A Correlation Technique for Estimating Traffic Speed from Cameras

Todd N. SCHOEPFLIN and Daniel J. DAILEY

Abstract—This paper presents a new algorithm to estimate mean vehicle speeds on a per-lane basis from sequences of roadside camera images. The algorithm is suitable for both congested and uncongested traffic levels as well as a variety of lighting and weather conditions. Individual vehicle lanes are identified and horizontal vehicle features are emphasized using a gradient operator. The features are projected into a one-dimensional subspace and transformed into a linear coordinate system using a simple camera model. A correlation technique is used to summarize the movement of features through a group of images and estimate mean speed for each lane of vehicles.

Index Terms—Video, speed sensor, vehicle tracking, cross-correlation methods, image processing.

INTRODUCTION

This paper presents a new algorithm to estimate mean vehicle speeds on a per-lane basis from sequences of roadside camera images. The algorithm is suitable for both congested and uncongested traffic levels as well as a variety of lighting and weather conditions. This algorithm can be used with any series of roadway images that have an inter-frame delay of 0.3 seconds or less. The only underlying assumptions are that the only moving objects in the images are vehicles, that the traffic moves largely toward or away from the camera, and that the roadway is approximately straight in the lower one-third of the image.

The algorithm, presented pictorially in Figure 1, consists of a set of image processing steps followed by an averaging process to map the vehicle features from a lane of vehicles into a one-dimensional space. The algorithm uses parameters from the camera model to transform the one-dimensional feature vector and thereby place features in the foreground and far field on the same scale. After creating a feature vector for each image in a sequence, the temporally adjacent vectors are cross-correlated to determine the distance moved between frames. In this way the features from all of the vehicles in the front one-third of the image are used to estimate the mean traffic speed. This approach takes advantage of a variety of features from many vehicles to improve the speed estimate and as a result it works well in uncongested, congested, and stop-and-go traffic.

There are several approaches to tracking vehicles in images published in the literature, many of which focus on tracking individual vehicles. The simplest method is to identify the background and subtract it from the current image, revealing vehicle blobs of interest ((1), (2)). Such approaches fail when the algorithm cannot identify individual vehicles due to occlusion, e.g., congested traffic. The work of (3) extracts vehicle corner features, tracks the features through the image sequence, and groups the features into vehicle tracks using spatial and velocity constraints. Whenever a portion of the vehicle is occluded this approach is more robust than methods that track the entire vehicle. It also seems to offer reasonable performance at night when only the headlights and taillights of the vehicle are visible. However, it has difficulty tracking features when the object moves a large distance between frames, and the feature grouping can become computationally expensive when many vehicles are present in the scene. Other researchers have accumulated information from the sequence in a spatiotemporal image ((4), (5), (6)) or volume (7), which is then used to extract velocity information about the object. While this approach does perform velocity estimation, it is unclear how it might handle congested traffic scenes. In yet another approach, researchers (8) advocate use of the velocity Hough transform to simultaneously extract the velocity and position of vehicles. However, this method probably cannot handle images containing many objects. In addition, fitting this method to the problem of vehicle tracking would require a perspective warp of the image followed by examination of each pixel in a given lane to accumulate the Hough variables. This is a computationally expensive proposition, particularly when the velocity range is unknown.
Only one approach currently in the literature shares significant common ground with the research presented below. In (9), the authors describe the processing of an omnidirectional image to enable a mobile robot to navigate. Their algorithm transforms the image into polar coordinates, obtains an edge image, and projects the edge image, creating a one-dimensional signal. Their primary interest is in tracking each spike in the one-dimensional signal using a correlation method in order to estimate the position of the robot within the scene. Thus, they perform frame-by-frame feature tracking of individual features, whereas the current approach uses correlation to integrate information over the entire scene to overcome the problems of tracking individual vehicles in congested traffic. The algorithm now presented is suitable for an automated system and it is computationally inexpensive.

**IMAGE PROCESSING**

In the initial steps of the work presented here, a series of image processing algorithms are applied to a sequence of images. The first step in the algorithm of Figure 1 is to analyze the scene to locate the roadway using vehicle motion cues. To do this, an activity map image is obtained from the video sequence by subtracting each of several hundred median-filtered images from its immediate temporal neighbors. After binarizing the image difference using an automatic threshold, the nonzero pixels are accumulated. This step highly resembles a concept presented in (10) where the authors describe how to generate an activity map and use it to obtain very rough lane masks using morphological operators. The sample activity map for a normal traffic scene in Figure 2 shows that the activity values increase toward the top of the image because of the reduced space between vehicles. Horizontal variations in the activity map indicate the middle of the lanes and their boundaries.

