Raspberry Pi YOLO Autostart & Hotspot Deployment Guide

This guide walks you through setting up a Raspberry Pi to:

- Run YOLO object detection using the PiCamera

- Act as a Wi-Fi hotspot

- Send detection payloads (JSON + image) to a connected laptop

- Start everything automatically on boot

---

1. Requirements

Hardware

- Raspberry Pi 5 ([buy here](#))
- Camera ([buy here](#)) and Raspberry Pi Camera FPC 200mm Adapter Cable
- Memory Card ([buy here](#))
- Monitor, HDMI Cable, HDMI to USB Type C Converter, Keyboard and Mouse and a laptop for the TCP server

Hardware Assembly

Connect the thicker side of the cable to the camera, and the thinner side to the Pi 5. These connectors have a tab on them - lift them up, then insert the cable into the slot. Once it is sitting in there nice and square, push the tab back down to clamp the cable into place. Just keep an eye out as these connectors only work in one orientation, and they can be fragile so avoid bending them tightly (a little bit is okay). Use C cable slot for power supply to the Pi and HDMI connector for display on monitor, there are USB slots for peripherals.

System Assembly

We need to install Pi OS onto the micro SD card. Using the select Raspberry PI 5 as the Device and Raspberry Pi OS (64-bit) as the Operating system <u>Raspberry Pi Imager</u>, and your microSD card as the storage device.

NOTE: INSTALLING PI OS ONTO THE MICROSD CARD WILL WIPE ALL DATA ON IT.

This process may take a few minutes to download the OS and install it. Once the process has finished, insert it into the Pi and boot it up. Your Pi will run through a first-time installation and just ensure that you connect it to the internet. (<u>guide on how to flash on a fresh pi</u>)

Now we will be setting up Python and Virtual Environment with the libraries. Python Installation could be done with

```
sudo apt update

sudo apt install python3-pip -y
```

We then install

```
pip install -U pip
```

Now that we are working in a virtual environment, we can start installing the required packages.

To create a virtual environment, open a new terminal window and type in:

```
python3 -m venv --system-site-packages yolo_object
```

This will create a new virtual environment called "yolo_object". You can find the folder of this virtual environment under *home/pi* and it will be called "yolo_object".

After creating the venv, enter into it by typing in:

```
source yolo_object/bin/activate
```

Install required packages
```
sudo apt install libcamera-dev libopencv-dev python3-opencv
```

```
pip install ultralytics picamera2
```

---

2. Client setup on Pi for YOLO Inference

Save model_1.pt or model_2.pt (one is a lighter version) in: `~/dev/` same as where you save yolo.py

```python
import cv2
import socket
import json
import time
import base64
from datetime import datetime
from picamera2 import Picamera2
from ultralytics import YOLO

# Network settings
LAPTOP_IP = '10.42.0.250'  # Your laptop IP
PORT = 5000

# Detection targets
INTEREST_CLASSES = {
    "camouflage_soldier",
    "weapon",
    "military_tank",
    "military_truck",
    "military_vehicle",
    "soldier",
    "military_artillery",
    "military_aircraft",
    "military_warship"
}

# Initialize camera
picam2 = Picamera2()
```

```python
picam2.preview_configuration.main.size = (1280, 1280)
picam2.preview_configuration.main.format = "RGB888"
picam2.preview_configuration.align()
picam2.configure("preview")
picam2.start()

# Load model
model = YOLO("model_1.pt")


def encode_frame_to_base64(frame):
    success, buffer = cv2.imencode('.jpg', frame)
    if not success:
        raise ValueError("Could not encode frame.")
    return base64.b64encode(buffer).decode('utf-8')


def send_payload(label, confidence, encoded_image):
    payload = {
        "timestamp": datetime.isoformat(),
        "object_detected": label,
        "confidence": round(confidence, 3),
        "image_base64": encoded_image
    }
    try:
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
            print(f"[CLIENT] Connecting to {LAPTOP_IP}:{PORT}...")
            s.connect((LAPTOP_IP, PORT))
            s.sendall(json.dumps(payload).encode())
            print(f"[CLIENT] Sent detection: {label} ({confidence:.2f})")
    except Exception as e:
        print("[CLIENT] Failed to send payload:", e)


while True:
    frame = picam2.capture_array()
    results = model(frame)[0]

    for box in results.boxes:
        cls_id = int(box.cls[0])
```

```python
            confidence = float(box.conf[0])
            label = model.names[cls_id]

            if label in INTEREST_CLASSES and confidence > 0.70:
                try:
                    encoded_image = encode_frame_to_base64(frame)
                    send_payload(label, confidence, encoded_image)
                except Exception as e:
                    print("[CLIENT] Error preparing/sending image:", e)

        # Annotate and show
        annotated_frame = results.plot()
        fps = 1000 / results.speed['inference']
        cv2.putText(annotated_frame, f'FPS: {fps:.1f}', (10, 30),
                cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
        cv2.imshow("Camera", annotated_frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

cv2.destroyAllWindows()
```

