Report on YOLO-Based Object Detection in Video

Amirreza Soltani 3121668

Amirreza.soltanii@yahoo.com

1. Introduction

This project implements an object detection system using the YOLOv8 (You Only Look Once) model to process a video file from SRH building. The system detects objects frame by frame, highlights them with bounding boxes, and generates an output video with visualized detections. Additionally, the program identifies unique objects in the video and displays them at the end for 10 seconds.

---

2. Implementation Details

2.1. YOLO Model & Dependencies

The YOLOv8n model from the Ultralytics library is used for object detection.

The implementation requires:

cv2 (OpenCV) for video processing

torch for deep learning model execution

numpy for handling image arrays

2.2. Video Processing Workflow

Video Loading: The input video (IMG_7870.mp4) is opened using cv2.VideoCapture().

Frame Extraction & Processing:

Each frame is read, and YOLO detects objects in it.

Bounding boxes and labels are drawn on detected objects.

Frames are written to a new output video file (output_video.mp4).

Unique Object Detection:

A set (unique_objects) is used to track detected object labels.

This ensures each type of object is recorded only once.

Video Speed Adjustment:

The playback speed is slowed down by modifying the frames per second (FPS).

The slow_factor parameter (set to 1.1) adjusts the speed.

Displaying Unique Objects at the End:

After all frames are processed, a black screen frame is created.

The names of all detected objects are displayed.

This screen is shown for 10 seconds in the final video.

2.3. Output

A processed video (output_video.mp4) with detected objects.

A list of unique detected objects is printed in the console.

The final 10-second display of detected objects in the output video.

---

3. Unique Objects Detected

During the video processing, the following unique objects were detected:

['couch', 'boat', 'traffic light', 'car', 'keyboard', 'microwave', 'person',

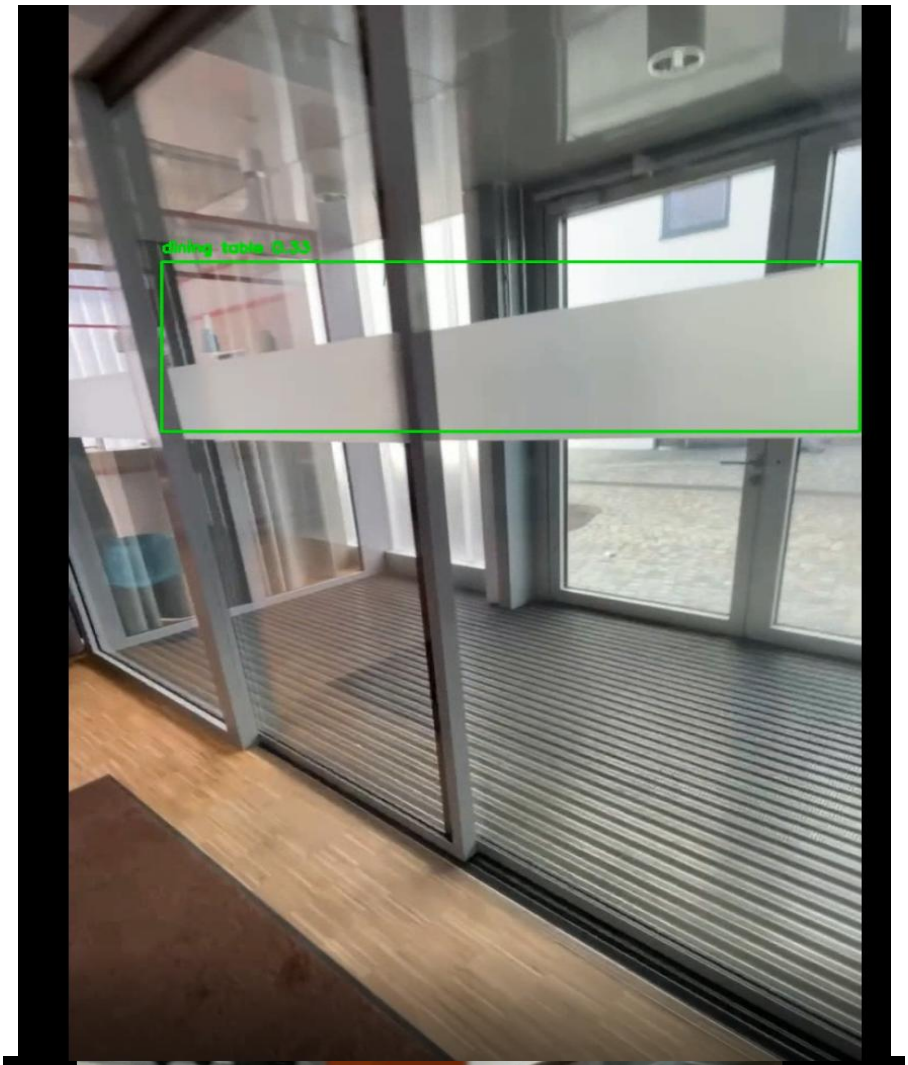
'cell phone', 'potted plant', 'cake', 'laptop', 'bench', 'bed', 'oven',

