A Vision System for Detection and Tracking of Stop-Lines

Young-Woo Seo

CMU-RI-TR-14-09

June 2014

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

This paper presents a computer vision algorithm that detects, by analyzing lane-marking detection results, stop-lines and tracks, using an unscented Kalman filter, the detected stop-line over time. To detect lateral and longitudinal lane-markings, our method applies a spatial filter emphasizing the intensity contrast between lane-marking pixels and their neighboring pixels. We then examine the detected lane-markings to identify perpendicular, geometry layouts between longitudinal and lateral lane-markings for stop-line detection. To provide reliable stop-line recognition, we developed an unscented Kalman filter to track the detected stop-line over frames. Through the testings with real-world, busy urban street videos, our method demonstrated promising results, in terms of the accuracy of the initial detection accuracy and the reliability of the tracking.

Contents

1 INTRODUCTION

To be deployed in real world driving environments, self-driving cars should be capable of complying with the traffic rules, i.e., understanding rules in place and executing its driving maneuvers as dictated. For example, an autonomous vehicle should be able to recognize a stop-line and stop at the detected stop-line. Such a capability is a crucial one that self-driving cars must acquire, to drive along with manually-driven cars. However, a development of such an intelligent driving behavior is challenging as it requires a seamless and reliable execution of multiple relevant sub-tasks; recognition of the traffic rule in place (e.g., recognizing a lateral lane-marking as a stop-line), decision-making of appropriate driving behavior (e.g., slowing down to stop at the line), motion-planning and vehicle acutation control (e.g., following the computed trajectory with appropriate speed). Each of these tasks needs to be flawlessly executed to comply with the traffic rule. As a successful completion of these tasks begins a recognition of stoplines, this paper presents an effort, as a first step toward developing of such an intelligent driving behavior, of developing a computer vision algorithm that detects stop-lines and tracks them over time.

A successful recognition of on-road-surface traffic devices such as stop-lines and other road-markings is important for successful development of advanced-driver assistance systems or self-driving cars because it can be used to localize ego-vehicle position [2, 17], and build a map of urban driving environment [2, 5, 8]. For instance, Barth et al manually marked stop-line locations to build a map of stop-lines for localization [2]. Wu and Ranganathan transfomed stereo camera images into an inverse perspective image to detect road-markings (e.g., directional arrows, railroad crossings, etc.) and used the features extracted from the detected road-markings to localize the position of ego-vehicle [17]. Choi et al. also converted an input perspective image into an inverse perspective image and applies multiple steps of morphological operators to detect a pedestrian-crossing [5]. Most works on detecting on-road-surface traffic devices including ours use an inverse perspective image because the spatial layout between lane-markings is recovered, e.g., two longitudinal lane-markings are appeared to be parallel. Our approach is different from existing ones in that 1) we examine the geometric layouts between the detected, longitudinal and lateral lane-markings, 2) we develope a Bayes filter to track the detected stop-lines, and 3) our testing scenes are more complex and challenging than those of two previous works [2, 5]. In particular, our testing images are more challenging to detect stop-lines in that the lateral and longitudinal lane-markings for stop-lines and for lane-boundaries are not always visible because of occlusions by neighboring vehicles and other urban structures, and painting qualities.

The pipeline of our stop-line recognition algorithm begins with a lane-marking detection. A stop-line is a traffic device painted on roads (mostly and nearly) orthogonally to the vehicle's travel direction and intended to dictate the point where cars should stop [14]. Its width varies, but the color and material are mostly identical to longitudinal lane-marking. Based on this fact, we extended our longitudinal lane-marking detection algorithm [11, 12] to identify lateral lane-markings. Many excellent works have been done in the field of lane-marking detection. We refer to [7, 9] for a comprehensive literature survey and here we briefly discuss only the work relevant to ours. To de-

tect longitudinal lane-markings, some investigated lane-markings' appearances (e.g., regularity in shapes [13] and homogeneity in color [4]). Others including ours have utilized the fact that there are intensity contrasts between lane-marking pixels and their neighboring pixels [1, 3, 10]. Other methods have used extra information, such as geometric structures of road lanes or road scenes, to improve lane-marking detection results [16, 18]. Similarly, we utilize the result of vanishing point detection to improve an initial result of longitudinal, lane-marking detection [12].

