Taller de drones: Guía del profesor

1. Presentación

Esta guía está destinada al profesorado que quiera utilizar el material de este taller en su propia docencia. Aquí pueden encontrarse aclaraciones sobre puntos clave del taller (por ejemplo, sobre los diferentes retos) y materiales adicionales para abordar retos cada vez más ambiciosos, como, por ejemplo, aumentar las capacidades de la librería propuesta para controlar el dron, montar y configurar un dron real que pueda ser controlado mediante la estación de tierra implementada en Python, o incorporar a la aplicación funcionalidades para el reconocimiento de objetos mediante redes neuronales.

2. Observaciones sobre el taller

2.1 BETA updates

Se ha observado que algunas versiones de Mission Planner no ponen correctamente en funcionamiento la simulación de un enjambre de drones. El problema se resuelve instalando las BETA updates, tal y como muestra el video del paso 1 de la parte 2 del taller. Es importante tener esto presente porque, con frecuencia, los estudiantes no se fijan en esta cuestión, ansiosos por ver lo antes posible el enjambre en acción.

2.2 Enjambre de 1

En algunas ocasiones, dependiendo de las versiones del software y del portátil, se ha observado un mal funcionamiento del simulador. En concreto, el dron se detiene a medio camino de su destino. Este problema se muestra en el vídeo del paso 2 de la parte 2 de este taller. Muchas veces el problema desaparece si se hace volar el dron a mayor altura. Por otra parte, el problema no se pone de manifiesto si ponemos en marcha la simulación de un enjambre de 1 solo dron, tal y como se muestra en el vídeo del paso 3.

2.3 Altitude Angel

Las versiones más actuales de Mission Planner incluyen un plugin llamado Altitude Angel, que muestra alertas en el caso de que se intente volar en zonas restringidas. Estas alertas pueden ser incómodas si lo que estamos haciendo es simular, porque se presentan en forma de grandes áreas de color rojo que impiden ver el mapa de la zona implicada.

La forma de desactivar esta molesta opción no se muestra en los vídeos pero si en el texto de la guía del estudiantes. En concreto, se explica cuando se describe el paso 1. La explicación está acompañada por la figura 7, que debe resultar suficientemente clarificadora.

2.4 Mal funcionamiento de geofences

Trabajando directamente con Mission Planner no resulta fácil poner en evidencia el funcionamiento del mecanismo de geofence. Tal y como se muestra en el paso 4, si intentamos hacer volar el dron a un punto fuera del geofence de inclusión en dron no se va a mover, porque detecta que el punto está fuera de la zona de vuelo. Es un comportamiento razonable, pero impide observar el mecanismo de geofence en acción. En cambio, sí que podemos ver ese mecanismo en acción si intentamos atravesar un geofence de exclusión, tal y como se muestra en el vídeo de ese paso.

También se muestra en ese video que, a menos que la velocidad de vuelo del dron sea muy pequeña, la inercia que tiene el dron hará que entre claramente en la zona de exclusión, en lugar de bloquearse justo en el límite. Es importante, por tanto, hacer pruebas con velocidades bajas (1 o 2 metros por segundo).

Es posible establecer un margen para el geofence. Esto se consigue configurando adecuadamente el parámetro FENCE_MARGIN. Si ese parámetro tiene, por ejemplo, el valor 5, entonces el dron se bloqueará 5 metros antes de llegar al límite del geofence. No obstante, se ha observado que el simulador parece ignorar este parámetro porque el dron solo se bloquea al llegar al límite. El mecanismo sí que funciona perfectamente cuando operamos el dron desde la aplicación que se propone en la parte 3 del taller. Puede comprobarse ahí que, habiendo fijado desde Mission Planner el valor del parámetro FENCE_MARGIN, y haciendo navegar al dron con los botones de la interfaz, el dron se bloquea a la distancia de los límites del geofence establecida por el parámetro.

2.5 Recursos bloqueados

En algunas ocasiones, cuando se cierra de manera inapropiada la aplicación en Python (por ejemplo, la que se presenta en la parte 3 de este taller) algunos recursos quedan bloqueados, lo cual impide poner de nuevo en marcha correctamente la aplicación.

Esta incidencia es fácil de detectar. Tal y como muestra la figura 1, el sistema indica “Waiting for process to detach”. El problema se resuelve clicando en el símbolo x (tal y como indica la figura) hasta que el mensaje desaparece.

