

Documentación para el Control de Drones con Python y MUSE Headbands

Índice

1. Introducción
 2. Requisitos Previos
 3. Instalación de Python
 4. Instalación de un Entorno de Desarrollo (IDE)
 5. Instalación de Bibliotecas Necesarias
 6. Configuración del Dron
 7. Código de Control del Dron
 8. Conclusión
-

1. Introducción

Este proyecto tiene como objetivo mover drones utilizando señales de ondas cerebrales leídas por bandas para la cabeza MUSE. La comunicación con el dron se realiza a través de un script en Python que envía comandos al dron basados en la entrada del usuario.

2. Requisitos Previos

Antes de comenzar, asegúrate de tener lo siguiente:

- Un ordenador con acceso a internet.
- Un dron Tello de DJI.
- Una banda para la cabeza MUSE.
- Conocimientos básicos de Python.

3. Instalación de Python

Windows

1. Descarga el instalador de Python desde la página oficial: [Python.org](https://python.org)
2. Ejecuta el instalador y sigue las instrucciones en pantalla. Asegúrate de marcar la opción "Add Python to PATH".

macOS

1. Abre la Terminal.

2. Usa Homebrew para instalar Python:

bash

```
brew install python
```

Linux

1. Abre la Terminal.
2. Usa el administrador de paquetes de tu distribución para instalar Python. Por ejemplo, en Ubuntu:

```
sudo apt-get update
```

```
sudo apt-get install python3
```

4. Instalación de un Entorno de Desarrollo (IDE)

Un entorno de desarrollo (IDE) facilita la escritura y depuración de código.

Visual Studio Code (recomendado)

1. Descarga e instala Visual Studio Code desde: code.visualstudio.com
2. Instala la extensión de Python desde el Marketplace de Visual Studio Code.

PyCharm

1. Descarga e instala PyCharm desde: jetbrains.com/pycharm
2. Sigue las instrucciones en pantalla para la configuración inicial.

5. Instalación de Bibliotecas Necesarias

Abre la terminal o el símbolo del sistema y ejecuta los siguientes comandos para instalar las bibliotecas necesarias.

easytello

easytello es una biblioteca que simplifica la comunicación con el dron Tello.

Comando de instalación: `pip install easytello`

6. Configuración del Dron

1. Enciende tu dron Tello.
2. Conéctate a la red Wi-Fi del dron desde tu ordenador.

7. Código de Control del Dron

A continuación, se presenta el código para controlar el dron basado en los comandos del usuario:

```
from easytello import tello
import time

# Crear instancia del dron
drone = tello.Tello()

def move_drone(command):
    """
    Función para mover el dron según el comando recibido.

    Args:
        command (str): El comando recibido.
    """
    if command == "a":
        drone.left(50)
    elif command == "d":
        drone.right(50)
    elif command == "w":
        drone.forward(50)
    elif command == "s":
        drone.back(50)
    elif command == "p":
        drone.down(50)
    elif command == "u":
        drone.up(50)
    elif command == "stop":
        drone.stop()

def main():
    """
    Función principal.
    """
    drone.takeoff()
    print("Bienvenido al control del dron. Usa 'w', 'a', 's', 'd' para moverte.")
    print("Para detener el movimiento, presiona 'stop'.")
    print("Para aterrizar, presiona 'q'.")
```

```
while True:
    command = input("Ingresa un comando: ").lower()
    if command == 'q':
        break
    move_drone(command)
    time.sleep(0) # Pausa de 0 segundos después de cada comando
de movimiento

drone.land()

if __name__ == "__main__":
    main()
```

Descripción del Código

- **Importación de Bibliotecas:** Se importa la biblioteca `easytello` para interactuar con el dron y `time` para manejar pausas.
- **Instancia del Dron:** Se crea una instancia del dron utilizando `tello.Tello()`.
- **Función `move_drone`:** Controla el movimiento del dron según el comando recibido.
- **Función `main`:** Gestiona la lógica principal del programa, incluyendo el despegue, la recepción de comandos y el aterrizaje.

