TRAFFIC ROAD MONITORING

Idoia Ruiz, Roque Rodríguez, Lidia Talavera, Onofre Martorell

Universitat Politècnica de Catalunya

ABSTRACT

Traffic accidents are one of the most common causes of death in the world. However road monitoring has been proved to reduce these accidents. To do that, a system based on computer vision techniques is implemented. Several processes are performed on traffic sequences so the different vehicles are detected, tracked and measured on terms of speed. For detection, the background is modeled as a gaussian distribution with and without adaptation and also with a mixture of gaussian approach. For the tracking, kalman filters and mean shifts algorithms are used. Some stabilization techniques based on optical flow are also performed. Finally, different evaluation methods are used to validate the system with ground truth annotations.

Index Terms— Traffic monitoring, computer vision, car tracking, kalman filter, mean shift, video stabilization, speed estimation.

1. MOTIVATION

In 2016 there were 1.160 dead on the Spanish roads, 29 more than in 2015 [1], and most of them due to speeding. This data, along with the great advances that are emerging in the field of computer vision justifies the creation of roads monitoring systems in order to ensure greater control and, therefore, greater road safety.

2. STABILIZATION

Cameras for traffic monitoring are subjected to the environment conditions in which they are placed. For instance, weathering can introduce camera motion. Therefore, the first point considered in our pipeline is to stabilize the video, so that the background estimation explained in section 3 is not affected by the camera motion. To compensate the camera motion, first it must be estimated, computing the optical flow.

2.1. Block matching optical flow estimation

The optical flow consists on computing the movement between two images or video frames. The most simple algorithm to compute optical flow is called block matching and consists on dividing the image in a grid of blocks of size N, and for each block search for another block in the surrounding area which minimizes:

$$\sum_{r \in R} ||DFD(\vec{r}, \vec{D}(\vec{r}))|| \tag{1}$$

where the DFD is the displaced frame difference.

More specifically, the area of search is the square of sides $(2P + N) \times (2P + N)$ centered at the center of the $N \times N$ square used to do the search.

At this point, a simple approach to perform stabilization is to compute the mean optical flow (u_x, u_y) for the full frame, that is the mean optical flow for all the blocks, and shift the next frame $-(u_x, u_y)$, where u_x and u_y are the components of the mean optical flow.

However, this method is computationally expensive and it can be affected by the foreground motion. Thus, a more efficient method is proposed in section 2.2

2.2. ROI tracking stabilization

This method, which corresponds to the Matlab example [2] for video stabilization, is based on tracking a target region that belongs to the background.

For each frame, the selected target is searched in the next frame, in a small area around it. Since the search area is limited, the method is less computationally expensive. The translation of the Region Of Interest (ROI) between both frames, is used to translate the second frame in the opposite direction so that the motion is compensated.

However, the drawback of this method is that we must specify, using our knowledge of the video, the coordinates of the target region. This makes this method difficult to generalize.

3. BACKGROUND ESTIMATION

Once we have stabilized the video, the next step is to determine which pixels of each frame of the video corresponds to whether background or foreground. In this section we will present three approaches to solve this problem based on the modeling of the background points with Gaussian distributions.

3.1. Gaussian modeling

In this approach, each pixel of the background in the scene is modeled as a random variable: its mean μ represents its actual value and its variations with respect to the mean value is due to the noise introduced by the camera sensor. This camera sensors are usually assumed to be independent and identical distributed random variables with Gaussian distribution $N(0, \sigma^2)$.Therefore, a pixel will be part of the foreground if its value its outside an interval centered on the mean of the background points. More specifically,

$$p \in \text{foreground if } I(p) \notin (\mu - \alpha(\sigma + 2), \mu + \alpha(\sigma + 2)),$$

where p is the incoming pixel, $I(p)$ its values, μ and σ the parameters of the background and α a parameter to be set. Otherwise, the point is classified as background.

In order to obtain a background model, some frames of the video are taken. More specifically, for each pixel of the image, a mean and a standard deviation is computed based on the value of the frames of the training stage.

3.2. Adaptive Gaussian

The model of this section is an improvement of the previous model and is based in the following statement: if a pixel is classified as background its value can be used to improve the model of the background. With that, if the pixel p is classified as background, the mean and the variance of the background model are updated as

$$\begin{cases} \mu = \rho \cdot I(p) + (1 - \rho)\mu \\ \sigma^2 = \rho(I(p) - \mu)^2 + (1 - \rho)\sigma^2 \end{cases}$$

where ρ is a parameter to be set.

