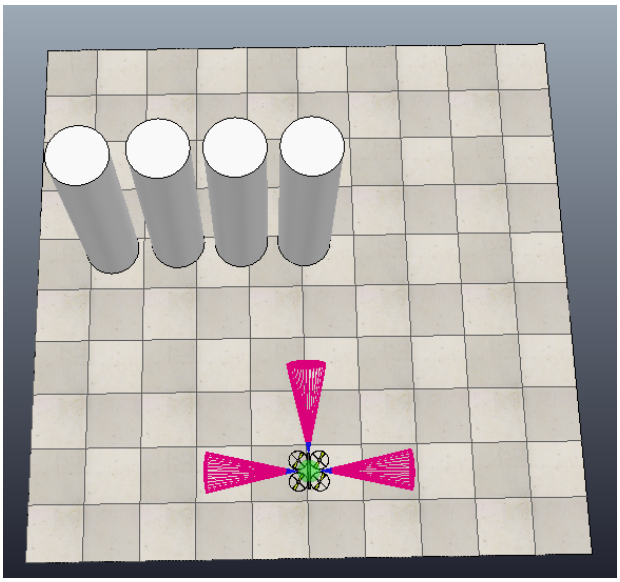

Fig 17: Simulación 5

- 6: El dron es capaz de rodear toda la hilera de cilindros.

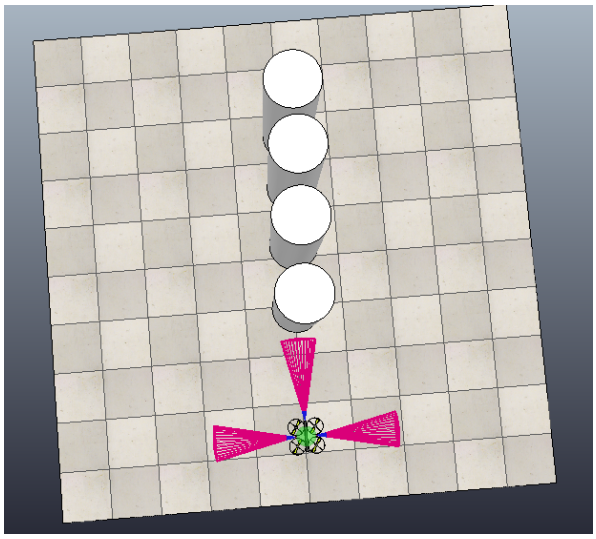

Fig 18: Simulación 6

- 7: El dron es capaz de evitar los distintos árboles que se encuentran en la simulación.

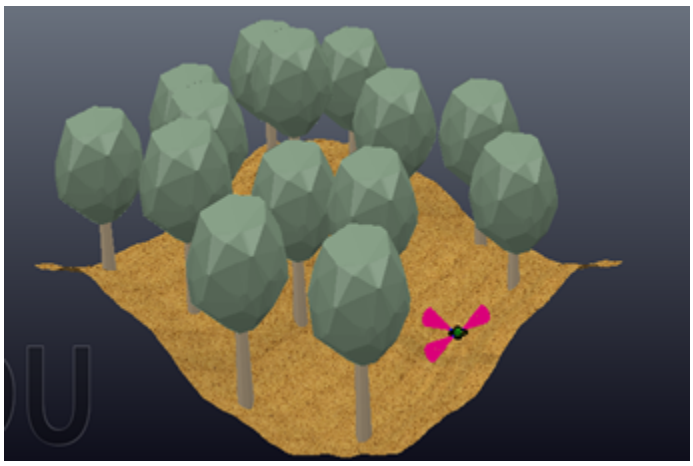

Fig 19: Simulación 7

- 8: El dron es capaz de pasar todo el laberinto llegando hasta al final.

