Near Real-Time Vehicle Detection and Tracking in Highways

Ebrahim Soroush[1], Ali Mirzaei[2], Shiva Kamkar[3]
1-Amirkabir University of Technology (e.soroush@aut.ac.ir)
2-Amirkabir University of Technology (ali_mirzaei@aut.ac.ir)
3-Amirkabir University of Technology (sh.kamkar@aut.ac.ir)

Abstract

In this paper we present an approach for detection and tracking of vehicles in highways. The Aggregated Channel Features (ACF) is used to detect vehicles and the Kalman filter is employed to track the detected objects. The proposed scheme enjoys high accuracy in both detection and tracking. Moreover, it can be run at near real-time speed on an ordinary computer (both detection and tracking take about 140ms for each frame). The proposed approach was the best algorithm in Amirkabir Artificial Intelligence Competition in 2015 (AAIC2015).

Key Words: **Car Detection, Car Tracking, ACF, Kalman Filter**

1 Introduction

With daily increasing of number of vehicles and highway traffic, the Intelligent Transportation Systems (ITS) become more important than before. Obtaining the traffic parameters such as number of vehicles and average speed for any type (light and heavy) can help the responsible organizations with better monitoring of the traffic. The vision-based systems are better than other existing systems (like laser guns for obtaining the speed or magnetic loops for counting and classification of vehicles) from several point of view: First, the cost of cameras are far less than the other technologies. Secondly, maintenance of the cameras are much easier and finally all traffic parameters can be obtained with a single camera in a specific metropolis area.

In this paper we propose a scheme to detect and track vehicles in a video. Our proposed scheme works on a single camera mounted on a high place and it is recording video from rear of vehicles. The Aggregated Channel Features (ACF) is used for detection and a Kalman filter is employed to track the vehicles. The proposed approach can detect and track the vehicles in a near real-time speed (about 5 frame per second) on an ordinary computer[1]. The presented approach was the best method in Amirkabir Artificial Intelligence Competition in 2015

[1] Intel i5-4460, 8GB RAM

(AAIC2015[2]) and it ranked second in that competitions (no team announced as the first rank).

The rest of this paper is organized as follows: Section II illustrates the existing algorithms for detection and tracking in the literature. In section III, the used detection algorithm is described. In section IV, the configuration of our Kalman filter as a tracker is explained and section V concludes the paper .

2 Related Works

Detection and tracking of vehicles can be considered as a core of any vision-based intelligent systems. In the following subsections the most well-known methods for both detection and tracking are reviewed. Moreover, it is explained why we prefer the selected methods against the other ones.

A. Detection

All existing detection algorithms can be classified into two categories: motion-based and appearance-based approaches. Motion-based methods [1],[2],[3] try to detect vehicles using their motions. However, these methods are fast and easy (they do not need to be trained), they are too vulnerable to shadows, luminance changes and shaking of the camera. In the other hand appearance-based approaches detect vehicles with a pre-trained model according to the intrinsic features of the objects. Generally these kind of methods are more computationally demanding than motion-based approaches but they do not have the mentioned disadvantages of motion-based methods.

As mentioned, the appearance-based detections are more robust against luminance changes, camera shaking and shadows (problems that are common on highways). Because of these reasons we chose this family of methods for detection task .

One of the most well-known methods in the object detection are part-based models such as Deformable Part Model (DPM) [4]. Although DPM has a convincing performance for vehicle detection, it suffers from a high computations in test phase. The time consumption of DPM in test phase (about 9 second per frame) prevent us to have a real-time or even near real-time system for detection and tracking of vehicles. There are some methods which try to

[2] http://autcup.aut.ac.ir

accelerate run time of DPM ([5], [6] and [7]) but all these algorithms reduce the performance or they are not real-time.

