

CO-ORDINATE MAPPING AND ANALYSIS OF VEHICLE TRAJECTORY FOR ANOMALY DETECTION

Satyam Srivastava, Ka Ki Ng, Edward J. Delp

Video and Image Processing Laboratory (VIPER)
School of Electrical and Computer Engineering
Purdue University
West Lafayette, Indiana, USA

ABSTRACT

Behavioral information can be inferred from the trajectory of a rigid object. We propose a method to detect anomalies in the approach of a vehicle by observing the patterns in its velocity and describe methods for more effective analysis of the velocity trajectory. First we define a hypothetical co-ordinate system in which the axes are specified with respect to the road and distances are true ground measurements. In this co-ordinate system the (axial) velocity is a one dimensional quantity. We estimate the “normal” trends in the velocities by activity path modeling after scaling the velocities by the vehicle’s average speed. We detect an anomaly if a vehicle’s velocity does not fall in the available path models. Finally, we use the shape of the trajectory to determine turns and other significant maneuvers. We also detect an anomaly if a vehicle’s velocity falls in a path model which is inconsistent with the shape of its trajectory. We observe that our proposed co-ordinate system also improves trajectory shape analysis by suppressing false turns.

1. INTRODUCTION

Video surveillance of vehicles has emerged as the preferred technology for traffic monitoring, public safety, and law enforcement applications [1, 2, 3]. The range of behavioral analysis for vehicles is limited when compared with human subjects [4] due to rigid shapes and regularity of motion. Still, timely detection of anomalies can potentially prevent mishaps and damage to life/property [5]. Traditionally such anomaly detection is achieved using trajectory clustering and identifying deviations from the clusters [6, 7, 8]. Although the trajectory given by a tracker includes both position and velocity estimates, many approaches end up discarding the velocity changes which can be highly informative. Furthermore, path learning via clustering is affected by issues such as unequal trajectory lengths, choice of path models, and effects of distance metric.

We approach the task by hypothesizing that the velocity and shape of the trajectory can be analyzed separately and that in most situations, patterns in the velocity provide enough information to detect anomalies. Basharat *et al.* describe modeling of motion patterns which does not require clustering [9]. We illustrate this with an example of vehicles approaching a check point. Medioni *et al.* use a clustering-based method to detect if a vehicle tries to evade the check point [7]. However, their approach does not detect if the vehicle slows down or stops and then proceeds toward the check point or

picks up speed as it approaches indicating an intention of crashing through it. In contrast, a simple velocity differential [2] can detect these patterns.

We propose to form path models of velocities as functions of position rather than time. We also estimate the velocity models for finite sub-regions of the field of view. This is different from identifying sub-paths after clustering. In each such division, we represent the velocity with Gaussian distributions. The use of finite spatial divisions prevents the issue of unequal trajectory lengths. This allows an anomaly detection system to utilize the patterns in the velocity and also allows better modeling of the decision function than the simple thresholding used in [2]. These methods are described in Section 3.

Next we observe that detecting deviations from “normal” velocity would be easier if there is only one set of normal driving patterns. This is not true in general. For instance, a vehicle would slow down from its original speeds when making a turn or taking a highway exit. Thus, if a vehicle’s velocity matches a cluster corresponding to vehicles that made a turn, this vehicle should also be making a turn. We use template matching to identify significant maneuvers from the trajectory shape in Section 4. By separating the temporal and spatial analyses of the trajectory, we obtain better flexibility to define the rules of anomalous behavior.

The above analyses assume that vehicles are point objects moving in a 2D plane. In Section 2 we define a co-ordinate transformation in which the motion of the vehicle consists of two components – motion along the direction of the road and motion perpendicular to it. This transformation allows us to represent the velocity with a 1D descriptor (by ignoring the perpendicular component). It also improves the shape analysis by discarding the curvature caused by naturally curved roads.

In this paper, we describe the above analyses with a goal of achieving real-time operation. More specifically, we would like to process the state of the vehicle (as given by a tracker) as soon as it is received. Our methods are compared with other techniques which do not offer such operation capability. The results of testing our methods are provided in Section 5.

