Operar Drone Tello DJI con Keyboard/mando de xbox one s

Román Gabriel Velásquez Reyes

October 6, 2024

Abstract

Este proyecto se basa en la implementación de un nodo de teleoperación que permite controlar un dron Tello utilizando el teclado de un computador o un mando de xbox one s.

Entorno en el que se probo:

- Ubuntu 20.04.6 LTS
- ROS Noetic Ninjemys
- Dron Tello DJI Ryze

1 Operar el Dron Tello DJI con Pycharm

Métodologia para programar´ el drone en el IDE de Pycharm
Este experimento se trabajo con python 3.8.10 en ubuntu 20.04

1. Instalación del entorno
   - Instalar pycharm

     ```
     sudo snap install pycharm-community {claasic
     ```

   - Creación de entorno
     Crear un nuevo proyecto en una dirección para mejor organización.
   - Instalar bibliotecas
     Seleccionar “File”, posterior en settings y agregar las siguientes bibliotecas.
     - Djitellopy 2.4.0
     - numpy 1.24.4
     - opencv-python 4.4.0.46
     - pillow 10.4.0
     - pygame 2.1.3

Todas las bibliotecas contienen en su descripción la referencia del repositorio de git-hub original. Nota: Algunas se agregan en automatico.

2. Conexión al drone

   - Crea dentro del nuevo entorno un archivo.py que nos permita conectarnos al drone e imprima el porcentaje de bateria.
   - Agrega el siguiente código:

     ```
     #Importa la clase Tello de la biblioteca
     from djitellopy import tello  djitellopy
     #Instancia del dron
     tello = tello.Tello()
     #Establece la conexión entre el código y el drone
     tello.connect()
     #utilizando el método get_battery(), imprime el % de bateria.
     print(tello.get\_battery())
     ```

   Nota: Al dar mantener la tecla "ctrl" + click derecho en alguna función del drone, aparecera una venta que incluye todas las funciones con su respectiva explicación, variables y parámetros que requiere para su implementación.

3. Establecer conexión al drone DJI tello Ryze
   Al encender el drone, iniciara una conexión ip del drone a la que debemos conectarnos, de esta forma tenemos acceso a sus puertos con el protocolo UDP, finalmente ejecuta el script de python de conexion y debera imprimir el porcentaje de bateria del drone.

4. Operar el Drone con teclado (movimientos basicos)

   - Crea un nuevo script de python e implementamos las funciones importantes para realizar movimientos:

Movimientos Drone	
Movimiento	Tecla
Despegue	T
Aterriza	L
Adelante	Flecha adelante
Atrás	Flecha atrás
Izquierda	Flecha izquierda
Derecha	Flecha derecha
Rotación sentido antihorario	A
Rotación sentido horario	D
Arriba	W
Abajo	S

Funciones básicas del drone	
connect	Ingresa al modo SDK. Llámalo antes de cualquiera de las funciones de control.
streamon	Activar la transmisión de vídeo.
get_battery	Obtener el porcentaje actual de batería
takeoff	Despegue automático.
land	Aterrizaje automático
move	Tello vuela arriba, abajo, izquierda, derecha, adelante o atrás con distancia x cm.

5. Acceso, captura, transmisión de video del drone Utilizando las funciones de video, se implementan 2 códigos distintos para las siguientes situaciones:

   - recording.py: Transmite video en tiempo real del drone y almacena el video para posteriormente procesarlo con ORB-SLAM, además permite operarlo con funciones de movimiento con el teclado.
   - control-streaming-photo.py: Tiene las mismas funciones de recording.py, la diferencia es que al oprimir la tecla p, toma una fotografia con vista a la cámara del drone y la almecena en la dirección del IDE. Esto para aplicaciones de calibración.

2 Operar el Dron Tello DJI con ROS

Utilizamos el paquete de ROS tello_driver http://wiki.ros.org/tello_driver

1. Crear un ROS Workspace:

   ```
   mkdir -p ~/catkin_ws/src
   cd ~/catkin_ws/
   catkin_make
   ```

2. Instalación

   ```
   En caso de no tener instalado:
   pip3: sudo apt install python3-pip
   av: pip3 install av

   cd <catkin_ws/src>
   git clone https://github.com/anqixu/TelloPy.git
   cd TelloPy
   sudo -H pip3 install -e .
   cd .
   git clone https://github.com/anqixu/h264_image_transport.git
   git clone https://github.com/anqixu/tello_driver.git
   cd ..
   rosdep install h264_image_transport
   NOTA: Accede a la carpeta de workspace e ingresa a la dirección
   del paquete de tello_drive para modificar en la carpeta src los
   archivos .py que se ejecutan en python2 por python3|
   catkin_make
   ```

3. Comprobar conexión con drone tello

   - Enciende y conecta al punto de acceso de la conexión wifi del drone
   - Ejecutamos los siguiente comandos para los nodos del drone:

     ```
     Terminal 1: roscore
     Terminal 2:roslaunch tello_driver tello_node.launch
     ```

3 Configuración y uso de un joystick (xbox one s) compatible con Linux con ROS.

Se utiliza el paquete Tello-Driver para controlar el drone con ROS. Este paquete incluye un nodo para comunicarse con el drone a través de Wi-Fi y otro nodo para controlar el drone con un mando de xbox one, con el paquete joy. Esta entrada se convierte en un mensaje ROS, enviado al controlador a través del topico joy.

1. Reconocimiento del mando

   - Instalar paquete joy.