The And-Or-Graph (AOG) [8] is another popular method which models the cars as a graph. This algorithm proposed a method of learning reconfigurable hierarchical And-Or models to integrate context and occlusion for car detection. Although this method has a better accuracy than DPM, its running time almost is the same as DPM. So it also cannot be used for real-time applications. Recently the Convolutional Neural Networks (CNNs) outperform other existing methods in object detection tasks. For example Densebox method \cite{huang2015densebox} could be considered as the best representative of this family which outperformed all other existing methods in Kitti dataset challenge in car detection[3]. Usually CNN-based methods need to run on Graphical Processor Units (GPU) for fast applications. The cost of GPUs is much more than CPUs and they are not available on ordinary personal computers.

In 2014 a fast object detection method was presented in [9] called Aggregated Channel Features (ACF). This method accelerates the detection with predicting of features instead of calculating them. The accuracy of ACF is almost the same as DPM and AOG, while its run time is almost 30 times faster than these methods [9] .

As we want to design a system to easily can be used in real applications, we chose the ACF among all mentioned methods as the detection algorithm of our scheme.

B. Tracking

The Kalman filter can model the movements of an object and predict the object location based on that model. The paper [10] used this method to track cars. Particle filter can be considered as an extension of Kalman filter which in addition of movement model it can use the appearance features of the tracked object. This method can be better than Kalman filter in the crowd situations where objects are much likely occluded. Tracking-Learning-Detection (TLD) [11] is another well-known tracking method which learn the appearance of object and track the object based on its appearance .

[3] http://www.cvlibs.net/datasets/kitti/eval_object.php

As we have a reliable detector which detects vehicles in each frame, we used the simplest algorithm, Kalman filter, for tracking. The Kalman filter has few complexity and its time consumption is negligible compared to detection time .

3 Detection

This section provides a brief overview of ACF detection [9] method. This method has a high accuracy for object detection tasks (especially solid objects) comparing with other competing algorithms [12]. Moreover, it enjoys a very high speed in both training and test stages. In the subsequent sections, feature types of this method are introduced and also it is explained how to extract these features in a fast way.

A. Aggregated Channel Features (ACF)

The ACF method can be summarized into four major steps: First step includes computing several channels for a given image I with a transformation function $C = \Omega(I)$. In the second step, each channel feature is partitioned to some blocks and summation in every block leads to a lower resolution channel. Finally the aggregated features will be obtained with applying a smoothing filter on the lower resolution channels. Next step is concatenating of all pixels in the aggregated channels in a vector. In the last step, a boosted trees of classifier is trained to distinguish desired objects from background. Fig 1 demonstrates these steps .

The main features of ACF detector can be summarized as:

- **Channels:** 10 channels of features are extracted as: LUV color channels (3 channels), normalized gradient magnitude (1 channel) and histogram of oriented gradients (6 channels). After smoothing these channels with a $[1\ 2\ 1]/4$ filter, the channels divided into 4×4 blocks and pixels in each block are summed up. Finally the channels are smoothed again with the same filter.
- **Classifier:** For classifying of cars and non-cars, an AdaBoost classifier [13] is trained which uses 128 depth-two trees as weak classifiers.
- **Sliding Window Detection:** As traditional object detectors, sliding window scheme is employed to find the bounding boxes of car candidates. In this approach extracted features in multiple scales, called feature pyramid, fed into a boosted classifier and desired objects are

detected in multi-scale. Usually calculating of the feature pyramid is a bottleneck for running time of detection phase. In the next section this subject will be covered in detail.

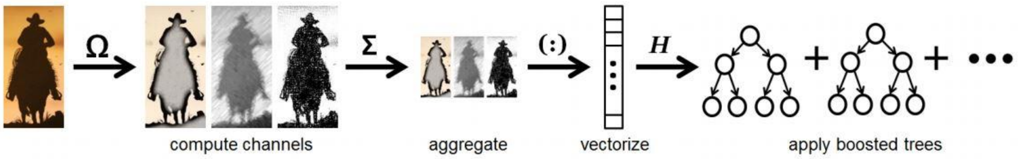

Figure 1 Outline of Aggregated Channel Features framework. First step: Computing several channels of given image I with Ω . Second step: Sum every blocks of channel features and smooth the resulting aggregated channels. Third step: Concatenation of feature channels in a vector. Fourth step: Training a boosted classifier [9]

B. Fast Feature Pyramids

One of the most challenging problems in object detection is that different instances of one object could be appeared in different sizes. To tackle this problem, detectors use sliding window method in a pyramid of multiple scale images. In traditional methods it was needed to calculate features in every scales. Therefore, constructing the feature pyramid resolves the multi-scale problem by increasing the computational cost.