2. TRAJECTORY ESTIMATION AND CO-ORDINATE MAPPING

The tasks of robust object detection and tracking are central to many image analysis problems. Over the years many interesting methods have been proposed to detect moving objects and to track them. Haritaoglu *et al.* [10] and Stauffer and Grimson [11] describe popular methods for object detection and tracking. A comprehensive survey on tracking is provided by Yilmaz [12].

This paper is based upon work supported by the U.S. Department of Homeland Security’s VACCINE Center under Award Number 2009-ST-061-CI0001. Address all correspondence to E. J. Delp (ace@ecn.purdue.edu).

In this paper, we only provide some qualitative features of our tracking system and refer to appropriate publications for further details. Our system does object detection and tracking in a continuous manner. Foreground objects are detected by adaptive background subtraction as described in [13]. The method consists of adaptive computation of classification thresholds and background model updates at variable learning rates. The output of detection (foreground blobs) is characterized by connected components analysis and is used to create targets which are tracked using a particle filter framework as described in [14]. Particle filter is a Bayesian framework which is used to estimate the “state” of a target being tracked. In our system, the object’s state includes its position and size. The filter generates several hypotheses (particles) of the new state and computes the weighted average of these hypotheses as the estimate of the new state. The weights are computed by determining the similarity of the hypotheses with respect to some chosen object features. We use color and edge orientation as the objects features. Arulampalam *et al.* provide a detailed tutorial on the application of particle filters in object tracking [15].

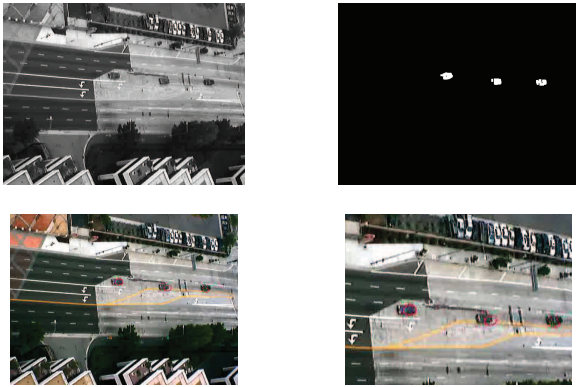


Fig. 1. An example of results produced by object detection and tracking – from left to right: original frame and the foreground mask (top), three tracked vehicles and a close-up of the scene (bottom).

Figure 1 shows a snapshot from one of our test sequences. The corresponding outputs from background subtraction and object tracking are also provided. For the rest of this paper, we assume that the vehicle’s position in the video frame is available. More precisely, a vehicle’s “image trajectory” is defined as its position in the frame as a function of time (or frame number), and is provided by the tracker. Let this image trajectory be represented by

$$\Phi_c^i = \{(x_t^i, y_t^i) : 1 \leq t \leq T_c\}, \quad (1)$$

where the subscript c denotes a particular object. For the sake of brevity, this subscript is not used in further discussion. Similarly, T_c is the lifetime of the object in number of frames, t is the time index, and the superscript i denotes image co-ordinates. In our convention x increases from left to right and y from top to bottom.

For any object, one would like to transform the trajectory into actual ground co-ordinates in order to remove the distortion due to camera perspective. Let this transformed trajectory be represent by Φ^g with definition similar to Φ^i . In our application scenario where an object can be modeled as a point, an ideal transformation would result in an aerial view of the trajectory. This can be achieved with a planar homography [16] or an approximation of the same. Our preferred method of perspective correction uses look-up tables because

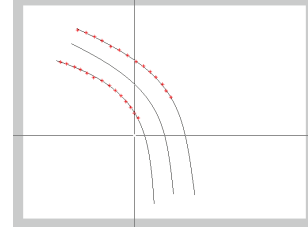


Fig. 2. An illustration of user specified co-ordinate transformation. The points marked with red crosses divide the “road” into equal distance segments.

it is fast and requires minimal calibration. However, from our experiments we have observed that ground co-ordinates are often not the most efficient representation for trajectory analysis. For example, vehicles moving along a naturally curved road traverse a curved trajectory in ground co-ordinates whereas their motion is a “straight line” with respect to the road.

