Video Surveillance for Road Traffic Monitoring

Yi Xiao Ferran Carrasquer Juan Felipe Montesinos
Universitat Politcnica de Catalunya
Computer Vision Center
{yixiaoyixiao123,ferrancarrasquer,jfmontgar}@gmail.com

Abstract

With this work it is introduced an affordable computer vision based traffic monitoring system. The objective of this system is to track and predict the speed of the vehicles along the road. Different computer vision techniques are used such as video stabilization, foreground segmentation, region tracking and deep learning networks for detecting cars. The system can run on a regular CPU.

1. Introduction

As it is widely known, car accidents are one of the main mortality causes in the world. From the biggest to the small companies and governmental institutions there have been seen lot of efforts on decreasing this fatal cause of mortality by starting projects and campaigns to make the society more aware of the real hazard that it has to do the task that most of the people does every day, driving.

Despite what can be thought to be the most important factor of car accidents, velocity plays an important role when there is a crash. Its been proved that driving at speeds above 50 km/h the provability of death on a car accident increases exponentially with the velocity. This is the main point for the introduction of a methodology to capture the speed of the different cars passing through the same road where there is supposed to be a surveillance camera.

Although typical speed controls are done using RADARS, there is presented an alternative for those RADARS based on computer vision techniques, making it way more cheap and affordable for the institutions in charge of the safety of the roads.

The techniques used for developing the system are: video stabilization, foreground segmentation, region tracking and speed estimation. After developing the first version of tracking it has been improved by means of a Neural Network and computed the speed detector that has been extended to a collision detector evaluator based on the current position of the cars, speed and physic characteristics.

The project code can be found on GitHub[1].

2. Related Work

During the past two decades road traffic monitoring has been an active topic of research. With the benefit of the recent advances in automated video analysis, current speed estimation systems based on video cameras are becoming a reality.

With the intention to name a few related work, in [4], the authors perform region based detection and featured based tracking. In [1] the authors detect moving vehicles through background/foreground segmentation techniques and estimate vehicles speed per class using feature tracking and nearest neighbors algorithms, similar to the previous mentioned. In [3], vehicle detection is based on motion detection, using kernel density estimation in a pixel-based technique. Besides, the authors use a 3D pose estimation to take into account occlusions, and a Kalman filter to perform the tracking. As a last work,[5] presents a pyramidal approach of Lucas-Kanade algorithm to estimate motion vectors, and tracking of moving objects is performed using Kalman filter, tracking single or multiple moving objects over the frames recorded.

3. System and Methods

In this section, the algorithms and methods were used in this project will be described. Two datasets have been used in this project, the Highway and UAB Road.

3.1. Video Stabilization

Since the cameras are usually placed in traffic signs, poles or cars, the monitoring videos which are captured could be jittery, which may result in a bad object detection. Thus, it is important and necessary to do stabilization on these videos.

The main idea of video stabilization is taking one region of the frame which is not moving in reality as reference, fix-

[1] https://github.com/mcv-m6-video/mcv-m6-2018-team6

ing all the other frames to an appropriate location. More in details, the optical flow with block matching with respect to the reference frame was computed, and then the translation matrix need to be defined and applied to each frame for the rectification. Video Stabilization was applied in the UAB Road dataset which is not stable enough to implement a tracking. Results are shown in Fig. 1. The yellow boxes indicate the reference region.

Figure 1. Results of Video Stabilization.(left: Before; right: After)

3.2. Vehicles Tracking

Vehicles tracking is the process of locating a moving car (or multiple cars) over time using a camera. Two approaches have been implemented to achieve tracking in this project: Kalman Filter in Sec. 3.2.1 and Deep Learning approach (Recurrent YOLO Network) in Sec. 3.2.2.

3.2.1 Kalman Filter Approach

Kalman filter is an optimal method in case of a linear dynamic model with gaussian noise. Having any uncertain information about a dynamic system, an educated guess about what the system is going to do next can be made. It has many uses, including applications in control, navigation, computer vision, and time series econometrics.

Since Kalman Filter should be implemented after object detection, background subtraction has been performed firstly. A function from OpenCV which is based on gaussian mixture algorithm can be used. It selects the appropriate number of gaussian distribution for each pixel, and provides good adaptability to varying scenes due illumination changes. You can see an example of the result frame after background subtraction in Fig. 2

Figure 2. Results of background subtraction.(left: original image; right: result image)

After background subtraction, the main idea next is to find out the centroids of the objects which indicate the positions of the cars, and take them as the input of the kalman filter. For each frame, the steps can be summarized as below:

1. Detect tracks and obtain centroids and bounding box;
2. Create tracks and assign track ID;
3. Calculate cost between track prediction and detection;
4. Use Hungarian Algorithm determine and assign every detected cars to correct track ID;
5. Maintain tracks and handle unassigned track predictions and detections;
6. Apply Kalman Filter to predict and update tracks.

