Speed Estimation and Abnormality Detection From Surveillance Cameras

Speed Estimation and Abnormality Detection from Surveillance Cameras

Panagiotis Giannakeris
CERTH-ITI
Thessaloniki, Greece
giannakeris@iti.gr

Vagia Kaltsa
CERTH-ITI
Thessaloniki, Greece
vagiakal@iti.gr

Konstantinos Avgerinakis
CERTH-ITI
Thessaloniki, Greece
koafgeri@iti.gr

Alexia Briassouli
Dept of Data Science and Knowledge Engineering
Maastricht University, Maastricht, Netherlands
alexia.briassouli@maastrichtuniversity.nl

Stefanos Vrochidis
CERTH-ITI
Thessaloniki, Greece
stefanos@iti.gr

Ioannis Kompatsiaris
CERTH-ITI
Thessaloniki, Greece
ikom@iti.gr

Abstract

Motivated by the increasing industry trends towards autonomous driving, vehicles, and transportation we focus on developing a traffic analysis framework for the automatic exploitation of a large pool of available data relative to traffic applications. We propose a cooperative detection and tracking algorithm for the retrieval of vehicle trajectories in video surveillance footage based on deep CNN features that is ultimately used for two separate traffic analysis modalities: (a) vehicle speed estimation based on a state of the art fully automatic camera calibration algorithm and (b) the detection of possibly abnormal events in the scene using robust optical flow descriptors of the detected vehicles and Fisher vector representations of spatiotemporal visual volumes. Finally we measure the performance of our proposed methods in the NVIDIA AI CITY challenge evaluation dataset.

1. Introduction

Smart city technologies for assistive transportation and safe driving, make up one of the most intriguing domains of computer science and have attracted significant attention during the last decade. Video surveillance, along with various other types of monitoring infrastructure provide a huge amount of exploitable data for extracting optimal traffic management rules, increasing safety in busy streets, detecting or predicting and preventing accidents and numerous other applications of traffic monitoring. Moreover, increasing industry trends towards autonomous driving, vehicles, and transportation in general, is changing the landscape of traffic analysis. The visual content from traffic cameras will, in the near future, also be used to manage autonomous vehicle navigation, by sending information about events elsewhere in the city, traffic conditions, pedestrian congestion, to optimally guide vehicles.

The automated analysis of visual traffic data is necessary to extract useful information in a reasonable amount of time and with minimum human involvement in these cumbersome and extremely time consuming tasks. Many computer vision algorithms have already been developed for the automated analysis of traffic video data. Examples such as automatic vehicle detection and tracking, speed and traffic flow analysis, detection of abnormal events, have been developed and their levels of accuracy are continuously increasing. A big challenge, however, lies in the development of fast and computationally efficient methods to be used in actual real world scenarios that demand near real time solutions.

In this work we develop a traffic analysis framework combining two separate modalities. At first, we propose a cooperative detection and tracking algorithm based on deep CNN features to retrieve vehicle locations in surveillance videos and discover their trajectories in subsequent frames. The vehicle speeds are then approximated by an accurate fully automatic camera calibration algorithm. Additionally, abnormal events are detected based on optical flow descriptors of the detected vehicles and a learned GMM visual vo-

cabulary. Fisher vector representations of spatiotemporal volumes aid in the retrieval of possible abnormal events in the scene such as car crashes and stalled vehicles.

2. Related work

2.1. Speed estimation

Traffic flow analysis from surveillance cameras can be decomposed to many different aspects of traffic understanding, such as vehicle detection and tracking, counting, traffic level classification and speed estimation. We focus here on a brief review of the existing methods for speed estimation. This task involves the translation of the displacement of pixels that belong to vehicles, into the real distances traveled and so, it relies heavily on proper camera calibration. As a result, most of the proposed algorithms focus on techniques for accurate retrieval of camera intrinsic and extrinsic parameters, as well as inference of the scene scale since we are only interested in the analysis of videos taken from a single monocular camera.

Methods are generally categorized into semi-automatic and fully-automatic. In the first case most of the calculations are performed automatically, but a user's manual input is required usually in the form of some known distance in the scene. In a method from this category, [18] detected and tracked the vehicles using GMM background subtraction and Kalman filters, calibrated assuming a zero pan pinhole camera model. In [9] the calibration is based on patterns of lane markings on the road and image rectification to cope with perspective projection. A simple method using optical flow to compute displacement of pixels and relaxation of the perspective projection effect in [16] measured the speed inside a rectangle region of interest using known lane width as reference.