We propose analyzing vehicle trajectories in a hypothetical co-ordinate system in which the distances are arbitrarily close to the distances in ground co-ordinates but the axes are defined with respect to the shape of the roads. Thus, a vehicle moving parallel to the center median is considered as moving along a straight line. This transformation results in two immediate benefits – (1) the vehicle’s velocity can be specified with a single component (in the direction of the road) and (2) false curve/turn detection along gently curved roads can be suppressed. Again, we utilize 2D look-up tables to achieve this transformation from ground to the hypothetical co-ordinate system. This table is constructed interactively by user input. This is explained next with an example.

Consider the situation in Figure 2 where the parallel curves represent the boundaries of a curved road. This image is assumed to be perspective corrected. We next assume that this corresponds to a road in the hypothetical space whose length and width matches the actual section of the road but which is not curved. We establish an approximate point correspondence by asking a user to divide the two curves into equal intervals (equal in the ground co-ordinates). Depending on the number of divisions, corresponding points are located on the hypothetical road. This completes the look-up table where the input is a position in the ground co-ordinates and the output is the corresponding position in the hypothetical system. For points not aligned with the input grid, 2D Shepard interpolation [17] is used. This look-up table is later used to transform the vehicle’s trajectory into the desired co-ordinate system.

Note that a single look-up table can be created to map the object position from the image co-ordinates directly to the hypothetical co-ordinates. That is, no explicit perspective correction step is required. This is done by taking the user input (or an equivalent method) to divide the road boundaries in the image co-ordinates. This is our recommended approach and has been used in our experiments. Let the final trajectory be represented as

$$\Phi^h = \{(x_t^h, y_t^h) : 1 \leq t \leq T\}, \quad (2)$$

where the position x^h is along the road and y^h is perpendicular to it.

3. VELOCITY ANALYSIS

We first introduce our proposed representation of the velocity as a function of the vehicle’s position. Note first that we only consider

the axial component of the velocity (parallel to the road) in this section. Let $x(t)$ and $v(t)$ represent the (axial) position and velocity estimates of a vehicle at time t . At this point we make the following key observations:

- Even in the absence of obstacles there is natural variation in “normal” driving speeds due to curves, turns, and inclinations/declinations. However there will not be many variations when looking at a small portion of the roadway. Therefore, patterns of normal velocities should be estimated for different portions of the road.
- A fast moving vehicle will have fewer samples in its trajectory while a slow moving vehicle will have more samples. Thus, the k^{th} sample of these vehicles’ trajectories would not correspond to the same portion of the road.
- Resampling of trajectories to get the same number of samples may solve this problem of mis-registration. For instance, if a vehicle A drives twice as fast as another vehicle B we can upsample A’s trajectory by a factor of 2. If the sampling time (inverse of the frame rate) is τ then the k^{th} sample in A’s resampled trajectory corresponds to time $k\tau/2$ (since A’s entry) while in B’s trajectory it represents time $k\tau$. However, their spatial position would be very similar due to the inverse effect of velocities.

The third observation motivated us to represent the velocities as function of position $u(x)$. This representation removes the need for resampling because the velocities of two vehicles at same position *are* comparable. To allow easier indexing we quantize the position x by dividing the road into M smaller segments. The velocity can be specified using the index of the segment in which x lies. Let the length of the road in the field of view be L (feet/meter) where we follow the convention that x increases from 0 to L as the vehicle gets closer to the observation point. We represent the segments by a set of points $\{s_0, s_1, \dots, s_M\}$ such that $s_0 = 0, s_M = L$, and

$$s_i - s_{i-1} = \frac{L}{M}, \quad \forall i \in \{1, 2, \dots, M\}. \quad (3)$$

With the length of the segments $\Delta = \frac{L}{M}$, the velocity $u(x)$ can be represented using $\bar{u}(i)$ where

$$i = \lfloor x/\Delta \rfloor \quad (4)$$

indicates the i^{th} segment and $\lfloor \cdot \rfloor$ denotes integer floor operation. The length Δ (or equivalently, M) would be chosen small (large) enough to prevent velocity behavior changes within a segment while not over-segmenting because that would discard spatial averaging.