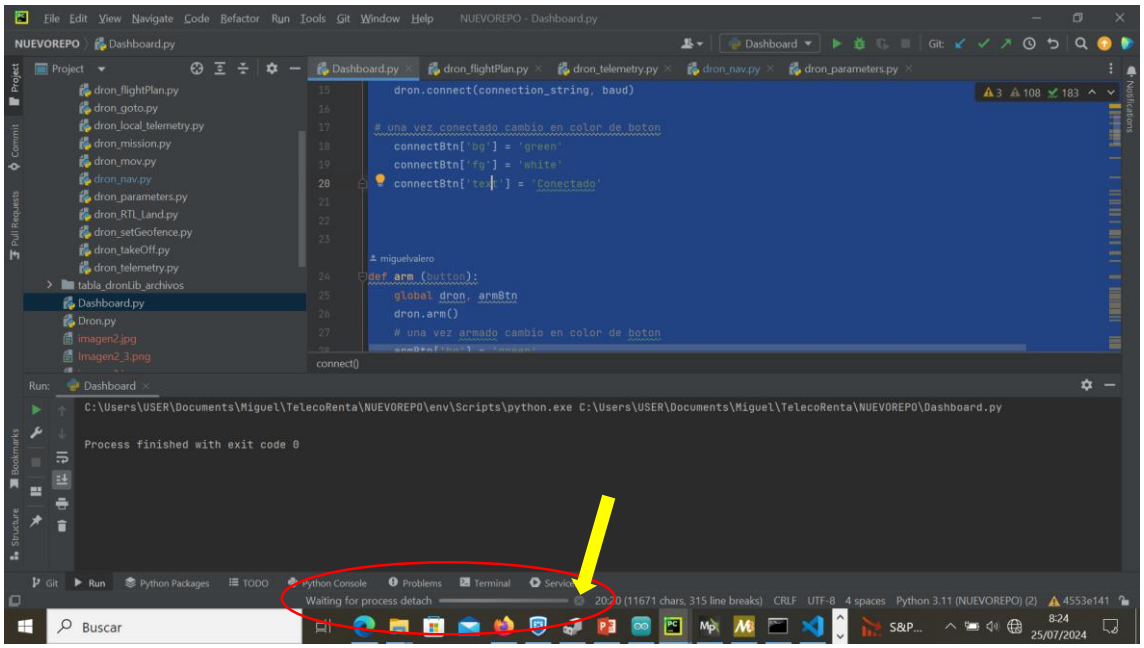


Figura 1: Aplicación que ha dejado algunos recursos del sistema bloqueados

2.6 Solución a los retos de la parte 2

La tabla siguiente contiene enlaces a los vídeos que muestran cómo se resuelven los retos propuestos en la parte 2 del taller.

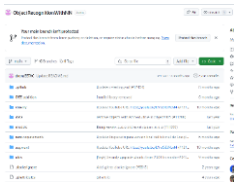
Reto 2	https://www.youtube.com/watch?v=N0gqlSieEVs
Retos del paso 3	https://www.youtube.com/watch?v=sYpFNfTxoAo
Reto 4.1	https://www.youtube.com/watch?v=NsHva36y5yU
Reto 4.2	https://www.youtube.com/watch?v=gGinRjdKYXk
Reto 5.1	https://www.youtube.com/watch?v=ZSTrHjwJ0eU
Reto 5.2	https://www.youtube.com/watch?v=7rvtuEtjs08
Reto 6	https://www.youtube.com/watch?v=AQx4XauOH9c
Reto 7	https://www.youtube.com/watch?v=Q5xnPN3R1kI

3. Reconocimiento de objetos mediante redes neuronales

Uno de los usos de los drones que más se están extendiendo es el reconocimiento de objetos desde el aire (por ejemplo, para contabilizar el número de personas en una manifestación o el número de ovejas en el rebaño). Las técnicas más utilizadas actualmente para este propósito son las redes neuronales convolucionales, cuyo desarrollo se ha visto favorecido por las mejoras de los algoritmos, el aumento de la capacidad de computación accesible para todos y el aumento de colecciones de imágenes necesarias para el entrenamiento de las redes. Aprender cómo funcionan las redes neuronales para reconocer objetos e incorporar esa funcionalidad a la aplicación desarrollada en este taller puede ser una aventura apasionante.

Este es un tutorial básico sobre cómo crear una red neuronal e integrarla en una aplicación en Python.

Este es el repositorio con el código que se usa en el video.



4. Modelos de comunicación

La aplicación desarrollada en la parte 3 de este taller utiliza un modo de comunicación que denominamos directo. El propio código de la interfaz de usuario es el que ejecuta las operaciones de control del dron, llamando a los métodos adecuados de la clase Dron, dependiendo, por ejemplo, del botón pulsado por el usuario. Naturalmente, si queremos controlar un dron real (no el simulador), el portátil en el que se ejecuta la aplicación tiene que estar conectado directamente al dron a través de la radio de telemetría.