There are fewer works on fully automatic camera calibration methods. In [6] vanishing point detection from vehicle movement using a diamond space accumulator is performed and scale inference is computed by matching statistically detected vehicles' dimensions to mean dimensions of real vehicles. [23] extends the previous work by matching pre-made 3D vehicle models to the detected vehicles' 3D bounding boxes. An evolutionary algorithm for camera parameter extraction is used in [7], assuming constant speed of vehicles, and its accuracy is increased by license plate detection.

2.2. Anomaly detection

Methods dealing with anomaly detection in traffic videos can roughly be separated into two main categories. The first category comprises of methods that apply their models on raw image data, such as pixel location or other low level features. One recent work in this category is that of [4], using hierarchical feature representations and a Gaussian Process Regression (GPR) framework to build a low-level and a high-level codebook respectively. Anomalies are then detected, after the integration of local and global anomaly detectors. Probabilistic topic models are also proposed in a variety of works to capture spatiotemporal changes in traffic scenarios. The most typical works include the hierarchical Bayesian models of [25] which model the scene in two layers, the new Markov Clustering Topic Model of [11], the Probabilistic Latent Sequential Motifs introduced by [24] and the Dependent Dirichlet Process-Hidden Markov Model (DDP-HMM) framework proposed in [15]. In all cases anomalies are determined in a probabilistic global framework. The main drawback of all these methods is their computational cost, which is usually high due to the complexity of their models. At the same time they deal with modeling at the pixel level, ignoring more complicated structures such as the objects themselves, thus missing important information.

The second category involves methods based on trajectory extraction and analysis. Objects, or even pixels are firstly localized and tracked to obtain their patterns, which are then clustered or modeled to represent the dominant underlying motions. A work in this category is that of Salemi et al. [21] where object trajectories are modeled using kernel density estimations, while a unified Markov Chain Monte Carlo (MCMC) sampling-based scheme is then used to generate the most likely paths. Anomalies are detected based on the estimated probability density of the next state by comparing the actual measurements of objects with the predicted tracks. A different approach is followed in [14], where three different levels of semantics are considered after tracking all moving objects in the video. Rules of normal events are automatically extracted at each level and anomalies are defined as the events deviating from these rules. In [13] a collection of trajectories is sent as an input to a two-stage inference model based on a probabilistic framework, while in [27] trajectory segmentation and multi-instance learning are used for the detection of local anomalies. Finally, trajectory clustering and a single class Support Vector Machine (SVM) framework is used by [19].

3. Traffic flow analysis

A robust algorithm for detection and tracking of moving vehicles from surveillance camera footage operates at the basis of our traffic flow analysis system. Passing vehicles have to be successfully detected in each frame and then subsequently tracked as they follow their full trajectory on screen. The task is simultaneously performed in a cooperative manner by two separate modalities: (a) a generic object detector, specifically trained to discover bounding boxes of vehicles at an adjustable detection rate, and (b) a tracker which accepts new image patches as input queries and is assigned to discover the most probable position of each query

Figure 1. Illustration of three orthogonal vanishing points detected in a highway scene.

in subsequent frames. This finally leads to the incremental construction of the movement trajectories for each individual vehicle that has been captured by the detector at some point in the frame sequence.

To accurately compute the velocities of a detected vehicle, known pixel coordinates of the corresponding trajectory in the image plane have to be back-projected in real world coordinates in the road plane. Then, given a vehicle's traveled distance in meters and the time duration in seconds, velocities can be calculated. The frame per second capture ratio of the recorded videos is known and can be used to compute time intervals in seconds between consecutive frames. What is not directly available however, is a way to translate displacement of pixels in the image plane to the real distance in meters a particular vehicle has traveled in a given amount of time. In order to calculate this precisely and effectively, an algorithm for automatic camera calibration is needed. Once camera parameters have been discovered and a few basic assumptions tailored to this specific application have been integrated, pixel coordinates in the image plane can be successfully back-projected to 3D world coordinates in the road plane and therefore real vehicle displacements can be measured.