In the final pre-processing stage, we propose to scale the velocities with respect to each vehicle’s average speed. This allows us to emphasize the *changes* in the velocity rather than the absolute speeds. While certain techniques (such as laser speed guns) allow more accurate estimation of velocities than video, such accuracy is not critical in our velocity analysis approach due to this scaling step. Further, it also leads to smaller variances in the normal behavior distributions which we describe next. Note that our goal is to achieve real-time operation. Thus, the average speed must be estimated with only part of the observations. We estimate the average speed u_{avg} over the velocities observed when $x < L/2$. This was motivated by the assumption that a visible change in behavior is more likely to occur when a vehicle is near the observation point (e.g. a check point). Hence it will exhibit more “typical” behavior when in the far field of view. The final form of the velocity is given by

$$c(i) = \bar{u}(i)/u_{avg}. \quad (5)$$

The normal velocity behavior for each segment i is modeled with a mixture of Gaussian distributions specified by the mean $\mu_{i,k}$ and variance $\sigma_{i,k}^2$ of vehicle velocities for the k^{th} Gaussian component. One can use clustering to obtain these parameters. We use a simpler context-based approach in which the number of Gaussian components is related to the number of maneuvers observed in the segment. For instance, a “straight-only” lane with no curvature and/or grade will require only one component while other sections may require components for turns and straight line course.

In the testing phase, the scaled velocity $c(i)$ of an oncoming vehicle is compared with the components in the i^{th} segment. An “anomaly” is detected if

$$|c(i) - \mu_{i,k}| > \alpha \cdot \sigma_{i,k} \quad \forall k. \quad (6)$$

Here α is a scalar that determines the threshold. Under Gaussian assumption, a value of $\alpha = 2$ would imply approximately 5% outliers. In the case when there exists some component such that the velocity difference is within the threshold, we validate the selection of this component by estimating the maneuver with shape analysis (described next). This ensures, for example, that if a vehicle’s velocity matches the component obtained for turning vehicles then that vehicle should also turn. A mismatch in the anticipated and observed maneuver would also result in an anomaly detection.

4. SHAPE ANALYSIS

We now describe our methods to characterize the shape of a vehicle’s trajectory. In this exercise our goal is to develop techniques to identify significant maneuvers like turns and lane changes. We construct a library of shapes associated with different maneuvers and match the vehicle’s trajectory to these templates. Note that we perform template matching in the hypothetical co-ordinates defined earlier and both x^h and y^h are used to define the vehicle position. In order to simplify discussion (and implementation) we use a complex number representation of the trajectory in this section.

Let a spatial trajectory be defined as:

$$P = \{p_t = x_t^h + jy_t^h : 1 \leq t \leq N\}, \quad (7)$$

where N is the number of samples in the trajectory and j denotes the square root of -1 . We obtain a trajectory template for each significant maneuver. Thus for k maneuvers we obtain the templates P_1, P_2, \dots, P_k .

4.1. Procrustes Analysis

According to Dryden [18], “shape is all the geometrical information that remains when location, scale, and rotational effects are filtered out from an object.” For analysis purposes we represent the shape of an object by a finite number of pixels on the object’s surface. There are different methods to estimate the similarity of object shapes. Procrustes shape analysis is one such method which is invariant to scale, translation, and rotation [19]. Our use of Procrustes analysis for turn detection is inspired by a similar work by Harguess and Aggarwal [20]. Our approach is different because we do not pre-segment the trajectories resulting in a real-time analysis.