To accelerate feature pyramid extraction, the ACF detector approximates features in different scales instead of calculating them. For clarifying the subject, let $I_k(x, y) = I(x/k, y/k)$ is the up-sampled version of original image and desired features are gradient of the image. Following the simple derivative rules:

$$h_{k_x} = \frac{\partial I_k}{\partial x}(x, y) = \frac{1}{k}\frac{\partial I}{\partial x}\left(\frac{x}{k}, \frac{y}{k}\right)$$

$$h_{k_y} = \frac{\partial I_k}{\partial y}(x, y) = \frac{1}{k}\frac{\partial I}{\partial y}(x/k, y/k)$$

With a simple calculus we can deduce $h_k = k \times h$.

Besides above formulas, experiments also show approximation $h_k \approx k \times h$ for gradient histogram features is true. In down-sampled images, these formulas are no longer valid because there are some information loss during down-sampling of image. But experiments show this loss of information is consistent and could be approximated with $h_{\frac{1}{k}} \approx \mu\, h$, where μ is a constant value smaller than $1/k$.

So the gradient features can be approximated for up-sampled and down-sampled images based on calculated features of original image. The authors of [9] show that besides the gradient features, other related features of image pyramid like HOG can be approximated based on features of original image. As a result the image pyramid can be approximated instead of calculating and this procedure accelerate constructing the image pyramid. More details and result of experiments are presented in [9].

4 Tracking: Kalman Filter

We use the Kalman filter framework for tracking of a single object. In first subsection the model specification of this Kalman filter is explained and in the second part we introduce an assignment algorithm to extend the single object tracking to a multi-object one.

A. Kalman Model

We consider position (x, y) and size of bounding box (w, h) and their velocity as the state of Kalman filter. So the state of the kalman filter will have 8 variables $(x, y, w, h, v_x, v_y, v_w, v_h)$.

With this defining of these variables the dynamic model will be:

$$\begin{bmatrix} x \\ y \\ w \\ h \\ v_x \\ v_y \\ v_w \\ v_h \end{bmatrix}^{t+1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \\ h \\ v_x \\ v_y \\ v_w \\ v_h \end{bmatrix}^{t} + n_d$$

where n_d is a vector and is called model noise. The measurement model can be written as:

$$\begin{bmatrix} x \\ y \\ w \\ h \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \\ h \\ v_x \\ v_y \\ v_w \\ v_h \end{bmatrix} + n_m$$

where n_m is a vector called the measurement noise .

B. Assignment of Detections and Tracks

The following cost matrix is used to assign the detected objects to existing tracks:

$$c(i,j) = \frac{Area(d_i \cap t_j)}{\sqrt{Area(d_i) * Area(t_j)}}$$

where d_i and t_j are the bounding box of i^{th} detection and j^{th} track, respectively. The Hungarian algorithm [14] is used to assign the detections and tracks according to defined cost matrix. There are three types of assignments:

- **Assigned Pairs**
  In this case the Kalman filter of the assigned track would be corrected with its assigned detection.
- **Unassigned Detections**
  An unassigned detection is probably a new object which is appeared in the current frame. So a new track will be initiated. At first the created track is not a valid track. If the number of assignment gets greater than a threshold, the track will be valid. This validation procedure help the tracker to remove the noisy false alarms of detection.
- **Unassigned Tracks**
  In this case the counter of unassigned tracks will be incremented. If this counter for a particular track gets greater than a predefined threshold, that track will be killed .

5 Implementation and Performance Evaluation

In this section we demonstrate the performance of our method on AAIC dataset[4] and explain some implementation details.

