# ENGINEER FOR A DAY

Engineering is about
more than just
technical knowledge
it's about innovation,
problem-solving
and transforming ideas
into reality.

Prof Michele Crispoltoni

# DRONE TELLO EDU

Prof Michele Crispoltoni

Drone Programming with Tello — Tello Programming 0.0.1 documentation

## What are drones?

A drone is an unmanned aerial vehicle (UAV), meaning that is a type of aircraft that is capable of moving through the air without a pilot on board.

They can be controlled remotely, or by specific programming using software and GPS to operate autonomously.

The most common are RPAs (Remotely Piloted Aircraft) which are operated from a remote ground station via a communications link, which sends signals to the drone to perform the manoeuvres the pilot wants it to perform.

What is a drone and what is it used for? All the details!

(185) Real-life applications of drones - YouTube

## Applications of drones:

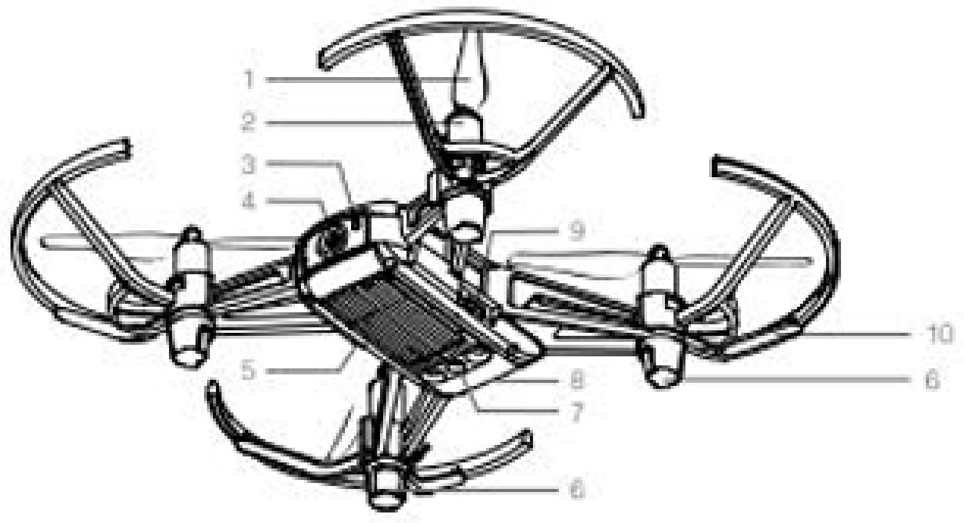
- **Agriculture:** Precision farming, crop monitoring, and pest control.
- **Delivery services:** Fast and efficient package delivery by companies like Amazon and UPS.
- **Photography and videography:** Capturing aerial views for films, events, and journalism.
- **Search and rescue:** Locating missing persons or assessing disaster areas.
- **Environmental monitoring:** Wildlife tracking, pollution control, and climate studies.
- **Construction and surveying:** Site inspections, mapping, and progress tracking.
- **Education:** STEM learning, programming, and robotics training using drones like **Tello EDU**.

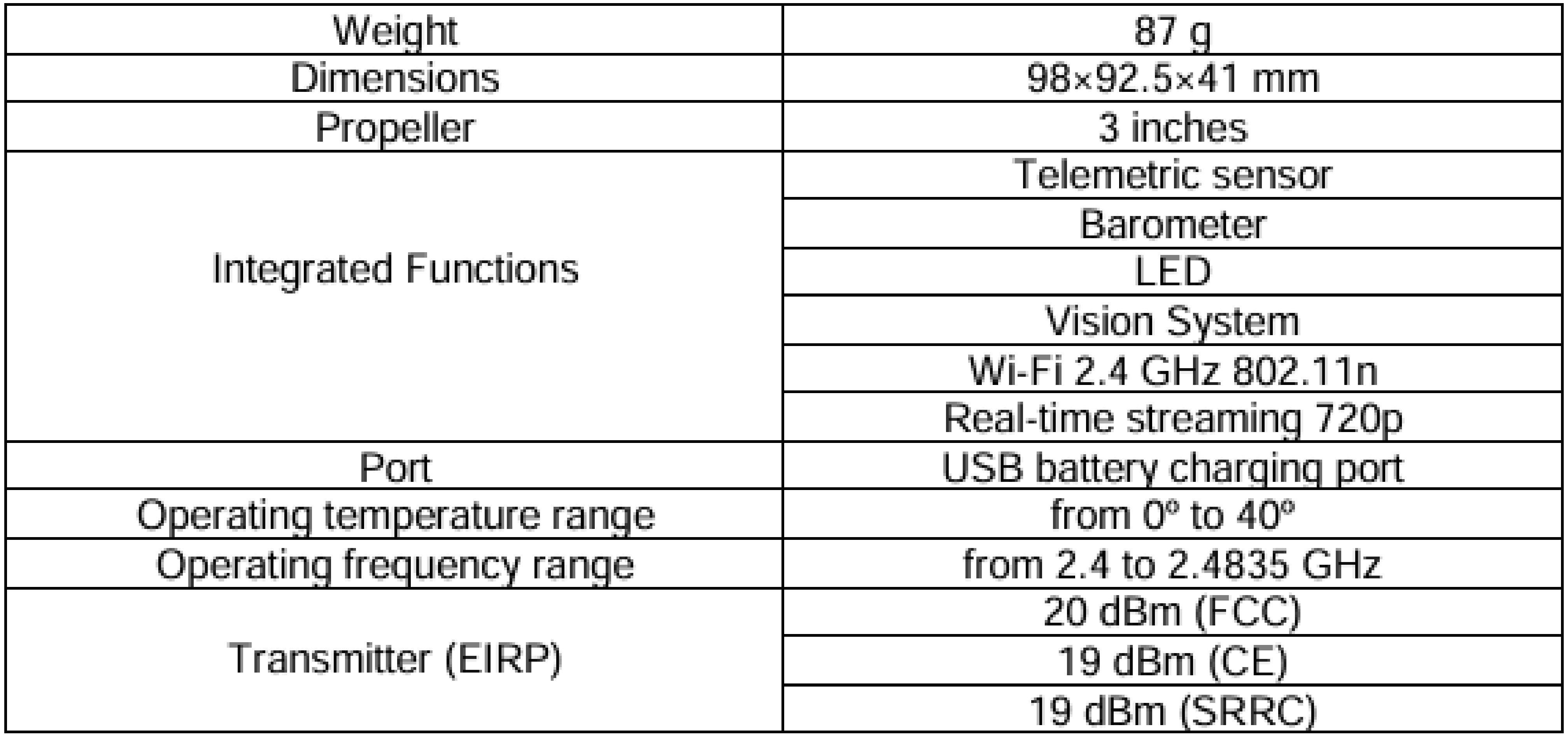
## What is the Tello EDU?

- ➢ A programmable educational drone designed by DJI and Ryze Tech.
- ➢ Lightweight, compact, and safe for indoor and outdoor use.