Two important pieces of information are extracted from the activity map generated by the process described in Figure 1: (1) an estimate of the horizon line and (2) individual masks for each lane of the roadway. Both of these objectives are achieved in the following way. Observing the line structure present in the image, line templates are used to sample the activity image values at different locations and orientations. In (11), we showed that rigid affine templates are an effective way to search for and extract known boundaries in an image. In the present research, the templates are simple lines centered at $v = v_c$, chosen about one-third up the image; orientations are sampled in increments of one degree. For each horizontal position $u_c$, the orientation with the largest average activity map value is recorded. Figure 3 presents the intermediate results showing the best lines at each location for a scene where water is on the camera lens. Clearly, many of the lines are invalid. For example, lines that intersect the bottom of the activity map image where the image has a small value, should be removed. Lines that are between lanes or outside the main roadway area are also to be removed. Lines where the activity map image has a low value at the centroid $(u_c, v_c)$, relative to values at the other centroids where $v = v_c$, are also removed. After removing the lines just mentioned, the remaining lines are grouped by lane using the gaps in the horizontal position $u_c$. Next, the vanishing point is estimated by finding the intersection of each line with all lines outside its group; internal intersections are discarded due to the small angle between the lines. The average coordinates of the intersection point of the lines estimates the vanishing point $(u_{vp}, v_{vp})$ for all lines parallel to the roadway.

Next, the algorithm of Figure 1 generates lane masks from the activity map by drawing a line from the vanishing point through the centroid of the best template in each group. The $u$-coordinate for each of these lines on the bottom row of the image is recorded as an estimate of the centroid of each lane. Linear interpolation is then used to estimate the $u$-coordinate of the lane boundaries on the bottom row. Finally, the lane masks are generated by connecting two lane boundary points on the bottom row with the vanishing point. Figure 4 superimposes these lines on an image in a challenging rainy scene where water drops lie on the camera lens. This method is fairly effective at isolating the lanes of a road in the bottom half of the image, even if the road is somewhat curved.

The lane masks then make it possible to obtain the vehicle features belonging to a particular lane of traffic. The algorithm of Figure 1 generates these features by emphasizing the horizontal vehicle lines using a standard technique from (12). This “Gradient Edge Operator” block consists of a vertical derivative-of-Gaussian FIR filter with $\sigma = \sqrt{2}$ and a kernel length of 11, i.e., [-2, -19, -82, -191, -202, 0, 202, 191, 82, 19, 2]T. Similarly, the image is horizontally low-pass-filtered with a Gaussian FIR filter of [1, 9, 55, 191, 403, 518, 403, 191, 55, 9, 1]T. Applying the mask using the “AND” operator in Figure 1 for a given lane isolates horizontal edge features as illustrated in Figure 5. These feature values are then averaged horizontally across the length of the mask, resulting in a one-dimensional signal along the $v$-axis. Stated more precisely, this averaging must occur in a direction orthogonal to the lane of traffic. In typical scenes where the camera is aimed approximately in the direction of the traffic, this orthogonal direction is essentially horizontal in the image. Although features from the occasional tall vehicle in an adjacent lane may overlap the lane of interest, the averaging used by the tracking process greatly reduces this effect. The following two sections describe how the remaining steps from the algorithm in Figure 1 remove the signal’s nonlinear scale and correlate it with other signal samples to estimate the traffic speed.
PROJECTION AND SCALING

The camera re-scales the 3D spatial information from the roadway when capturing 2D images for processing by the algorithm in Figure 1. To place the features throughout the image on a single scale, a transformation from the image plane to the 3-D world is necessary. The “Inverse Perspective Projection Transformation” block in Figure 1 is defined by the inverse of the camera model that projects 3-D objects onto the image plane. The scene is modeled as a road viewed through a pinhole camera as shown in Figure 6, assuming that the camera is located at a height $h$ above the ground plane and a perpendicular distance $d$ from the road. The camera is oriented at a pan angle $\theta$ and tilt (down) angle $\phi$ such that a point in the earth system ($X, Y, Z$) is transformed into the camera coordinate system ($X_c, Y_c, Z_c$) by only a translation and a rotation. The camera is oriented along the negative $Z_c$ axis (hence the negative sign associated with the focal length) and its line of sight intersects the ground plane a distance $F = h \csc(\phi)$ away. This model is accurate when the road can be modeled as a plane and the rotation of the ground plane about the $Z_c$ axis is negligible, i.e. no roll. The following equations describe the perspective projection of points in the pinhole camera’s coordinate system onto the image, given a focal length $f$:

$$
\begin{align*}
    u &= -f \frac{X_c}{Z_c} \\
    v &= -f \frac{Y_c}{Z_c}
\end{align*}
$$

Lai and Yung (13) presented an invertible 2-D to 3-D coordinate transformation. The work presented here derives a similar transformation whereby the (centered) image coordinates $(u, v)$ of points lying on the ground plane, in Figure 6, are transformed into their real-world coordinates $(X, Y, 0)$. This paper extends (13) by including a scale factor that relates world coordinates to image coordinates (meters per pixel). Starting with the ground-plane $X$-$Y$-$Z$ coordinate system, expressions for the $X_c$-$Y_c$-$Z_c$ coordinates are developed. First, the $U$-$V$-$W$ system is obtained by rotating an angle $\phi$ around the $X$-axis:

$$
\begin{align*}
    U &= X \\
    V &= Y \cos(\phi) - Z \sin(\phi) \\
    W &= Y \sin(\phi) + Z \cos(\phi)
\end{align*}
$$