Let each object be represented by a $N \times 1$ vector where N is the number of 2D points on the object’s surface (in our case, the number of points on the trajectory). We represent the two trajectories as $U = (U_1, U_2, \dots, U_N)^T$ and $V = (V_1, V_2, \dots, V_N)^T$ where $U, V \in \mathbb{C}^N$. That is, each 2D point is represented by a complex number as in Equation 7. The Procrustes distance is a minimum

sum of squared distances shape metric that requires shapes with one-to-one corresponding point pairs. After filtering the location, scale, and rotational effects the minimum Procrustes distance configurations that is used in this paper is defined by:

$$d_F(U, V) = \left(1 - \frac{V^* U U^* V}{U^* U V^* V}\right)^{1/2}. \quad (8)$$

4.2. Shape Matching with Sliding Windows

Our motivation for using a sliding window based method arises from the need for real-time operation. We observe that the approach used by Harguess [20] creates significant delay in the analysis because it requires the complete trajectory (or at least a large part of it) in order to detect steps, ramps, and impulses. Thus we devise a method to use template matching without a need for explicit segmentation of the trajectory.

Let us represent a window W as an index set defined by the two end points τ_1 and τ_2 :

$$W = \{\tau_1, \tau_1 + 1, \dots, \tau_2 : 1 \leq \tau_1 < \tau_2 \leq N\}. \quad (9)$$

We ensure sufficient samples in the windows by the constraint that $\tau_2 - \tau_1 > n_{min}$ ($n_{min} = 20$ in our experiments). The shape to be matched is constructed from the trajectory points located in the window. Thus,

$$\beta = \{p_t \in P : t \in W\}. \quad (10)$$

The shape descriptor thus obtained is a complex vector of length $\tau_2 - \tau_1 + 1$. In order to compute the Procrustes distance with respect to the k templates described above, two operations are needed. First, the vector is centered by subtracting the median value. Next, the centered vector must be resampled so that it contains exactly the same number of elements as the l^{th} template for each $1 \leq l \leq k$. The resampling is achieved using linear interpolation. Let $\hat{\beta}$ represent the centered and length-adjusted shape vector corresponding to a given window W .

Let D represent the Procrustes distances for $\hat{\beta}$ for each of the template. That is,

$$D = \{\delta_l : 1 \leq l \leq k\}, \quad (11)$$

where δ_l is the distance from the l^{th} template shape. We declare the occurrence of a particular maneuver when the distance of its template is significantly smaller than the other distances. From our experiments, we observed that a 25 – 35% difference works well. We use 30% as the decision threshold in our experiments. An event is declared by marking p_{τ_2} as the position of occurrence.

Since the only two parameters determining a window are the end points, we now explain how these points get updated.

- If a turn is detected, all but the very latest parts of the current window are discarded, and a new window is created with the smallest permissible size (n_{min}). Thus,

$$\tau_1^{new} = \tau_2^{old} - \frac{n_{min}}{2} \text{ and } \tau_2^{new} = \tau_2^{old} + \frac{n_{min}}{2}, \quad (12)$$

where the superscripts *new* and *old* have been inserted for illustration purpose.

- If a turn is detected and is the same type as the last detected turn, a u-turn is inferred if the two events occurred within a chosen distance from each other. We declare a u-turn if two identical turns occur within 50 samples of each other. The end points are updated as earlier.

- If the event detected is a straight line course or no event is detected, only τ_2 is modified – increased by one if the end of the trajectory has not been reached. The analysis terminates when the end is reached.
- If the event detected is a straight line course or no event is detected, one can optionally choose to discard the oldest points in the window if the window exceeds a certain size threshold. We discard the older half of the samples if the window becomes larger than 100 samples.