Una organización alternativa es lo que llamamos comunicación global. En este caso, existe un servicio especial, que solemos denominar autopilotService, que está separado de la interfaz gráfica y que es el responsable de controlar el dron. En este caso, cuando el usuario pulsa un botón la interfaz gráfica hace una petición al autopilotService que es el que finalmente da la orden al dron.

El modelo de comunicación global es el adecuado cuando el dron tiene capacidad de computación a bordo (por ejemplo, porque le hemos instalado una Raspberry Pi). En ese caso, el autopilotService se ejecutaría en la Raspberry Pi y la aplicación con la interfaz gráfica en el portátil. Ambos dispositivos estarían conectados a internet de manera que la aplicación enviaría sus peticiones (y recibiría resultados) a través de internet.

También es el adecuado cuando queremos que sean varios usuarios los que puedan controlar el dron, cada uno desde su portátil, de manera que cualquier usuario podría enviar su petición al autopilotService, que podría estar ejecutándose en el portátil de cualquiera de ellos.

Entre los diferentes modelos de comunicación a través de internet que pueden usarse, uno especialmente adecuado para este contexto es la comunicación a través de un bróker que implementa el protocolo MQTT. De acuerdo con este modelo, los dispositivos/procesos que quieren comunicarse (en este caso la aplicación y el autopilotService) lo hacen mediante un mecanismo de subscripción/publicación. Todos los implicados se conectan al bróker y cuando uno de ellos hace una publicación sobre un tema determinado (por ejemplo, un comando para el autopilotService) todos los subscritos a ese tema reciben del bróker una notificación. El protocolo MQTT es uno de los más usados en el mundo del internet de las cosas (IoT) porque es muy intuitivo y ligero.

En estos vídeos puede encontrarse una explicación de todos estos conceptos, que se ilustran con un ejemplo.

Y aquí puede encontrarse una explicación sencilla de cómo usar un bróker MQTT en Python.

Se abre aquí una puerta al apasionante mundo de las comunicaciones a través de internet y a la posibilidad de plantear interesantes y divertidos proyectos. El más obvio de todos es construir un sencillo autopilotService y modificar la aplicación desarrollada en este taller para que trabaje en modo global, siguiendo el ejemplo que se muestra en el vídeo anterior. Y a partir de ahí es fácil imaginar retos divertidos como, por ejemplo, implementar un juego en el que diferentes usuarios cooperan para completar una misión, turnándose para llevar el dron de un sitio a otro. O incluso un juego de competición en el que cada usuario tiene que llevar el dron a un destino concreto y el sistema va asignando turnos de manera aleatoria para que cada usuario disponga de un periodo de tiempo (quizá también aleatorio) para acercar el dron a su objetivo antes de perder el turno.

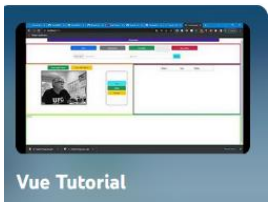
5. Web Apps

Para poder usar la aplicación que se ha desarrollado en este taller es necesario instalarla en el portátil que se va a conectar al dron. Imaginemos ahora que queremos controlar el dron desde nuestro teléfono móvil y que, además, no queremos tener que instalar ninguna aplicación en el teléfono (que lo tenemos a tope de apps). Ese interesante objetivo nos empuja al mundo de las web Apps.

Una web app no es más que un servidor web que se está ejecutando en algún servidor conectado a internet. Desde el móvil podemos conectarnos a esa web exactamente igual que nos conectamos a cualquier otra (como, por ejemplo, un periódico digital). Pero ahora lo que nos va

aparecer en la pantalla del móvil es un conjunto de botones para controlar el dron. Incluso lo que vemos en el móvil puede parecerse bastante que lo que muestra la interfaz gráfica de la aplicación desarrollada en este taller.

La web app solo puede trabajar en modo de comunicación global (ver el apartado anterior), de manera que al pulsar un botón la web app hace una publicación en el bróker para que éste notifique la petición al autopilotService, que ejecutará la operación correspondiente con el dron.



El mundo de las web apps no es fácil, pero es altamente motivador. Nos aleja de la programación en Python porque ahora hay que programar en JavaScript, TypeScript, HTML, CSS, etc. Aquí puede accederse a un tutorial que puede ayudar a introducirse en el tema, y que incluye aspectos tales como comunicaciones a través de un bróker MQTT.

El código que se desarrolla a lo largo de ese tutorial puede encontrarse aquí.

6. El dron

La aplicación que se desarrolla en este taller controla un dron simulado. Esto permite tener en un portátil todas las herramientas necesarias para desarrollar la actividad (incluido el simulador SITL). Como es natural, un paso más es hacer que la aplicación controle un dron real, para lo cual, en general, bastará modificar un par de líneas de código, precisamente las que especifican si la aplicación debe conectarse al simulador o al dron real.

