Visual traffic monitoring at a road intersection project

Computer Vision Project 2 – LARA TOMEH 507

June 2023

Task 1 - Classify the road lanes as being occupied or not.

For this task, I experimented with an approach of dividing the street lines into 9 lanes, as required. I used a completely empty image with no vehicles. Then, after converting the empty lane with the required lane to be predicted to grayscale, and applied Gaussian blur, I subtracted the test image from the original image using the OpenCV function (absdiff). However, this approach did not produce satisfactory results. In addition, there was a lack of empty images taken at different times of the day as templates. The only empty image I found was mostly captured during sunset. When subtracting images taken during daylight, the results showed significant differences due to lighting variations, indicating the presence of a vehicle in that row. Therefore, this approach was incorrect. Therefore, I utilized YOLO (You Only Look Once) in version 8 from Ultralytics framework. After detecting the vehicles in the image, I identified points for the nine polygons (lanes) that need to be tested. I then defined conditions for these polygons and predicted bounding boxes, so that I could predict the actual vehicle's location using the coordinates of the bounding box surrounding the vehicle.

Task 2 - Track a specific vehicle from the initial frame to the final frame of the video

For this task, the desired car was simply tracked using Python trackers, specifically Boosting. All the trackers available in Python were tested, including Boosting, MIL, KCF, TLD, Median Flow, GOTURN, and MOSSE. Each tracker has its own features and drawbacks, and they differ in terms of characteristics and algorithmic approaches. However, the Boosting tracker had the best performance for our special case.

Task 3 - Count the trajectories of vehicles in the given video.

For the third task, YOLO v8 was also used to detect vehicles in the video and three regions were defined as boundaries for the areas where vehicles move to and from. The approach is similar to the first task, but this time we need to use a tracker to assign a unique ID to each vehicle and not consider it as a new vehicle, we used Euclidian distance tracker. Conditions were set for each vehicle, and these conditions may vary from case to case. However, in general, a common condition is that if the vehicle's center moves from the first polygon to the second one, it indicates that the vehicle has transitioned from the first region to the second region, for example.