VEHICLE SPEED ESTIMATION BY LICENSE PLATE DETECTION AND TRACKING

Diogo C. Luvizon, Bogdan T. Nassu and Rodrigo Minetto

Department of Informatics - DAINF
Federal University of Technology - Paraná (UTFPR), Curitiba, Brazil
diogo@luvizon.com, {nassu,rminetto}@dainf.ct.utfpr.edu.br

ABSTRACT

We describe a novel system for vehicle speed estimation from videos captured in urban roadways. Our system uses text detection to locate the license plates of passing vehicles, which are then used to select stable features for tracking. The tracked features are then filtered and rectified for perspective distortion. Vehicle speed is estimated by comparing the trajectory of the tracked features to known real world measures. In experiments performed on videos captured under real operation conditions, our system attained a precision of 0.87 and a recall of 0.92 for license plate detection. Vehicle speeds were estimated with an average error of 0.59 km/h, staying inside the +2/-3 km/h limit, determined by regulatory authorities in several countries, in over 75% of the cases.

***Index Terms*—** Vehicle speed estimation, license plate detection, text detection, feature tracking.

1. INTRODUCTION

Determining vehicle speed is an important task for urban traffic surveillance. The information may be used not only to issue fines when drivers exceed speed limits, but also to feed systems such as traffic controllers.

Vehicle speed can be measured by intrusive or non-intrusive technologies. Intrusive systems, usually based on inductive loop detectors, are highly sensible and accurate, but have high installation costs and can damage the roadway. Non-intrusive systems are mostly based on laser sensors, infrared sensors, Doppler radar, or audio based sensors [1, 2]. They can be portable, but are expensive and require frequent maintenance. In many cities, inductive loop detectors are installed along with cameras, which are used to identify the license plates of vehicles that exceed the speed limit. If the speed information can be extracted from the already available image data, we may obtain a non-intrusive system with significantly reduced costs.

In this paper, we describe a novel system for estimating vehicle speed from videos captured in urban roadways. The system's pipeline is shown in Fig. 1. First, a text detector [3, 4] is used to detect the license plates of passing vehicles. Stable features inside the detected regions are then tracked, using a combination of the SIFT [5] and KLT [6] algorithms. After filtering out inconsistencies, the vehicle speed is estimated by comparing feature trajectories to known real world measures, which allow us to rectify the perspective distortion and obtain a meter-per-pixel relation. To our knowledge, our system is the first to estimate vehicle speed by tracking corner and region features extracted from the license plate.

To evaluate our system, we used videos captured under real operation conditions associated with ground truth data obtained by an inductive loop detector. Our system attained 0.87 precision and a 0.92 recall for license plate detection. Vehicle speed was estimated with average error of 0.59 km/h, staying inside the +2/-3 km/h limit determined by regulatory authorities in several countries [7], in over 75% of the cases.

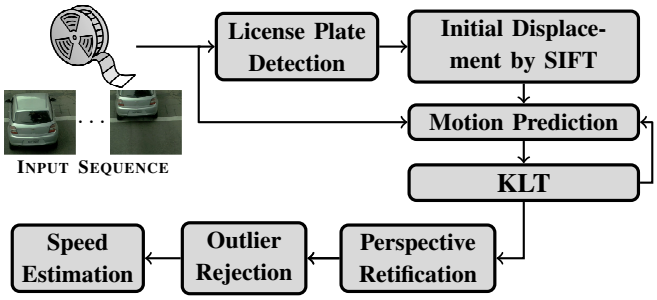


Fig. 1. Overview of the proposed system.

This paper is divided as follows. In Section 2, we discuss previous work. Our approach is described in Section 3, and its experimental evaluation is reported in Section 4. Section 5 concludes the paper and gives directions for future work.

