Capítulo 5

Integracion Dron Tello en Kibotics

En este capítulo se describe el desarrollo realizado para poder programar el dron Tello. A diferencia del Mbot, donde el proceso era similar para cualquier sistema operativo, en este caso el proceso va a variar.

5.1. Drone Tello

Tello es un mini dron distribuido por la empresa Ryze Technology (Figura 5.1). Debido a la facilidad de su uso, está destinado tanto a niños como a adultos que quieran aprender a utilizar drones. Puede alcanzar una velocidad de hasta 28 km/h, su radio de control abarca hasta los 100 metros y su señal de vídeo se transmite a una frecuencia de 2,4Ghz. Posee una unidad procesadora de Intel, un barómetro para controlar de altura, motores de tipo escobilla y una cámara de 720p. Además, sus hélices están protegidas y es resistente a las caídas.

Entre sus puntos fuertes destacan su gran estabilidad durante el vuelo y que utiliza un sistema de posicionamiento por visión (VPS). El sistema VPS utiliza un mínimo de dos cámaras para medir la distancia al suelo y la posición a la que se encuentra, compensando así los cambios en la posición que puedan darse (Castro, 2019).

Existen una variedad de software que permiten programar al Tello, entre los que destacan.

- Tello EDU App, es una aplicación diseñada para móviles o tables (Ios y Android), creada por la propia empresa que distribuye a este dron (RYZE Technology), que permite programarle utilizando bloques. Además incluye una opción para programar a un dron Tello simulado.
- El IDE (Integrated Development Environment) de Scratch ofrece la posibilidad de programar a este dron utilizando este lenguaje, aunque requiere para su uso, instalaciones previas en el ordenador del usuario.
- DroneBlocks además de permitir programar el Tello, ofrece la posibilidad de utilizar otros drones (Phantom 3, Phantom 4, Mavic Pro, Mavic Air, Spark). Utiliza también un lenguaje de bloques basado en Scracth y esta disponible en iOS App Store , Google Play Store y Chrome App Store.

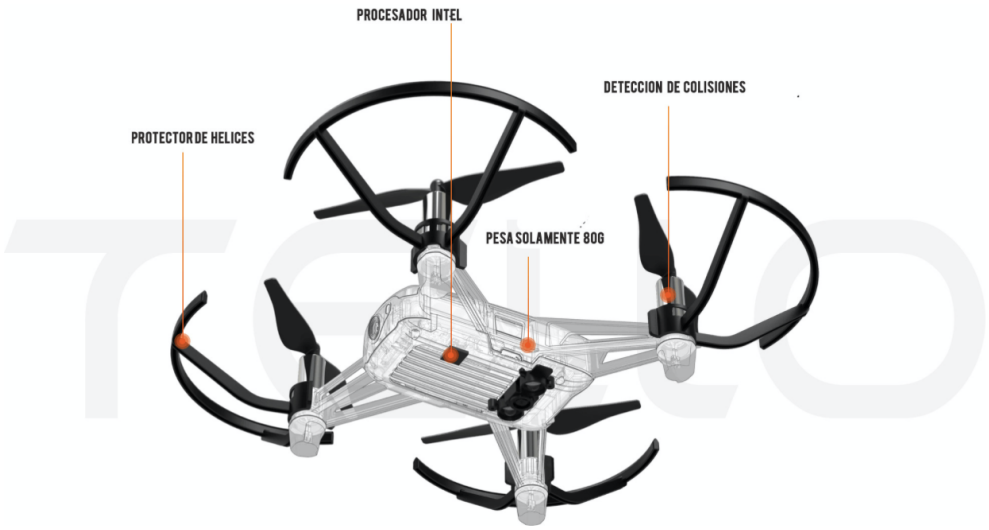

Figura 5.1: Esquema del dron Tello

Figura 5.2: IDE de Arduino (arriba a la izquiera), DroneBlocks (arriba a la derecha) y Tello EDU App (abajo)

5.2. Diseño

En la Figura 5.3, se muestra cuál es la arquitectura necesaria para conseguir el envío del programa al dron Tello.

