Visual Computing Report

Nicolas Bourriez

March 14, 2023

1 Introduction

The purpose of this project was to track vehicles in a series of videoframes without using deep learning techniques. In this report, I will discuss the main problems I faced in this project, the datasets I used, the techniques explored, fine-tuning and hyperparameters, difficulties encountered, and finally the additional techniques that I explored but failed to make it work.

2 Main Problematics to Tackle

The project faced several challenges.

First, the fact that we face *scale variation*, sometimes *occlusion*, and *deformation* of the cars on the highway makes it difficult not to get many false positives.

Similarly, a lot of wheeled working machines, tires, and signs that make it even more difficult for the classifier.

Finally the fact that the frames come from an *on-board* dashcam i.e. rendering a constanstly *moving background*, made it impossible to use techniques such as **Background Subtraction** efficiently...

3 Data

The datasets used in this project were the GTI, KITTI, and Udacity datasets. The GTI dataset contains vehicle images from several different viewpoints, while the KITTI dataset includes images of vehicles from a moving camera. The Udacity dataset contains annotations for object detection and tracking.

I extracted vehicle images from the train set initially provided, but two main problems came out of this use:

- The fact that using this additional data source might have **bias** the dataset, adding way too many *vehicle images* and making it ***unbalanced***.
- Some extracted images were unsable, because of some very tiny bounding boxes

4 Techniques explored

Several techniques were explored, including

- background subtraction using `cv2.createBackgroundSubtractorMOG2()`
- Dense optical flow
- Multiple scales sliding window over frame using a classifier using HOG features extraction

After many tests trials, the HOG features extraction seemed to be the best working solution. I thus tested several configs, such as trying to get the right number of features per image (not too few, but not too many to avoid overfitting). I found that using Spatial Bins + Colors bins + HOG with specific configuration (`pixel_per_cell=(7,7)`) seems to be a good compromise.

Once the features were extracted, I needed to find a good classifier, that would be able to face the main challenge of **high dimensional** features. After having trying mainly SVM, RandomForest and AdaBoost, it appeared that empirically RF works best. This classifier was applied to the vehicle and non-vehicle images combined feature vectors to identify the cars in the dataset images.

Once trained on the train dataset, the `VehicleDetector` divides each frame with different grids of windows, using smaller grids on smaller `y_axis` to reduce useless extraction and be able to detect smaller cars as they appear smaller away.

5 Fine-tuning and Hyperparameters explored

Several fine-tuning and hyperparameters were explored in this project, which resulted each time in a *better* performance or a *worse* one. These techniques included the use of external datasets (*better*), extraction of vehicle images from the original dataset (*worse*), using `spatial_bin` and `color_hist` as additional features (*better*), investigating the use of `canny_edge` application before hog extraction (*worse*), building an adaptive threshold for the heatmap (*better*), having different window sizes for the window search (*better*, building a Dice score to test on the original dataset (*usefull*), and filtering bounding boxes by size (*better*). Data augmentation techniques, including center crop, horizontal flip, and brightness change, were also explored and gave slightly *better* results.

6 Difficulties encountered

I encountered several difficulties in this project, but the main ones were finding the right position of the sliding windows and the overlap, avoiding false positives over the artefacts such as tyres and signs, and identifying a suitable threshold for the heatmap that would adapt on every frame (i.e. solved using pythonnp.max()*threshold).

Another main overall difficulty was the **lack of computational power** on my computer, which slowed down my progress considerably: comparing my computational time (Apple Intel i5) with a similar kind of detection algorithm (SVM Classifier — features: 6700) on all the test frame with one of my friends (Apple M2), the difference was outrageous:

- Apple i5: 1h36
- Apple M2: 18min

7 Additional techniques explored but failed

The first additional technique that I wanted to explore was to perform a GridSearch over a whole set of parameters (i.e. see the `params.json` file, but a lack of power on my computer forced me to try by hand.
The second additional technique that I tried is ***frame aggregation***:

- over each frame, check a minimim distance between newly detected bounding box and current Objects. If they appear to be similar objects, I smooth the bounding box of Object with the new bounding box and return it. If no similar, I add the new bounding box as new Object. IMPORTANT: I parametrized the `min_centroid_dist` to be proportional to the size of the Object, since a closer object will move more between each frame !
- over X frames, prune and return only current Objects

The main difficulty was the **fine-tuning** of the algorithm, which boils down to finding a good enough minimum detection count over multiple frames, and aggregating the combined window dimensions for overlapping detections.

8 Conclusion

In this project I used several computer vision techniques and machine learning algorithms to track vehicles in a video. Several datasets were used, and several techniques were explored, including background subtraction and particularly HOG features extraction with classifier classifier. I faced several challenges, including scale variation, occlusion, deformation, and moving background, and several fine-tuning and hyperparameters were explored. Despite encountering some difficulties and trying several additional techniques, the project achieved the objective of tracking vehicles in a video without using deep learning techniques.