2. RELATED WORK

Several methods were proposed to estimate the speed of vehicles in highways [1, 8, 9, 10, 11, 12, 13]. All these methods are based on background subtraction and blob detection techniques, with the speed being estimated from the displacement of blobs (considering the centroid or part of the contour). This

This work was supported by Fundação Araucária, CNPq and CAPES.

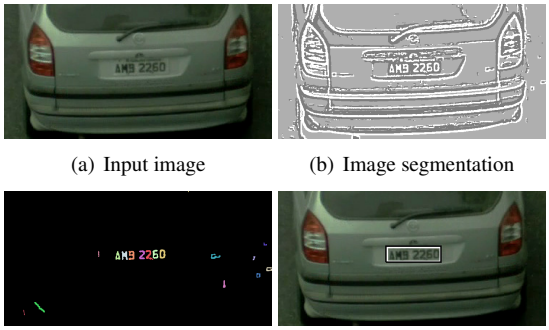
kind of approach requires that the camera is distant enough from the scene so that it is possible to identify the entire vehicle. Our method works in a different scenario, with the camera being closer to the vehicles. That makes blob analysis very sensitive to lighting variations and shadows, with the limits of the vehicles becoming hard to determine. On the other hand, the closer view makes it possible for our approach to detect and track stable features inside the license plate region. These differences prevent a direct comparison between our system and the above mentioned work.

Many methods were proposed for detecting license plates [14, 15, 16]. These methods analyze connected components, edges, straight segments and textures; detecting the boundaries or the characters of the license plate. References [15, 16] give an overview of the problem and existing solutions. Although our system depends on license plate detection, this paper concentrates on the problem of determining vehicle speed based on features extracted from inside the license plate region. Other license plate detectors could be used without changing the structure of our system.

3. PROPOSED APPROACH

3.1. License Plate Detection

License plate detection is crucial to our system's performance. For this task we use SNOOPERTEXT [3], publicly available at [4], a state-of-the-art algorithm for detecting text in urban scenes (such as building numbers, billboards, etc.). Several parameters of the detector were selected to improve its performance for detecting license plates. As shown in Fig. 2, the SNOOPERTEXT detector consists of four main modules: image segmentation, character filtering, character grouping, and text region filtering [17].



(a) Input image (b) Image segmentation

(c) Character filtering (d) Grouping

Fig. 2. License plate detection by SNOOPERTEXT.

The segmentation algorithm used by SNOOPERTEXT [18] is a modified version of Serra's *toggle mapping* [19], an operator for local contrast enhancement and thresholding that uses the local foreground and background levels (Fig. 2 (b)). In order to find bright text on dark background, the segmentation is repeated on the negative (pixel-wise complemented) image. This second stage can be ignored if the license plates always have dark characters on bright background (that varies from country to country). That can considerably decrease the computation effort, and also helps avoiding false detections.

The segmented foreground regions are filtered based on simple geometric criteria (area, width and height) and classified as character/non-character, based on shape descriptors trained on a dataset of segmented letters (Fig. 2 (c)). The remaining regions are then grouped to form the candidate license plate regions (Fig. 2 (d)).

All these steps are performed in a multi-scale fashion, in order to efficiently handle different character sizes, to suppress irrelevant detail and to avoid the use of overly large kernels in the segmentation. Note that, depending on the camera position and video frame size, we do not need more than 2 pyramid levels to detect license plates — opposed to the 5 levels needed in a free context scenario.

In the last step, candidate text regions are validated by a binary text/non-text region classifier that rejects candidate regions that do not appear to contain a single line of text. This classifier uses the T-HOG descriptor [17], which is based on the multi-cell *Histogram of Oriented Gradients* (HOG) [20].

3.2. Feature Extraction and Tracking

After a vehicle's license plate is detected in a frame, our system extracts features from the license plate region, and tracks these features across frames. The license plate region is used only once for each vehicle, to determine the set of features to be tracked. To extract and track features, we combine the Kanade-Lucas-Tomas (KLT) [21, 6] and the Scale-Invariant Feature Transform (SIFT) [5] algorithms.