5. EXPERIMENTAL RESULTS

The results are provided under three categories – evaluation of the co-ordinate transformation technique, evaluation of the velocity analysis method, and the evaluation of turn identification using shape analysis. Three video sequences were used in our experiments. These included the (publicly available) Lankershim Boulevard, California dataset generated for the NGSIM project [21] (denoted by LANK in this section) and two videos recorded by the authors (denoted by ANON1 and ANON2). A snapshot of the video sets is provided in Figure 3. In order to illustrate the role of the hypothetical co-ordinate system, we also provide a satellite image of the road in the field of view, using Google Maps [22]. Due to the nature of our outputs, the figures in this section are best viewed in color.

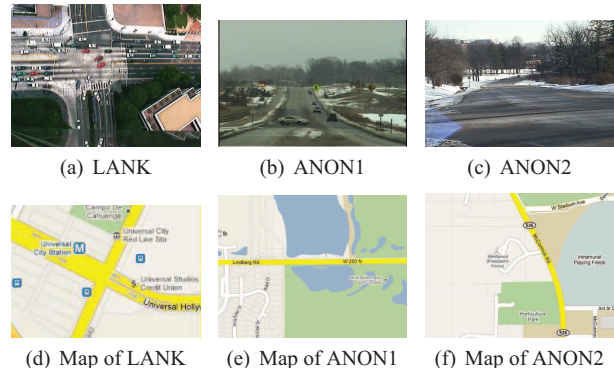


Fig. 3. Sample images from our test videos (a-c) and the corresponding map (d-f) (courtesy: Google Maps [22]). The field of view is highlighted in yellow.

We present two cases in which the proposed look-up table based method has been used on curved trajectories. These are shown in Figure 4. The first case is a synthetic trajectory created for the curved road used to illustrate the training process in Section 2. The second case is a trajectory from the ANON2 dataset. The linearized trajectories (in the hypothetical co-ordinate system) are shown in the second row. Note that in the second case, a single 2D LUT is used to transform from the image co-ordinates to the desired co-ordinates. The perturbation seen in the upper part of the transformed trajectory (in case 2) is due to the sensitivity of the transformation very deep in the field of view. This error can be improved by better user input or with alternative camera calibration methods although the non-linearity will always cause greater errors in the far field of view.

Our method for analyzing the approach velocity of a vehicle was used with the ANON1 dataset. Different scenarios were realized by driving a designated vehicle in various manners. In Figure 5, we present the decision vectors corresponding to four cases – driving

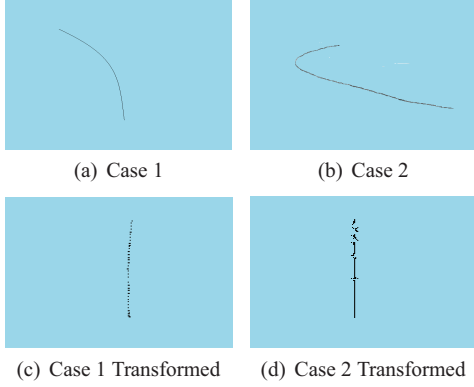


Fig. 4. Results of the LUT-based co-ordinate transformation to curved trajectories. Case 1 pair is a synthetic test case while the Case 2 pair is from the dataset ANON2.

at nearly uniform velocity, unexpected slowing, unexpected sudden acceleration, and making a u-turn. Recall that we treat velocity as a one-dimensional signal (in the direction of the road) as a function of the position. Thus, the vertical axis represents the scaled velocity at different points in the roadway. The blue circles indicate normal approach while the red crosses denote (atomic) anomalies. The dashed black line represents the mean (scaled) velocity estimated from the “normal” vehicles. It is clear to see that this treatment and the resultant method for visualization of the decisions would enable very fast decision-making by an operator.

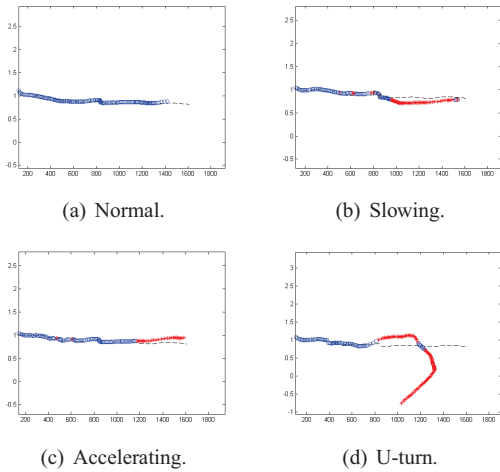


Fig. 5. Plot of the decision vectors corresponding to different driving behaviors. The red crosses indicate occurrence of atomic anomalies.