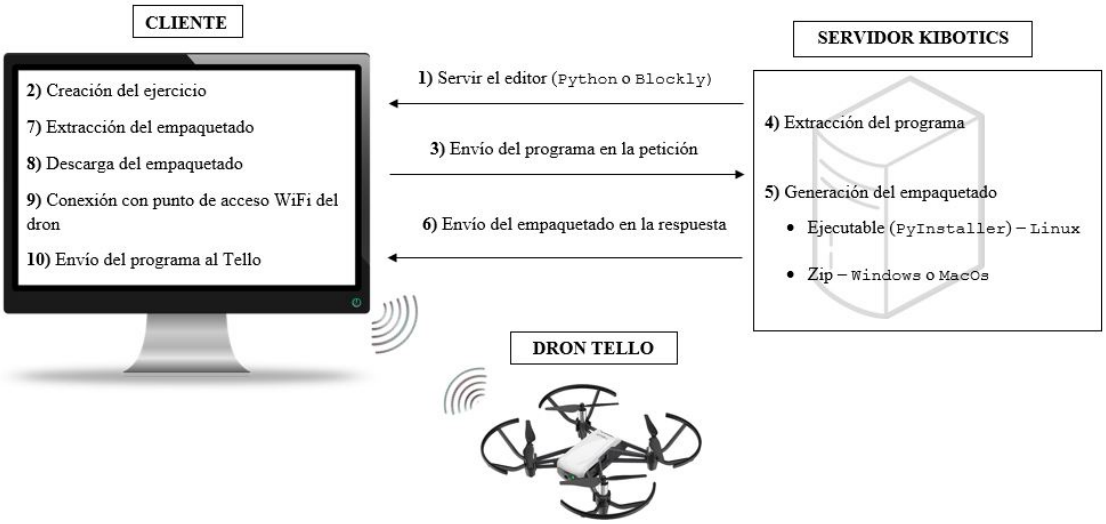

Figura 5.3: Arquitectura del desarrollo para el Tello

A continuación hablaremos en detalle sobre los pasos necesarios para conseguir la integración del Tello en Kibotics en los sistemas operativos Linux, Windows y MacOs.

5.3. Lado cliente

En la parrilla de Kibotics se encuentran dos unidades dedicadas al dron Tello , una dedicada a su programación utilizando Python y otra con Blockly. En la unidad se encuentra una sección que proporciona información sobre los requisitos para este robot y cómo utilizarlo.

Para el Mbot no se requería ninguna instalación y, además, el proceso de envío del programa era el mismo para los diferentes sistemas operativos. En el caso del Tello, existen peculiaridades para los diferentes sistemas operativos al no tener control sobre la controladora que posee el robot y al necesitar un proxy intermediario. Aunque en el robot no es necesario instalar nada, si que se necesitarán unos requisitos previos para los ordenadores que sean Windows y MacOs.

5.3.1. Preparación del anfitrión

Para MacOs y Windows, es indispensable que el usuario tenga instalado Python en su versión 2.7 y los drivers necesarios para el uso del Tello. La empresa Dji-sdk posee un repositorio en GitHub

(https://github.com/dji-sdk/Tello-Python) en el que se facilitan instaladores para su uso en los diferentes sistemas operativos. Por lo que el usuario tendría que instalar estos requisitos. Aunque para el caso de MacOs, cuando se carga el programa a el robot si no dispone de estos requisitos los instalará (sección 5.3.6).

En Linux, sin embargo, no es necesaria ninguna instalación previa para poder utilizar este dron. Esto es gracias a PyInstaller, una librería de Python, y a que el servidor de Kibotics se encuentra en una máquina Linux. En las próximas secciones se profundizará en este aspecto.

5.3.2. Editor

En la unidades se tiene también un editor (Python o Blockly), adaptado a la porgramación del Tello, donde el usuario podrá realizar su programa. El aspecto es similar al los utilizados para el Mbot.