For most of cases, our lane-marking detection works reasonably well. Thus, given a set of detected lateral and longitudinal lane-markings, one might think it would be easy as just picking up one of the detected lateral lane-markings. In practice, however, this is not the case because (lateral and longitudinal) lane-markings are not always clearly visible, due to occlusions and low quality of painting. In other words, for the busy urban street scenes, the lane-marking detector would fail to correctly detect all of the true lane-markings. As results, some of the detection results may include erroneous and partial detections. To effectively handle such potentially erroneous detection of lane-markings, we further examine the spatial layout between the detected lateral and longitudinal lane-markings. In particular, we seek for the lateral lane-marking that has the strongest orthogonal layout to longitudinal lane-markings, to detect a stop-line.

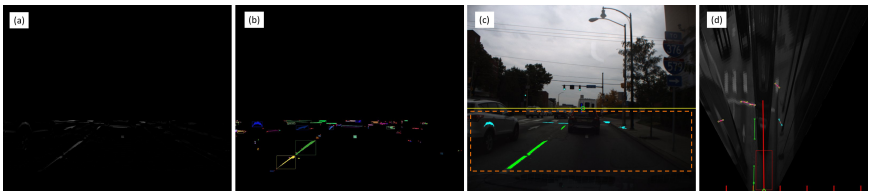

Figure 1: Examples of lateral and longitudinal lane-marking detection. (a) To produce an intensity image about lane-markings, a spatial filter is applied to emphasize the intensity contrasts between lane-marking pixels and their neighboring pixels. (b) An intensity thresholding and a connected-component grouping algorithm are applied to the resulting intensity image shown (a), to produce a set of lane-marking blobs. The detected lane-marking blobs are depicted in different colors. (c) The results of our lane-marking detection depicts longitudinal lane-markings in green and lateral lane-markings in cyan. The green-circle at the mid of the image shows the detected vanishing point and the yellow line indicates the estimated horizon line. (d) The image sub-region delineated by an orange dashed rectangle in (c) is, through an inverse perspective mapping, transformed into a ground image for a stop-line detection.

On busy street images, occasionally no lateral lane-markings appear on the input images, due to occlusions by neighboring cars. For such extreme, but not uncommon cases, our lane-marking detector would completely fail to detect lane-markings because no relevant image features are available. To effectively tackle such cases, we develop an unscented Kalman filter (UKF) to track the detected lane-marking over frames. This assumes that the images with no relevant features will appear only after the image with relevant features, where our detector is able to detect stop-lines. Once our detector detects stop-lines, the UKF tracks it over frames to fill the potential gaps of stop-line

detections.

Our contributions include 1) a development of vision-based stop-line detection based on a lateral and longitudinal lane-marking detection, 2) a development of stop-line tracking using an unscented Kalman filter, and 3) an empirical evaluation of the proposed algorithms.

In what follows, we first explains how our lane-marking detector works in Section 2.1. We detail how our detector, through an investigation of the detected, lateral and longitudinal lane-markings' geometric layouts, identifies a stop-line in Section 2.2 and tracks the detected stop-line over frames in Section 2.3. We then discuss the findings from experiments in Section 3.

2 RELIABLE STOP-LINES DETECTION OVER TIME

To reliably identify a stop-line from a given input image, we first detect lateral and longitudinal lane-markings [11, 12]. We examine the characteristics of each detected lane-marking and their geometric relations to identify lateral lane-markings as stop-line candidates. A detection of stop-lines initiates a Bayes filter to track them over the frames.

