Robust Vehicle Detection for Tracking in Highway Surveillance Videos using Unsupervised Learning

Birgi Tamersoy and J.K. Aggarwal
Computer and Vision Research Center
The University of Texas at Austin
Austin, Texas 78712-0240, U.S.A.
[birgi,aggarwaljk]@mail.utexas.edu

Abstract

This paper presents a novel approach to vehicle detection in highway surveillance videos. This method incorporates well-studied computer vision and machine learning techniques to form an unsupervised system, where vehicles are automatically "learned" from video sequences. First an enhanced adaptive background mixture model is used to identify positive and negative examples. Then a classifier is trained with these examples. In the detection phase, both background subtraction and the classifier are used to achieve very accurate results while not compromising efficiency. We tested our method with very low-, medium- and high-quality, crowded and very crowded surveillance videos and got detection accuracies ranging between 90% to 96%.

1. Introduction

As the time spent on daily commutes continues to increase, accurate and efficient traffic monitoring systems are being installed in order to improve the quality of these commutes. For the last decade, there has been a significant amount of research on both highway monitoring systems and Intelligent Transportation Systems (ITS). In this paper the focus is on vehicle detection in highway surveillance videos which is a crucial part of the traffic monitoring systems.

Notable work in this area can be traced back to the early 1990s, where Koller *et al*. [8] proposed a deformable contour based tracking algorithm. They introduced an "occlusion reasoning" step, where the partial occlusions are resolved by the motion prediction information. In the case of crowded scenes, where the vehicles enter the scene partially occluded, this tracking algorithm will have significant errors.

In order to address the partial occlusion problem, Coifman and Beymer [4] proposed a feature based tracking algorithm. They detected the "corner" features and then grouped them according to a "common motion constraint". Similarly, Jun *et al*. [7] incorporated feature based segmentation to background subtraction based detection. They over-segmented the "irregular blobs" and then cluster the pieces according to the common motion constraint of the extracted features. They reduced the computational costs by limiting the feature analysis only to "irregular blobs". However, both algorithms depend solely on the accuracy of feature detection and matching, which makes them error prone in noisy, low resolution videos. Common motion constraint is not applicable in very crowded scenes, where the vehicles are forced to move at similar speeds.

Using 2D and 3D models for detection has also been explored by several authors. Song and Nevatia [11] incorporated a 3D model-based detection approach with background subtraction. They created 2D templates from the 3D vehicle models and used them to generate multiple hypotheses for a given foreground mask. This approach only makes use of the template contours, so like the feature-based approaches, its performance on noisy, low resolution and crowded scenes is uncertain.

Papageorgiou and Poggio [10] approached a more general problem, object detection, from a machine learning perspective. They presented a trainable system which can be used for detection of several different object classes, including vehicles in a highway surveillance video. Their training is viewpoint-dependent and requires different sets of "labeled" positive and negative examples for each particular viewpoint. Forming these training sets may be costly.

This paper presents a vehicle detection algorithm which addresses most of the aforementioned issues related to low quality, crowded surveillance videos. In Section 2 the proposed approach is explained in detail. In Section 3 experimental results are presented. Discussion and future work is included in Section 4.

978-0-7695-3718-4/09 $25.00 © 2009 IEEE
DOI 10.1109/AVSS.2009.57

2. Methodology

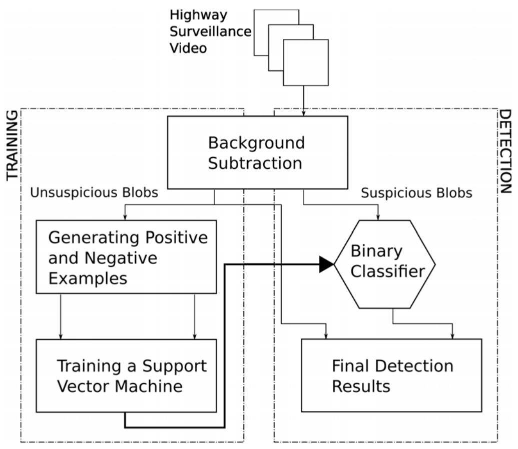


Figure 1. System overview.