## Key features:

- ➢ **Programmability:** Compatible with Python, Swift, and Scratch.
- ➢ **Camera capabilities:** Real-time video streaming and photo capture.
- ➢ **STEM learning:** Great for learning coding, robotics, and aerodynamics.
- ➢ **Swarm programming:** Coordinate multiple drones for collaborative tasks.

| | |
|---|---|
|  | 1.Propellers |
| | 2.Engines |
| | 3.Drone Status Indicator |
| | 4.Camera |
| | 5.Power Button |
| | 6.Antennas |
| | 7.Vision Positioning System |
| | 8.Battery |
| | 9.Micro USB port |
| | 10.Propeller protectors |



| | |
|---|---|
| Weight | 87 g |
| Dimensions | 98×92.5×41 mm |
| Propeller | 3 inches |
| Integrated Functions | Telemetric sensor |
| | Barometer |
| | LED |
| | Vision System |
| | Wi-Fi 2.4 GHz 802.11n |
| | Real-time streaming 720p |
| Port | USB battery charging port |
| Operating temperature range | from 0° to 40° |
| Operating frequency range | from 2.4 to 2.4835 GHz |
| Transmitter (EIRP) | 20 dBm (FCC) |
| | 19 dBm (CE) |
| | 19 dBm (SRRC) |

| | |
|---|---|
| Maximum distance of flight | 100 m |
| Maximum speed | 8 m/s |
| Maximum flight time | 13 min |
| Maximum flight height | 30 m |

| | |
|---|---|
| Removable | Yes |
| Capacity | 1100 mAh |
| Voltage | 3.8 V |
| Type | LiPo |
| Energy | 4.18 Wh |
| Net Weight | 25 ± 2 g |
| Temperature range when charging | from 5° to 45° |
| Maximum Load Power | 10 W |

| | |
|---|---|
| Photo | 5 MP (2592x1936) |
| Field of view | 82.6° |
| Video | HD 720p 30 fps |
| Format | JPG (Photo) |
| | MP4 (Video) |
| Electronic stabilization | Yes |

## Why Tello EDU for Education?

**Ideal for:**

- Beginner programmers
- Schools and STEM programs
- Learning AI and computer vision basics

We want to teach drones to see, understand, and interact

The focus is on working with drones, like the **Tello EDU**, and creating intelligent systems using tools like **Python**, **YOLOv8**, and **NVIDIA GPUs.**

## Why Tello EDU for Education?

**TASKS:**

1. REMOTE CONTROL FROM KEYBOARD
2. MANAGE STREAMING VIDEO
3. SIMPLE GUI WITH TKINTER
4. COMPUTER VISION ALGORITHM (CANNY EDGE DETECTOR – HOUGH CIRCLES)
5. YOLO NEURAL NETWORKS WITH GPU TO DETECT PEOPLE

We’re going to explore an exciting topic:
**how to program a drone using Python.**

We’ll learn to control the Tello drone, use its live video stream, and even apply artificial intelligence to detect people in a room.

Along the way, I’ll show you how Python works for different tasks, how we can manage complex projects, and how to use some powerful tools like OpenCV.

Let’s get started!

**Before diving into the code, we need to create something called a virtual environment. But what is it, and why do we need it?"**

*A virtual environment is like a private workspace for Python. It keeps all the tools and libraries you need for your project in one place. Think of it as creating a small, isolated lab for your code.*

**how to create it in Windows:**

```
python -m venv myEnv

cd myEnv/Scripts

activate
```

**Install the required library**:

*For controlling the Tello drone, we'll use a library called djitellopy.*

*Let's install it inside our environment:*

```
pip install djitellopy
pip install numpy==1.23.5
pip install opencv-python
```

## Create a Virtual Environment (AFTER INSTALL PYTHON 3.9 - ADD TO PATH)

To manage dependencies and isolate the project.

- Open Command Prompt:
- Navigate to the folder where you want to set up your project.
- Create the Virtual Environment:
- Run the following command to create a virtual environment named myenv:

```
python -m venv myenv
```

- Activate the Virtual Environment:
- Activate myenv to isolate packages:

```
.\myenv\Scripts\activate
```

- You'll see (myenv) at the beginning of the prompt, indicating it's active.

## Install Required Python Libraries

- Install NumPy: pip install numpy==1.23.5
- Install DJITelloPy for drone control: pip install djitellopy
- Install OpenCV for video processing: pip install opencv-python
- Install PyTorch with CUDA Support: pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu116
- Install Ultralytics YOLOv8 for object detection: pip install ultralytics

## Object-Oriented Programming: Using Classes

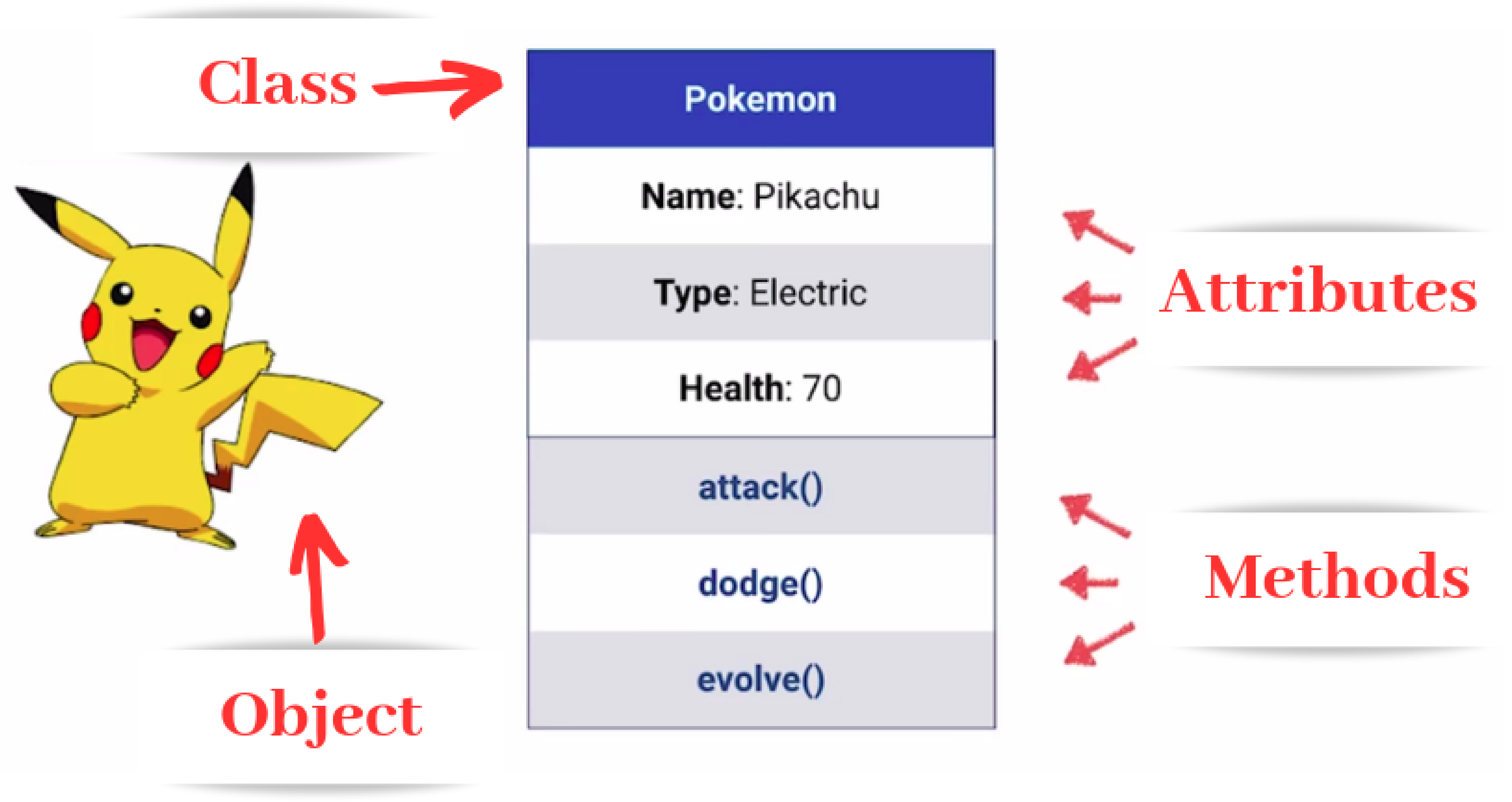
Next, let’s talk about object-oriented programming, or OOP.
It’s a way to organize code into reusable parts called classes.
Instead of creating our own class, we’ll focus on how to use existing ones, like the Tello class from djitellopy.