Features are extracted as described by Shi and Tomasi [22]. A "good" feature is a region with high intensity variation in both x and y directions, such as textured regions or corners. Let $[I_x\ I_y] = \nabla I = [\partial I/\partial x\ \partial I/\partial y]$ be the image derivatives in the x and y directions of I and Z a 2×2 matrix given by

$$Z = \sum_{W} \begin{bmatrix} {I_x}^2 & I_x I_y \\ I_x I_y & {I_y}^2 \end{bmatrix} \quad (1)$$

where W is a window with $n \times n$ pixels centered on some pixel within the license plate region. The region covered by the window is selected if both eigenvalues of Z are above a given threshold (set as 1 in our system).

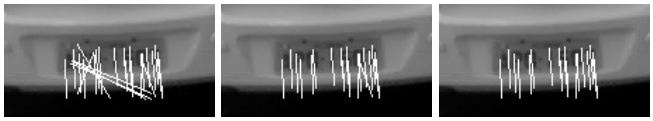
To track the selected features with subpixel accuracy we have used the pyramidal KLT algorithm [23]. Let I and J be two successive video frames. The KLT algorithm takes I, J and a set of n template regions $T_{\{1,2,\ldots,n\}} \in I$ with $n \times n$ pixels covered by a small image window W that contains the features to be sought. For each selected template T_i centered at $u = (x, y)$, it returns an adjusted position $d = (x', y')$ on J such that the neighborhood W of $u + d$ in J is most

similar to T_i. That is, it finds the displacement d which minimizes $\sum_W [J(u+d) - I(u)]^2$. The maximum pixel displacement d that the pyramidal KLT can handle is given by $d = (2^{\ell+1} - 1)\delta$, where ℓ is the number of pyramid levels and δ is the pixel motion allowed by elementary optical flow computation (of the order of one pixel). That is, as noted by Bouguet [23], for $\ell = 3$ the maximum displacement that can be found is about 15 pixels. However, depending on the vehicle speed and the video frame rate, this can be insufficient. To circumvent this problem, we need an initial estimation of the vehicle speed. This estimation is obtained by computing SIFT features within the first occurrence of the license plate region and matching them in the next frame.

The average region displacement found by the SIFT matching is used as a guessed position for the KLT algorithm. That is, we compute a rough prediction of the vehicle displacement d in the next frame by using the SIFT, this displacement is further refined with sub-pixel accuracy by the KLT algorithm, in order to find the best displacement d. Note that the SIFT features are used only for the initial estimate, when the license plate is first detected — for the remaining frames, the prediction is computed from the average region displacement found by KLT, since we already have a coarse estimative of the vehicle speed.

3.3. Outlier Rejection

As described in Section 3.2, from the motion of each template region T_i in a consecutive pair of frames, we extract a motion vector $d_i = (x, y)$. In order to discard motion vectors that correspond to mismatches, i.e. outliers, we compute the mean and standard deviation of the displacements in the x and y axes, discarding motion vectors outside the three-sigma deviation in any direction. This procedure is repeated until the standard deviation in both x and y axes become smaller than 0.5 pixel. The process is exemplified in Fig. 3.



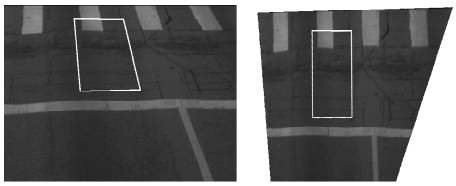
(a) (b) (c)

Fig. 3. Outlier rejection: (a) motion vectors; (b) outliers elimination (fourth iteration); (c) final result (fifth iteration).

3.4. Vehicle Speed Estimation

Vehicle speed estimation starts from the second frame, after the detection of the license plate, and is based on the features tracked by the KLT algorithm. For each new frame, each tracked feature (except the outliers) will result in a motion vector $d_i = (x, y)$ in pixels per frame. In order to convert the motion vector to a velocity vector — in meters per second — we have to determine a relationship between the pixel motion in the image and the motion in the real world.