Ejecutar el Código

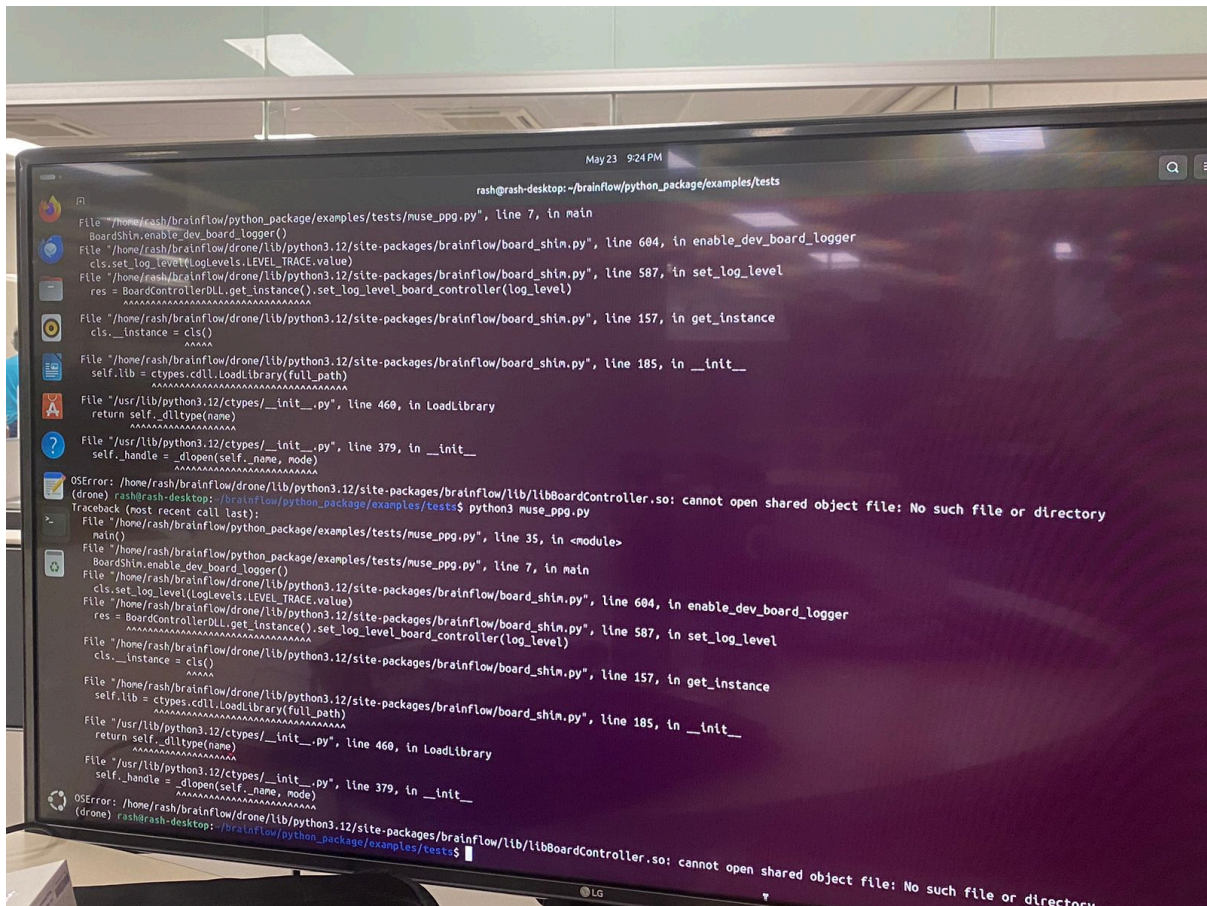
1. Guarda el código en un archivo llamado `control_dron.py`.
2. Abre la terminal o el símbolo del sistema.
3. Navega hasta el directorio donde guardaste `control_dron.py`.