2.1 Longitudinal and Lateral Lane-Marking Detection

We detect lane-markings by applying a spatial filter to an input image. The filter is designed to identify the intensity contrast between lane-marking pixels and their neighboring pixels [10]. We define a region of interest (ROI) for lane-marking detection and apply this spatial filter to the ROI, to obtain a new intensity image about lane-markings. Figure 1 (a) shows a resulting intensity image about lane-markings. To identify a set of lane-marking blobs, we first do an intensity thresholding to this new intensity image, to produce a binary image of lane-markings and then apply a connected-component grouping. Figure 1 (b) shows the identified lane-marking blobs. To simplify the representation of each pixel blob, we fit a line segment to the pixel blob. To this end, we compute the eigenvalues and eigenvectors of the pixel coordinates' dispersion matrix. The eigenvector, $\mathbf{e}_1$, associated with the largest eigenvalue is used to represent the orientation of a line segment and its length, $\mathbf{l} = (\phi, \rho) = (atan2(\mathbf{e}_{1,2}, \mathbf{e}_{1,1}), \bar{x}\cos\phi + \bar{y}\sin\phi)$, where $\bar{x} = \frac{1}{n}\Sigma_i x_i, \bar{y} = \frac{1}{n}\Sigma_i y_i$. A pixel blob, lb_l, is then represented as a triplet of the coordinates of its centroid, x_l and y_l, and its orientation, ϕ_l. For this line-segment fitting, we tried three methods: the line-fitting based on eigen-analysis, the probabilistic, and the standard Hough transform. We found the eigen-analysis method to work best in terms of the number of resulting lines and representation fidelity to the patterns of low-level features. Figure 1 (c) shows an example of the lane-marking detection results where green pixel blobs represent longitudinal lane-markings and cyan ones represent lateral lane-markings. We then project this lane-marking detection results onto a ground image (or an inverse perspective image). We do this to remove any perspective effects, e.g., make parallel longitudinal, lane-marking appear to be parallel (or nearly parallel). Figure 1 (d) shows an inverse perspective image that is acquired by transforming the image region delineated by orange dashed line in Figure 1 (c).

2.2 Stop-Line Detection

A stop-line is a lane-marking painted perpendicularly to the driving direction of a road. The orientations of some stop-lines are slanted, but for most cases, the orientation of a stop-line is nearly orthogonal to that of longitudinal lane-marking – the one delineates the boundary of a road-lane. The previous section described how our algorithm detects lateral and longitudinal lane-markings. This section details how our algorithm further analyzes the detected lane-markings to detect stop-lines.

Our lane-marking detector works reasonably well for most of urban driving scenes [12]. However, for given busy urban streets, it would fail to correctly detect all of the true lane-markings. This is primarily because lane-markings are not clearly visible, due to occlusions and obsolete paintings. This makes the problem of stop-line detection more challenging. To effectively handle this challenge, we define, using the longitudinal, lane-marking detection results, a region-of-interest (ROI) and analyze the geometric relation between the detected, lateral and longitudinal lane-marking. In particular, we define a region of interest to filter out some of the detected lane-marking blobs irrelevant to stop-line detection and reduce the area of the image region to search for stop-lines. Clearly the optimal ROI would be the boundary of the road-lane that our vehicle happens to be driving on. However, at busy urban streets, it is challenging to even clearly detect longitudinal lane-markings about left and right boundary of a road-lane. Thus we roughly define a rectangular ROI based on the results of our longitudinal lane-marking detection results and a prior knowledge about a road-lane width in pixel. An cyan, dashed-line rectangle at Figure 2 shows an example of the ROI about the current road-lane. When the longitudinal lane-marking detection returns a solid, white lane-markings, this would produce a laterally-narrower rectangular ROI.

For the detected lateral, lane-markings within the ROI, our algorithm groups them in terms of longitudinal or y-axis distance between lane-markings. This is because some of the detected, lateral lane-markings belonging to a true lateral lane-marking appear to be separate, due to occlusion by other neighboring vehicles.

For each of the lateral, lane-marking groups, we compile a list of points that each point is either end-point of a lateral, lane-marking. And then we fit a line segment to the list of points, as we did it for representing a lane-marking blob, and represent a lateral, lane-marking, lm_m, as a vector of the coordinates of its two end-points ($\mathbf{p}_{j=1,2} = \left\{x_j^m, y_j^m\right\}$), its orientation, ψ_m, and its length, ρ_m. A score function, $f(lm_m)$ is used to compute the likelihood of a lateral, lane-marking group for being a stop-line.