With just these two commands, we’re using methods defined in the Tello class to control the drone.

By using pre-built classes, we don’t need to reinvent the wheel.

Instead, we can focus on what we want to achieve, like making the drone fly or process video.

## Why Tello EDU for Education?



## Connecting to the Tello Drone

Now that we have everything set up, let’s connect to the Tello drone and send it some simple commands.

```
from djitellopy
import Tello
drone = Tello()
drone.connect()
print(f Battery: { drone.get_battery() } %)
```

Here, we’re using the djitellopy library to connect to the drone and check its battery level.

Who wants to see the drone take off? Let’s write another command to do that!

```
drone.takeoff()
drone.land()
```

## First steps in Python:

**1) Import the library:**

```
from djitellopy import Tello
```

**2) Connect to the drone:**

```
tello = Tello()

tello.connect()

print(fBattery level: { tello.get_battery() })
```

**3) Take off and land:**

```
tello.takeoff()

tello.land()
```

**4) Basic movements:**

```
tello.move_up(50) → Move up 50 cm

tello.move_down(30)

tello.move_forward(100)

tello.move_back(100)
```

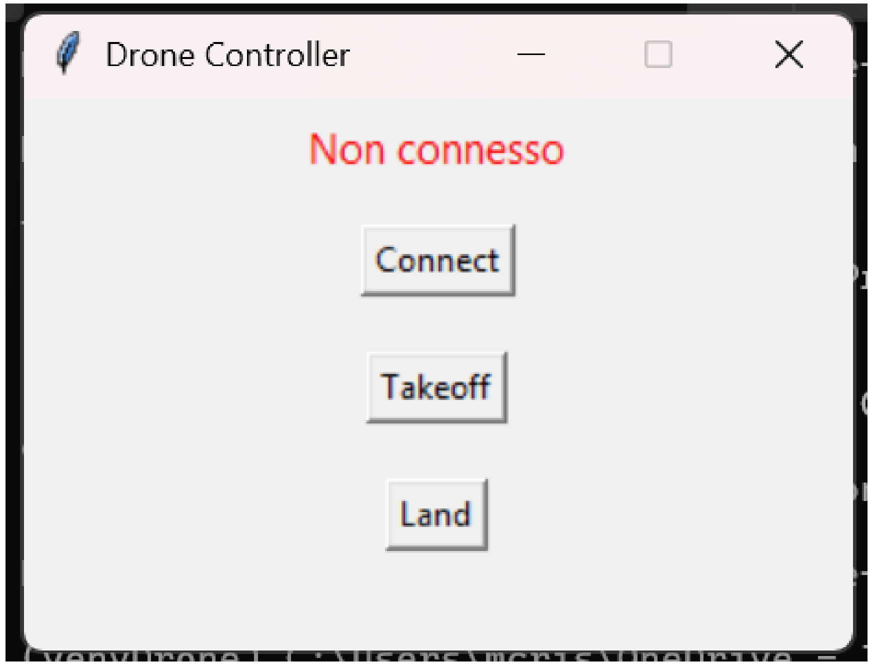
**5) Rotations**:

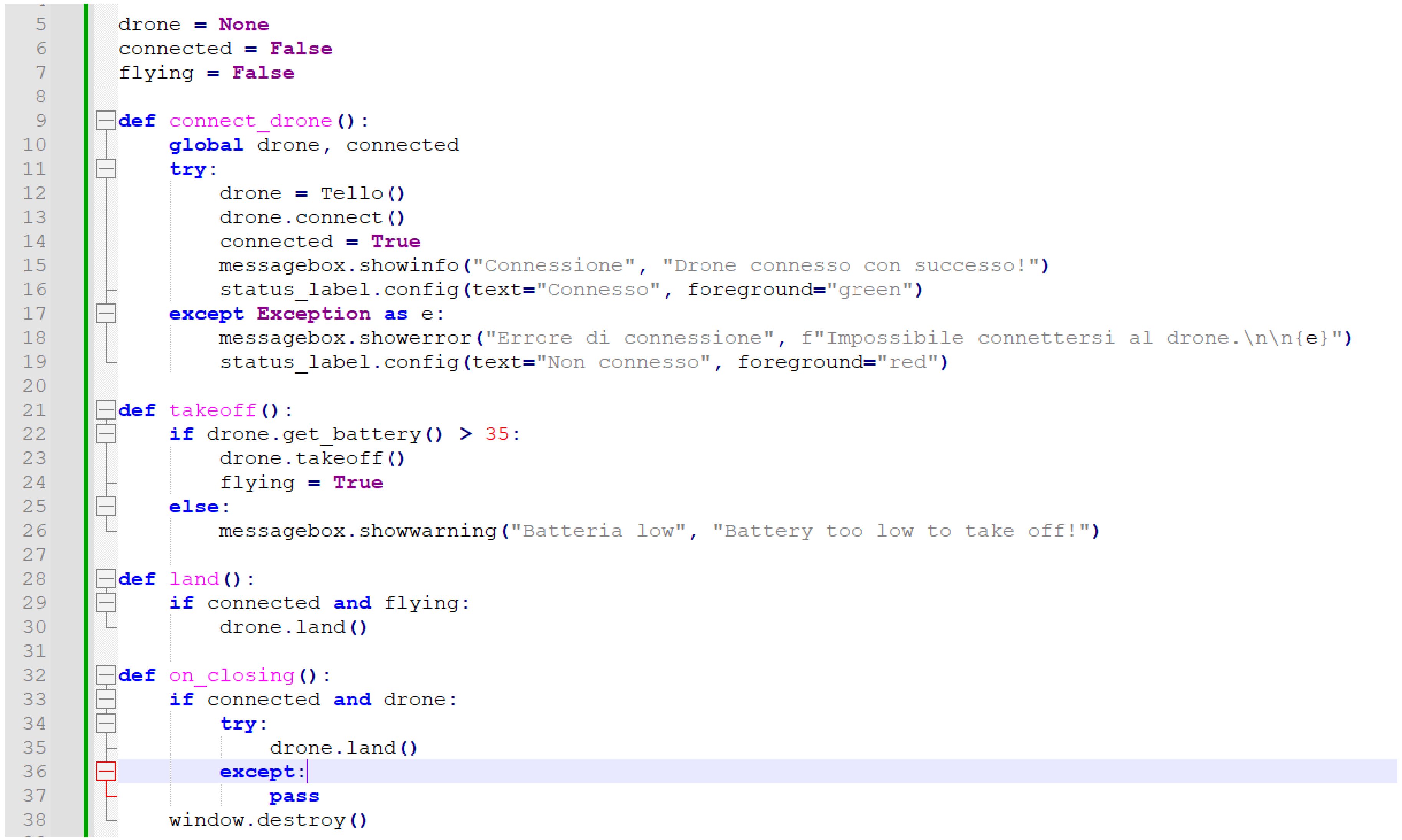
```
tello.rotate_clockwise(90) → 90° clockwise

tello.rotate_counter_clockwise(90)
```