A. Implementation

We trained ACF detector with open-source piotr-toolbox [15]. For convenience we used a pre-trained DPM model (trained on PASCAL VOC2012) to extract car patches from the given training videos. In training phase we extracted 1657 car patches and 19188 non-car patches which were resized in [64 64]. Training phase, including reading all images, extracting of feature pyramids with aforementioned method and training of boosted classifiers, takes 17 seconds.

AAIC dataset includes 12 video of unstable cameras from different views, Fig 2 shows all 12 samples of this dataset. In training stage, we extracted positive and negative samples from 1000 random frames of each training video.

Table 1 COMPARISON OF OUR RESULTS WITH OTHER TEAMS IN AAIC COMPETITION

Team	Recall	Precision	FAR	MT	ML	FP	FN	IDs	FM	MOTA	MOTP
Ours	**51.27**	**81.49**	2.93	**97**	**133**	17961	**73144**	**91**	**380**	**39.57**	**73.54**
FanAssa	43.45	64.59	6.01	95	157	36737	87210	667	1072	19.20	73.49
Basir	12.95	54.56	**2.72**	27	383	**16632**	134260	127	426	2.08	68.3

B. Results

Table 1 shows result of first 3 teams of AUTCUP2015. As shown the proposed approach outperforms other participating teams in most metrics of this competitions. Particularly, our method gets MOTA of 39.57 which is almost twice better than the second team. More details of detection and tracking metrics can be found at [16].

The proposed algorithm works in near real-time (5 frame per second) on a 960×540 video on an ordinary PC. The output of tracking on all 12 videos are shown in Fig 2.

[4] Available in November 2015 at: http://ceit.aut.ac.ir/~nikabadi/AAIC_traffic_dataset/