3.3. Stauffer & Grimson

The algorithm proposed by Stauffer and Grimson [3] models the distribution of colors of each backgrounds pixel as a Mixture of Gaussians (GMM). We decided to use the implementation provided by MATLAB, adjusting the appropriate parameters for our system. In this case, for every pixel several gaussians are used to model the different color appearances of the scene.

4. FOREGROUND COMPENSATION

Once we have determined the objects of the screen, we will try to improve the detection in order to make the algorithm more robust. The way to improve it will we through postprocess techniques: hole filling, area filtering and morphological operators.

The hole filling consists on applying some postprocessing which closes those small holes on the connected components within each frame. Matlab implementation is used for this matter. On the other hand, the area filtering consists on eliminating all those connected regions that have an area lower than a certain threshold P.

In some cases, not the whole vehicles were detected during the background estimation. Since the morphological operators are strongly dependent on the images, we decided to apply a different set of morphological operators to each of the sequence, according to what we want to eliminate.

• *Fall sequence*. For this sequence, it is hard to remove the noise without removing part of the detected objects and vice versa. Therefore, at first, an opening (with diamond size 3) is performed and then a closing (with diamond size 7) is applied. An *imfill* operation is done between them so the holes are filled for sure.

• *Highway sequence*. It was clear that the segmentation presented holes within the detected objects and that some noise remained. Therefore, at first the remained holes were closed (by a closing with a diamond of 6), then the noise was removed with an opening (with diamond sized 2).

• *Traffic sequence*. This sequence was perturbed by the presence of noise on a line which cross the image through a 45 angle.Thus, a first opening was performed with a perpendicular line. Then, in order to fill the holes of the car, a closing was performed with a big diamond (sized 20).

5. REGION TRACKING

The last step of our system is based on car tracking which can be divided into two parts [4]: the first part is detecting moving objects in each frame and the second part is based on associating the detections corresponding to the same object over time.

To carry out this, it is necessary to calculate the object position and speed for each instant. The main difficulties in reliable tracking are: handle appearance changes caused by image noise, illumination changes, nonrigid motion; deal with occlusions and cluttered background and the maintenance of objects identities when multiple objects merge into a single detection.Although there are several approaches, we have decide to use Kalman filter and Mean Shift methods to accomplish this function.

5.1. Kalman filter

The Kalman filter [5] is a recursive method used to track linear dynamic models that in that each updated infers the next state of an object knowing the current one using Gaussian distributed errors. To use the Kalman filter, it is supposed the object must be moving at constant velocity or constant acceleration. The Kalman filter algorithm involves two steps:

1. **Prediction**: uses previous states to predict the current state of the object.

2. **Correction**: locates the object (which is in the neighborhood point predicted in the previous stage) and uses its real position (measurement) to carry out the state correction.

In [5], we can see examples of Kalman Filter performance. We use the implementation provided by MATLAB.

5.2. Mean shift

Mean shift method is based on finding the object in the next frame using the color distribution of the object (histogram). Given a new zone, compare both histograms and weight the pixels with the probability to belong to the tracked object. Then, compute the centroid with weights and move zone (mean shift). We decided to use the implementation provided by Team 1 of 2016 curse [6] , but we have several problems as there are many parameters that need to be tuned.

5.3. Speed estimation

The speed of an object is the ratio between the traveled distance and the time within this distance is gone through. Therefore, the speed is computed as quotient the pixels traveled by the centroid of the objects and the number of frames. However, the real distance is usually hard to compute because of the perspective on the videos and the differences among pixels and meters. Our main assumption is to create a coefficient which establishes the relation of our measure and kilometers per hour. The main drawback is that this parameter is not changing along the sequence so the perspective is not considered and that these parameters must be tuned depending on the sequence. In terms of visualization, the tracking box will be red if some speed limit is over reached.

6. RESULTS AND EVALUATION

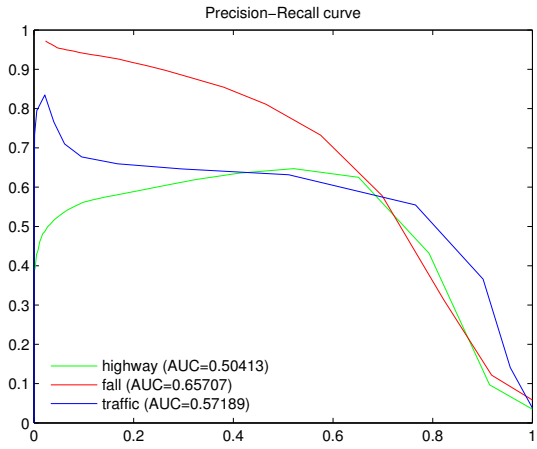
6.1. Background estimation and morphological processing

To evaluate the foreground segmentation results, we use video test sequences from the Change Detection Video Dataset, which have ground truth of the motion areas.