Figure 1 illustrates the overview of the proposed system. In the training phase, some positive vehicle examples are identified directly from the background subtraction. This is followed by locating the possible negative examples. Then these two sets of image patches are used to train a SVM. The result is a video-specific binary classifier, which is used to further process the blobs that are suspected to have more than one vehicle. The details of each step is explained in the following sections.

2.1. Road Orientation and Entrance Regions

This is an optional pre-processing step where the entrance region of a traffic surveillance video is determined without any supervision. In this step, the Canny edge detection algorithm [2] followed by a Hough line transform [6] is used to obtain the dominant lines of a frame. The entrance region is determined using this road orientation and a predefined upper limit as illustrated in Figure 2. This is a robust method for straight roads since the number of cars in the frame do not affect the final results.

Identifying this region both decreases the computational costs and improves the detection accuracy. Since the corresponding region in real world is the closest region to the surveillance camera, vehicles in this region appear to have relatively more space in between them.

2.2. Background Subtraction

For background subtraction, an enhanced version [7] of the adaptive background mixture model [12] is used. The adaptive background mixture model uses a mixture of Gaussians (MoG) to model the background of a scene. In the enhanced MoG method, a 3D (time being the 3^{rd} dimension) connected component analysis is applied on the foreground masks obtained from the MoG method. This 3D connected component analysis incorporates both spatial and temporal information in the background model, and hence results in extremely good foreground masks. Additionally, the background model "adapts" itself to the changing lighting and weather conditions. Figure 3 illustrates a sample input and the foreground mask obtained from that input.

2.3. SVM Training

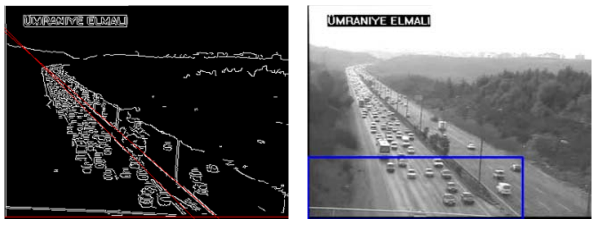
SVM training has two major steps: 1) forming positive and negative training sets and 2) feature extraction. In many applications the former step requires manual labeling which can be a costly operation. In this section we will explain how to obtain these samples automatically from the video sequences *without* using any human supervision.

2.3.1 Forming the Training Sets

Figure 3 illustrates that foreground masks have blobs of varying sizes. One key observation is that highway surveillance videos are usually captured from a distance, so vehicles in the same video have similar sizes (exceptions are long buses and trucks). Hence, the area of a blob which contains a single vehicle is significantly different from the area of a blob that contains two or more vehicles.

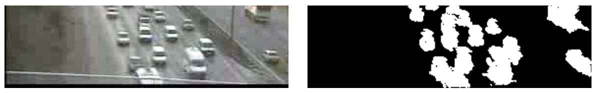
Figure 4(a) shows the blob areas seen in four different datasets. As the figure illustrates, it is relatively easy to determine where the blob areas show a steeper change. Therefore the area threshold (τ_{area}) can be selected from the range where this change is not significant. Additionally, since this is just the training phase, a "safe" threshold may be chosen, which would *not* capture all existing single vehicle blobs.

With these observations, patches corresponding to "small" blobs are stored as positive examples. Figures



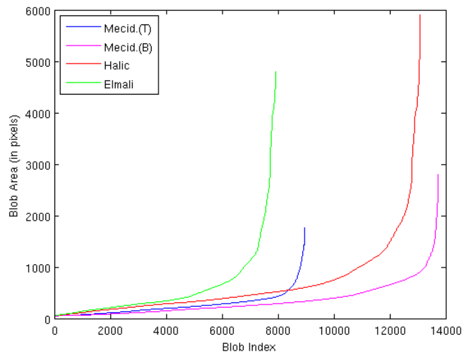
(a) Dominant lines (in red). (b) Entrance region.

Figure 2. Determining the entrance region (Best viewed in color).

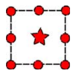


(a) Input frame. (b) Foreground mask.

Figure 3. Results of the enhanced adaptive background mixture model.