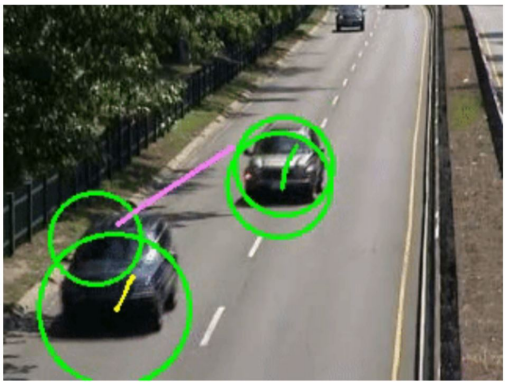

Figure 3. Result of Kalman Filter Tracking

The result of Kalman Filter tracking in Fig.3 was not as good as we expected, since the object detection and foreground segmentation were not good enough, resulting in a bad tracking. The tracker did not stay for many frames and the Object detection was not stable. With this thinking, a new approach based on Deep Learning has been tried in Sec. 3.2.2 to perform a better object detection for cars tracking.

3.2.2 Recurrent Neural Network

Nowadays deep learning achieves high accuracy rates in object detection task. If this accuracy is high enough to detect frame by frame all the vehicles inside, it is possible to skip foreground detection replacing it by an object detection network. At this point it is possible to apply any classic technique to assign IDs and perform tracking. In this specific case Yolo has been used as object-detection NN. As tracker an algorithm based in Intersection over Union has been developed. This algorithm uses an all vs all technique to compare detections between two different frames, assigning the ID corresponding to maximum IoU if this value is higher than a arbitrary threshold. Yolo is an accurate and

fast object detection network, but it may miss some detections. That's why the already explained recurrence explores deeper in the past (N frames, arbitrary parameter), applying the algorithm the detections whose ID has not been assigned yet. In case of after comparing these N frames there were no matching, a new ID would be assigned. Results, though

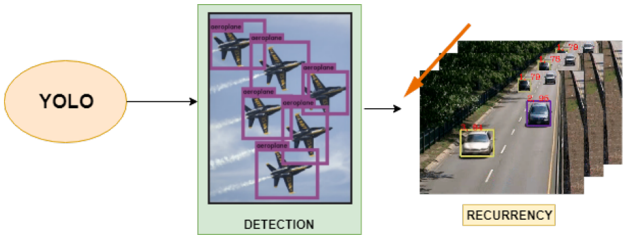

Figure 4. Working scheme proposed for tracking.

qualitative, are better than using classical tools with a high FPS (as high as object detection NN).

3.3. Speed Estimation

Speed estimation is a difficult-to-generalize task in Computer Vision. It is necessary to have a real world measurement or any assumption to build a feasible model. What's more camera perspective provokes non-linearities in the real space each pixel represents.

In this case the real-world measurement is distance between dashed lines (12 m). For solving non-linearity problems it is done an rectification of the road handpicking 4 points. This way a front view of the road is obtained. In this front view each pixel of the image represents the same amount of real space, being now able to compute $v = \frac{\Delta s}{time}$. The proper way to obtain speed in [km/h] is:

$$v = \frac{\Delta s \cdot FPS \cdot 3.6}{rate[pixels/meters]} \tag{1}$$

As you can see in the Figure 5, since car is not contained by

Figure 5. Rectification performed handpicking 4 points.

the plane formed by the road it suffers an unwanted deformation. For calculating pixel displacement between frames (which in the end is proportional to speed), two main ways were proposed. The first one is just computing the displacement of bounding boxed.

- It's computationally cheap
- Single point provokes error propagation of a bad bounding box estimation
- If yolo fails detecting a vehicle at some frame it's not possible to calculate it

The second one is computing optical flow in this rectified space (which is the chosen option) using DeepFlow V2:

- Computationally expensive
- Averaged measurement which compensates any error
- Detection-independent

Results are qualitative successful for this approach. Cars at the left way of the road are faster than ones on the right way, which is coherent. Speeds are in a logical range for a motorway. Tracking is stable (with some mistakes due to yolo, which miss detection for several frames (12) in a few cases)

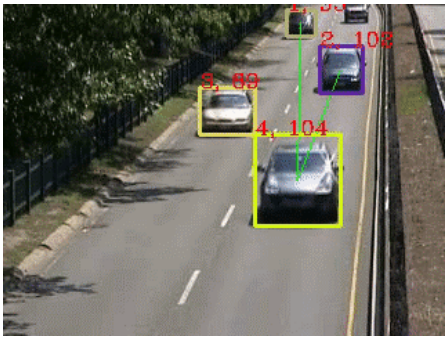

Figure 6. General result of tracking and speed estimation. ID at left, speed [km/h] at right

3.4. Collision Detector

Once the velocity of a car can be determined by means of computer vision techniques, we implement our own case study focused on evaluating the risk of collision from 2 different cars on the road one in front the other.