## Building a GUI with Tkinter

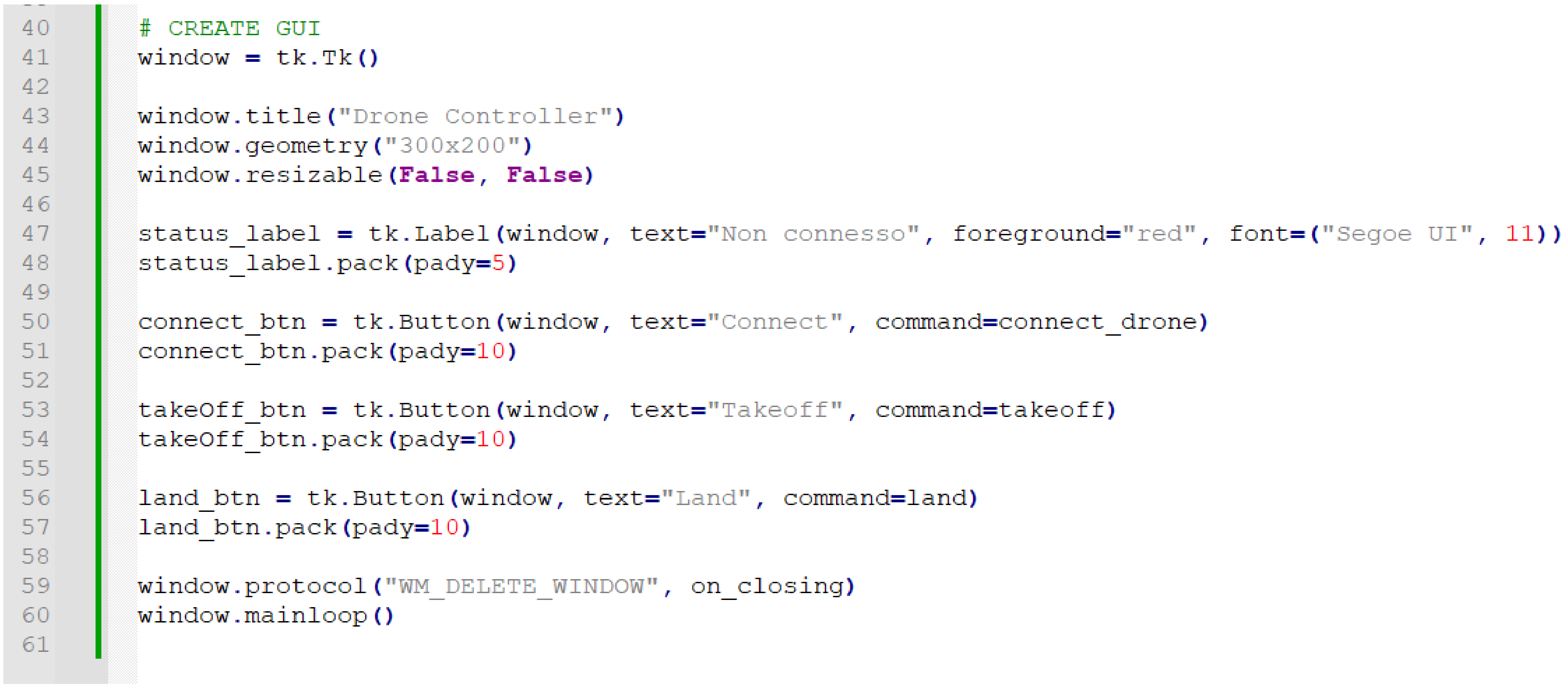
Wouldn’t it be cool if we could control the drone with buttons?
Let’s create a graphical user interface, or GUI, using Tkinter in Python.

```python
import tkinter as tk
from tkinter import ttk, messagebox
from djitellopy import Tello

```





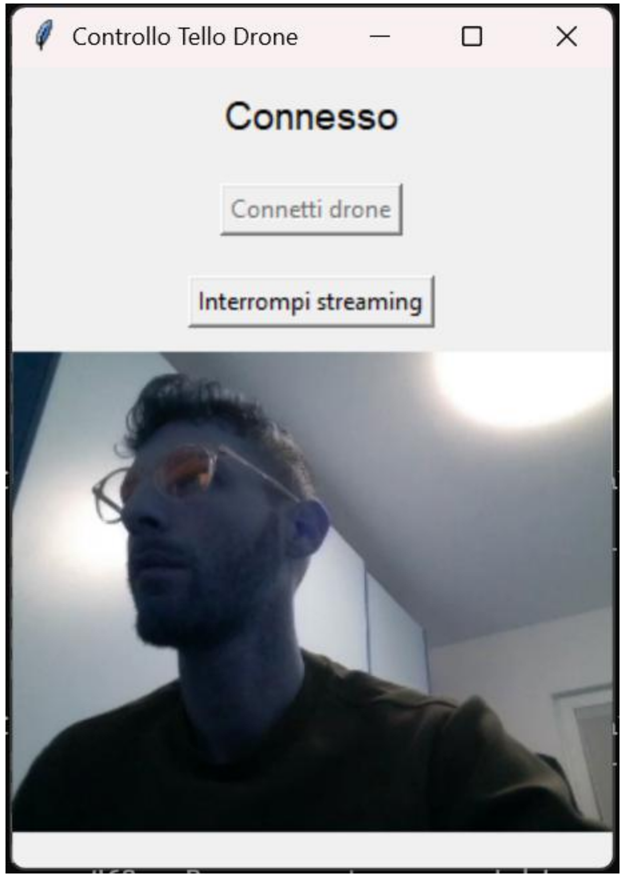
```python
drone = None
connected = False
flying = False

def connect_drone():
    global drone, connected
    try:
        drone = Tello()
        drone.connect()
        connected = True
        messagebox.showinfo("Connessione", "Drone connesso con successo!")
        status_label.config(text="Connesso", foreground="green")
    except Exception as e:
        messagebox.showerror("Errore di connessione", f"Impossibile connettersi al drone.\n\n{e}")
        status_label.config(text="Non connesso", foreground="red")

def takeoff():
    if drone.get_battery() > 35:
        drone.takeoff()
        flying = True
    else:
        messagebox.showwarning("Batteria low", "Battery too low to take off!")

def land():
    if connected and flying:
        drone.land()

def on_closing():
    if connected and drone:
        try:
            drone.land()
        except:
            pass
    window.destroy()
```