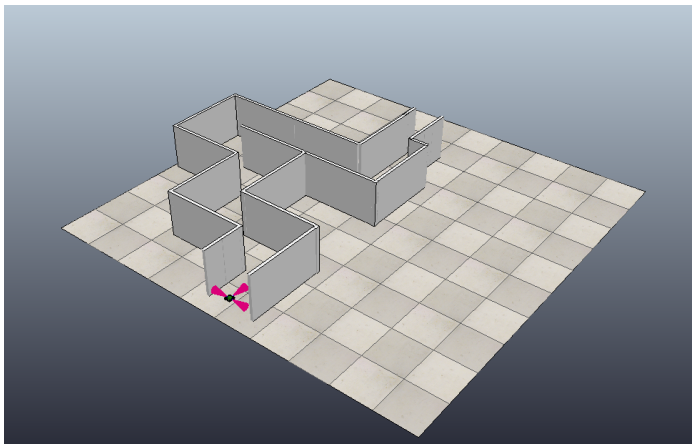

Fig 20: Simulación 8

PATH PLANNING:

Ideas Preliminares

Antes hacer el proyecto teníamos diferentes ideas para implementar path planning , estas ideas las concretamos anteriormente con los módulos:

- **COVERAGE PATH PLANNING***:* en este módulo se implementará el algoritmo de coverage path planning que permitirá encontrar la mejor ruta en cuanto a distancia y giros, que son los dos parámetros que más influyen en el consumo de batería del dron.
- **DESCOMPOSICIÓN EN CELDAS** para optimizar el número de giros que hace el dron
- **SPANNING TREES** que nos permitirán optimizar la distancia mediante rutas generadas con GRAFOS HAMILTONIANOS, que por definición ya son óptimas en cuanto a la distancia**.**

Sin embargo, la implementación que hemos realizado finalmente no han sido estos módulos sino que nos hemos basado en el algoritmo *A**.

Básicamente se compone de dos funcionalidades principales:

- La **SELECCIÓN DEL ÁREA DE TRABAJO** que nos permitirá seleccionar por que área se deberá mover nuestro dron. Antes de cualquier tipo de ejecución, al dron se le define el área rectangular en el que va a trabajar así tener fronteras y adquirir todos los datos necesarios del área para poder tomar decisiones. Por ejemplo, en el trabajo adjunto de SM, tomamos las diferentes altimetrías de un área en concreto, de esta manera sabremos qué obstáculos vamos a encontrar en el área cuando todavía no ha despegado el dron.
- La propia implementación del algoritmo en base a las heurísticas comentadas con anterioridad. Investigando algoritmos de path planning, hemos visto técnicas para definir obstáculos con plots, por ejemplo un algoritmo define a través de grafos una ruta de un punto A a un punto B en un espacio bidimensional.

  Conjuntamente con el proyecto de SM (Sistemas Multimedia) podemos obtener las altimetrías y tomar decisiones haciendo una posible representación tridimensional del terreno. Parte del código de este algoritmo se adjunta en este sprint como prueba1.m.

Algoritmo Implementado

Finalmente decidimos aplicar A* al ser una estrategia rápida computacionalmente y eficiente para encontrar una posible solución. En el código se le definen los obstáculos, que va a tener que evitar y el algoritmo busca una posible ruta óptima para poder ir al destino.

Ciertos problemas que hemos encontrado implementando el algoritmo en el dron por ejemplo cuando se presentan obstáculos en la simulación (adjunta en el GitHub Rlp_ppla_00.ttt), consta de un laberinto que tiene que completar, el algoritmo al hacer el camino óptimo a veces se puede acercar demasiado a la pared, para evitar

este tipo de problemas los obstáculos (las paredes) las definimos en el programa un poco más anchas para evitar colisiones. El código algoritmo en sí, lo hemos obtenido de una librería de algoritmos que se define en los adjuntos del proyecto.