As the images acquired by the camera have perspective distortion, we first rectify the image [24], as shown on Fig. 4. For each motion vector d_i we compute a new motion vector r_i using an homography matrix and a set of known reference distances measured from the roadway (highlighted in Fig. 4). The rectified motion vector has a constant meter-per-pixel factor M. Then, for each adjacent pair of frames, a rectified motion vector $R = (x_r, y_r)$ is calculated by the average of all r_i. The norm of R is given by $\|R\| = \sqrt{x_r^2 + y_r^2}$, and real displacement in meters by $D = \|R\| / M$, where D is the displacement in meters. Finally, the vehicle speed s in km/h is computed by $s = 3.6\frac{D}{T}$, where T is the frame interval in seconds. Note that rectification is performed only for the motion vectors — the features are still tracked in the original input images.



(a) (b)

Fig. 4. Image rectification: (a) original image with perspective effect; (b) image rectified.

4. EXPERIMENTAL EVALUATION

For our tests, we collected a dataset with 75 vehicle sequences from a urban road lane with associated ground truth speed. Our dataset was acquired from a video, with frame resolution of 768×480 pixels and 31.25 frames per second. The ground truth speeds were obtained from a high precision speed meter inductive loop detector, properly calibrated and approved by a national metrology agency. The dataset vehicle speed distribution is shown in Fig. 5.

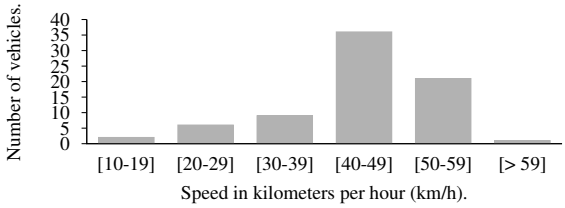


Fig. 5. Dataset vehicle speed distribution.

4.1. License Plate Detection Performance

We measure the license plate detection performance in terms of precision (P) — the proportion of detected objects that

were indeed license plates — and recall (R) — the proportion of license plates that were detected. As only the first occurrence of each license plate is used by our system to extract and track features, all measures consider only the frame in which the first detection have occurred.

Let r be the rectangular region detected as a license plate, and s be the real license plate region in the image. A true positive (TP) is counted if $A(r \cap s)/A(r \cup s) > 0.7$, where $A(t)$ is the area of the smallest rectangle enclosing a set t. Otherwise, we have a false positive (FP). A false negative FN refers to a missed license plate and a true negative TN refers to a vehicle without license plate. From these indicators, we have $P = TP/(TP + FP)$ and $R = TP/(TP + FN)$.

The overall precision and recall of our system, considering all the vehicle sequences, were $P = 0.87$ and $R = 0.92$ respectively (for $TP = 60$, $FP = 9$, $FN = 5$ and $TN = 1$). The F-measure (the harmonic mean of P and R), given by $F = 2/(1/P + 1/R)$, was $F = 0.90$. Some samples of license plates found by character detection are shown in Fig. 6.

Fig. 6. Samples of license plates found by character detection.

The detection errors were mainly due to: motorcycles, with license plate characters too small to be detected (less than 9 pixels tall); false positives detected together with the correct plate; license plates in poor condition. See Fig 7.

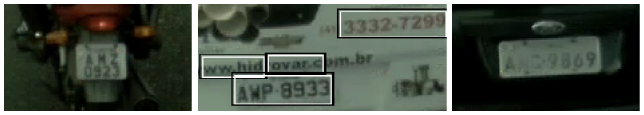


Fig. 7. License plate detection errors.

4.2. Speed Estimation Performance

The speed performance was computed by comparing the estimated speed, returned by our system, with the ground truth speed. See Fig. 8. The average error, for the whole dataset, was 0.59 km/h with a standard deviation of 1.63 km/h. Some samples of speed estimation are shown in Fig 9.