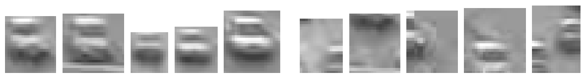


(a)

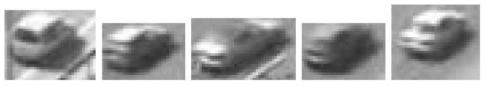


(b)

Figure 4. (a) Blob Areas. (b) Relative positive and negative example centers.



(a) Elmali positive examples. (b) Elmali negative examples.



(c) Halic positive examples.

(d) Halic negative examples.

Figure 5. Positive and negative training examples from Elmali and Halic datasets (see Section 3).

5(a) and 5(c) illustrate some automatically captured positive training examples from the Elmali and Halic datasets explained in Section 3. Once a positive example is located, corners and edge midpoints on the patch boundary are taken as negative example centers as illustrated in Figure 4(b). The generated negative examples are "hard" negatives which is better for the training of the classifier (illustrated in Figures 5(b) and 5(d)). With these hard negatives the classifier is forced to learn structured clutter, rather than just plain background, which is beneficial for accurate localization of vehicles in blobs that are suspected to have more than one vehicle.

2.3.2 Feature Extraction

As Figure 5 illustrates, color and shape of the patches are not discriminative enough to separate the positive examples from the negative ones. On the other hand, positive and negative examples have significantly different line distributions and orientations. For this reason a histogram of oriented gradients (HOG) based method, similar to [9] and [5], is used to extract features from the patches.

First each patch is divided into four cells by a 2×2 grid. Then for each cell, an 8-bin HOG is computed. These histograms are normalized with respect to the number of pixels in each cell. And finally, these 8-bin histograms are concatenated and a SVM is trained by the resulting 32-dimensional feature vectors.

This method has two advantages: 1) Since the training examples are automatically generated from the background subtraction, they have various sizes. This method does not require any resizing or alignment, which is very important since these operations would not be very accurate in this uncontrolled setting. 2) By using a 2×2 grid, some spatial information is also captured.

2.4. Vehicle Detection

Vehicle detection is done in two steps. First, the background subtraction explained in Section 2.2 is used to get the foreground masks. As mentioned in Section 2.3.1, blobs with relatively small areas are assumed to contain only a single vehicle. Hence, these vehicles can be detected directly by using the foreground masks. Contrary to this, some blobs are suspected to have two or more vehicles, so they need to be further processed.

This is done with the help of the binary classifier generated in the training step (see Section 2.3). Given a patch descriptor, this classifier decides if the corresponding patch contains a vehicle or not.

In order to detect the vehicles in these suspicious blobs, a sliding window approach is used. The window size is determined by the median width and height of the positive training examples of Section 2.3. At each pixel location along the suspicious blob, a patch centered at that location is extracted. Then the 32-dimensional feature vector for that patch is computed and fed into the binary classifier. This process is illustrated in Figure 6. The leftmost image shows a blob which is suspected to have more than one vehicle. The middle image is the corresponding patch from the frame, and the rightmost image is the output of the pixel-wise sliding binary classifier. Each pixel in this binary output indicates whether there is a vehicle centered at that pixel or not.

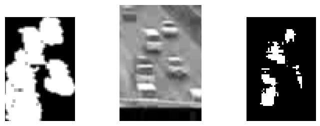
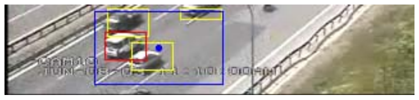


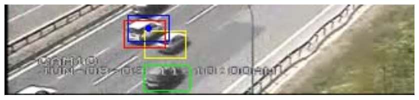
Figure 6. Processing a suspicious blob.

As Figure 6 suggests, results of the binary classifier should further be processed in order to eliminate some false positives. This is handled by the constraint that the distance

(a) Initializing the tracking.



(b) The first match (without position prediction).



(c) Following matches (with position prediction).

Figure 7. Tracking a vehicle. Colors represent; red: best match, yellow: other match candidates, blue: predicted position and green: other detections (Best viewed in color).

between two detected vehicle centers cannot be less than a particular threshold (τ_{dist}). The system is relatively more sensitive to this threshold since it directly affects the number of false positives and false negatives. However, a good threshold can still be determined using the window size and the average single vehicle blob area (see Section 2.3.1). So, for every suspicious blob, results of the binary classifier are sorted in decreasing order of areas. Then each result center is compared with all other result centers with larger areas. If the Euclidean distance is less than the threshold, this result is assumed to be a false positive and is eliminated.