2. Setting Up Hotspot

Enable and configure NetworkManager

```
sudo apt install network-manager
sudo systemctl disable dhcpcd
sudo systemctl stop dhcpcd
sudo systemctl enable NetworkManager
sudo systemctl start NetworkManager
```

Create the hotspot

```
nmcli device wifi hotspot ifname wlan0 ssid PiHotspot password 12345678
```

Pi will now be `10.42.0.1`.

Laptop (when connected) will get `10.42.0.xxx` (Could be checked with *ifconfig* terminal)

---

5. Server Script on Laptop

Save as server.py and run this on your laptop:

```python
import socket
import json
import base64
import os
from datetime import datetime

HOST = '0.0.0.0'
PORT = 5000
IMAGE_DIR = "received_images"
LOG_FILE = "detection_log.jsonl"

os.makedirs(IMAGE_DIR, exist_ok=True)

def save_image(base64_string, timestamp, label):
    try:
        image_data = base64.b64decode(base64_string)
        safe_timestamp = timestamp.replace(":", "_").replace(".", "_")
        filename = f"{IMAGE_DIR}/received_{safe_timestamp}_{label}.jpg"
        with open(filename, "wb") as f:
            f.write(image_data)
        print(f"[SERVER] Image saved to: {filename}")
    except Exception as e:
        print(f"[SERVER] Error saving image: {e}")

def append_log(payload):
    try:
        with open(LOG_FILE, "a") as log_file:
```

```python
                json.dump(payload, log_file)
                log_file.write("\n")
            print(f"[SERVER] Logged detection to {LOG_FILE}")
        except Exception as e:
            print(f"[SERVER] Error logging payload: {e}")


    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        s.bind((HOST, PORT))
        s.listen()
        print(f"[SERVER] Listening on port {PORT}...")

        while True:
            conn, addr = s.accept()
            with conn:
                # print(f"[SERVER] Connected by {addr}")
                print(f"[SERVER] Connected by 10.42.0.1")
                try:
                    chunks = []
                    while True:
                        chunk = conn.recv(4096)
                        if not chunk:
                            break
                        chunks.append(chunk)
                    data = b"".join(chunks)

                    if data:
                        payload = json.loads(data.decode())
                        print(f"[SERVER] Detection: {payload['object_detected']} ({payload['confidence']}) at
{payload['timestamp']}")
                        save_image(payload["image_base64"], payload["timestamp"],
payload["object_detected"])
                        append_log(payload)
                except Exception as e:
                    print(f"[SERVER] Error processing data: {e}")
```

6. Autostart YOLO on Boot

Create a wrapper script

```
nano ~/start_yolo.sh
```

Paste:

```
#!/bin/bash
source /home/pi/yolo_object/bin/activate
cd /home/pi/dev
python yolo.py
```

Make executable:

```
chmod +x ~/start_yolo.sh
```

Create systemd service

```
sudo nano /etc/systemd/system/yolo.service
```

Paste this:

```
[Unit]
Description=YOLO Object Detection Autostart
After=network-online.target
Wants=network-online.target

[Service]
ExecStart=/home/pi/start_yolo.sh
WorkingDirectory=/home/pi/dev
StandardOutput=inherit
StandardError=inherit
Restart=always
```

```
User=pi
Environment=DISPLAY=:0
Environment=XAUTHORITY=/home/pi/.Xauthority

[Install]
WantedBy=multi-user.target
```

Enable and start:

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable yolo.service
sudo systemctl start yolo.service
```

Check status:

```
sudo systemctl status yolo.service
```

Final Workflow Summary

1. Raspberry Pi boots and becomes a hotspot.

2. Laptop connects to PiHotspot (`10.42.0.1` gateway).

3. `yolo.py` autostarts and captures from PiCamera.

4. Pi sends payloads (JSON + base64 image) to the laptop server.

5. Laptop saves images and logs JSON per detection.