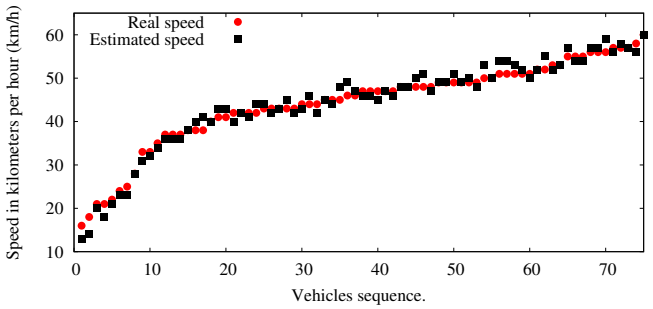


Fig. 8. Vehicle speed performance: estimated speed (black squares) and the ground-truth (red circles).

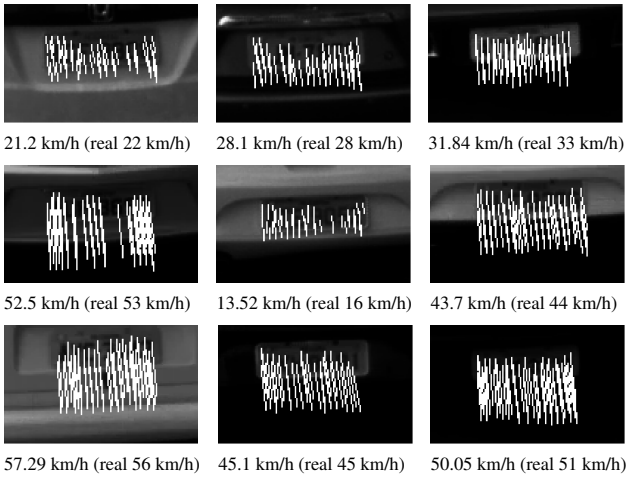


Fig. 9. Speed estimation for those vehicles shown in Fig. 6. Their real speed, obtained by an inductive loop detector, are shown in brackets. The magnitude of the motion vectors (white lines) are proportional to the vehicle speed.

The maximum nominal speed error values for the whole dataset were -3.24 km/h and $+3.91$ km/h.

5. CONCLUSIONS

In this paper, we described a novel system for vehicle speed estimation from videos captured in urban roadways. Our experiments have shown that, using an ordinary camera, with resolution of 768×480 pixels and 31.25 frames per second, the proposed system was able to successfully detect most license plates, robustly tracking features and estimating vehicle speeds using a combination of the SIFT and KLT algorithms.

As future work, we intend to perform character recognition on the detected license plates in order to create a traffic speed control system with integrated surveillance tools, e.g. to analyze the traffic flow, to identify stolen vehicles, etc.

6. REFERENCES

[1] Xiao C. H. and Yung N.H.C., "A Novel Algorithm for Estimating Vehicle Speed from Two Consecutive Images," *IEEE Workshop on Applications of Computer Vision (WACV)*, p. 12, 2007.

[2] F. Pérez-González, R. López-Valcarce, and C. Mosquera, "Road Vehicle Speed Estimation from a Two-microphone Array," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, pp. 1321–1324, 2002.

[3] R. Minetto, N. Thome, M. Cord, N. J. Leite, and J. Stolfi, "SnooperText: A Text Detection System for Automatic Indexing of Urban Scenes," *Computer Vision and Image Understanding (CVIU), Elsevier (to appear)*, pp. 1–14, 2014.

[4] R. Minetto, "Snoopertext Source Code," `http://www.dainf.ct.utfpr.edu.br/~rminetto/projects/snoopertext/`, 2013.

[5] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision (IJCV)*, vol. 60, no. 2, pp. 91–110, 2004.