Una vez que el usuario ha desarrollado el programa, debería pulsar el botín azul llamado "Ejecutar en Tello" (Figura 5.2) para iniciar el proceso de envío, que variaría dependiendo del sistema operativo utilizado.

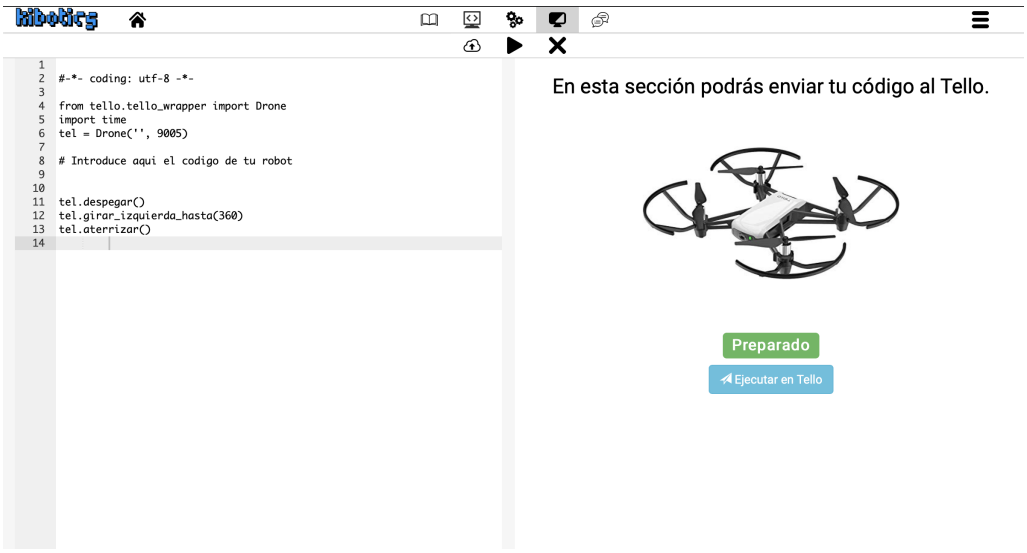

Figura 5.4: Editor para programar el dron Tello en Python

5.3.3. Envío del código fuente al servidor

Una vez iniciado el proceso de envío, el primer paso es enviar el código escrito por el usuario a el servidor de Kibotics, para generar el ejecutable que permita enviar el programa al Tello.

Se extrae el código escrito en el editor y se envía desde el navegador al servidor en un query parameter de una petición GET (Fragmento 5.1).

```
function send_code_tello() {
    var editor = ace.edit("ace");
    let code = editor.getValue();
    console.log(code);
    const message = {
       method: "GET"
     };
     var = url = '/get_code_to_tello?python_code=' + JSON.stringify(
         code);
     fetch(url, message)
           . . . . .
}
```

Fragmento 5.1: Envio del programa desde el navegador al servidor

5.3.4. Recepción del ejecutable

El navegador queda a la espera de recibir la respuesta a la petición que envió al servidor. Y tras obtener esta respuesta, extraerá un ejecutable en el caso de que se use una máquina Linux, o un fichero comprimido en el caso de usar Windows o MacOS, que preparó el servidor y que permitirá realizar la carga del programa. Posteriormente, el fichero extraído se descargará en el ordenador de usuario.

```
. . .

var = url = '/get_code_to_tello?python_code=' + JSON.stringify(code)
    ;
fetch(url, message)
    .then(response => {
       ld.style.display = "none";
       if (response.ok) {
          download_executable(response);
       } else {
          console.error("Bad Response");
       }
    })
   .catch(err => console.error(err));
}

function downloadExecutable(response) {
    response.blob().then(blob => {
```

```
            var down = document.createElement("down");
            document.body.appendChild(down);
            down.style = "display: none";
            url = window.URL.createObjectURL(blob);
            down.href = url;
            down.download = 'tello_code';
            down.click();
            window.URL.revokeObjectURL(url)
        })
    }
```

Fragmento 5.2: Extracción empaquetado y descarga

5.3.5. Conexíón con el drone

Para poder enviar el programa el Tello, necesitamos estar conectados con él. El dron, una vez encendido, emite un punto de acceso WiFi, que tendrá el nombre Tello seguido de un número (Figura 5.5), al que se tiene que conectar el usuario para posteriormente cargarle el programa. Necesitamos de esta pasarela para comunicarnos con él.