$$f(lm_m) = \alpha h(lm_m) + (1-\alpha) g(lm_m) \tag{1}$$

where h examines how likely the appearance of a given lateral, lane-marking group, lm_m, is to be a stop-line and g investigates how strong the lane-marking group, lm_m is supportive by longitudinal, lane-markings. In particular, $h(lm_m)$ computes how perpendicular the lateral, lane-marking's orientation is with respect to the road's driving direction by computing $\sin(\psi_m)$, where $\sin(\psi_m) \in [0, \pi]$. To be supportive, a longitudinal lane-marking lb_l should be located between two end-points of the lateral, lane-marking group, lm_m. To check this, our algorithm performs a x-coordinate overlap test. Suppose that x_1^m and x_2^m from a lateral lane-marking group, where $x_1^m < x_2^m$, and x_3^l and x_4^l from a longitudinal lane-marking, where $x_3^l < x_4^l$. We can deter-

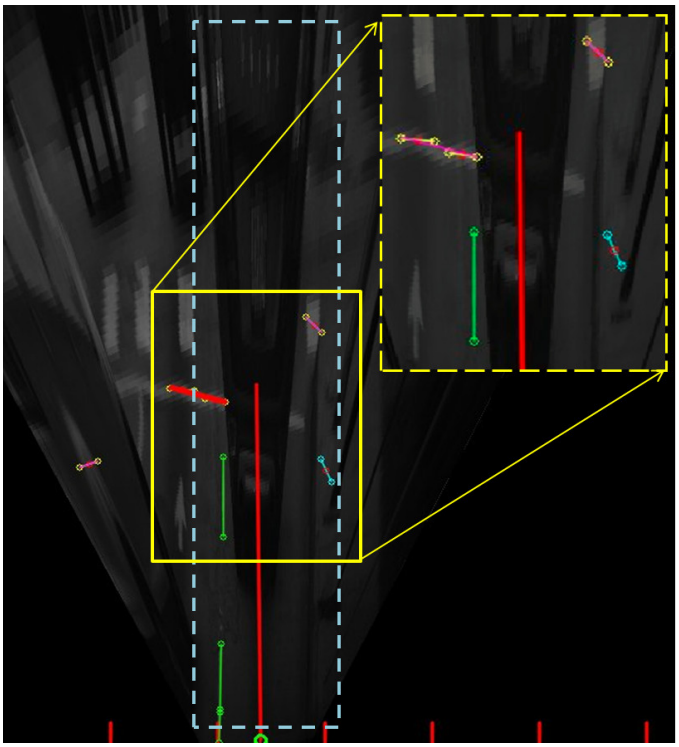

Figure 2: An example of stop-line detection result. The inlet image at the top right enlarges the image subregion by the yellow rectangle.

mine that there is no overlap between these two x ranges if $x_1^m > x_4^l$ and $x_2^m < x_3^l$. Once a detected longitudinal lane-marking is regarded to be supportive for a lateral, lane-marking group, the function, $g(lm_m)$ examines 1) how perpendicular they are, 2) how long the longitudinal lane-marking is, and 3) how far they are each other. Our algorithm chooses then the lateral, lane-marking group that maximizes the likelihood function, $f(lm_m)$. Figure 3 shows some examples of the stop-line detection where the detected stop-lines are painted in red.

2.3 Stop-Line Tracking

To reliably and consistently identify a stop-line, we track, using a Bayes filter, the detected stop-line. Due to the current setup of our vision system (e.g., resolution, CCD size, lens, mounting height, etc.), our stop-line detector begins to work when a stop-line appears about 30 meters in the front of ego-vehicle. For most time, within such a short distance, the relative motion between ego-vehicle and the stop-line is linear because ego-vehicle drives on the same lane until it passes the stop-line. Occasionally, however, the motion could be nonlinear when ego-vehicle changes lane. Based on this observation, we implement a nonlinear Kalman filter, unscented Kalman filter (UKF)

