VIDEO SURVEILLANCE SYSTEM FOR ROAD TRAFFIC MONITORING

Anna Martí Aguilera, Santi Puch Giner and Xènia Salinas Ventalló

Universitat Autònoma de Barcelona
{anna.martiagu, santiago.puch, xenia.salinas}@e-campus.uab.cat

ABSTRACT

We present an end-to-end road traffic monitoring system that estimates vehicle speeds as well as simple road statistics such as vehicle count, mean speed and traffic density. The system consists of a pipeline of simple processing blocks based on common computer vision techniques that each solve a specific task: background modeling, filtering with morphological operators, Kalman filter tracking, among others. A total of five road traffic video sequences have been used to develop and validate the complete system under different road scenarios, and we show that our system is capable of providing reasonably good results considering the simplicity and accessibility of our approach.

Index Terms— traffic monitoring, road monitoring, background subtraction, morphological operators, Kalman Filter, computer vision

1. INTRODUCTION

With the growing number of vehicles and users, monitoring road and traffic within cities and road networks has became a huge research challenge.
Current techniques for road traffic monitoring rely on sensors which have limited capabilities, and often, both costly and disruptive to install. The use of video surveillance cameras, along with computer vision techniques offers an attractive alternative to current sensors.

2. RELATED WORK

Traffic monitoring is a broad area and has been a topic of interest for a long time. The first camera-based traffic monitoring scheme was introduced by Koller et al. [1], where they used Kalman filter and contour trackers for traffic scene analysis and classification. In [2], they presented an approach for detecting vehicles in urban traffic scenes by means of rule-based reasoning on visual data. The vehicles are detected using background subtraction, binarization, gradients and so on, and classified road condition using rule-based reasoning. The computation involved in detecting vehicles is high and time consuming and hence the method is not suitable for real-time operation.

Since deep learning appeared outperforming handcrafted features, there are many implementations that rely on deep learning to [3] detect and [4] track vehicles in the context of road traffic monitoring.
Our approach consists of a low-cost road traffic monitoring system based on computer vision techniques that provides basic road statistics such as mean speed per lane or traffic density.

3. METHODS

The proposed system, depicted in Figure 1, consists of a sequence of processing blocks based on simple computer vision methods.

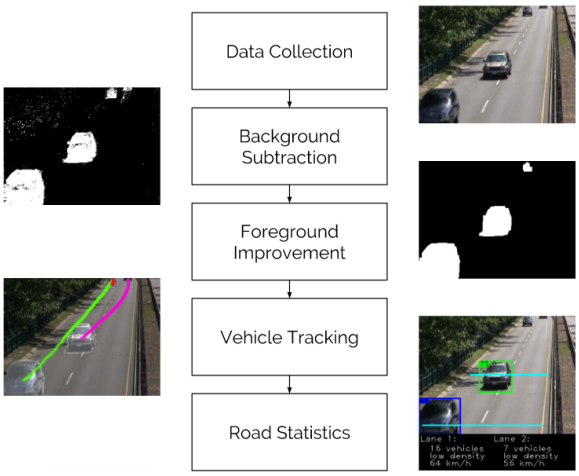

Fig. 1: System overview

The first processing block of the system is Background Subtraction. The aim of this block is to subtract the background of the images in the sequence in order to obtain a simple segmentation of the objects of interest, i.e. the moving vehicles. The general approach in this block is to create a statistical model of the background given a subset of the images of the sequence, and then use this statistical model and simple decision rules to extract a segmentation of the foreground for all the images in the sequence.
The next block in the processing pipeline is Foreground Im-

provement. This block expects a segmentation mask as input and performs a series of filtering operations to obtain a more refined segmentation that represents the objects of interest as blobs and removes the regions that are not of interest for monitoring. We base all the filtering operations in mathematical morphology operators.
The third block, Vehicle Tracking, is designed to detect and track vehicles using the refined segmentation from the previous block as input. Unlike in the two previous blocks, in which the processing was performed for each frame independently, in this block we obtain the path that follows each vehicle along the sequence of images. The main functionality of the Vehicle Tracking block is based on Kalman Filters.
Finally, the fourth block extracts different statistics of the road. Given the tracks of the vehicles estimated in the previous block, it estimates the speed of each vehicle and checks if its i below the speed limit of the road. It also computes some statistics per lane, specifically it computes the total number of vehicles on each lane, the vehicle density at each time step and the global average speed.