Figura 5.5: Conectarse al punto de acceso WiFi emitido por el dron

5.3.6. Ejecución

El fichero descargado en el ordenador permitirá enviar el programa al Tello, como este fichero es diferente para cada uno de los sistemas operativos, el proceso que hay que seguir para ejecutarlo no es el mismo. A continuación se describe cual es el proceso de ejecución para cada uno de los sistemas operativos:

- **Linux.**

  1. Se descargará un ejecutable.
  2. Dirigirse al directorio donde se descargo, darle permisos de ejecución y ejecutarlo (Fragmento 5.3).

```
chmod +x send_code.sh &&
./send_code.sh
```

Fragmento 5.3: Comandos para ejecución del envío en MacOs

- **Windows.**

  1. Se descargará un fichero comprimido.
  2. Descomprimir el fichero y dirigirse a directorio extraído.

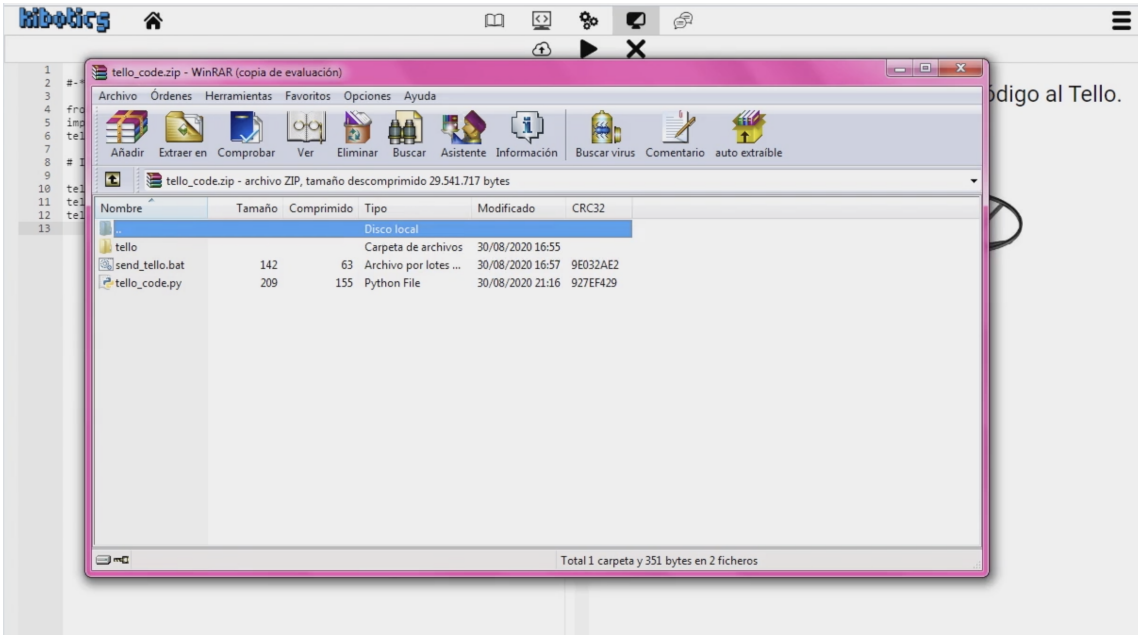

Figura 5.6: Descompresión en Windows

  3. Haciendo doble "click" en el ejecutable send_tello.bat, el programa será enviado al Tello.

- **MacOs.**

  1. Se descargará un fichero comprimido.
  2. Descomprimir el fichero y dirigirse a directorio extraído.

3. Dentro del directorio se encuentra un fichero ejecutable llamado "send_code.sh", al que se tendrá que dar permisos de ejecución y, posteriormente, ejecutarlo para enviar el programa al dron (Fragmento 5.4).

```
chmod +x send_code.sh &&
./send_code.sh
```