For computing the stopping distance of a car it is based on [2] where it is introduced the stopping distance= reaction distance + breaking distance. While the reaction distance can be approximate as a 0.5 times the current velocity at x time, the breaking distance can be approximated using the Newton's laws of motion, where the cinematic energy that the car has is going to be transformed to the friction between the tires and the road as it is showed below:

$$\frac{1}{2}mv^2 = F.d = \mu N.d \rightarrow \mathbf{d} = \frac{v^2}{2\mu g} \tag{2}$$

Once the stopping distance is known, it is necessary to set

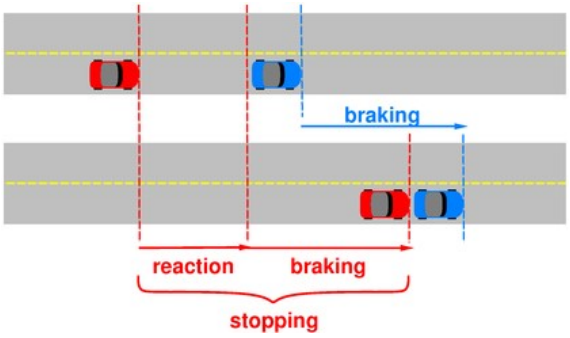

Figure 7. Illustration of the Stopping Distance.

which of the cars from the scene are going to be evaluated in function of their position and driving lane. For doing so, it is proposed a method to know when two cars are one in front of the other based on the parallelism between the lines that connect each pair of cars and the contour lines of the road.

Following what before mentioned, plotting the lines between cars such as the orange ones, 1 and 2, those can be studied by computing the angle between 1 and 3, 1 and 4, 2 and 3 and 2 and 4. After this, if some of the combinations have an angle below than a given threshold (similar to 30) those lines are parallel and the stopping distance is evaluated. From the illustration below this would be extrapolated to evaluate the case of the line '1' due is the only orange line parallel to some of the blue lines, the 1 parallel with the 3.

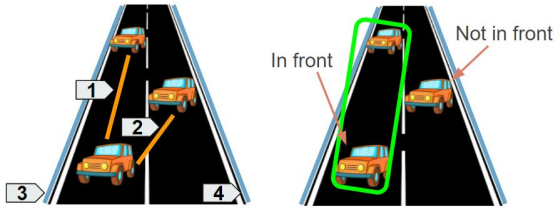

Figure 8. Driving Lanes separation for collision study.

Once the two cars are selected it is studied if the distance between these two cars is shorter than the stopping distance and it is plotted a color line between these 2 cars with the legend of green, orange and red corresponding to the degree of danger.

4. Conclusions and Future Work

In this paper it has been shown a road traffic monitoring system capable of computing the speed of the cars from a fixed camera and evaluate the risk of collision between two co-linear cars on the road lane. A first approach was based on kalman filter after the foreground segmentation was done. To make the system more robust we proposed a method based on detecting the cars with a pre-trained Neural Network over the frames. Furthermore, a collision detector was performed in order to asses when the distance between two cars was smaller than the stopping distance and therefore there is a high risk of collision if the car in front stops suddenly.

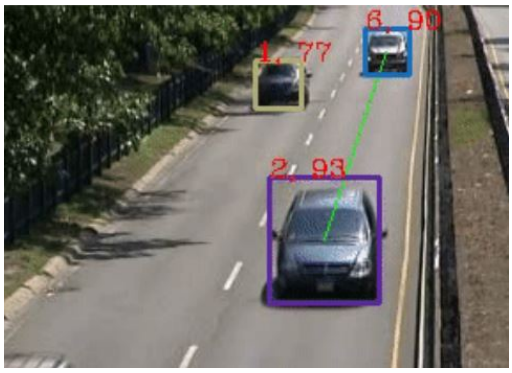

Figure 9. Green line plotted between 2 cars with no risk.

On the other hand, there are some parts of the system that could be studied in the future to make the system more robust. The first one is to train a Neural Network taking into account the time dimension of each car detected, another improvement could be related to computing the velocity of the cars since when doing the projective rectification the results of 2 cars with the same GT velocity can differ depending on the shape of those. Finally it could be improved the decision of the cars to study for the risk of collision detecting each line road over the video based on the white lines of the road.

References

[1] F. G. S. Agustn Yabo, Sebastian I Arroyo and . D. Oliva. Vehicle classification and speed estimation using computer vision techniques. *in XXV Congreso Argentino de Control Automatico*, 2016.

[2] P. Delaigue and A. Eskandarian. A comprehensive vehicle braking model for predictions of stopping distances. pages 1409–1417, 2004.

[3] R. Khilar and S. Chitrakala. A novel method for effective vehicle detection and tracking eliminating occlusion. *Asian Journal of Research in Social Sciences and Humanities, vol. 6, no. cs1, pp. 714727*, 2016.

[4] T. S. Mohamed H Zaki and S. E. Ibrahim. "Demonstrating the viability of automated video analysis for mid-block traffic data collection: a study in the city of edmon". *Tech. Rep.*, 2017.

[5] B. B. More and M. Pathade. Moving object detection, tracking, object counting and speed measurement. *IRJET*, 2017.