Stabiler Code vor mqtt ergänzung

```python
import cv2
from ultralytics import YOLO

# ------------------ Einstellungen ------------------

MODEL_PATH = r"C:\Cam\yolo26n.pt"   # Pfad zu deinem .pt
SOURCE = 0                          # 0 = Standard-Webcam

# Linie (Türschwelle) im Bild anpassen!
LINE_P1 = (300, 400)   # Punkt 1 (x1, y1)
LINE_P2 = (200, 100)   # Punkt 2 (x2, y2)

# Minimale Bewegung, damit ein Seitenwechsel gezählt wird
MIN_MOVE_PIXELS = 5

# ------------------ Logik-Variablen ------------------

Room_count = 0
last_positions = {}  # {track_id: (x, y)}

# ------------------ Hilfsfunktionen ------------------

def point_side_of_line(px, py, x1, y1, x2, y2):
    """
    Gibt das Vorzeichen der Punktlage relativ zur Linie zurück.
```

```
    >0: eine Seite, <0: andere Seite, 0: genau auf der Linie.
    """
    return (x2 - x1) * (py - y1) - (y2 - y1) * (px - x1)


def draw_info(frame):
    """Zähler und Linie ins Bild zeichnen."""
    cv2.line(frame, LINE_P1, LINE_P2, (0, 255, 255), 2)
    cv2.putText(frame, f"IN: {Room_count}", (10, 30),
                cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)


# ------------------ Hauptprogramm ------------------

def main():
    global Room_count, last_positions

    # Modell laden
    model = YOLO(MODEL_PATH)

    print("Starte Kamera... Drücke 'q' zum Beenden")

    # Tracking-Loop
    for result in model.track(
        source=SOURCE,
        show=False,
        stream=True,
        persist=True,
        classes=[0],       # nur "person"
        verbose=False,
    ):
```

```
        # Bild holen (mit eingezeichneten Boxen)
        frame = result.plot()
        draw_info(frame)

        # Boxen / Tracks auswerten
        if result.boxes is not None and len(result.boxes) > 0:
            boxes = result.boxes
            track_ids = boxes.id

            # WICHTIG: track_ids kann None sein, wenn keine Tracks vorhanden
            if track_ids is not None:
                for box, tid in zip(boxes.xyxy, track_ids):
                    track_id = int(tid.item())
                    x1, y1, x2, y2 = box.tolist()
                    cx = int((x1 + x2) / 2)
                    cy = int((y1 + y2) / 2)

                    # Schwerpunkt einzeichnen
                    cv2.circle(frame, (cx, cy), 4, (255, 0, 0), -1)
                    cv2.putText(frame, f"ID {track_id}", (cx + 5, cy - 5),
                                cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 1)

                    # Vorherige Position holen
                    if track_id in last_positions:
                        prev_cx, prev_cy = last_positions[track_id]

                        # Nur auswerten, wenn sich die Person genug bewegt hat
                        dist_sq = (cx - prev_cx) ** 2 + (cy - prev_cy) ** 2
                        if dist_sq >= MIN_MOVE_PIXELS ** 2:
                            # Seite relativ zur Linie vorher / nachher
                            prev_side = point_side_of_line(prev_cx, prev_cy,
```

```
                                                    LINE_P1[0], LINE_P1[1],
                                                    LINE_P2[0], LINE_P2[1])
                        curr_side = point_side_of_line(cx, cy,
                                                    LINE_P1[0], LINE_P1[1],
                                                    LINE_P2[0], LINE_P2[1])

                        # Seitenwechsel -> Linie überquert
                        if prev_side * curr_side < 0:
                            # Richtung über y-Bewegung (an Kamera anpassen!)
                            if cx > prev_cx:
                                Room_count -= 1
                                if Room_count < 0:
                                    Room_count = 0
# Verhindert negativen Zähler
                                print(f"ID {track_id} ENTERED, IN: {Room_count}
")
                            else:
                                Room_count += 1
                                print(f"ID {track_id} EXITED, IN: {Room_count}"
)

                    # aktuelle Position speichern
                    last_positions[track_id] = (cx, cy)

        # Bild anzeigen
        cv2.imshow("YOLO Room Counter", frame)

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

    cv2.destroyAllWindows()
```

```
    print(f"\nFinale Zählerstände: IN={Room_count}")



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

Revision #1
Created 24 March 2026 07:33:20 by Nico
Updated 24 March 2026 07:33:35 by Nico