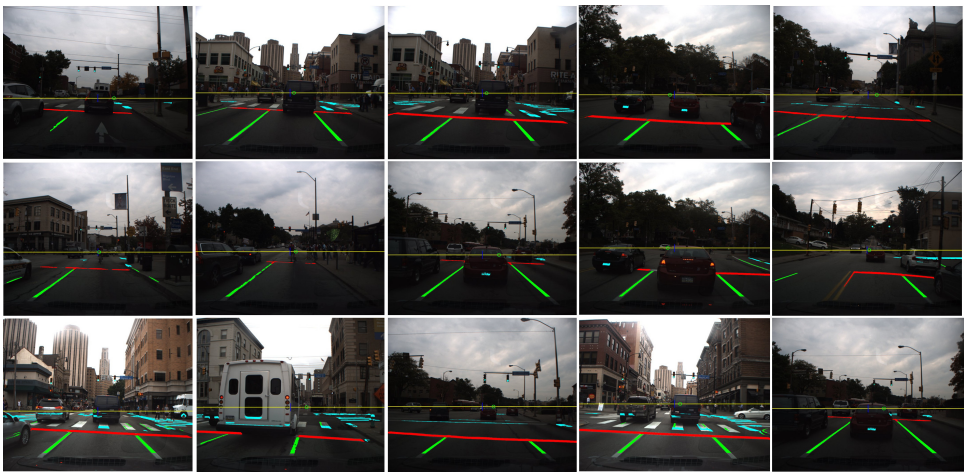

Figure 3: Examples of stop-line detection results. The red pixel blobs represent the detected stop-lines, the green (cyan) ones represent the detected longitudinal (lateral), lane-markings. The red, lateral lane-markings are detected stop-lines.

[15].

We define the state of a detected stop-line as, $\mathbf{x}_k = [y_k, \theta_k]^T$. We define the state such a way because of two observations: 1) a true stop-line moves toward the bottom of an ground image, as ego-vehicle drives toward a stop-line and 2) although the x-coordinate of a stop-line changes, it is not critical to track the x-coordinate. Figure 4 shows the coordinate of a ground plane where our UKF tracks the detected stop-line over frames.

Given the previously estimated state, $\hat{\mathbf{x}}_k$, the location of a stop-line at the next time step is predicted by the following prediction model:

$$\mathbf{x}_{k+1}^{-} = \begin{bmatrix} y_k + v\Delta t \\ \theta_k \end{bmatrix} \tag{2}$$

where $\mathbf{x}_{k+1}^{-}$ is the state predicted by the above motion model, v is the speed of ego-vehicle and Δt is a discretized time unit. The value of v is in fact interpreted into the number of pixels that approximates, on the ground plane, the speed of ego-vehicle in miles per hour.

A motion model is used to predict the location of a detected stop-line at the next time step. At the predicted state, $\mathbf{x}_{k+1}^{-}$, our filter also predicts an expected measurement, $\hat{\mathbf{z}}$.

$$\hat{\mathbf{z}} = \mathbf{h}(\mathbf{x}_{\mathbf{k+1}}^{-}) = \begin{bmatrix} y_k + \tan\theta(x_j - x_0) \\ \theta \end{bmatrix} \tag{3}$$

where x_j is the jth detected, lateral lane-marking's x-coordinate and x_0 is the predefined location of the x-coordinate of a stop-line being tracked.

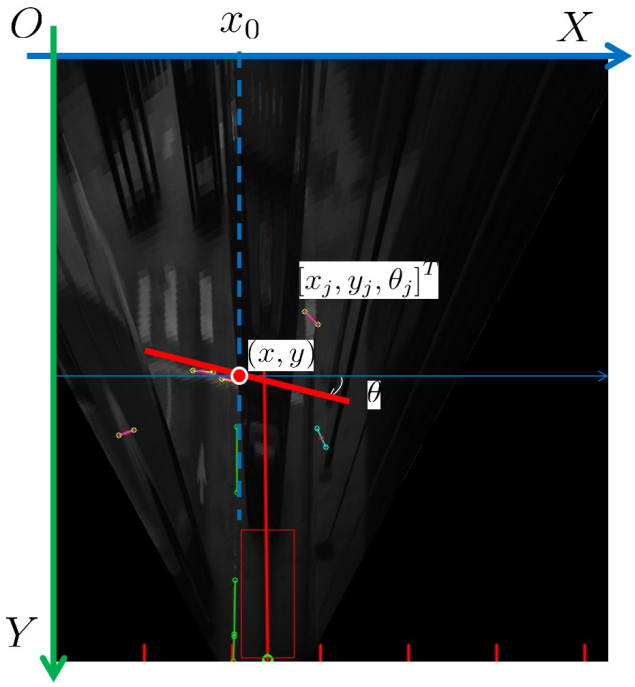

Figure 4: A ground plane where our stop-line tracker works. A stop-line being tracked is represented by a triplet of the coordinates of its centroid and its orientation. A solid line with two circles represents a projection of a lane-marking that is detected from an input perspective image. Notice that the origin is at the top left.