Fragmento 5.4: Comandos para ejecución del envío en MacOs

El ejecutable que se utiliza en MacOS para el envío (send_code.sh), con el objetivo de facilitar las instalaciones requeridas para el Tello, instalará todas las dependencias necesarias en el caso de que no se tengan en el ordenador.

```
#!/bin/bash

########## Requirements ##########

env=~/.virtualenvs/tello-env
if [ -d $env ];
then
   source ~/.virtualenvs/tello-env/bin/activate
else
   if ! type "pip" > /dev/null; then
      sudo easy_install pip
    fi
    if ! type "brew" > /dev/null; then
       /usr/bin/ruby -e "$(curl -fsSL https://raw.
           githubusercontent.com/Homebrew/install/master/install
           )"
       brew update
    fi
    sudo brew install cmake
    sudo brew install boost
    sudo brew install boost-python
    sudo brew install ffmpeg
    sudo brew install tcl-tk

    mkdir ~/.virtualenvs
    pip install virtualenv
    virtualenv ~/.virtualenvs/tello-env --python=python2.7
    source ~/.virtualenvs/tello-env/bin/activate
    pip install SimpleWebSocketServer
    pip install numpy
    pip install matplotlib
    pip install opencv-python==3.1.0.1
```

```
fi

echo "#####################################"
echo "### Sending program to tello ###"
echo "#####################################"
python tello_code.py
echo "Program finished"
deactivate
```

Fragmento 5.5: Ejecutable de envio para MacOs

5.4. Lado servidor

Como ocurría en el Mbot, el servidor de Kibotics, además de ser el responsable de servir las páginas, también es el encargado de adaptar el código que ha escrito el usuario para que el programa pueda ser enviado al dron.

En este caso, la adaptación consistirá en realizar un empaquetado que será devuelvo a el navegador. En los siguientes apartados se describen los pasos seguidos para conseguirlo.

5.4.1. Recepción del código fuente

Una vez que el usuario ha realizado el programa y se ha iniciado el proceso de envió, el navegador envía el código escrito por el usuario al servidor. Este extrae dicho código del query parameter de la petición recibida (Fragmento 5.5).

```
def get_code_to_tello(request):
    . . . .
    python_code = json.loads(request.GET.get('python_code', None))
    . . . .
```

Fragmento 5.6: Extracción programa en el servidor

5.4.2. Empaquetado

El dron Tello entiende el lenguaje Python, y como el código extraído ya se encuentra escrito en este lenguaje, no será necesaria realizar ninguna conversión. Por lo tanto, una vez extraído el código se procede a su empaquetado, que será diferente en función del sistema operativo utilizado por el usuario.

- ***Linux***. El empaquetado consistirá en realizar un ejecutable que contenga todas las dependencias necesarias para el envío. Esto es posible gracias a una librería de Python llamada PyInstaller, un módulo que permite empaquetar ficheros Python, generando así un ejecutable e incluyendo dentro del propio ejecutable el intérprete de Python. PyInstaller no permite

una compilación cruzada, es decir, si este empaquetado se realiza en una máquina Linux, el ejecutable solo podrá utilizarse en una máquina Linux. Debido a esto, la integración para el dron Tello varía según el sistema operativo; como el servidor de Kibotics está alojado en una máquina Linux, solo los usuarios de Linux se beneficiarán de esta utilidad.

```
def create_executable_with_pyinstaller(exercice_path):

    call(". kibotics-drivers/tello/tello_env/bin/activate;
        pyinstaller -F --distpath " +
    exercice_path + "dist --workpath " + exercice_path + "
        build --specpath " + exercice_path + " --clean " +
        exercice_path +
    "tello_code.py", shell=True)
  . . .
```