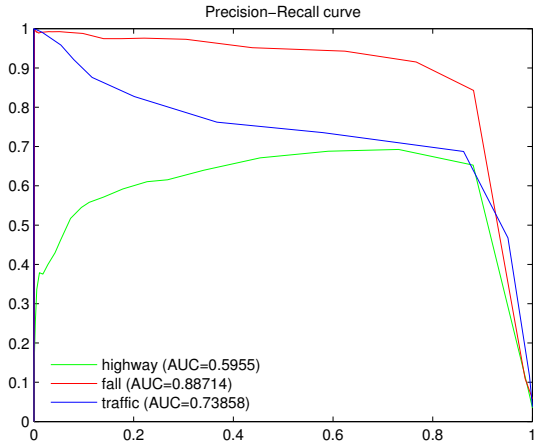
Figure 1 shows the Precision - Recall curve and the Area Under The Curve (AUC) value, for the foreground segmentation on *fall*, *traffic* and *highway* sequences before and after applying the morphological processing described in section 4. There is a gain in the AUC value of 18.12% for the *highway* sequence, 35.01% for the *fall* sequence and 29.15% for the *traffic* sequence.

6.2. Stabilization

Stabilization can avoid detecting background as foreground, as a consequence of the camera motion. Figure 2 shows this



(a) Without morphological processing



(b) With morphological processing

Fig. 1. Precision - Recall curve for foreground segmentation (a) before and (b) after applying morphological processing. Using a range for α from 0 to 30 with step 1 and ρ (*highway*) = 0.2, ρ (*fall*) = 0.18 and ρ (*traffic*) = 0.1.

(a) Non stabilized (b) Stabilized

Fig. 2. Example of avoiding a false detection with stabilization, due to the camera motion.

(a) *traffic* sequence (b) Own video, low light

Fig. 3. Tracking examples using Stauffer & Grimson for foreground segmentation and Kalman filtering.

effect on an own video which suffers jittering. This comparison also shows that the speed is better estimated for the stabilized case, since at the road the video was filmed, there is no way the cars actually went 79 km/h, which is the speed predicted in the non-stabilized video. However, 17 km/h is likely to be close to the real speed.

6.3. Speed estimation

To evaluate the speed estimation, we measured the real speed of a vehicle by measuring the time it takes to cover a certain measured distance, assuming the speed is constant in this road section.

Our measured speed for a certain car is 17 ± 3.61 km/h while the estimated speed by our system is ~ 20 km/h. Therefore, our system predicts in the error range, although more measurements should have been carried out to obtain a more exhaustive evaluation.

6.4. Region tracking

The Kalman tracking algorithm described in section 5.1 accurately detect vehicles' trajectories on the test sequences used for evaluating in section 6.1. Although in custom videos it has some limitations to detect small objects like vehicles from a certain distance, or bikes.

However, the Mean shift method, explained in section 5.2 does not work properly. It might be because the parameters are not the optimal ones.

Figure 3 shows two examples of Kalman tracking. On example (a), the speed display turns red above a certain chosen speed, and the upper left number indicates the number of vehicles per minute. Example (b) shows how our system also can work in low light conditions.

The video results of the tracking algorithms are published in our workshop page [7].

7. CONCLUSIONS

Our proposed system, is able to identify and count vehicles while tracking their trajectories and estimating their speed. It also works on low light conditions.

However, since the parameters, as well as the morphological processing, are sequence dependent, makes our system to have a lack of generalization that should be improved in further work.

There also are limitations when detecting small objects, for instance vehicles from a certain distance, or bikes.

8. REFERENCES

[1] DGT, "Accident rate in Spain," `http://www.dgt.es/es/`.

[2] MathWorks, "Video Stabilization," `http://es.mathworks.com/help/vision/examples/video-stabilization.html`.

[3] Chris Stauffer and W Eric L Grimson, "Adaptive background mixture models for real-time tracking," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.* IEEE, 1999, vol. 2, pp. 246–252.

[4] MathWorks, "Multiple Object Tracking," `https://nl.mathworks.com/help/vision/examples/motion-based-multiple-object-tracking.html`.

[5] Rudolph Emil Kalman et al., "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[6] Team1_2016, "Mean Shift algorithm," `https://github.com/mcv-m4-video/mcv-m4-2016-Team1/blob/master/week5/functions/matlab_mean_shift_avi.m`.

[7] I. Ruiz, R. Rodríguez, L. Talavera, and O. Martorell, "Video Surveillance For Road Traffic Monitoring, Workshop," `https://mcv-m4-video.github.io/mcv-m4-2017-team5`.