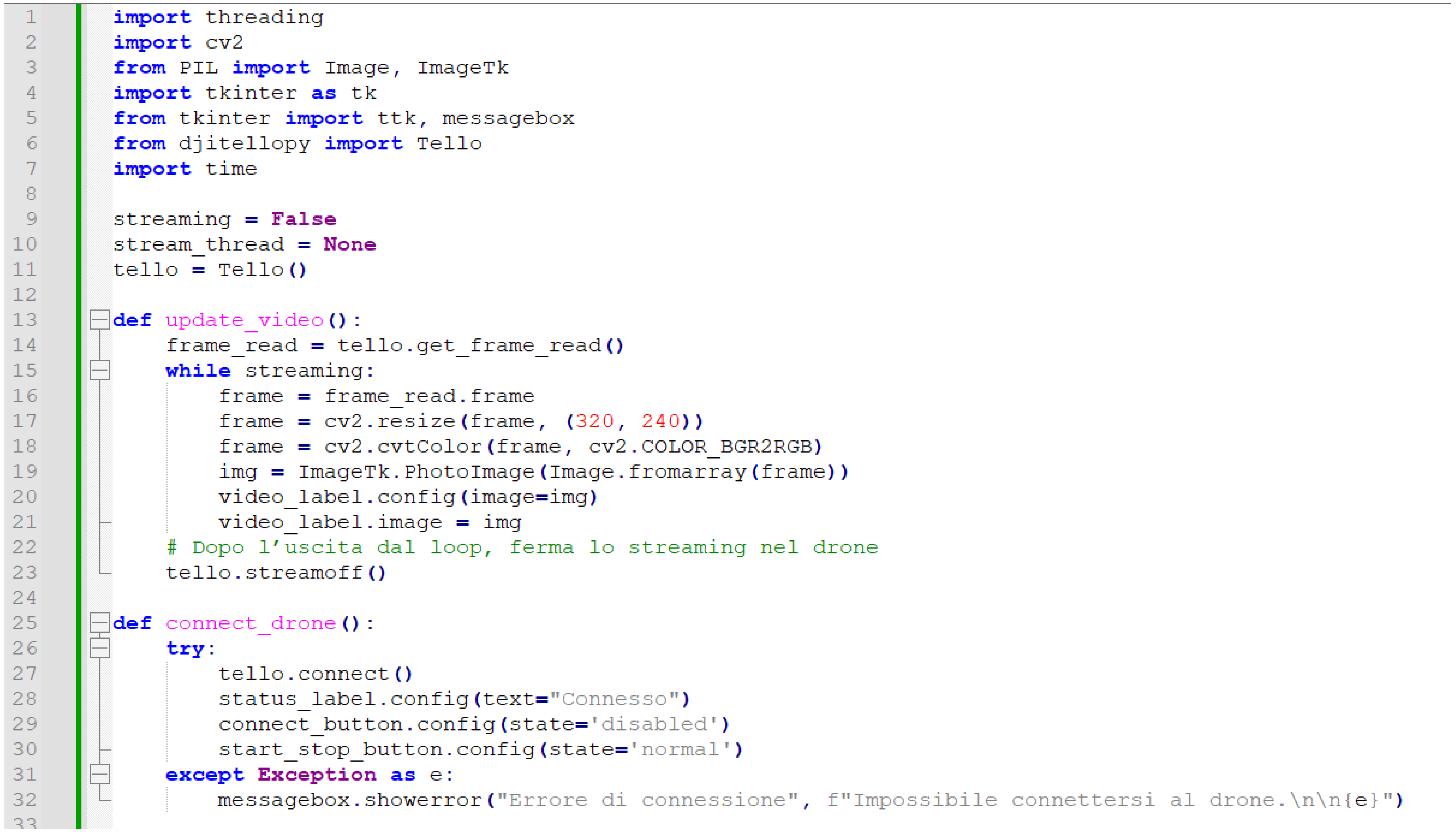


```python
# CREATE GUI
window = tk.Tk()

window.title("Drone Controller")
window.geometry("300x200")
window.resizable(False, False)

status_label = tk.Label(window, text="Non connesso", foreground="red", font=("Segoe UI", 11))
status_label.pack(pady=5)

connect_btn = tk.Button(window, text="Connect", command=connect_drone)
connect_btn.pack(pady=10)

takeOff_btn = tk.Button(window, text="Takeoff", command=takeoff)
takeOff_btn.pack(pady=10)

land_btn = tk.Button(window, text="Land", command=land)
land_btn.pack(pady=10)

window.protocol("WM_DELETE_WINDOW", on_closing)
window.mainloop()