Fragmento 5.7: Creación ejecutable con PyInstaller

- ***Windows y MacOs***. El proceso es similar en ambos, pero diferente al caso anterior. Se empaqueta el programa con el directorio que engloba todas las librerías y las dependencias necesarias y que, además, contiene el ejecutable .bat (en Windows) o .sh (en MacOS) que se encargará del envío (Fragmento 5.7). La única diferencia entre ambos, es que al necesitar dependencias especificas por sistema operativo, se utilizan directorios diferentes en el empaquetado.

```
if operative_system == 'Mac':
    shutil.move(exercise_dir + "tello_code.py", os.getcwd() +
        "/kibotics-drivers/tello/MacOs")
    shutil.make_archive('output-tello-mac', 'zip', os.getcwd()
        + "/kibotics-drivers/tello/MacOs")

elif operative_system == 'Windows':
    shutil.move(exercise_dir + "tello_code.py", os.getcwd() +
        "/kibotics-drivers/tello/Windows")
    shutil.make_archive('output-tello-windows', 'zip', os.
        getcwd() + "/kibotics-drivers/tello/Windows")
```

Fragmento 5.8: Creación empaquetado y respuesta en Windows y MacOs

5.4.3. Envío del ejecutable

Una vez generado el empaquetado, ejecutable en Linux y fichero comprimido en MacOs y Windows, se devuelve al navegador en la respuesta a la petición que envió. Esto permitirá que sea descargado en el ordenador del usuario y pueda ejecutarlo, consiguiendo así el proceso de envió del programa al dron Tello.

```
create_executable_with_pyinstaller(exercice_path)
response = FileResponse(open(exercice_path + "dist/tello_code", 'rb
    '), content_type='application/octet-stream')
shutil.rmtree(exercise_dir + "dist/")
shutil.rmtree(exercise_dir + "build/")
os.remove(exercise_dir + "tello_code.spec")
```

Fragmento 5.9: Respuesta a la petición para Linux

```
if operative_system == 'Mac':
    ...
    with open('output-tello-mac.zip', 'rb') as f:
        response = HttpResponse(f, content_type=guess_type('output-
            tello-mac.zip')[0])
        response['Content-Length'] = len(response.content)
        os.remove('output-tello-mac.zip')

elif operative_system == 'Windows':
    ...
    shutil.move(exercise_dir + "tello_code.py", os.getcwd() + "/
        kibotics-drivers/tello/Windows")
    shutil.make_archive('output-tello-windows', 'zip', os.getcwd()
        + "/kibotics-drivers/tello/Windows")
    os.remove(os.getcwd() + "/kibotics-drivers/tello/Windows/
        tello_code.py")
    with open('output-tello-windows.zip', 'rb') as f:
        response = HttpResponse(f, content_type=guess_type('output-
            tello-windows.zip')[0])
        response['Content-Length'] = len(response.content)
        os.remove('output-tello-windows.zip')
```

Fragmento 5.10: Creación empaquetado y respuesta en Windows y MacOs

5.5. Validación experimental

La tecnología utilizada en este caso es distinta a la del Mbot debido a las diferencias en las controladoras del robot; con la placa mCore, el control sobre ella es mayor, mientras que en la que utiliza el Tello no tenemos ningún control.

Otro aspecto importante es que para el envío del programa al Tello se necesita una pasarela, es decir, para poder comunicarnos con él necesitamos conectarnos al punto de acceso WiFi que emite el dron, por lo que no es posible un envío directo desde el navegador, algo que sí se podía hacer en el Mbot con Web Serial.

Por otro lado, la tecnología PyInstaller permite que el usuario interaccione con el dron sin necesidad de instalar previamente nada, y también simplifica el envío al lanzar el ejecutable descargado. El problema que surge es que, con la restricción de compilación cruzada, este proceso solo puede ser factible para Linux, ya que el servidor de Kibotics, que es el encargado de preparar el ejecutable, está alojado en una máquina Linux.