To optimally estimate the state, these predictions should be corrected by the image features obtained from the image frame where the predictions are made. To do so, we use the detected, lateral lane-markings. In particular, each of the detected, lateral lane-marking is used as a measurement for our UKF, $\mathbf{z}_j = [y_j, \theta_j]^T$. Note that the lateral lane-marking used as a measurement is one of the lateral lane-markings in the lateral lane-marking group used for the stop-line detection.

At a state update step, the state, $\hat{\mathbf{x}}_{k+1}$, (and its covariance matrix $\mathbf{P}_k$) is estimated by examining the difference between the expected measurement, $\hat{\mathbf{z}}$ and the actual measurement, $\mathbf{z}$, $\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + K_k(\mathbf{z}_j - \hat{\mathbf{z}}_k)$, ($\mathbf{P}_k = \mathbf{P}_k^- - K_k \mathbf{P}_z K_k^T$). Where K is the Kalman gain that determines how much the innovation $(\mathbf{z}_k - \hat{\mathbf{z}}_k)$ is used to compute the estimate, $\hat{\mathbf{x}}_k$.

3 EXPERIMENTS

To evaluate the effectiveness of the proposed algorithms, we tested them against 485 urban street images. These images were acquired while driving on 15 intersections (i.e.,

15 instances of stop-lines) at busy urban streets where the roads were crowded with cars and pedestrians. The vision sensor installed on our vehicle is PointGrey's Flea3 Gigabit camera, which can acquire an image frame of 2448×2048 maximum resolution at 8Hz. The CCD size of the camera is 2/3 inches and our lens is 8 milimeter, resulting in horizontal (vertical) field of view 57.6° (44.8°). For a faster, real-time processing, we rescaled the original resolution into half and used a predefined ROI, $x_1 = 0$, $x_2 = \mathbf{I}_{width-1}$, $y_1 = 1300$ and $y_2 = 1800$ for the inverse-perspective transformation. These y-values are in the original resolution. If the image is scaled to a half of the original, these y-values are scaled as well.

We implemented the proposed algorithms in C++ with OpenCV libraries. Our implementation of stop-line detection and tracking ran about 5 Hz on a 2.7GHZ quad core. For the UKF, we used the Bayes++ package[1] to implement our boundary tracker and initialized the state and its error (or covariance) matrix, $\mathbf{x}_0 = [y_0, \theta_0]^T = [300, 0]^T$ and $\mathbf{P}_0 = diag\left([10^2, 0.01^2]\right)$, where the values of x is in pixels, θ is in radian. In addition, the noises of the process model, $\mathbf{Q} = diag\left([10^2, 0.01^2]\right)$ and the measurement model, $\mathbf{R} = \left[5^2, 0.01^2\right]$.

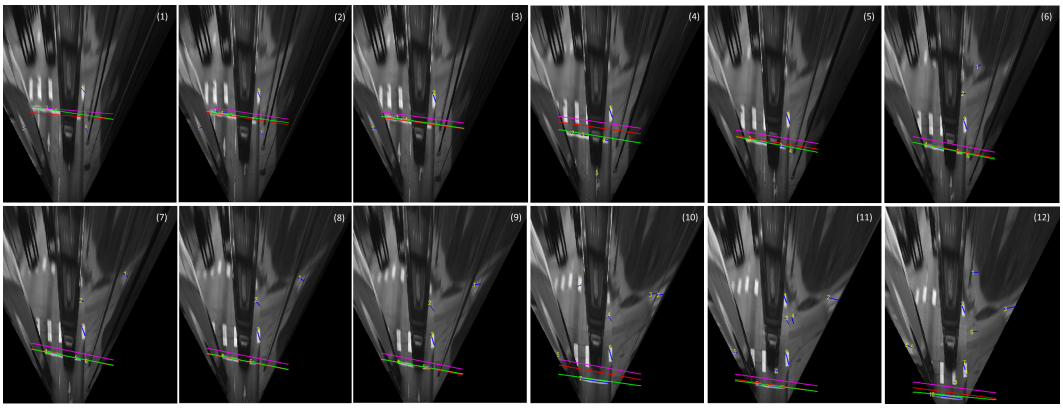

Figure 5: Examples of stop-line tracking. The blue lines represent the detected (lateral and longitudinal) lane-markings. The magenta lines are stop-line detection results, the red lines are predictions of UKF, and the green lines are the result of UKF tracking.