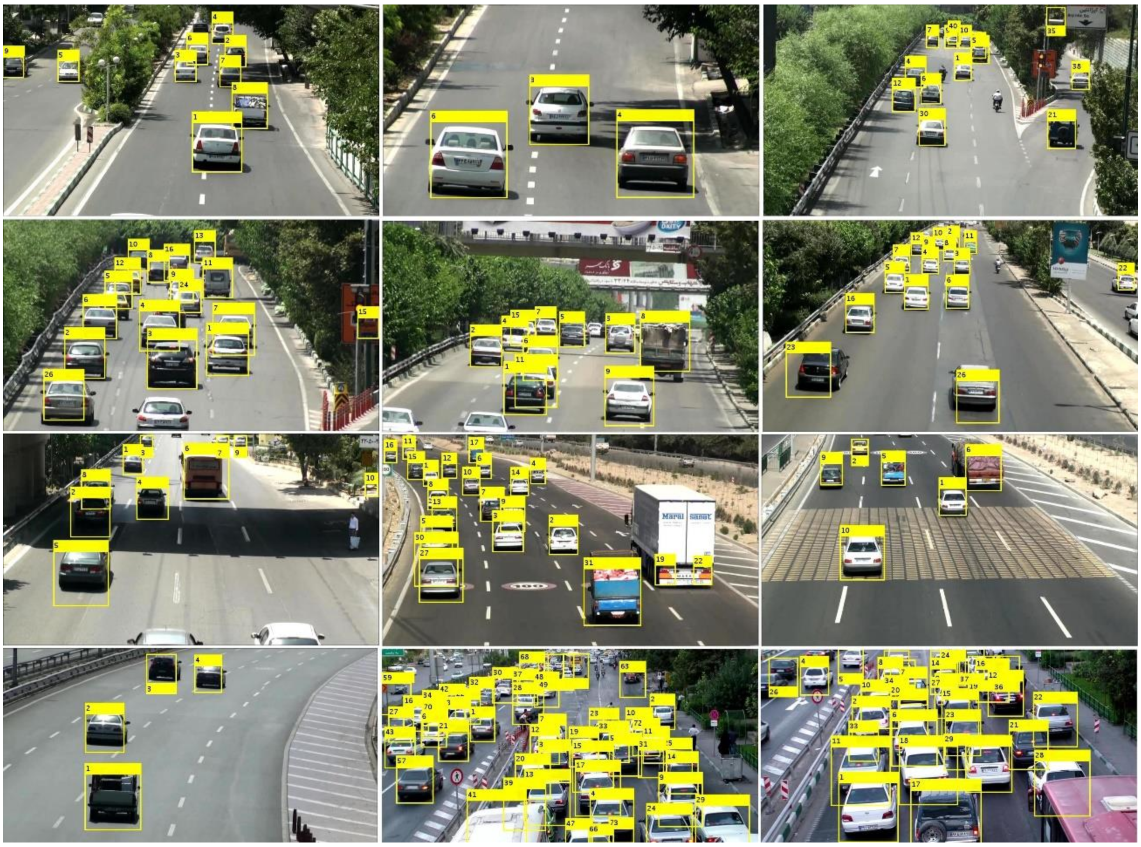

Figure 2 Result of our method in 12 videos of AAIC dataset

6 Conclusion

In this paper we presented a feasible paradigm for detection and tracking of vehicles in highways. This method enjoys high accuracy and near real-time speed property on an ordinary PC. In more details, the running time of our algorithm in test phase is about 145 ms for each frame and it got MOTA of **39.57** and MOTP of **73.54** in AUTCUP dataset. These obtained accuracies is the best results in AUTCUP2015.

7 References

[1] S. C. Sen-Ching and C. Kamath, “Robust techniques for background subtraction in urban traffic video,” in *Electronic Imaging 2004*. International Society for Optics and Photonics, 2004, pp. 881–892.

[2] N. Sirikuntamat, S. Satoh, and T. H. Chalidabhongse, “Vehicle tracking in low hue contrast based on camshift and background subtraction,” in *Computer Science and Software Engineering (JCSSE), 2015 12th International Joint Conference on*. IEEE, 2015, pp. 58–62.

[3] X. Lu, T. Izumi, T. Takahashi, and L. Wang, “Moving vehicle detection based on fuzzy background subtraction,” in *Fuzzy Systems (FUZZ-IEEE), 2014 IEEE International Conference on*. IEEE, 2014, pp. 529–532.

[4] P. Felzenszwalb, D. McAllester, and D. Ramanan, “A discriminatively trained, multiscale, deformable part model,” in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[5] J. Yan, Z. Lei, L. Wen, and S. Z. Li, “The fastest deformable part model for object detection,” in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014, pp. 2497–2504.

[6] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, “Cascade object detection with deformable part models,” in *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*. IEEE, 2010, pp. 2241–2248.

[7] M. A. Sadeghi and D. Forsyth, “30hz object detection with dpm v5,” in *Computer Vision–ECCV 2014*. Springer, 2014, pp. 65–79.

[8] B. Li, T. Wu, and S.-C. Zhu, “Integrating context and occlusion for car detection by hierarchical and-or model,” in *Computer Vision–ECCV 2014*. Springer, 2014, pp. 652–667.

[9] P. Dollár, R. Appel, S. Belongie, and P. Perona, “Fast feature pyramids for object detection,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 8, pp. 1532–1545, 2014.

[10] R. Zhang, P. Ge, X. Zhou, T. Jiang, and R. Wang, “An method for vehicle-flow detection and tracking in real-time based on gaussian mixture distribution,” *Advances in Mechanical Engineering*, vol. 5, p. 861321, 2013.

[11] Z. Kalal, K. Mikolajczyk, and J. Matas, “Tracking-learning-detection,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 7, pp. 1409–1422, 2012.

[12] P. Dollár, S. Belongie, and P. Perona, “The fastest pedestrian detector in the west.” in *BMVC*, vol. 2, no. 3. Citeseer, 2010, p. 7.

[13] J. Friedman, T. Hastie, R. Tibshirani *et al.*, “Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors),” *The annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.

[14] H. W. Kuhn, “The hungarian method for the assignment problem,” *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[15] P. Dollár, “Piotr’s Computer Vision Matlab Toolbox (PMT),” http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html.

[16] K. Bernardin and R. Stiefelhagen, “Evaluating multiple object tracking performance: the clear mot metrics,” *Journal on Image and Video Processing*, vol. 2008, p. 1, 2008.