Para Windows y MacOs la situación cambia completamente. El usuario sí necesita tener instalado en su ordenador el intérprete Python2.7 y los drivers para poder usar el Tello, aunque en MacOs el ejecutable de envío será el que lo instale si el usuario no lo tiene, facilitando, por tanto, su uso. Además, el envío del programa no es tan directo, ya que en primer lugar hay que descomprimir el fichero descargado, que tiene las dependencias necesarias para el envío y, después, se lanza el ejecutable.

Para su experimentación se ha probado en diferentes sistemas operativos (*Ubuntu 18.04*, *Ubuntu 16.04*, *Windows 10*, *Windows 7*, *MacOs Catalina Versión 10.15.6*, *MacOs Mojave Versión 10.14.6*) y navegadores (*Safari Versión 14.0*, *Google Chrome Versión 85.0*, *Firefox Versión 68.9*), verificando así que la integración permite ser multiplataforma.

Por último, en lo siguientes punteros a vídeos, se muestra un ejemplo de realización y envió de un programa al dron Tello, para los principales sistemas operativos, desde la plataforma de Kibotics.

- En Linux: *https://www.youtube.com/watch?v=b_msB5vBw4A&t=8s*
- En Windows: *https://youtu.be/vfRo9dGXbBw*
- En MacOs: *https://youtu.be/SAa1XO8Cp_o*

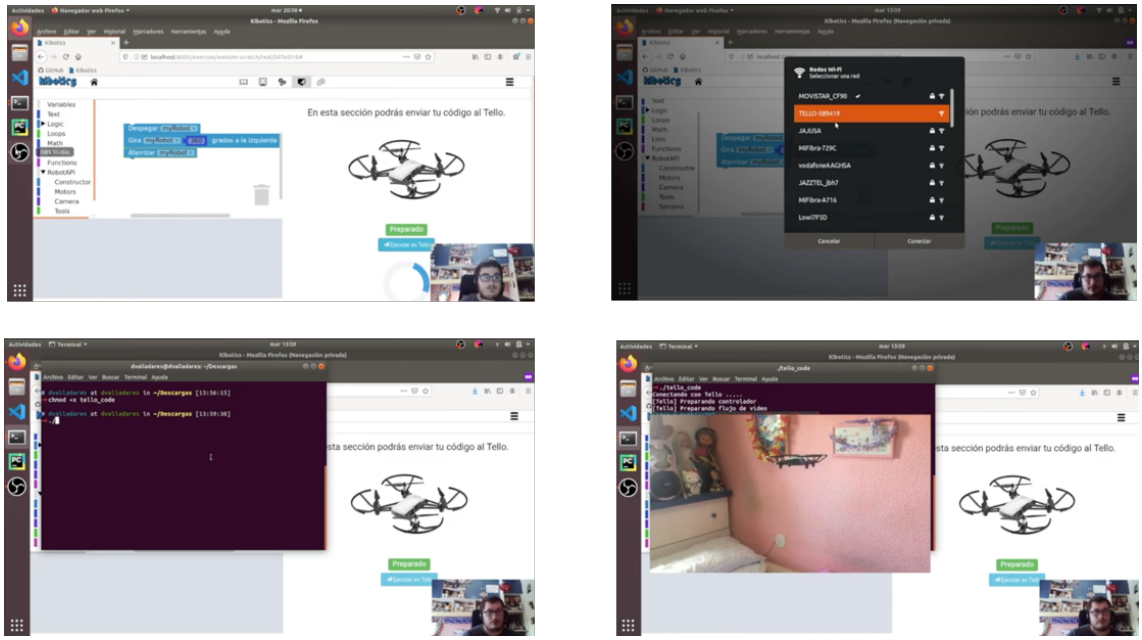

Figura 5.7: Fotogramas del vídeo de ejemplo de realización de un programa para el Tello en Linux