	Precision	Recall
Detection Only	0.89	0.92
Detection-Tracking	0.94	0.95

Table 1: An error-metric comparison between the detection and the tracking of stop-lines.

Table 1 shows experimental results that compare the performance between the detection and the the tracking. For the "Detection-Only," the stop-line detection method

[1] `http://bayesclasses.sourceforge.net/Bayes++.html`

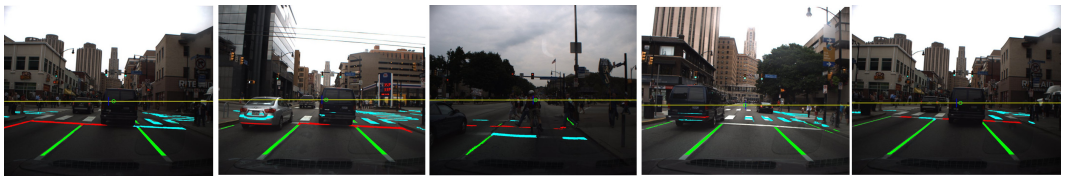

Figure 6: Examples of testing images both detection and tracking of stop-lines failed.

described in Section 2.2 was used to detect stop-lines from individual input frames and its performance was evaluated by examining whether its outputs correpond to the true stop-lines. Whereas for the "Detection-Tracking" method, the "Detection-Only" method was used to detect stop-lines from input image frames, if any, and then the tracking method described in Section 2.3 was used to track the detected stop-lines. Because that it is difficult to define the number of true negatives, we measure the performance of the "Detection-Only" and "Detection-Tracking" using precision and recall where the precision is the ratio of the correct outputs over the total number of outputs and the recall is the ratio of the correct outputs over the total number of true stop-lines. Note that 430 out of 485 test images contains the true stop-lines.

The "Detection-Only" method produced 445 outputs against 485 testing images where 396 of them were correct and 49 were incorrect, and 34 true stop-lines were not detected, resulting in the precision, 0.89 ($= \frac{396}{396+49}$) and the recall, 0.92 ($= \frac{396}{396+34}$). The performance of the "Detection-Tracking" method was the precision, 0.94 ($= \frac{408}{408+26}$) and the recall, 0.95 ($= \frac{408}{408+22}$). As expected, the "Detection-Tracking" method outperformed the "Detection-Only" method. But the same time, the "Detection-Only" method worked relatively well, even at the challenging urban street images. The most mistakes the "Detection-Only" method made are from the middle of the image sequences. In particular, the "Detection-Only" methods failed to correctly detect a stop-line when no relevant image features are present at that image frame. However, the tracking method was able to correctly predict the location of the stop-line based on earlier detection results and based on the prediction model. The performance gain by the tracking method explained the fact, whenever the detector initiated the tracker to track the detected stop-lines, the tracker. worked well.

Although the performance of "Detection-Tracking" is promising, there are some images that both the detection and the tracking of stop-line failed. Figure 6 shows some of erroneous detections where a part of the true stop-lines was not detected; a part of neighboring cars was detected; the part of cross-walk was detected.

4 CONCLUSIONS AND FUTURE WORK

This paper presented a new computer vision method that detects, through an analysis of the detected lateral and longitudinal lane-markings, stop-lines and tracks, using an unscented Kalman filter, the detected stop-lines over frames. To detect stop-lines, the proposed algorithm detects, utilizing intensity contrasts between lane-marking pixels and their neighboring ones, lateral and longitudinal lane-markings and then examines

orthogonal relation between the detected lane-markings. A stop-line detection algorithm might fail to consistently detect the stop-line that its appearance vary over consecutive image frames. To effectively tackle with such a problem, we developed an unscented Kalman filter to track the initially detected stop-line. Through the testings with images about busy, urban streets, our algorithm demonstrated promising results.