```

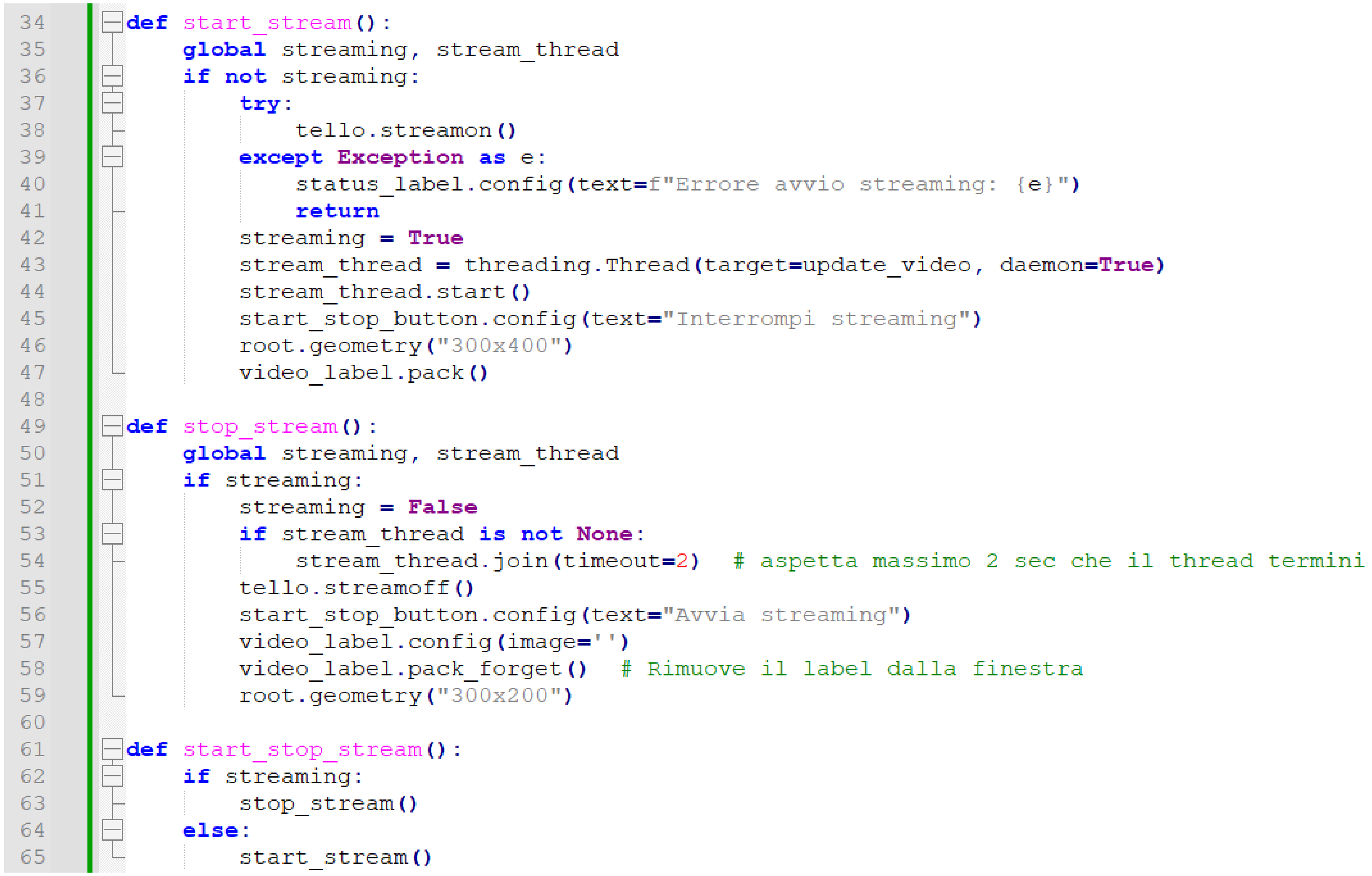
## OpenCV: Processing Video Streams

Now let's move on to another important tool called OpenCV. OpenCV stands for Open Computer Vision, and it's a library for working with images and videos. We'll use it to capture and process the drone's video stream.

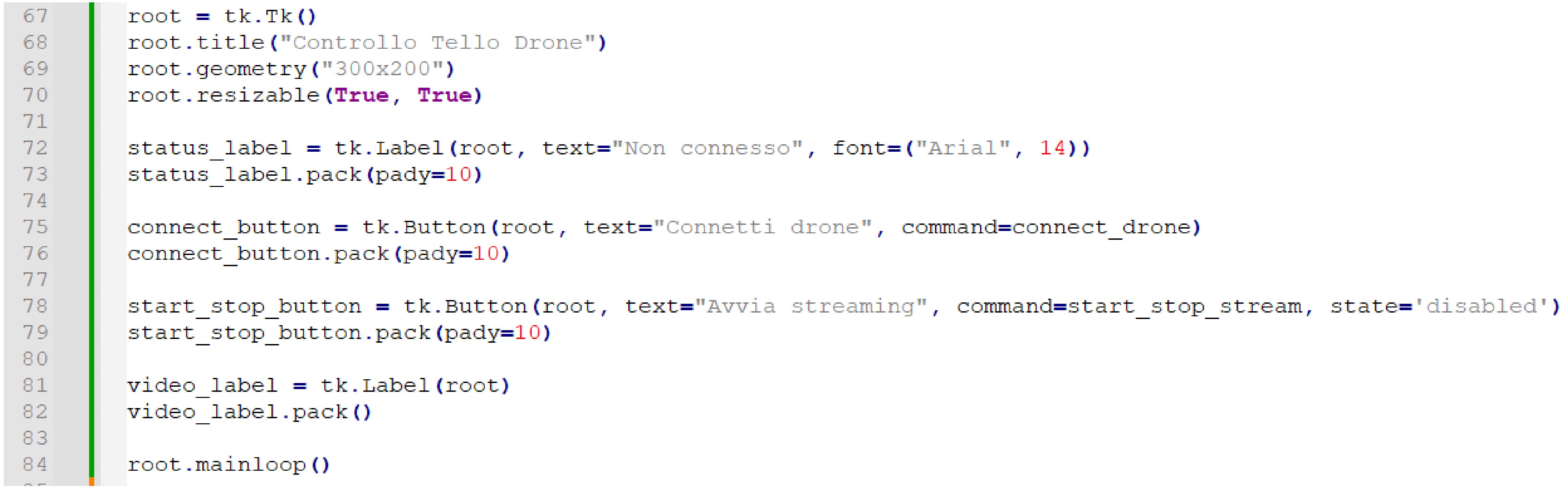




```python
import threading
import cv2
from PIL import Image, ImageTk
import tkinter as tk
from tkinter import ttk, messagebox
from djitellopy import Tello
import time

streaming = False
stream_thread = None
tello = Tello()

def update_video():
    frame_read = tello.get_frame_read()
    while streaming:
        frame = frame_read.frame
        frame = cv2.resize(frame, (320, 240))
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        img = ImageTk.PhotoImage(Image.fromarray(frame))
        video_label.config(image=img)
        video_label.image = img
    # Dopo l'uscita dal loop, ferma lo streaming nel drone
    tello.streamoff()

def connect_drone():
    try:
        tello.connect()
        status_label.config(text="Connesso")
        connect_button.config(state='disabled')
        start_stop_button.config(state='normal')
    except Exception as e:
        messagebox.showerror("Errore di connessione", f"Impossibile connettersi al drone.\n\n{e}")
```



```python
def start_stream():
    global streaming, stream_thread
    if not streaming:
        try:
            tello.streamon()
        except Exception as e:
            status_label.config(text=f"Errore avvio streaming: {e}")
            return
        streaming = True
        stream_thread = threading.Thread(target=update_video, daemon=True)
        stream_thread.start()
        start_stop_button.config(text="Interrompi streaming")
        root.geometry("300x400")
        video_label.pack()

def stop_stream():
    global streaming, stream_thread
    if streaming:
        streaming = False
        if stream_thread is not None:
            stream_thread.join(timeout=2)  # aspetta massimo 2 sec che il thread termini
        tello.streamoff()
        start_stop_button.config(text="Avvia streaming")
        video_label.config(image='')
        video_label.pack_forget()  # Rimuove il label dalla finestra
        root.geometry("300x200")

def start_stop_stream():
    if streaming:
        stop_stream()
    else:
        start_stream()
```



```python
root = tk.Tk()
root.title("Controllo Tello Drone")
root.geometry("300x200")
root.resizable(True, True)

status_label = tk.Label(root, text="Non connesso", font=("Arial", 14))
status_label.pack(pady=10)

connect_button = tk.Button(root, text="Connetti drone", command=connect_drone)
connect_button.pack(pady=10)

start_stop_button = tk.Button(root, text="Avvia streaming", command=start_stop_stream, state='disabled')
start_stop_button.pack(pady=10)

video_label = tk.Label(root)
video_label.pack()

root.mainloop()
```

## Canny Edge Detector

- Smooth by Gaussian

$$S = G_\sigma * I \qquad G_\sigma = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