2.5. Vehicle Tracking

In every frame vehicles are detected independently. Tracking in this setting is simply linking these independent detections in consecutive frames. For every vehicle, the position in the next frame is predicted using a constant acceleration dynamic model [6] as follows:

$$\mathbf{p}_i = \mathbf{p}_{i-1} + (\triangle t)\mathbf{v}_{i-1} \tag{1}$$

$$\mathbf{v}_i = \mathbf{v}_{i-1} + (\triangle t)\mathbf{a}_{i-1} \tag{2}$$

$$\mathbf{a}_i = \mathbf{a}_{i-1} \tag{3}$$

where vector $\mathbf{p}$ gives the position, vector $\mathbf{v}$ the velocity and vector $\mathbf{a}$ the acceleration. In the next frame, all the detections that overlap with this predicted position are considered to be matching candidates. The best match is determined by the correlation coefficient (R_{coeff}) between the detection from the previous frame (T) and a candidate (I_i), as in Equation 4. When computing the correlation coefficient, occlusions are not a problem since between two consecutive frames, the amount of occlusion is not expected to alter significantly.

Dataset	Training		Test		Accuracy
	+	-	+	-	
Mecid. (T)	1001	1002	837	993	91.20%
Mecid. (B)	200	1501	135	1180	95.13%
Halic	700	7700	317	436	98.008%
Elmali	1100	11500	403	924	87.18%

Table 1. Training statistics. "+" and "-" are the number of positive and negative examples, respectively.

$$R_{coeff} = \sum_{x,y} [T'(x,y) I'_i(x,y)] \tag{4}$$

where

$$T'(x,y) = T(x,y) - \frac{\sum_{x',y'} T(x',y')}{wh} \tag{5}$$

$$I'_i(x,y) = I_i(x,y) - \frac{\sum_{x',y'} I_i(x',y')}{wh} \tag{6}$$

and w, h are the width and the height of the detection window. Tracking of a vehicle is illustrated in Figure 7.

3. Experiments and Results

Experiments are performed on three different traffic surveillance videos taken in Istanbul, Turkey. The names Elmali, Halic and Mecidiyekoy correspond to the locations that the videos were taken. Elmali is a very low-quality video with 384 × 288 resolution. Halic and Mecidiyekoy are middle- and high-quality videos with 320 × 240 resolutions each. For Elmali and Halic, the "region of interests" are automatically determined. For Mecidiyekoy, the

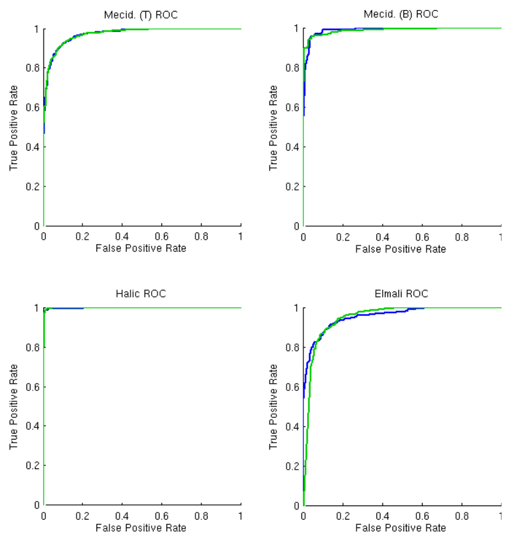


Figure 8. ROC curves for the classifiers.