     ```
     sudo apt-get install ros-noetic-joy
     ```

   - Conocer los dispositivos activos en la computadora,conecte el mando de xbox a la computadora para verificar si reconocio el dispositivo, con el siguiente comando, debera verlo impreso:

     ```
     ls /dev/input/
     o también
     ls -l /dev/input/jsX
     ```

     Los dispositivos joystick se denomian js"X" Ejemplo: js0 Para verificar lectura del datos de control, coloque el siguiente comando y vera un cuadro de asignación de botones del joystick, al presionar los botes, su valor inicial varia.

     ```
     jstest /dev/input/js0
     ```

2. Control del Drone con ROS

   - Crear un ROS Workspace

     ```
     mkdir -p ~/catkin_ws/src
     cd ~/catkin_ws/
     catkin_make
     ```

     También utilizamos el paquete image_transport para decodificar la transmisión de video H.264 enviada por Tello. image_decompressed incluido en este paquete decodifica el mensaje de entrada sensor_msgs/Image type y lo vuelve a publicar con un nombre de tema arbitrario.

   - Instalar bibliotecas:

     ```
     sudo apt install ros-noetic-cv-bridge
     sudo apt install ros-noetic-image-transport
     sudo apt install ros-noetic-camera-info-manager
     sudo apt install ros-noetic-codec-image-transport
     ```

   - Instalamos los repositorios requeridos en el workspace:

     ```
     cd ~/catkin_ws/src
     #Paquete Tello-Driver para controlar y comunicarse con el drone Tello.
     git clone --recursive https://github.com/appie-17/tello_driver.git
     #Paquete image_transport
     git clone https://github.com/ros-perception/camera_info_manager_py.git
     ```

   - Realizar modificaciones al código.
     Modificar los nodos en la dirección:

     ```
     cd ~/catkin_ws/src/tello_driver/nodes
     Cambiar:
     #!/usr/bin/env python2 por: #!/usr/bin/env python3.
     ```

- Agregar la siguiente función aproximadamente en la línea 90 a gamepad_teleop_node en la dirección /catkin_ws/src/tello_driver/nodes/

```
    def parse_my_joy(self, msg):
if len(msg.buttons) != 11 or len(msg.axes) != 8:
raise ValueError('Invalid number of buttons (%d) or axes (%d)' % (
len(msg.buttons), len(msg.axes)))
self.L1 = msg.buttons[1] #Boton B Despegar
self.R1 = msg.buttons[0] #Boton A Aterriza
self.LX = msg.axes[3] #Joy-stick Derecho X Rotación izquierda/derecha
self.LY = msg.axes[4] #Joy-stick Derecho Y Arriba/abajo
self.RX = msg.axes[0] #Joy-stick izquierdo X derecha/izquierda
self.RY = msg.axes[1] #Joy-stick izquierdo Y adelante/atras

    def parse(self, msg):
err = None
try:
return self.parse_ps3_usb(msg)
except ValueError as e:
err = e
try:
return self.parse_ps3_usb2(msg)
except ValueError as e:
err = e
try:
return self.parse_my_joy(msg)
except ValueError as e:
err = e
raise err
```

  La función parse_my_joy está diseñada para interpretar los mensajes del joystick y actualizar el estado del gamepad.

Tabla de operaciones para el mando xbox en ROS		
msg.buttons		
Índice	Botón	Operación
0	A	Aterrizar
1	B	Despegar
msg.axes (ejes)		
Índice	Joy-stick	Operación
0	izquierdo X	derecha/izquierda
1	izquierdo Y	adelante/atras
3	Derecho X	Rotación izquierda/derecha
4	Derecho Y	Arriba/abajo

- Modificar archivo "launch"

```
<?xml version="1.0"?>
<launch>
<arg name="tello_ip" default="192.168.10.1" />
<arg name="tello_cmd_server_port" default="8889" />
<arg name="local_cmd_client_port" default="8890" />
<arg name="local_vid_server_port" default="6038" />
<arg name="camera_calibration" default="$(find tello_driver)/cfg/960x720.yaml" />
<arg name="namespace" default="tello" />

<group ns="$(arg namespace)">

<node pkg="tello_driver" name="tello_driver_node"
type="tello_driver_node" output="screen">
<param name="local_cmd_client_port" value="$(arg local_cmd_client_port)" />
<param name="local_vid_server_port" value="$(arg local_vid_server_port)" />
<param name="tello_ip" value="$(arg tello_ip)" />
<param name="tello_cmd_server_port" value="$(arg tello_cmd_server_port)" />
<param name="connect_timeout_sec" value="10.0" />
<param name="stream_h264_video" value="true" />
<param name="camera_calibration" value="$(arg camera_calibration)" />
</node>

<node pkg="image_transport" name="image_decompressed"
type="republish" args="compressed in:=image_raw raw out:=image_raw">
<param name="image_transport" value="h264"/>
</node>

</group>

</launch>
```

- Actualizar workspace

```
cd ~catkin_ws
catkin make
source devel/setup.bash
```

3. Iniciar Nodo Ejecute tello_driver_node y gamepad_teleop_node e intente controlar el dron con el control. Además, ejecute rviz para comprobar si puede recibir la imagen de la cámara del Tello.

```
Terminal 1:
roscore
Terminal 2:
rosrun rviz rviz
Terminal 3:
cd ~/catkin_ws
```

```
source devel/setup.bash
roslaunch tello_driver tello_node.launch
Terminal 4:
cd ~/catkin_ws
source devel/setup.bash
roslaunch tello_driver joy_teleop.launch
```