Shape analysis for detecting turns was used on both synthetic and true trajectories. Figure 6 shows the examples of template matching output on four types of maneuvers. These trajectories were obtained by tracking individual vehicles in the LANK dataset [21]. The black circle indicates the starting end of the trajectory (since there is no temporal information in these plots). Right turns are denoted by red triangles and left turns by blue squares. The color of a u-turn marker is chosen based on the turn being left-handed or

right-handed. A total of 46 vehicle trajectories were analyzed and 45 maneuvers were correctly identified. These cases included vehicles from East-West and North-South traffic and roughly equal number of right, left, and no turns. The u-turn case in Figure 6 was the only such maneuver in the dataset. The failure case occurred when a vehicle made a turn after prolonged delay and our shape matching method did not detect the turn.

A synthetic trajectory was generated to simulate complex multi-turn scenarios to test the effectiveness of our methods. Figure 7 shows the result of turn detection on the synthetic trajectory. It can be seen that our methods not only detect all the turns but also declare the occurrence of turns at a logically correct point in the trajectory.

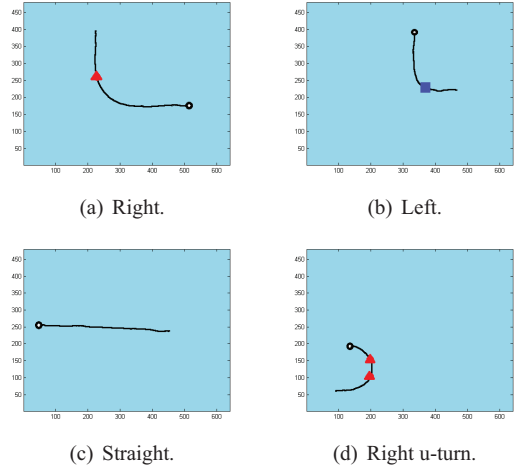


Fig. 6. Results of shape analysis on trajectories of different maneuvers from the LANK dataset. A black circle is used to denote the starting end of the trajectory.

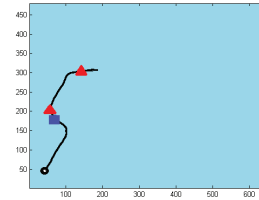


Fig. 7. Result of shape analysis on a synthetic multi-turn trajectory.

Finally, we demonstrate the usefulness of the hypothetical co-ordinate system by using shape matching for the two test cases shown in Figure 4. The window method was used to detect turns in the original (ground) and the transformed trajectories (in the hypothetical co-ordinates). The results are shown in Figure 8. Clearly, the linearization enforced by the co-ordinate transformation is effective in suppressing the false turn detections. Note that Figures 6, 7, and 8 have been enhanced to highlight the shape detection output.

6. CONCLUSIONS

In this paper we described novel methods for representing and analyzing a vehicle’s trajectory. We proposed a look-up table based

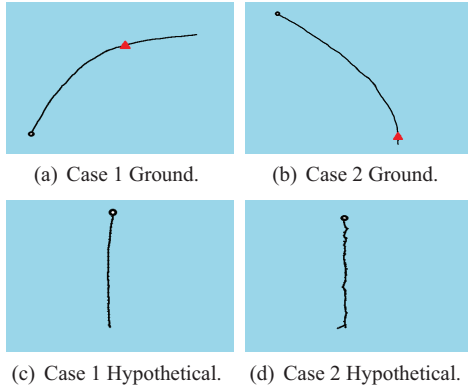


Fig. 8. Window method for turn detection applied to curved roads. The rows represent output of turn detection in ground (a-b) and hypothetical (c-d) co-ordinates. Note that false turns are detected only in the ground co-ordinates.

