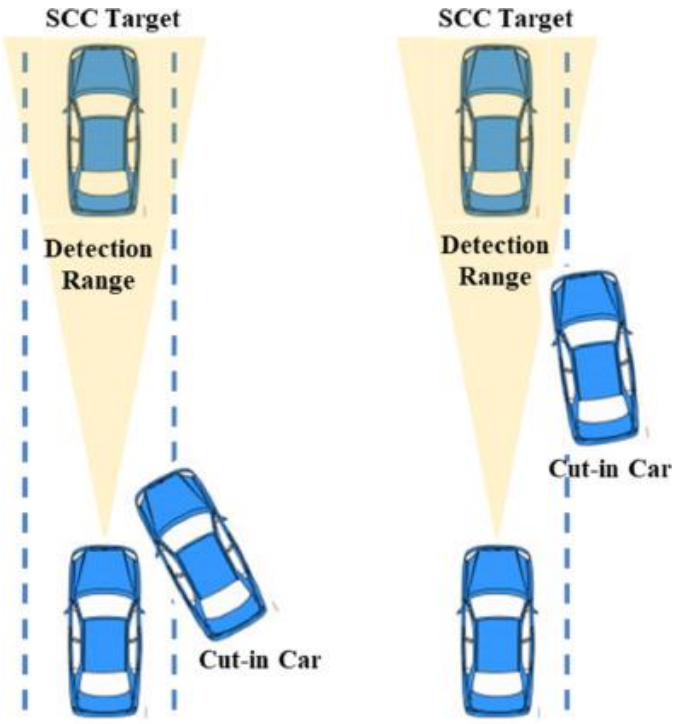
VEHICLE CUT-IN DETECTION

Team alt-tab



Technical Approach:

- Our team divided the problem into 4 steps:
    1. Detection
    2. Distance Estimation
    3. Time Calculation
    4. Warning Generation

- The video source used in the project is recorded using phone camera placed on the central region of the car dashboard.

- Software Used: Jupyter Notebook

- Detection: For detection, we trained the models using yolov8, python and coco dataset hence getting an accurate detection result. We set a detection area (called as guidelines), in the shape of a trapezoid, whose dimensions were calculated with respect to the width of the car bonnet. Each object in the frame is detected and is bounded by a diamond shape having dotted edges. The dotted edges are the centre points of each side of the apparent rectangular box which would have bounded it if not for the diamond. If any of the edges enters the trapezoid, the object is bounded by a box thus tagging the inbound object to the driver.

- Distance Estimation: As soon as any of the edges enter the detection area, the object is bound by a box. To calculate the distance, a horizontal line is drawn parallel to the x axis along the y coordinates of the lower base line of the bounding box bound which cuts the trapezoid at some point .The coordinates of the intersected point are fetched and corresponding Euclidean distance is calculated from the intersected point to the bottom corner coordinates of the trapezoid. Since the distance obtained is in pixels, it is converted into metre by an approximate value

- Time Calculation: For calculating time, we assumed speed of the car to be a constant value. Time can now be calculated using the formula ***TIME=DISTANCE/SPEED***. We had a different approach for live speed calculation of our car but it could not be implemented due to a setback we faced while working on the project. The problem shall be explained later in the PROBLEMS FACED section.

- Warning Generation: A warning message is generated for each object only if the calculated time falls below the threshold value, i.e. 0.7s. Counter measures are taken based on the warnings received.

Code Link:

- https://github.com/ryanmjball/project/tree/code

Problems Faced:

- We were having our End Semester Examination from June 6 to June 28 due to which we lost a large amount of time required for good planning and organising.

- The video output in yolo was laggy and sluggish due to which we could not work on live camera footage. We tried different methods to boost the PC performance but unfortunately, we could not arrive at a solution. This forced us to work on recorded footage.

- Our alternative method for speed calculation was to get the live coordinates of the device used from google maps using python. Since the device is used within the moving car, getting the coordinates of the device would be equivalent to getting the coordinates of the car. The change in coordinates with each frame time can be used to calculate the distance and ultimately the car's speed. However, this method could only be implemented on live footages which we could not work on due to the laggy and sluggish video output.

Results:

- If any object enters the trapezoidal detection area in front of our car, the object would we tagged immediately.
- The distance between the tagged object and our car is found out using intersection technique.
- Keeping the vehicle speed constant, instantaneous time is calculated using Time=Distance/Speed.
- If the calculated time is below threshold value (0.7s), the driver is instantly alerted and counter measures are taken.

Video Link:

- <Paste github video link here>