3.1. Camera calibration

In order to automatically obtain camera parameters of a traffic surveillance scene, we deploy the algorithm proposed by [6] which is based on detection of two vanishing points. As examined in [23], knowledge of two vanishing points is enough to calculate camera intrinsic parameters. Moreover, the third vanishing point position can be easily found by application of orthogonality. The model makes some basic assumptions for zero pixel skew, square shaped pixels and location of the principal point in the center of the image that produce tolerable errors. In this work we are interested in finding 3D coordinates belonging to vehicle trajectories, so the model is only used to retrieve points lying on the road plane, and is not applied to find arbitrary 3D locations in the images.

The method resorts to vehicle motion analysis in the scene as a means of retrieving the first vanishing point whose direction is parallel to the road and coincides with the stream of traffic. The detection algorithm uses the Hough transform on successfully tracked trajectory points based on parallel coordinates, mapping the projective plane onto a finite space, the so-called diamond space, as detailed in [5]. In order to detect good features to track, background subtraction is performed to limit the candidates to possibly only vehicle edges. Then, using the KLT tracker, features that are correlated with significant movement are interpreted as small straight fragments of valid trajectories and are allowed to vote in the diamond space accumulator. Based on the highest number of votes, the first vanishing point coordinates are retrieved. To discover the location of the second vanishing point, which is perpendicular to the first and parallel to the road plane, the diamond space accumulator is used again in the same manner but with the following constrains: edges supporting the first are excluded this time and an assumption of approximately horizontal scene horizon filters out nearly vertical edges. Again, the point with the most votes is selected as the second vanishing point.

Once the first two vanishing points $\mathbf{u}$, $\mathbf{v}$ have been found, we follow the calculations as in [23] to obtain the third vanishing point $\mathbf{w}$, the focal length $\mathbf{f}$ and the road plane normal vector $\overrightarrow{\mathbf{n}} = \frac{\mathbf{w}}{|\mathbf{w}|}$ which defines the road plane up to a scale. To back-project a 3D image plane point $\mathbf{p} = [p_x, p_y, f]$ onto the road plane $\mathbf{R}$ with the center of the camera projection in $\mathbf{O} = [0, 0, 0]$, we calculate the intersection $\overrightarrow{\mathbf{Op}} \cap \mathbf{R}$ using the normal plane vector to express $\mathbf{R}$. However, the distance of the road plane to the camera center is still unknown as $\overrightarrow{\mathbf{n}}$ only describes the direction of $\mathbf{R}$ up to a scale. This means that any distance we calculate from 3D points is not expressed directly in real units of distance, but rather is normalized by the scale $\mathbf{d}$. To overcome this we can apply back-projection of two 3D points in the scene with known distance in meters (or other unit) and solve for $\mathbf{d}$. The original paper adopts a different method for scale inference which includes fitting pre-made 3D car models in 3D bounding boxes in order to make the procedure fully automatic with no required user input. Figure 1 illustrates the three vanishing point orientations as detected from the algorithm, as well as the horizon line.

3.2. Vehicle detection and tracking

For the purpose of detecting vehicles in video frames we chose to extract deep image representations from a CNN and predict pixel coordinates of bounding boxes using a deep CNN object detector. We adopt a modification of the robust Faster-RCNN, originally introduced in [20] that is tailored for generic object detection in image samples. A more thorough evaluation of this model and comparisons

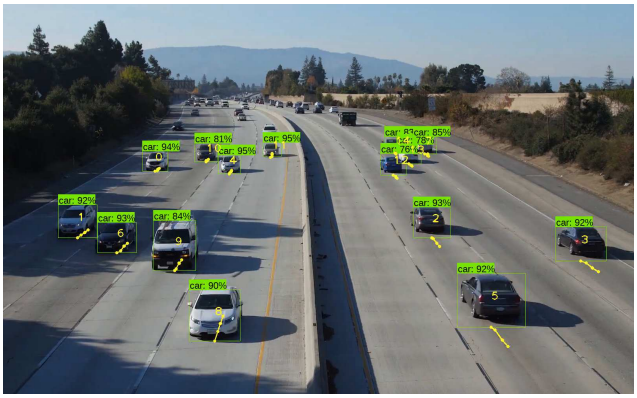

Figure 2. Visualization of detected vehicles and their trajectories.

with other SoA deep object detection architectures is presented in [12]. More specifically we adopted the Faster-RCNN-resnet101 architecture that achieves a good trade off between speed/accuracy. This model incorporates the resnet101 [8] deep feature extractor and a region proposal network along with a bounding box classifier and coordinate regressors. We chose this architecture because it achieves very fast object detection by using a single feed-forward convolutional network to directly predict classes and bounding boxes of objects.