method for mapping the trajectory from image co-ordinates to a hypothetical co-ordinate system in which the axes are selected with respect to the road. We then proposed a spatial representation of the velocity which helped localize the normal behavior definition. We detect deviations from the normal velocity for individual sections of the roadway where the normal velocity was modeled using a mixture of Gaussians (obtained by training). Next, we characterized the shape of the trajectory using template matching in order to detect turns and other maneuvers. Template matching was performed using sliding windows which does not require pre-segmentation of the trajectory. Thus it is very suitable for real-time operation. We provide results of testing our methods on real traffic videos. This work can be extended by comparing with more existing techniques for behavioral analysis and by studying the effects of road segment sizes (or number) on the trajectory modeling.

7. REFERENCES

- [1] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transportation Research: Part C*, vol. 6, no. 4, pp. 271–288, August 1998.
- [2] S. Srivastava and E. J. Delp, "Standoff video analysis for the detection of security anomalies in vehicles," *Proceedings of the IEEE Applied Imagery Pattern Recognition Workshop*, Washington, DC, October 2010.
- [3] L. Dlagnekov, "Video-based car surveillance: License plate, make and model recognition," Masters Thesis, University of California, San Diego, 2005.
- [4] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, vol. 34, no. 3, pp. 334–352, August 2004.
- [5] W. Hu, X. Xiao, D. Xie, and T. Tan, "Traffic accident prediction using vehicle tracking and trajectory analysis," *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, Shanghai, China, October 2003, pp. 220–225.
- [6] B. T. Morris and M. M. Trivedi, "A survey of vision-based trajectory learning and analysis for surveillance," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 8, pp. 1114–1127, August 2008.
- [7] G. Medioni, I. Cohen, F. Bremond, S. Hongeng, and R. Nevatia, "Event detection and analysis from video streams," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 8, pp. 873–889, August 2001.
- [8] T. Zhang, H. Lu, and S. Li, "Learning semantic scene models by object classification and trajectory clustering," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Miami, Florida, June 2009, pp. 1940–1947.
- [9] A. Basharat, A. Gritai, and M. Shah, "Learning object motion patterns for anomaly detection and improved object detection," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, June 2008, pp. 1–8.
- [10] I. Haritaoglu, D. Harwood, and L. S. Davis, "W4: Real-time surveillance of people and their activities," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 809–830, August 2000.
- [11] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, August 2000.
- [12] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, December 2006.
- [13] S. Gupte, O. Masoud, R. F. K. Martin, and N. P. Panikolopoulos, "Detection and classification of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 37–47, March 2002.
- [14] K. K. Ng and E. J. Delp, "Object tracking initialization using automatic moving object detection," *Proceedings of SPIE/IS&T Electronic Imaging: Visual Information Processing and Communication*, vol. 7543, San Jose, California, January 2010.
- [15] M. S. Arulampalam, S. Maskell, and N. Gordon, "A tutorial on particle filters for online nonlinear/non-Gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, February 2002.
- [16] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. New York: Cambridge University Press, 2004.
- [17] D. Shepard, "A two-dimensional interpolation function for irregularly-shaped data," *Proceedings of the ACM National Conference*, Princeton, New Jersey, January 1968, pp. 517–524.
- [18] I. L. Dryden and K. V. Mardia, *Statistical Shape Analysis*. New Jersey: Wiley, 1998.
- [19] J. C. Gower and G. B. Dijksterhuis, *Procrustes Problems*. Oxford, UK: Oxford University Press, 2004.
- [20] J. Harguess and J. K. Aggarwal, "Semantic labeling of track events using time series segmentation and shape analysis," *Proceedings of the IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009, pp. 4317–4320.
- [21] "The Next Generation Simulation Community," Online: <http://ngsim-community.org>, Accessed: August 2010.
- [22] "Google Maps," Online: <http://maps.google.com>, Accessed: August 2010.