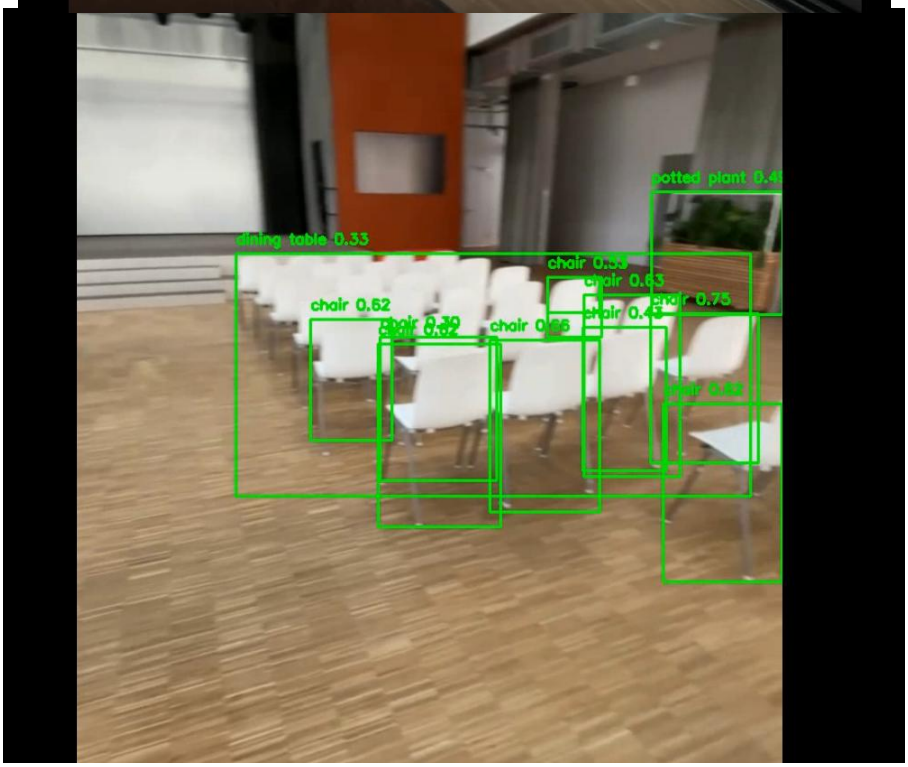
'umbrella', 'clock', 'chair', 'tv', 'dining table', 'toilet']

After checking detected objects we can see some errors (highligthted in red) in addition there are some obvious errors like recognizing door as dining table or recognizing a group of chair as dining table! It seems yolo8n model likes dining table.

With yolo11n model :

['clock', 'dining table', 'cup', 'orange', 'cake', 'chair', 'tennis racket', 'tv', 'surfboard', 'traffic light', 'car', 'potted plant', 'person', 'cell phone', 'fire hydrant', 'bench', 'donut', 'toilet', 'boat', 'refrigerator']


dining table 0.33


potted plant 0.4
dining table 0.33
chair 0.53
chair 0.63
chair 0.75
chair 0.62
chair 0.66
chair 0.83

---

4. Key Features

Efficient Object Detection: Uses YOLOv8 for real-time performance.

Video Speed Control: Slows down playback using FPS adjustments.

Bounding Box Visualization: Highlights objects in frames.

Unique Object Tracking: Prevents duplicate object entries.

Final Object Summary: Displays detected object names for 10 seconds.

---

5. Possible Enhancements

Improve performance by running YOLO on GPU for faster processing.

Add an option to save detected object data in a separate file (e.g., JSON or CSV).

Implement object tracking across frames for smoother results.

---

6. Conclusion

This project successfully integrates YOLOv8 with OpenCV to detect and track objects in a video. The system effectively visualizes detections, tracks unique objects, and presents a final summary screen. This approach is applicable in various fields, including security surveillance, autonomous vehicles, and smart monitoring systems.