The core functionality of our tracker is based on the KCF tracking algorithm that was proposed in [10]. The vehicle detector is used initially in order to detect vehicles every *r* video frames and initialize the new vehicle candidate database with new entries. Bounding box coordinates are stored over time so that full trajectories can be build. For every new ID its corresponding class label and a detection score is saved as well. Immediately after, the algorithm checks the new detections from the candidate pool for overlaps with already existing recent trajectories. Then, based on an IoU score check it rejects found boxes that exceed an overlap threshold to avoid creating multiple identities for the same vehicle. Next, we feed the KCF tracker with the remaining boxes in order to localize their position throughout sequential video frames. Future detections of already tracked vehicles are also utilized in order to rectify the bounding boxes of the monitored vehicles. When a detection is missed, we relocalize the bounding box relying only on KCF update coordinates, while when the algorithm does not not localize any tracked vehicle for *l* sequential video frames the vehicle is presumed to have traveled off the frame. To tackle overlaps between True Positive (TP) cases, which translates to when a correctly tracked vehicle passes in front of another confusing the tracker, we chose to merge the trajectories at the current frame and assign the oldest ID to the resulting trajectory. Figure 2 depicts bounding boxes and trajectories of vehicles successfully detected and tracked using our proposed algorithm.

3.3. Velocity estimation

To estimate the velocity of a tracked vehicle at frame f we feed the KLT tracker with points inside the previous box instance of frame *f-1* and produce several displacement calculations for each point. We then back-project all the displacement pixel pairs to the road plane according to the calibration parameters that have been found on that specific scene and select the median displacement as the true value. To calculate the velocity **v** in *m/s* we divide the true distance with the time duration of a frame which is equal to $1/fps$ *sec*.

4. Anomaly Detection in Traffic Scenes

4.1. Video Representation

In order to deal with the challenging nature of traffic videos derived from real surveillance systems, the proposed scheme should exhibit features such as generality, scalability, independence in external conditions (e.g. illumination changes, camera motion etc) and also simplicity for computational reasons. To this end, a framework based on the object detection described in the previous section is proposed. More specifically, an early descriptor for each track is formed in a pre-defined time window, while Fisher encoding follows to efficiently capture the whole frame's dynamics.

The early descriptor concerning each object in the scene consists of the concatenation of all the values describing object's speed and position in a specific spatiotemporal volume. More precisely, the early descriptor for a specific detected object in a window of t frames is given by:

$$D = \{OF_1, \ldots, OF_t, px_1, py_1, \ldots, px_t, py_t\} \quad (1)$$

where OF_t stands for the average value of the optical flow magnitudes corresponding to the object's bounding box at frame t, while px_t and py_t represent the position of the center of the bounding box in xy-axis at frame t.

Subsequently, all early descriptors extracted from a particular video sequence are led into a Fisher encoding scheme. This way, a visual vocabulary based on the most discriminating features of the whole video is built, and a more efficient representation is provided. The computation of the most discriminating samples is performed by applying unsupervised clustering (Gaussian Mixture Model (GMM)) in the shallow representation hyperspace, as formed by the feature collection of each video.

Let $\{\mu_j, \Sigma_j, \pi_j; j \in R^L\}$ be the set of parameters for L Gaussian models, with μ_j, Σ_j and π_j standing respectively for the mean, the covariance and the prior probability weights of the j^{th} Gaussian. Assuming that the D-dimensional early descriptor is represented as $\overline{x}_i \in R^D; i =$