3.1. Data Collection

We have collected a variety of traffic sequences in road environments for the development and evaluation of our system that present different characteristics and features. Concretely, we have used the following sequences:

1. Highway [5]: development sequence, standard traffic sequence with stable camera position and good point of view of the road, ideal for the first stages of development where we want to assess if the newly developed algorithm performs reasonably well.

2. Traffic [5]: development sequence, traffic sequence with a great amount of jittering that makes it more challenging than Highway, therefore ideal to test the robustness of the developed algorithm.

3. Alibi [6]: evaluation sequence, standard highway with stable camera position, ideal point of view of the road, and traffic variability. This sequence is the best fit for evaluating the system under almost ideal situations for road traffic monitoring.

4. Street Light [5]: evaluation sequence, normal road but different point of view from the rest of the sequences. Used to test the robustness of the tracking and road statistics blocks against changes in viewpoint.

5. Relaxing Traffic [7]: evaluation sequence, standard highway, good point of view but the only sequence in which both directions of traffic are included, therefore ideal for testing the motion and measurement models of the tracking system.

Figure 2 shows a representing frame of each sequence.

(a) Highway sequence

(b) Traffic sequence

(c) Alibi sequence

(d) Street Light sequence

(e) Relaxing Traffic sequence

Fig. 2: Sequences input

3.2. Background Subtraction

The statistical model of choice for background modeling that we have decided to implement at this stage has been an adaptive Gaussian model defined in RGB space. This model assumes all pixels are i.i.d multivariate Gaussian distributions with diagonal covariance matrices. At the implementation level this means that we have obtained the mean and standard deviation of the values of each RGB channel for the set of images selected to model the background. Mathematically this can expressed as:

$$\bar{\boldsymbol{x}}_{\boldsymbol{i,j}} = \frac{1}{|S_B|} \sum_{k \in S_B} \boldsymbol{x}^{\boldsymbol{k}}_{\boldsymbol{i,j}} \tag{1}$$

$$\boldsymbol{\sigma}^{\boldsymbol{2}}_{\boldsymbol{i,j}} = \frac{1}{|S_B|} \sum_{k \in S_B} (\boldsymbol{x}^{\boldsymbol{k}}_{\boldsymbol{i,j}} - \bar{\boldsymbol{x}}_{\boldsymbol{i,j}})^2 \tag{2}$$

where S_B is the set of indices of the images used to model the background, $\boldsymbol{x}^{\boldsymbol{k}}_{\boldsymbol{i,j}} = (x^R_{i,j}, x^G_{i,j}, x^B_{i,j})^k$ is a 3D vector representing the RGB values of pixel (i, j) of image x^k, $\bar{\boldsymbol{x}}_{\boldsymbol{i,j}}$ and $\boldsymbol{\sigma}^{\boldsymbol{2}}_{\boldsymbol{i,j}}$ are 3D vectors representing the mean and variance of the RGB values of pixel (i, j) for all images in S_B.
Using this statistical model of the background, we have implemented a simple decision rule based on thresholding in order to classify if a pixel belongs to background or foreground. The threshold used is $t_{i,j} = \alpha(\sigma_{i,j} + 2)$, and depends on 2 factors: α, which is a tunable parameter that controls the segmentation and acts as a trade-off between false positives (over-segmentation) and false negatives (under-segmentation); and σ, the variance of the model.

For each pixel, if any of the absolute values of each RGB channel is bigger or equal than the corresponding threshold for that channel, the pixel is classified as foreground, otherwise it is classified as background.

Finally we have included an adaptive component to this statistical model to enable the adaptation of the background model to the background changes over time. This adaptive approach updates the mean and the variance of the background model for each pixel that is classified as background at each time step using a running average controlled by parameter ρ, and leaves the model as it is for the pixels belonging to the foreground. This parameter ρ controls the rate at which new values of the background are incorporated into the model: if $\rho = 0$, the model is left static and does not change over time (not adaptive), and if $\rho = 1$, the model takes the new background value as the model and forgets of the previous value.

3.3. Foreground Improvement

In order to refine the segmentation we have obtained in the previous block, we have built a pipeline of morphological operations, depicted in Figure 3.