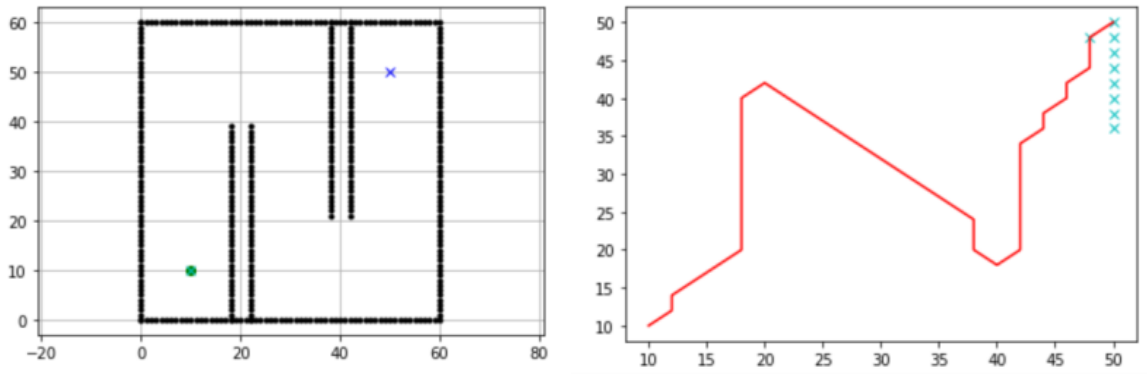

Fig 21: Laberinto especificado en el código i camino computado con A*

Una vez se ha programado el camino entre los obstáculos este se transfiere cada punto de coordenadas cada cierto tiempo a través de la API de Coppelia para que el dron pueda moverse y poder así definir también una velocidad de vuelo definido por un parámetro en el código.

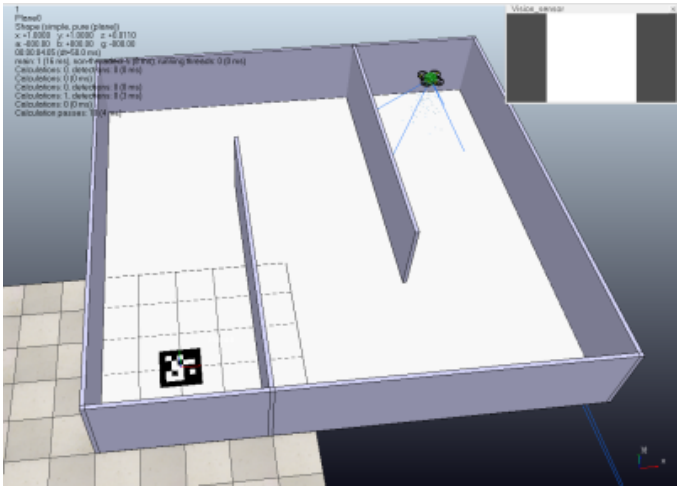

Fig 22: Dron simulado resolviendo un laberinto con A*

Finalmente para para que se vea visualmente en el simulador, hemos creado las paredes del laberinto con las mismas dimensiones que se definen en el código, de esta manera conseguimos que el dron consiga su objetivo.

Como se ha especificado anteriormente, conjuntamente con el proyecto de SM se pueden obtener las altimetrías de un área de trabajo. De esta manera, si por ejemplo queremos que el dron vuele a 1 metro del suelo en este área, podemos obtener todos los puntos de objetos a esta altura para tenerlos en cuenta como objetos con los que puede haber una colisión. Esta idea, es posible pero requeriría mucho más tiempo para implementar.

Foreseen risks and contingency plan

Risk #	Description	Probability (High/Medium/Low)	Impact (High/Medium/Low)	Contingency plan
1	desequilibrio del dron en el aire	High	High	PID para calibrar los motores
2	No llegar a la zona marcada como objetivo	Medium	Medium	Calibrar GPS/cámara
3	No esquivar obstáculos	High	High	NO TIENE SOLUCIÓN
4	NO detectar la h en el suelo	Medium	Medium	Mover de posición al dron para obtener otra vista de la marca de aterrizaje

Tabla 2. Riesgos y plan de contingencia

References

This project has been inspired by the following Internet projects:

https://github.com/miliaresis/GeoLogic_Shell
https://www.kaggle.com/kmader/dron-videos?select=DJI_0501.MP4
https://www.kaggle.com/c/hashcode-dron-delivery/overview
https://ieeexplore.ieee.org/document/9311993

Otros enlaces:

https://www.instructables.com/Ultimate-Intelligent-Fully-Automatic-dron-Robot-w/
https://create.arduino.cc/projecthub/user53873/autonomous-drone-uav-project-for-plane-5d8894
https://www.researchgate.net/publication/336642238_UAV_FLIGHT_CONTROL_BASED_ON_ARDUINO_BOARD_IMPLEMENTATIONS
https://www.youtube.com/watch?v=yFBvC_zRiek&ab_channel=KendinYap
https://www.youtube.com/watch?v=g_oLiW72wMs&ab_channel=JustinMarple

GitHub del Proyecto

Christian-Espinosa/Automated-Drone: This repository has all the code and simulations used to program a drone to work by it self. (github.com)