$\{1, \ldots, N\}$, with N denoting the total number of descriptors, Fisher encoding is then built upon the first and second order statistics:

$$f_{1j} = \frac{1}{N\sqrt{\pi_j}} \sum_{i=1}^{N} q_{ij} \sigma_j^{-1} (\overline{x}_i - \overline{\mu}_j)$$
$$f_{2j} = \frac{1}{N\sqrt{2\pi_j}} \sum_{i=1}^{N} q_{ij} [\frac{(\overline{x}_i - \overline{\mu}_j)^2}{\sigma_j^2} - 1] \quad (2)$$

where q_{ij} is the Gaussian soft assignment of descriptor x_i to the j^{th} Gaussian and is given by:

$$q_{ij} = \frac{exp[-\frac{1}{2}(x_i - \mu_j)^T \Sigma_j^{-1} (x_i - \mu_j)]}{\sum_{t=1}^{L} exp[-\frac{1}{2}(x_i - \mu_t)^T \Sigma_j^{-1} (x_i - \mu_t)]} \quad (3)$$

Distances as calculated by Eq. 2 are next concatenated to form the final Fisher vector, $F_X = [f_{11}, f_{21}, \ldots, f_{1L}, f_{2L}]$, characterizing each trajectory.

In order to deal with noise due to camera motion, each video is split into sequences of a one shot, and the proposed scheme is applied separately in each of them. Transitional clips are completely ignored by the process while clips consisted only of a small number of frames are discarded. The detection of camera motion is based on the estimation of global optical flow magnitudes per frame, and an experimentally defined threshold determines the presence of general motion. Despite the simplicity of the method, real world situations of this kind are managed in an efficient way, as proved by the experiments.

4.2. Anomaly detection

In order to infer about anomalous trajectories, the Support Vector Method For Novelty Detection of [22] is adopted. Support Vector Machines (SVMs) are chosen, as they generally exhibit good performance relatively to other machine learning methods at a low computational cost, while at the same time they are able to handle large data sets, which generally appear in real life situations.

The goal is to find an appropriate hyperplane that separates the training data from the origin, maximizing the distance and removing potential outliers. Thus, its purpose can be expressed in the terms of the following minimization problem:

$$min_{w, \dot{\xi}_i, \rho} \frac{1}{2} ||w||^2 + \frac{1}{\nu n} \sum_{i=1}^{n} \dot{\xi}_i - \rho \quad (4)$$

subject to:

$$(w - \phi(x_i)) \geq \rho - \dot{\xi}_i, \quad \forall i = 1, \ldots, n \quad (5)$$

$$\dot{\xi}_i \geq 0, \quad \forall i = 1, \ldots, n \quad (6)$$

where $x_i \in X$,$i \in [1, \ldots, n]$ stands for the n training data, w, ρ define the created hyperplane, $\dot{\xi}_i$ represents slack variables contributing to a soft margin and ν is a constant belonging to the space $[0, 1]$. The role of the parameter ν is quite important as it sets an upper bound on the fraction of outliers and at the same time defines a lower bound on the fraction of support vectors. The above minimization problem is solved by Lagrangian techniques, which lead to a consistent decision function that determines scores for each trajectory.

After the estimation of final scores, anomalous trajectories are defined as those that have a total score value greater than 3σ, with σ representing the standard deviation of all scores found in the particular video sequence. The proposed method is found to work efficiently in challenging traffic datasets, surpassing SoA methods, and detecting even the most challenging anomalous events.

5. Experiments and evaluation

We evaluated our traffic analysis methods in the NVIDIA AI CITY challenge [2]. Teams that entered the competition had the opportunity to experiment and evaluate their algorithms on three separate traffic analysis challenge tracks: (a) traffic flow analysis that focused on speed estimation of vehicles, (b) anomaly detection, for the detection of anomalous events such as car crashes or stalled vehicles and (c) multi-camera vehicle detection and re-identification. Moreover,a real-world evaluation dataset was made available for each track. In the sections that follow we present the experiments that took place and our evaluation scores for the first two tracks of the challenge.

5.1. Traffic flow analysis track

The dataset that was provided for the speed estimation track is composed of 27 videos and each one is a sequence of 1800 high definition resolution frames. The recorded videos depict highway traffic at several locations from various viewpoints. The camera in most of the videos is static, except at some locations where small trembling can be spotted, presumably due to windy conditions. Some videos also contain duplicate sequential frames at an unpredictable rate for an unknown reason which makes vehicles appear static. We presumed that when such frames appear no real time has passed and we chose to copy previous speed estimates.

