YONO web cam

!pip install ultralytics

# YOLOv8 Live Object Detection (notebook + desktop friendly)

# 1) Installs (keep these ONLY in notebooks/Colab)

# !pip install ultralytics opencv-python ipython

import os, sys, time

import cv2

from ultralytics import YOLO

# Notebook-safe display

IN\_NOTEBOOK = any(host in sys.modules for host in ("google.colab", "ipykernel"))

if IN\_NOTEBOOK:

    try:

        from google.colab.patches import cv2\_imshow  # Colab

    except Exception:

        from IPython.display import display, Image  # fallback (not used)

        cv2\_imshow = None

# 2) Load model (nano = fastest; switch to yolov8s.pt/m.pt for accuracy)

model = YOLO("yolov8n.pt")

# Optional: tweak default inference settings

CONF\_THRESH = 0.35  # ignore super-low confidence boxes

IOU\_THRESH = 0.45

# 3) Open camera (index 0). On Windows, CAP\_DSHOW can help open faster.

cap = cv2.VideoCapture(0, cv2.CAP\_DSHOW) if os.name == "nt" else cv2.VideoCapture(0)

if not cap.isOpened():

    print("Error: Cannot open webcam. Is it connected / accessible?")

    raise SystemExit(1)

# Set resolution (optional; higher res = slower)

cap.set(cv2.CAP\_PROP\_FRAME\_WIDTH,  640)

cap.set(cv2.CAP\_PROP\_FRAME\_HEIGHT, 480)

# 4) Track unique classes seen

all\_detected = set()

print("Starting live detection... Press 'q' (desktop) or stop the cell (notebook) to quit.")

prev\_time = time.time()

frame\_count = 0

fps = 0.0

try:

    while True:

        ok, frame = cap.read()

        if not ok:

            print("Failed to grab a frame. Exiting...")

            break

        frame\_count += 1

        now = time.time()

        dt = now - prev\_time

        if dt >= 1.0:

            fps = frame\_count / dt

            frame\_count = 0

            prev\_time = now

        # 5) Inference

        # Using 'model' directly is OK; conf & iou can be passed here

        results = model(frame, conf=CONF\_THRESH, iou=IOU\_THRESH, verbose=False)

        # 6) Draw boxes

        for r in results:

            if not hasattr(r, "boxes") or r.boxes is None:

                continue

            for b in r.boxes:

                cls\_id = int(b.cls[0])

                cls\_name = model.names.get(cls\_id, str(cls\_id))

                conf = float(b.conf[0])

                x1, y1, x2, y2 = map(int, b.xyxy[0])

                # Box

                cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)

                # Label

                label = f"{cls\_name} {conf:.2f}"

                cv2.putText(frame, label, (x1, max(20, y1 - 8)),

                            cv2.FONT\_HERSHEY\_SIMPLEX, 0.6, (0, 255, 0), 2)

                all\_detected.add(cls\_name)

        # 7) Show FPS

        cv2.putText(frame, f"FPS: {fps:.1f}", (10, 25),

                    cv2.FONT\_HERSHEY\_SIMPLEX, 0.7, (255, 255, 255), 2)

        # 8) Display frame

        if IN\_NOTEBOOK and cv2\_imshow:

            # In notebooks, reusing windows is messy; just show occasionally

            # (For real-time display, local desktop is better.)

            display\_frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

            cv2\_imshow(display\_frame)

            # WARNING: This will spam images; for Colab, prefer processing a video/file instead.

            break  # Show one frame then stop in notebook to avoid flooding

        else:

            cv2.imshow("Live Object Detection (press q to quit)", frame)

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

                break

except KeyboardInterrupt:

    print("\nStopped by user (KeyboardInterrupt).")

finally:

    print("\nCleaning up...")

    cap.release()

    cv2.destroyAllWindows()

    if all\_detected:

        print("\n--- Summary of All Unique Objects Detected ---")

        for i, name in enumerate(sorted(all\_detected), 1):

            print(f"{i}. {name}")

        print("----------------------------------------------")

    else:

        print("No objects detected.")