Fig. 3: Foreground improvement pipeline

The pipeline starts with a hole filling operation that in most cases fills the vehicles and reduces them to a single connected component. However this introduces a series of artifacts, which motivates the use of an area filtering operator that removes small regions that are not likely to belong to a vehicle. The pipeline consists of two more morphological operators: an opening, that reduces spurious artifacts and false positives that may have appeared because of the previous operations; and a closing, that helps connect the disconnected components of the segmentation that should be connected instead.

All the structuring elements and parameters of these morphological operators have been manually tuned for each sequence taking into account the nature of the sequence and the expected segmentation result.

3.4. Vehicle Tracking

Once each vehicle is represented by a different blob, we want to obtain the path followed by each vehicle. For that we have implemented a multi tracking algorithm based on Kalman Filter [8].

Once the multi tracking tool is initialized, for each frame, the following steps are executed:

1. *Detect vehicles and obtain centroid and bounding box:* From the mask obtained in the previous block we extract the connected components and we consider each connected component a different vehicle. For each of them we extract its bounding box and the corresponding centroid, which is used as the detection for the following steps.

2. *Assign detections to current track using Hungarian algorithm [9]:* Using the Hungarian algorithm, which is a combinatorial optimization algorithm that solves the assignment problem in polynomial time, we find the correspondences between detections and tracks. First we compute the cost of assigning each detection to all the tracks, represented by the distance between the detection and the predicted position of the track. Then each detection is assigned to the track with the minimum cost only if this cost is below the cost of non assignment (50). The cost of non assignment represents the maximum allowed distance between the current position of the vehicle and the previous one. The result of this step is a list of detections and tracks correspondences, a list of unassigned detections and a list of unassigned tracks.

3. *Update assigned tracks using detection:* For all the assigned tracks, we update the state of the Kalman filter (i.e. current position) using the corresponding detection.

$$C = M\Sigma_k M^T + \Sigma_m \tag{3}$$
$$G = \Sigma_k M^T C^{-1} \tag{4}$$
$$x = x + G\left[z - Mx\right] \tag{5}$$
$$\Sigma_k = \Sigma_k - GCG^T \tag{6}$$

, where M is the measurement model, Σ_k is the covariance matrix, Σ_m is the observation noise matrix, x is the current position and z is the observation.

4. *Update unassigned tracks using last state:* For all the unassigned tracks, we update the current position using the last state, which corresponds to the predicted position of the Kalman filter, $z = Mx$.

5. *Delete lost tracks and create new tracks from unassigned detections:* For each unassigned track we verify the age of the track, the number of frames since the first detection. If the track has an age below 15 frames, we verify the visibility percentage, the number of frames in which it has been detected with respect the total ones. If the visibility is below the 60% the track is deleted. Otherwise, if the track has an age over 15 frames, we verify the number of consecutive frames in which this track has not been seen. If it has not been seen for more

than 10 frames, the track is deleted. For the non assigned detections we create new tracks initializing them with the corresponding detection.

6. *Predict new location of tracks:* Finally, we predict the following location of each track using the Kalman filter.

$$x = Dx \tag{7}$$

$$\Sigma_k = D\Sigma_k D^T + \Sigma_d \tag{8}$$

, where D is the constant velocity motion model and Σ_d is the process noise matrix.

3.5. Road Statistics

The tracks extracted in the previous block provide information about the current location of each vehicle at each time step. With that we can estimate the instantaneous velocity of the car, according to fundamental physics, as $v = \dot{x} = dx/dt$. If we assume that the time difference between successive frames represents an infinitesimal time change dt, and we assume that the difference between the positions of each vehicle in this time difference is also infinitesimal, we can estimate the velocity as:

$$\boldsymbol{v_t} = \frac{\Delta \boldsymbol{x_t}}{\Delta t} = \frac{\boldsymbol{x_t} - \boldsymbol{x_{t-1}}}{t - (t-1)} \tag{9}$$

where $\boldsymbol{x_t}$ is the 2D position vector (x_i, x_j) of the vehicle at time instant t and $\boldsymbol{v^t}$ is the 2D velocity vector of the vehicle (v_i, v_j) at time instant t. The scalar velocity of the vehicle can be obtained computing the Euclidean distance of the velocity vector:

$$v_t = \sqrt{\langle v_t, v_t \rangle} = \frac{\sqrt{(x_{i,t} - x_{i,t-1})^2 + (x_{j,t} - x_{j,t-1})^2}}{t - (t-1)} \tag{10}$$