In order to tailor our model for vehicle detection we acquired the Faster-RCNN-resnet101 previously trained on the COCO dataset [17] and we further tuned it to the DETRAC dataset [26]. We choose to penalize the localization and classification losses of the second stage with the same weight, training for 140K steps using a momentum optimizer with a learning rate schedule which was initialized at 0.0005 and decreased progressively. We used random horizontal flips of the input bounding boxes during training as a

means of data augmentation. Furthermore, we experiment with the detection rate setting it to $r = [3, 5]$ frames per detection. Through empirical cross validation we set the target miss threshold l to 3 consecutive frames and the merge IoU threshold to 0.8.

The overall score $S1$ of this track incorporates a measure of the quality of the detections and the accuracy of the speed measurement and is defined as:

$$S1 = DR * (1 - NRMSE)$$

where, DR is the detection rate and NRMSE is the normalized root mean square error (RMSE) of speed estimation. A vehicle is said to be detected if it was localized in at least 30% of frames it appeared in and NRMSE is the normalized RMSE across all submissions, obtained via min-max normalization. Our vehicle detector and tracker manages to achieve a high detection rate score of $\mathbf{89}\%$ for both detection settings. The RMSE score of our speed estimator is $\mathbf{27.30}$ which appears to be higher that other teams participating in the challenge since our final Normalized RMSE score is 1. Further investigation needs to take place in order to detect the possible inefficiencies of our approach and further improve the accuracy of our estimations.

5.2. Anomaly detection track

The NVIDIA dataset for anomaly detection track comprises of 100 videos of 15 minutes each, at 800×410 resolution. It constitutes a challenging dataset as it contains a great variety of real traffic scenarios, severe camera motion, different weather conditions, illumination changes, occlusions and many low resolution shots.

In our effort to handle video sequences containing a variety of camera motions, such as zoom in/out or even a complete change of view, we choose to divide them into sub clips characterized by static view, so as to develop different models in each of them. This was deemed necessary, as scene dynamics may change dramatically and the existence of a single model seems to be completely inadequate. Camera motion is detected according to frame's average value of optical flow's magnitudes, with a value greater than 5 indicating that a severe motion is present and the frames that follow are ignored until a static view is restored. Then the training of a new model takes place.

In order to exploit temporal information, proposed algorithm is applied in a time window of 50 frames, to extract sufficient information from the video, while capturing anomalies on time. Trajectories whose length is smaller than 20 are discarded as possible sources of noise, while the rest of them are led to a Fisher-GMM scheme, with 25 cluster centers. Subsequently, a one class SVM with a Gaussian kernel is deployed with $\nu = 0.1$, so as to create a hyperplane characterizing existing trajectories. Anomalous instances arise from scores exceeding 3σ, with σ representing the standard deviation of all scores found in the particular video sequence.

Evaluation is based on anomaly detection performance, measured by the F1-score, and detection time error, measured by RMSE. More specifically the score is calculated as:

$$S2 = F1 * (1 - NRMSE)$$

where a true-positive (TP) detection will be considered as the predicted anomaly within 5 minutes absolute time distance of the true anomaly that has the highest confidence score, a false-positive (FP) is a predicted anomaly that is not a TP for some anomaly and a false-negative (FN) is a true anomaly that was not predicted. RMSE is calculated between the ground truth anomaly time and the predicted time for all TP predictions. NRMSE is the normalized RMSE obtained from min-max normalization across all submissions. We managed to reach an F1-score of 0.33 and our detection time RMSE was 227. Overall, we managed to surpass two other competing teams but we were outperformed by four.

6. Conclusion

We proposed a traffic analysis framework comprised of two major modularities: a cooperative detection and tracking algorithm for vehicle trajectory discovery which is ultimately used for the estimation of their speed and an anomaly detection algorithm that analyses visual content so as to discover in an efficient manner possible abnormalities such as car crashes and stalled vehicles from countless minutes of video surveillance footage that may be used in order to alert authorities of dangerous situations.

Acknowledgment

This work was supported by beAWARE [1] and ROBORDER [3] projects, partially funded by the European Commission under grant agreement No 700475 and No 740593.