In practice, it is very difficult to develop a perception algorithm that its performance is reliable enough to be used for a close-loop system, (i.e., begin to slow down a vehicle when a stop-line is detected.) Therefore a perception algorithm needs some guidance or help from other systems. For example, Fairfield and Urmson built a map of traffic lights to remove potential false-positive detection of traffic light bulbs while minimizing any misses of traffic lights [6]. Without such a map, it would be very hard for them to use their traffic light detector to control their self-driving cars. Their map helps the detector by providing the information about where and when to look for the traffic light bulbs. As a future work, we would like to build a map of stop-lines to enhance the performance of our stop-line detection and tracking algorithm. To be useful, the tracked stop-line should be associated with a real world coordinates for controlling ego-vehicle, e.g.) slowing down its speed as it approaches to the tracked stop-line. To this end, we would like to investigate a usage of homography and an alignment of active sensor measurements with images for computing the information about metric measurement.

5 Acknowledgments

The author would like to thank the team members of the General Motors-Carnegie Mellon University Autonomous Driving Collaborative Research Laboratory (AD-CRL) for their effort.

References

[1] M. Aly. Real time detection of lane markers in urban streets. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 7–12, 2008.

[2] A. Barth, J. Siegemund, and J. Schwehr. Fast and precise localization at stop intersections. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 82–87, 2013.

[3] A. Broggi, A. Cappalunga, C. Caraffi, S. Cattani, S. Ghidoni, P. Grisleri, P. P. Porta, M. Posterli, and P. Zani. Terramax vision at the urban challenge 2007. *IEEE Transactions on Intelligent Transportation Systems*, 11(1):194–205, 2010.

[4] K.-Y. Chiu and S.-F. Lint. Lane detection using color-based segmentation. In *IEEE Intelligent Vehicles Symposium*, pages 706–711, 2005.

[5] J. Choi, J. Lee, D. Kim, G. Soprani, P. Cerri, A. Broggi, and K. Yi. Environment-detection-and-mapping algorithm for autonomous driving in rural or off-road environment. *IEEE Transactions on Intelligent Transportation Systems*, 13(12):974–982, 2012.

[6] N. Fairfield and C. Urmson. Traffic light mapping and detection. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 5421–5426, 2011.

[7] A. B. Hillel, R. Lerner, D. Levi, and G. Raz. Recent progress in road and lane detection: a survey. *Machine Vision and Applications*, 2012.

[8] J. Levinson, M. Montemerlo, and S. Thrun. Map-based precision vehicle localization in urban environments. In *Proceedings of Robotics: Science and Systems*, 2007.

[9] J. C. McCall and M. M. Trivedi. Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation. *IEEE Transactions on Intelligent Transportation Systems*, 7(1):20–37, 2006.

[10] M. Nieto, J. A. Laborda, and L. Salgado. Road environment modeling using robust perspective analysis and recursive bayesian segmentation. *Machine Vision and Applications*, 22:927–945, 2011.

[11] Y.-W. Seo and R. Rajkumar. Use of a monocular camera to analyze a ground vehicle's lateral movements for reliable autonomous city driving. In *Proceedings of IEEE IROS Workshop on Planning, Perception and Navigation for Intelligent Vehicles*, pages 197–203, 2013.

[12] Y.-W. Seo and R. R. Rajkumar. Utilizing instantaneous driving direction for enhancing lane-marking detection. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 170–175, 2014.

[13] L.-W. Tsai, J.-W. Hsieh, C.-H. Chuang, and K.-C. Fan. Lane detection using directional random walks. In *IEEE Intelligent Vehicles Symposium*, pages 303–306, 2008.

[14] U.S. Department of Transportation, Federal Highway Administration. *Manual on uniform traffic control devices for streets and highways*, 2009 edition, 2009. http://mutcd.fhwa.dot.gov/.

[15] E. A. Wan and R. van der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of IEEE Symposium on Adaptive Systems for Signal Processing, Communications, and Control*, pages 153–158, 2000.

[16] Y. Wang, E. K. Teoh, and D. Shen. Lane detection and tracking using b-snake. *Image and Vision Computing*, 22:269–280, 2004.

[17] T. Wu and A. Ranganathan. Vehicle localization using road markings. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 1185–1190, 2013.

[18] S. Zhou, J. Gong, G. Xiong, H. Chen, and K. Iagnemma. Road detection using support vector machine based on online learning and evaluation. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 256–261, 2010.