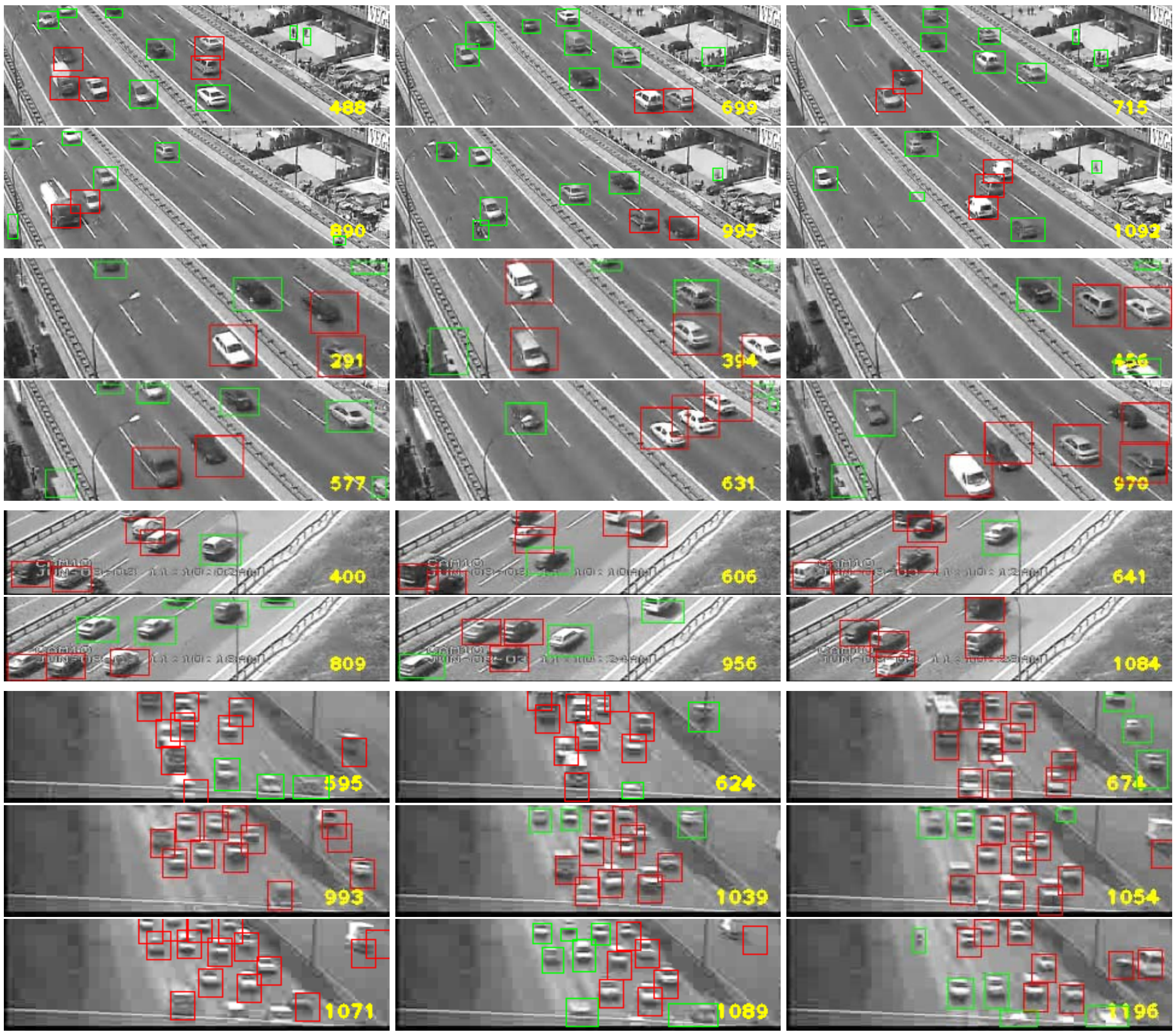


Figure 9. Good detection results. Colors represent; green: detection directly from background subtraction, red: detection by processing suspicious blobs, and blue: the blob was suspicious, but sliding window did not provide further information (Best viewed in color).

region of interest did not contain any suspicious blobs (due to viewpoint, quality and amount of clutter). Hence no region of interest is determined for Mecidiyekoy, and instead the frames are directly divided into two regions, top (T) and bottom (B). Experiments are done independently on these two regions.

For background subtraction, the MATLAB code provided by [7] is used. SVM related operations are handled by using the publicly available SVM library, LIBSVM [3]. The rest of the system is implemented in C++, using the open-source computer vision library, OpenCV [1].