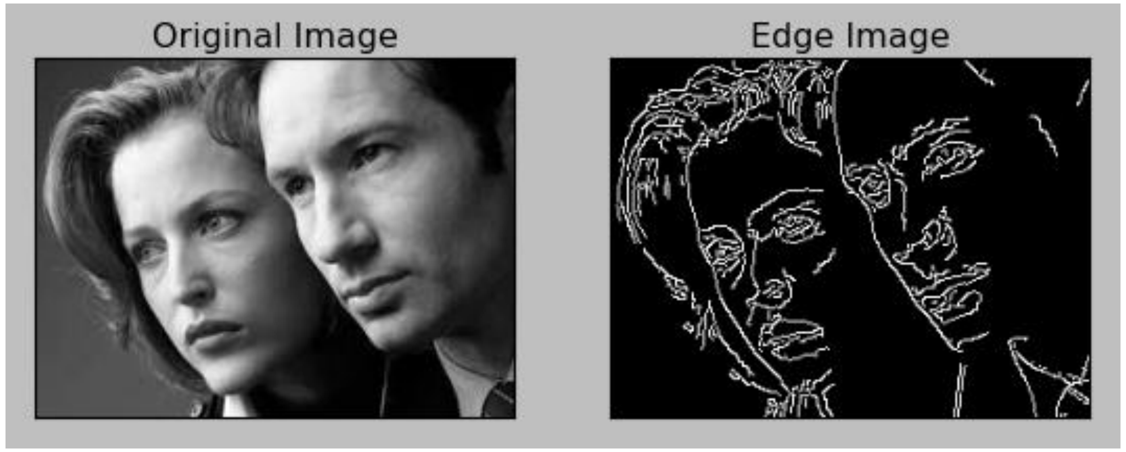
- Compute *x* and *y* derivatives

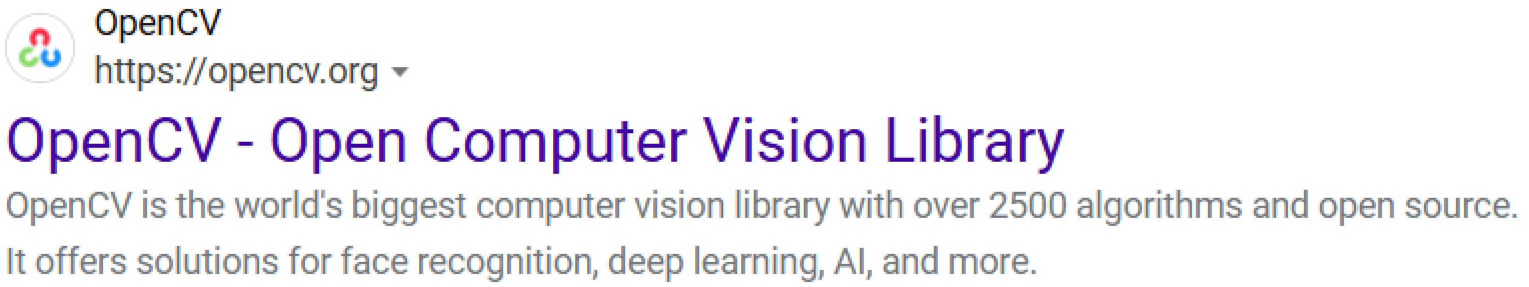
$$\nabla S = \begin{bmatrix} \frac{\partial}{\partial x} S & \frac{\partial}{\partial y} S \end{bmatrix}^T = \begin{bmatrix} S_x & S_y \end{bmatrix}^T$$

- Compute gradient magnitude and orientation

$$|\nabla S| = \sqrt{S_x^2 + S_y^2}$$

$$\theta = \tan^{-1} \frac{S_y}{S_x}$$

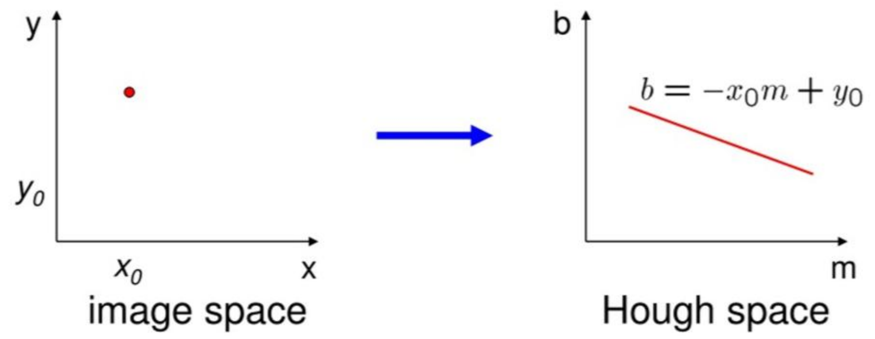



OpenCV
https://opencv.org

OpenCV - Open Computer Vision Library

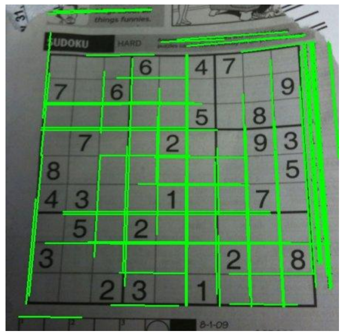
OpenCV is the world's biggest computer vision library with over 2500 algorithms and open source. It offers solutions for face recognition, deep learning, AI, and more.

## Finding lines in an image



Connection between image (x,y) and Hough (m,b) spaces

- A line in the image corresponds to a point in Hough space
- To go from image space to Hough space:
  - given a set of points (x,y), find all (m,b) such that y = mx + b
- What does a point ($x_0$, $y_0$) in the image space map to?
  - A: the solutions of $b = -x_0 m + y_0$
  - this is a line in Hough space



## Hough Transform for Curves

The H.T. can be generalized to detect any curve that can be expressed in parametric form:

- Y = f(x, a1,a2,…ap)
- a1, a2, … ap are the parameters
- The parameter space is p-dimensional
- The accumulating array is LARGE!