El dron recomendado es el que se muestra en las imágenes. Partiendo del kit que contiene todos los componentes, el dron puede montarse, configurarse y prepararse para el vuelo en unas 6 horas (dependiendo en parte de la habilidad manual de los implicados). Ese proceso es altamente formativo porque se toca temas relacionados con la electrónica, la mecánica, la física, la aerodinámica, etc.

Aquí puede encontrarse una guía que indica cómo adquirir el kit (cuyo precio es de aproximadamente 1500 euros) y describe paso a paso el proceso de montaje, configuración y preparación para el vuelo (y naturalmente también explica qué hay que cambiar en la aplicación para que actúe sobre el dron real).

Es importante tener presente que el dron resultante no puede volarse en cualquier sitio. La normativa actual impone importantes restricciones al respecto. Por este motivo,

recomendamos que antes de emprender la aventura de adquirir el kit, montar y configurar el dron se haga una valoración de las posibilidades existentes para volar el dron. Para ello, el contenido del último apartado de la guía (Dónde volar) puede ser de ayuda.

7. Raspberry Pi

En uno de los videos mencionados en el apartado 4 se sugiere un escenario interesante en el que el autopilotService se ejecuta en una Raspberry Pi, que naturalmente tiene que estar conectada a internet para recibir las notificaciones del bróker cuando nuestra aplicación solicite algún servicio.

En una situación real, la Raspberry Pi estaría instalada en el dron y conectada al autopiloto sobre el que operaría directamente para realizar la operación solicitada desde la aplicación.

Pero puede ser interesante hacer que la Raspberry Pi no actúe sobre el autopiloto real sino sobre el simulador. Esto puede ser útil para verificar que el código funciona correctamente antes de instalar la Raspberry Pi en el dron y hacer pruebas de vuelo real. También puede ser interesante si se dispone de la Raspberry Pi pero no del dron.

Aquí puede encontrarse una guía para la configuración de una Raspberry Pi, aunque hay muchos tutoriales en internet que explican bien ese proceso. Lo importante es que, para implementar el esquema que proponemos aquí, es necesario que la Raspberry Pi esté conectada a la misma wifi que el portátil en el que estará ejecutándose el simulador SITL, de manera que cuando el autopiotService se conecte use como connection_string `'tcp:xxxxxxx:5763'`, siendo `xxxxxx` la ip que tiene asignada el portátil en el que se ejecuta el SITL (que puede averiguarse con el comando de Windows `ipconfig`).


Configuración básica de la Raspberry Pi

1. Introducción

Con frecuencia un dron lleva computación a bordo, que le permite tomar decisiones durante el vuelo como, por ejemplo, cambiar el plan de vuelo previsto si detecta algo a través de los sensores del dron). Naturalmente, una cámara es uno de esos sensores más habituales.

Una opción ideal para esto es la Raspberry Pi (RPi), a la que se le puede instalar una cámara y además puede conectarse a internet. En esta guía vamos a describir paso a paso cómo configurar la de manera que pueda instalarse en su momento en el dron o para que pueda interactuar con el simulador SITL.

El kit RPi que vamos a utilizar es el que se muestra en la figura 1. Consiste en la placa con la RPi, un módulo de cámara, un cable de alimentación, cable para conectar la RPi a un monitor, un adaptador WiFi y una tarjeta microSD.

Figura 1: The RPi kit

2. Instalación del sistema operativo Raspberry Pi OS (antes conocido como Raspbian)

El sistema operativo (esa instalarse en una tarjeta microSD que posteriormente se insertará en

8. Y mucho más sobre programación

Tomando como base lo aprendido en la tercera parte de este taller puede llegarse realmente muy lejos en el ámbito de la programación de Python de aplicaciones para controlar el dron.


Programación de drones

1. Presentación

En este taller vas a aprender a desarrollar programas en Python para **controlar la operación de** una **interfaz gráfica** que use botones para ordenar al dron que despegue o vuele en una direcci presentará al usuario un **mapa** en el que mostrará la posición del dron en todo momento y per utilizando técnicas de reconocimiento de imagen.

En este repositorio encontrarás:

1. Los **códigos** de referencia para la realización del taller.
2. Material **escrito**.
3. **Vídeos** que te guiarán durante el proceso.

Aprenderás instalando esos códigos, analizándolos, modificándolos y ampliándolos.

En este repositorio puede encontrarse material que explica paso a paso cómo desarrollar una estación de tierra que muestre su propio mapa con el dron en movimiento e incluso permita enviar al dron órdenes a partir de la detección de ciertas poses del cuerpo, a través de la cámara del portátil.