However this assumes that the position of the vehicles is known in the real world and expressed in meters, but instead the position is estimated in the image plane and expressed in pixels. In order to estimate the real velocity in *km/h* we need to convert the velocity estimate from pixels to meters. The main problem with finding this pixels to meters correspondence is the projective distortion of the scene, i.e. the ratio is not constant throughout the road because of the camera viewpoint and its perspective. To address this problem we have manually defined a Region of Interest (ROI) in which the pixels to meters ratio can be assumed to be constant, and we have used known distances of specific elements in the road (e.g. median strip gaps) in order to establish this value for each sequence. An example of the speed ROI defined in Highway sequence is shown in Figure 4b.

We have also manually marked the boundaries of each lane for each sequence and used that information to determine the number of vehicles that have passed by a specific lane, as well as counting the current number of vehicles in that lane and estimating if that number represents low or high density traffic. Also, by having a correspondence between vehicles and lanes we have been able to accumulate the vehicle speeds over time and compute the mean speed for each lane.

Sequence	Lane 1	Lane 2	Lane 3	Lane 4
Highway	21 (19)	11 (13)	–	–
Traffic	45 (13)	25 (11)	–	–
Alibi	41 (35)	43 (34)	23 (23)	–
Street Light	25 (18)	29 (36)	68 (49)	55 (41)
Relaxing Traffic	37 (22)	34 (22)	15 (12)	14 (15)

Table 1: Vehicle counting results (real number of vehicles)

(a) Speed ROI (b) Lane ROIs

Fig. 4: Regions of Interest defined on Highway sequence

4. EVALUATION

Taking into account that the used sequences do not have ground truth for the road statistics that we have computed, we have done a visual inspection to evaluate the obtained results. For the vehicles speed we have seen that there are a lot of variances for the same vehicle while visually the speed is more or less constant. Nevertheless the computed speed are inside the expected range, we don't get extremely high or low speeds.
For the vehicle count the results are showed in Table 1. In some lanes we count a high number of false vehicles, this is caused by the errors in the detection phase.

5. CONCLUSIONS AND FUTURE WORK

The overall result of this project shows that through a convenient procedure, the cars can be properly tracked by using video surveillance cameras. However, due to the difficulty of mapping the road to a known surface has introduced noisy statistics to our proposed system.
In order to improve the reliability of the results, we propose to use deep learning techniques for obtaining a noiseless tracking of the vehicles, coupled with an automatic detection of the road lanes, and so improve the pixel to meter ratio computation.

6. REFERENCES

[1] Dieter Koller, Joseph Weber, T Huang, J Malik, G Ogasawara, B Rao, and S Russell, *Towards robust automatic traffic scene analysis in real-time*, vol. 1, 11 1994.

[2] R. Cucchiara, M. Piccardi, and P. Mello, "Image analysis and rule-based reasoning for a traffic monitoring system," *Trans. Intell. Transport. Sys.*, vol. 1, no. 2, pp. 119–130, June 2000.

[3] Jorge E Espinosa, Sergio A Velastin, and John W Branch, "Vehicle detection using alex net and faster r-cnn deep learning models: A comparative study," in *International Visual Informatics Conference*. Springer, 2017, pp. 3–15.

[4] Yingfeng Cai, Hai Wang, Xiao-qiang Sun, and Long Chen, "Visual vehicle tracking based on deep representation and semisupervised learning," *Journal of Sensors*, vol. 2017, 2017.

[5] F. Porikli J. Konrad N. Goyette, P.-M. Jodoin and P. Ishwar, "changedetection.net: A new change detection benchmark dataset," 2012.

[6] Supercircuits, "Alibi ali-ipu3030rv ip camera highway surveillance," 2014.

[7] Vlad Kiraly, "Relaxing highway traffic," 2017.

[8] R. E. Kalman, "A new approach to linear filtering and prediction problems," 1960.

[9] H.W. Kuhn, "The hungarian method for the assignment problem," 1955.