Ejecuta el script con el siguiente comando:

```
python control_dron.py
```

Notas adicionales:

Siguiendo el tutorial de la conexión de la MUSE con la librería de procesamiento de señales, encontramos este error, también se proporcionar links a errores similares y documentación:



```
May 23 9:24 PM
rash@rash-desktop: ~/brainflow/python_package/examples/tests

File ~/home/rash/brainflow/python_package/examples/tests/muse_ppg.py, line 7, in main
  BoardShim.enable_dev_board_logger()
File ~/home/rash/brainflow/drone/lib/python3.12/site-packages/brainflow/board_shim.py, line 604, in enable_dev_board_logger
  cls.set_log_level(LogLevel.TRACE.value)
File ~/home/rash/brainflow/drone/lib/python3.12/site-packages/brainflow/board_shim.py, line 587, in set_log_level
  res = BoardControllerDLL.get_instance().set_log_level_board_controller(log_level)
  ~~~~~
File ~/home/rash/brainflow/drone/lib/python3.12/site-packages/brainflow/board_shim.py, line 157, in get_instance
  cls._instance = cls()
  ~~~~~
File ~/home/rash/brainflow/drone/lib/python3.12/site-packages/brainflow/board_shim.py, line 185, in __init__
  self.lib = ctypes.cdll.LoadLibrary(full_path)
  ~~~~~
File ~/usr/lib/python3.12/ctypes/_init_.py, line 460, in LoadLibrary
  return self._dlopen(name)
  ~~~~~
File ~/usr/lib/python3.12/ctypes/_init_.py, line 379, in __init__
  self._handle = _dlopen(self._name, mode)
  ~~~~~
OSError: /home/rash/brainflow/drone/lib/python3.12/site-packages/brainflow/lib/libBoardController.so: cannot open shared object file: No such file or directory
(drone) rash@rash-desktop: ~/brainflow/python_package/examples/tests$ python3 muse_ppg.py
Traceback (most recent call last):
  File ~/home/rash/brainflow/python_package/examples/tests/muse_ppg.py, line 35, in <module>
    main()
  File ~/home/rash/brainflow/python_package/examples/tests/muse_ppg.py, line 7, in main
    BoardShim.enable_dev_board_logger()
  File ~/home/rash/brainflow/drone/lib/python3.12/site-packages/brainflow/board_shim.py, line 604, in enable_dev_board_logger
    cls.set_log_level(LogLevel.TRACE.value)
  File ~/home/rash/brainflow/drone/lib/python3.12/site-packages/brainflow/board_shim.py, line 587, in set_log_level
    res = BoardControllerDLL.get_instance().set_log_level_board_controller(log_level)
  File ~/home/rash/brainflow/drone/lib/python3.12/site-packages/brainflow/board_shim.py, line 157, in get_instance
    cls._instance = cls()
  File ~/home/rash/brainflow/drone/lib/python3.12/site-packages/brainflow/board_shim.py, line 185, in __init__
    self.lib = ctypes.cdll.LoadLibrary(full_path)
  File ~/usr/lib/python3.12/ctypes/_init_.py, line 460, in LoadLibrary
    return self._dlopen(name)
  File ~/usr/lib/python3.12/ctypes/_init_.py, line 379, in __init__
    self._handle = _dlopen(self._name, mode)
  OSError: /home/rash/brainflow/drone/lib/python3.12/site-packages/brainflow/lib/libBoardController.so: cannot open shared object file: No such file or directory
(drone) rash@rash-desktop: ~/brainflow/python_package/examples/tests$
```

<https://forums.raspberrypi.com/viewtopic.php?t=354207&sid=23c6673e34bb0fa7165bcab7bec27c79>

<https://www.baeldung.com/linux/solve-shared-object-error>