A linear kernel is used for SVM training. Table 1 shows the number of positive and negative examples used in training and testing. In each experiment, the system may extract hundreds of positive and negative examples by processing less than a minute of the input video. For Mecidiyekoy, only 600 frames ($\sim$ 20 seconds) were enough to generate the required number of examples. Classification accuracies ranged from 87% to an impressive 98% for the Halic dataset. The ROC curves of the four videos are illustrated in Figure 8.

Detection results are presented in Table 2 and Figure 9. From each dataset, 100 frames (with 10 frame periods to avoid including very similar frames) are manually inspected. The total number of vehicles, detections, false positives and false negatives are counted. As Table 2 shows, the proposed system has very few false positives and slightly more false negatives. Hence, extremely good detection accuracies, ranging between 90% to 96%, could be achieved in these three significantly different and challenging high-

Dataset	Frames	Vehicles	Detected Vehicles	Accuracy	False Positives	False Negatives
Mecid. (T)	100	659	633	96.05%	24	26
Mecid. (B)	100	335	313	93.43%	18	22
Halic	100	487	442	90.75%	33	45
Elmali	100	1197	1106	92.39%	85	91

Table 2. Detection results.

way surveillance videos.

It is worthwhile to emphasize the results of the Mecidiyekoy experiments. As Figure 9 shows (top 4 rows), when the video is divided into two regions, very accurate detection results could be achieved even though the vehicles in two regions look different. This is because we had trained one classifier for each region independent from the other region.

Figure 10 shows some shortcomings of our detection algorithm. The major shortcoming is the poor detection of large vehicles. This can be explained by the following; these vehicles look very different, training is done only with average-sized vehicles and the detection window size is too small for these vehicles. Other minor shortcomings are shadow-related localization problems and some false positives which could not be eliminated by the method explained in Section 2.4.

4. Discussion

We have presented a novel approach for vehicle detection in highway surveillance videos. The main contribution is the underlying unsupervised learning framework where the vehicles in a video are directly learned from the video without any prior knowledge or supervision.

With this framework, we achieved excellent detection results in three videos that are significantly different in terms of quality, viewpoint and clutter.

We are planning to extend this work in two ways:

The proposed method should be tested in a real world system and the accuracy should be evaluated in different weather and day/night conditions. We expect the system accuracy to be robust to these conditions to some extent since different illumination conditions do not affect the extracted features dramatically. However, the highway lighting conditions in the night may introduce a significant challenge to the system and may require re-training of the classifiers.

As mentioned in Section 3, separating a video into multiple regions and processing these regions independently may increase the overall accuracy. This approach may be applied to complex highway surveillance videos.

Acknowledgements. We would like to thank Dr. Muhittin Gokmen for providing us with the videos.

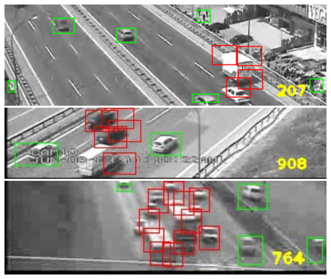


Figure 10. Bad detection results.

References

[1] *Open Source Computer Vision Library*, 2000. Software available at `http://sourceforge.net/projects/opencvlibrary/`.

[2] J. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[3] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/∼cjlin/libsvm`.

[4] B. Coifman and D. Beymer. A real-time computer vision system for vehicle tracking and traffic surveillance. *Transportation Res.: Part C*, 6(4):271–288, 1998.

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. IEEE Conf. on Computer Vision and Pattern Recognition, 2005.

[6] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach.* Prentice Hall, 2003.

[7] G. Jun, J. K. Aggarwal, and M. Gokmen. Tracking and segmentation of highway vehicles in cluttered and crowded scenes. IEEE Workshops on Applications of Computer Vision, 2008.

[8] D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. Proc. European Conf. on Computer Vision, 1994.

[9] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. of Computer Vision*, 60:91–110, 2004.

[10] C. Papageorgiou and T. Poggio. A trainable system for object detection. *Int. J. of Computer Vision*, 38:15–33, 2000.

[11] X. Song and R. Nevatia. A model-based vehicle segmentation method for tracking. IEEE Proceedings of the 10th Int. Conf. on Computer Vision, 2005.

[12] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. IEEE Conf. on Computer Vision and Pattern Recognition, 1999.