For circle: vote on $x_0$, $y_0$, r

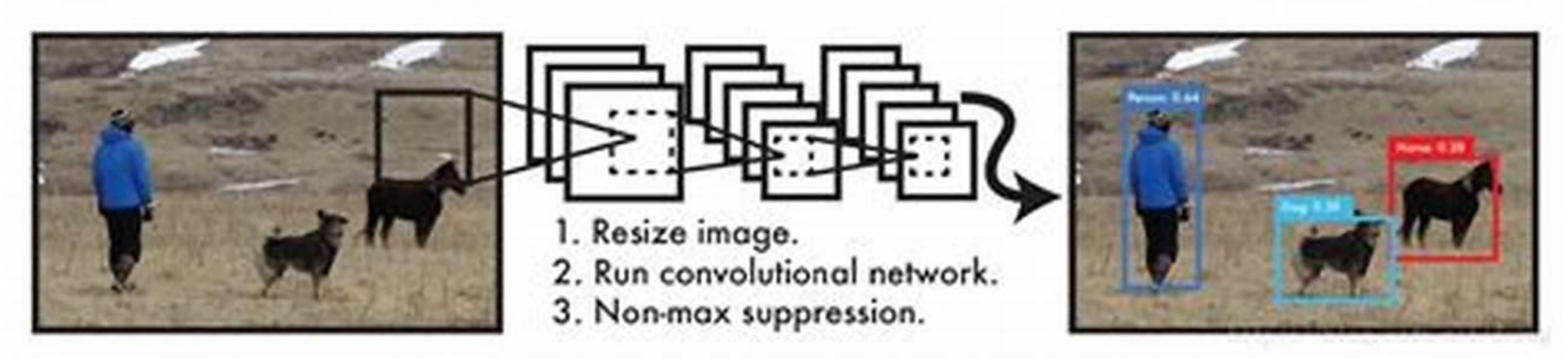
$$(x - x_0)^2 + (y - y_0)^= r^2$$

## What is YOLO?

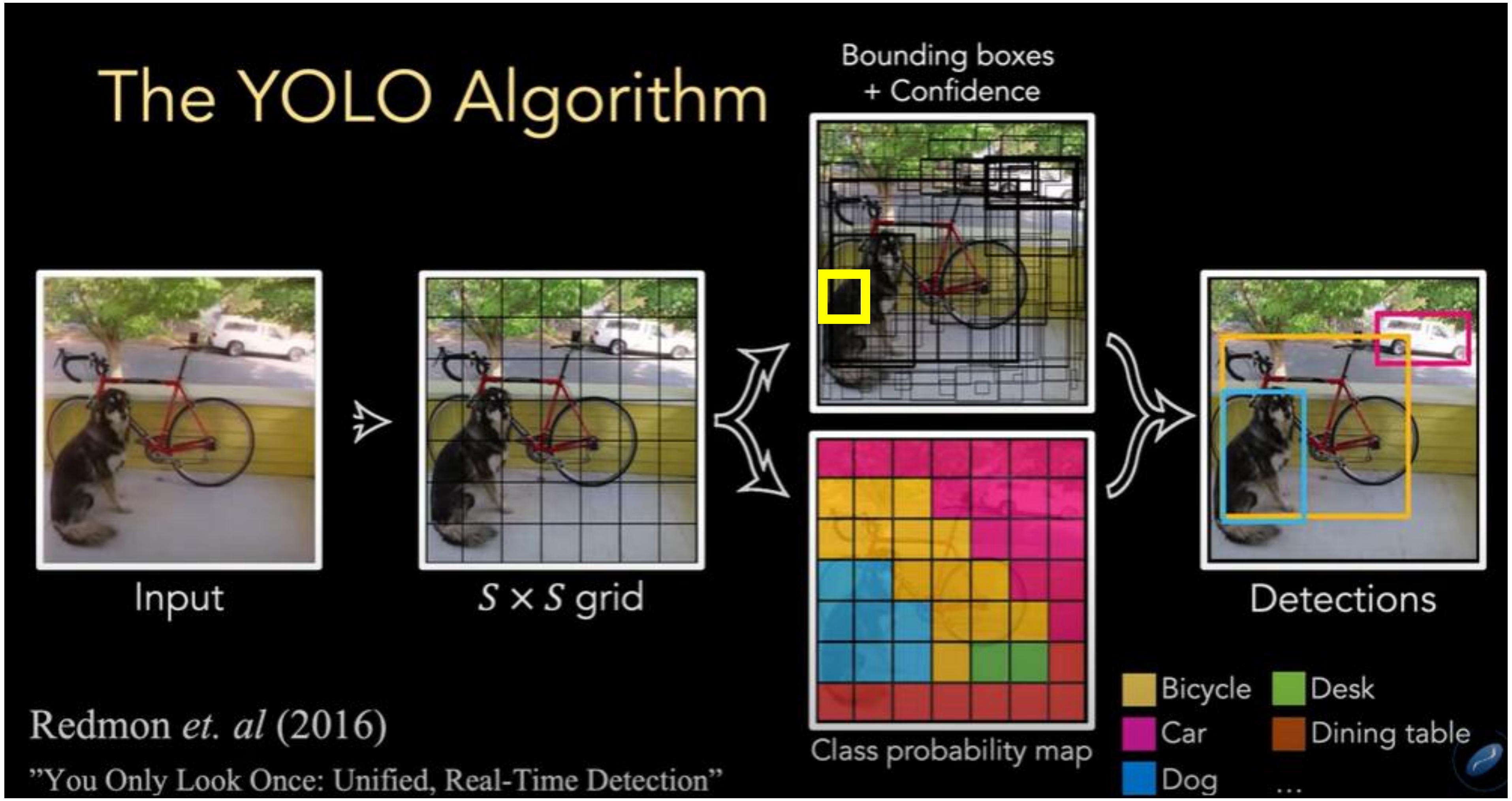
YOLO (You Only Look Once) is a real-time object detection algorithm introduced in 2015 that use a single convolutional neural network (CNN).

Instead of analyzing an image piece by piece, it processes the entire image simultaneously.

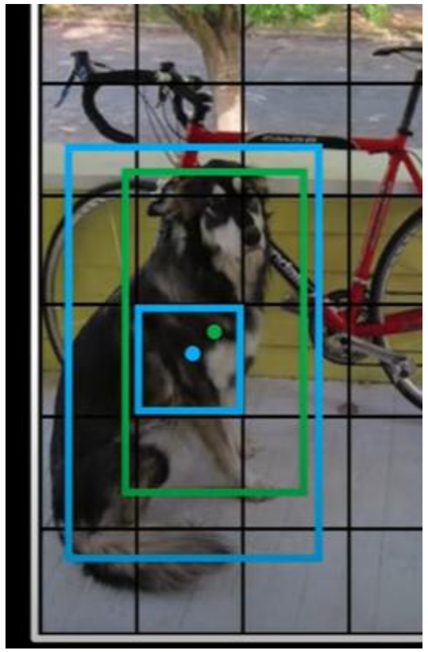
This single look approach makes YOLO incredibly fast, enabling real-time applications like self-driving cars and security systems.



## What is YOLO?



**GROUND TRUTH VS PREDICTION**



**Tasks:**

- **Programming with Python**: Writing code to control and automate drones.
- **YOLOv8 for Object Detection**: Teaching drones to recognize objects and make decisions.
- **Leveraging NVIDIA GPUs**: Using GPU power for real-time AI processing.

| HARDWARE |
|---|
| DELL LATITUDE 3540 |
| GPU NVIDIA GeForce MX550 |
| RAM 32 GB |
| INTEL CORE i5 |

nvidia-smi → supporting CUDA 11.6.

The Tello EDU establishes a **Wi-Fi connection** (hotspot) between the drone and your device **UP TO 10m**

| SOFTWARE |
|---|
| PYTHON 3.9 |
| CUDA 11.6 |
| Virtual Environment |
| Notepad++ |

## What is CUDA?

**CUDA** (Compute Unified Device Architecture) is a **parallel computing platform** and **application programming interface (API)** created by **NVIDIA**. It allows developers to use **NVIDIA graphics processing units (GPUs)** for general-purpose computing tasks.

## Traditional Computing vs. GPU Computing

Traditionally, the **CPU** handles tasks one at a time (serial processing), making it efficient for general-purpose computing.

In contrast, the **GPU** is built for parallel processing, handling many tasks simultaneously. This makes it ideal for **graphics rendering**, **computer vision**, and **artificial intelligence algorithms**, not just for images!!!