References

[1] beAWARE Project. http://beaware-project.eu. 6

[2] Nvidia ai city challenge. https://www.aicitychallenge.org. 5

[3] ROBORDER Project. http://roborder.eu/. 6

[4] K. W. Cheng, Y. T. Chen, and W. H. Fang. Gaussian process regression-based video anomaly detection and localization with hierarchical feature representation. *IEEE Transactions on Image Processing*, 24(12):5288–5301, Dec 2015. 2

[5] M. Dubská, A. Herout, R. Juránek, and J. Sochor. Fully automatic roadside camera calibration for traffic surveillance. *IEEE Transactions on Intelligent Transportation Systems*, 16(3):1162–1171, 2015. 3

[6] M. Dubská, A. Herout, and J. Sochor. Automatic camera calibration for traffic understanding. In *BMVC*, volume 4, page 8, 2014. 2, 3

[7] P. Filipiak, B. Golenko, and C. Dolega. Nsga-ii based auto-calibration of automatic number plate recognition camera for vehicle speed measurement. In *European Conference on the Applications of Evolutionary Computation*, pages 803–818. Springer, 2016. 2

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[9] X. C. He and N. H. Yung. A novel algorithm for estimating vehicle speed from two consecutive images. In *Applications of Computer Vision, 2007. WACV'07. IEEE Workshop on*, pages 12–12. IEEE, 2007. 2

[10] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015. 4

[11] T. Hospedales, S. Gong, and T. Xiang. Video behaviour mining using a dynamic topic model. *International Journal of Computer Vision*, 98(3):303–323, Jul 2012. 2

[12] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *IEEE CVPR*, 2017. 4

[13] H. Jeong, Y. Yoo, K. M. Yi, and J. Y. Choi. Two-stage online inference model for traffic pattern analysis and anomaly detection. *Machine Vision and Applications*, 25(6):1501–1517, Aug 2014. 2

[14] F. Jiang, J. Yuan, S. A. Tsaftaris, and A. K. Katsaggelos. Anomalous video event detection using spatiotemporal context. *Computer Vision and Image Understanding*, 115(3):323 – 333, 2011. Special issue on Feature-Oriented Image and Video Computing for Extracting Contexts and Semantics. 2

[15] D. Kuettel, M. D. Breitenstein, L. V. Gool, and V. Ferrari. What's going on? discovering spatio-temporal dependencies in dynamic scenes. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1951–1958, June 2010. 2

[16] J. Lan, J. Li, G. Hu, B. Ran, and L. Wang. Vehicle speed measurement based on gray constraint optical flow algorithm. *Optik-International Journal for Light and Electron Optics*, 125(1):289–295, 2014. 2

[17] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[18] A. Nurhadiyatna, B. Hardjono, A. Wibisono, I. Sina, W. Jatmiko, M. A. Ma'sum, and P. Mursanto. Improved vehicle speed estimation using gaussian mixture model and hole filling algorithm. In *Advanced Computer Science and Information Systems (ICACSIS), 2013 International Conference on*, pages 451–456. IEEE, 2013. 2

[19] C. Piciarelli, C. Micheloni, and G. L. Foresti. Trajectory-based anomalous event detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(11):1544–1554, Nov 2008. 2

[20] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 3

[21] I. Saleemi, K. Shafique, and M. Shah. Probabilistic modeling of scene dynamics for applications in visual surveillance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(8):1472–1485, Aug 2009. 2

[22] B. Schlkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001. 5

[23] J. Sochor, R. Juránek, and A. Herout. Traffic surveillance camera calibration by 3d model bounding box alignment for accurate vehicle speed measurement. *Computer Vision and Image Understanding*, 161:87–98, 2017. 2, 3

[24] J. Varadarajan, R. Emonet, and J.-M. Odobez. A sequential topic model for mining recurrent activities from long term video logs. *International Journal of Computer Vision*, 103(1):100–126, May 2013. 2

[25] X. Wang, X. Ma, and W. E. L. Grimson. Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(3):539–555, March 2009. 2

[26] L. Wen, D. Du, Z. Cai, Z. Lei, M. Chang, H. Qi, J. Lim, M.-H. Yang, and S. Lyu. Detrac: A new benchmark and protocol for multi-object tracking. *arXiv preprint arXiv:1511.04136*, 2015. 5

[27] W. Yang, Y. Gao, and L. Cao. Trasmil: A local anomaly detection framework based on trajectory segmentation and multi-instance learning. *Computer Vision and Image Understanding*, 117(10):1273 – 1286, 2013. 2