[6] Carlo Tomasi and Takeo Kanade, "Detection and Tracking of Point Features," Tech. Rep., 1991.

[7] U.S. Departament of Transportation, "Speed-measuring Device Performance Specifications: Across-the Road Radar Module," 2007.

[8] C. Maduro, K. Batista, P. Peixoto, and J. Batista, "Estimation of Vehicle Velocity and Traffic Intensity Using Rectified Images," *IEEE International Conference on Image Processing (ICIP)*, pp. 777–780, 2008.

[9] H. Zhiwei, L. Yuanyuan, and Y. Xueyi, "Models of Vehicle Speeds Measurement with a Single Camera," *International Conference on Computational Intelligence and Security Workshops (CISW)*, pp. 283–286, 2007.

[10] H. A. Rahim, U. U. Sheikh, R. B. Ahman, and A. S. M. Zain, "Vehicle Velocity Estimation for Traffic Surviillance System," *World Academy of Science, Engineering and Technology (WASET)*, p. 772, 2010.

[11] Schoepflin T.N. and Dailey D.J., "Dynamic Camera Calibration of Roadside Traffic Management Cameras for Vehicle Speed Estimation," *Intelligent Transportation Systems (ITS)*, 2003.

[12] L. Grammatikopoulos, G. Karras, and E. Petsa, "Automatic Estimation of Vehicle Speed from Uncalibrated Video Sequences," *Modern Technologies, Education and Professional Practice in Geodesy and Related Fields*, pp. 332–338, 2005.

[13] Ilkwang Lee, Hanseok Ko, and D.K. Han, "Multiple Vehicle Tracking Based on Regional Estimation in Nighttime CCD Images," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, pp. IV–3712–IV–3715, 2002.

[14] Kuan-Hui Lee, Yong-Jin Lee, and Jenq-Neng Hwang, "Multiple-kernel Based Vehicle Tracking Using 3-d Deformable Model and License Plate Self-similarity," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 1793–1797, 2013.

[15] C.N.E. Anagnostopoulos, I.E. Anagnostopoulos, I.D. Psoroulas, V. Loumos, and E. Kayafas, "License Plate Recognition From Still Images and Video Sequences: A Survey," *Intelligent Transportation Systems (ITS)*, vol. 9, pp. 377–391, 2008.

[16] Shan Du, M. Ibrahim, M. Shehata, and W. Badawy, "Automatic License Plate Recognition (ALPR): A State-of-the-Art Review," *Circuits and Systems for Video Technology*, vol. 23, pp. 311–325, 2013.

[17] R. Minetto, N. Thome, M. Cord, N. J. Leite, and J. Stolfi, "T-HOG: An Effective Gradient-Based Descriptor for Single Line Text Regions," *Pattern Recognition (PR), Elsevier,*, vol. 46, no. 3, pp. 1078–1090, 2013.

[18] J. Fabrizio, M. Cord, and B. Marcotegui, "Text Extraction From Street Level Images," in *ISPRS Workshop on City Models, Roads and Traffic (CMRT)*, 2009, number 38, pp. 199–204.

[19] J. Serra, "Toggle Mappings: from Pixels to Features," pp. 61–72, 1989, J.C. Simon (ed.), Elsevier.

[20] Navneet Dalal and Bill Triggs, "Histograms of Oriented Gradients for Human Detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2005, pp. 886–893, IEEE Computer Society.

[21] Bruce D. Lucas and Takeo Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.

[22] Jianbo Shi and Carlo Tomasi, "Good Features to Track," Tech. Rep., Ithaca, NY, USA, 1993.

[23] Jean-Yves Bouguet, "Pyramidal Implementation of the Lucas Kanade Feature Tracker," *Intel Corporation, Microprocessor Research Labs*, 2000.

[24] Gary Bradski and Adrian Kaebler, *Learnin OpenCV: Computer Vision with the OpenCV Library*, O'Reilly Media, Inc., Sebastopol, CA, USA, 2008.