However, these expressions may be further simplified (since $Z = 0$) because objects are assumed to lie on the ground plane. Next, there is a displacement $F$ to obtain camera-centered coordinates $X_c$-$Y_c$-$Z_c$.

$$
\begin{align*}
    X_c &= U = X \\
    Y_c &= W = Y \sin(\phi) \\
    Z_c &= -V - F = -Y \cos(\phi) - F
\end{align*}
$$

Applying Eq. (1) yields

$$
\begin{align*}
    u &= -f \frac{X_c}{Z_c} = -f \frac{X}{-Y \cos(\phi) - F} \\
    v &= -f \frac{Y_c}{Z_c} = -f \frac{Y \sin(\phi)}{-Y \cos(\phi) - F}
\end{align*}
$$

Solving for $X$ and $Y$ yields

$$
\begin{align*}
    X &= S \frac{u \cdot K}{K - v} \\
    Y &= S \frac{v \cdot K}{\sin(\phi) K - v}
\end{align*}
$$

where $S = \frac{h}{K} \sec(\phi)$ meters/pixel is a scale factor that corrects Lai and Yung’s result for real-world coordinates and $K = f \tan(\phi)$ is the $v$-coordinate of the vanishing line (also known as the horizon line) for the $X$-$Y$ ground plane in the image. That is, any arbitrary pair of parallel lines on the ground plane intersects at a point that lies on the line $v = K$.
in the image. It can be shown that points infinitely far from the camera map onto the line \( v = K \) in the image. Examining Eq. (5) and (6) closely also shows that it is possible to estimate both \( X \) and \( Y \) (within different scale factors) by extracting the horizon line in the image.

Two typical one-dimensional signals \( a_1(v) \) and \( a_2(v) \) are shown in Figure 7(a). They come from sequential images and were obtained by horizontally averaging the image features according to the block diagram of Figure 1. As the vehicles travel, their apparent velocity in the image plane changes due to the nonlinearity of the perspective projection transformation of Eq. (1). Using the techniques of the previous section to obtain \( K \), the signal \( a(v) \) is warped into \( b(y) \) using Eq. (6) and a cubic polynomial resampling is applied to \( b(y) \) to obtain uniform sampling through the range of \( y \). A typical result is contained in Figure 7(b), where the values for \( S \) and \( \phi \) are estimated using the methodology outlined in (14).

CROSSCORRELATION AND SPEED ESTIMATES

Using the results above, a one-dimensional signal \( b(y) \) with a linear scale can be obtained. Next, the algorithm of Figure 1 obtains a velocity estimate by using the cross-correlation between \( b(y) \) in sequential image frames. The peak in the cross-correlation function of \( b(y) \) provides an estimate of the mean spatial shift of the vehicles that occurs in the known time interval between the images. Specifically, given de-meaned signals \( b(y, t_1) \) and \( b(y, t_2) \), the circular cross-correlation function is found by computing

\[
R_{t_1,t_2}[y_R] = \frac{\mathcal{F}\{b(y, t_1)\} \mathcal{F}\{b(y, t_2)\}}{\sigma_{t_1} \sigma_{t_2}}
\]

(7)

where \( \mathcal{F}(\cdot) \) is the discrete Fourier transform, \( \ast \) denotes complex conjugation, \( \sigma_{t_1} \) is the standard deviation of \( b(y, t_1) \) and \( \sigma_{t_2} \) is the standard deviation of \( b(y, t_2) \). If tall vehicles from an adjacent lane noticeably affected the feature vector, one would expect to find two peaks in \( R_{t_1,t_2}[y_R] \). The mean spatial shift is estimated by averaging \( R_{t_1,t_2}[y_R] \) across many time samples as follows

\[
R_{\text{mean}}[y_R] = \frac{1}{N} \sum_{i=1}^{N} R_{t_1,t_2}[y_R]
\]

(8)

The vehicles are assumed to be moving with a constant speed over some short period of time, i.e., a stationary process. The error in the estimate of the cross-correlation function of a stationary process is proportional to \( \sqrt{1/N} \). By assuming ergodicity it is then possible to improve the estimate of the cross-correlation function by time-averaging across a number of realizations as described in Eq. (8). Figure 8 illustrates a typical result for \( R_{\text{mean}}[y_R] \) when \( N = 100 \) in a scene containing free-flowing traffic under sunny conditions. Fitting a parabola to the points surrounding the peak of \( R_{\text{mean}}[y_R] \) enables the analytic calculation of the location of the peak of the cross-correlation function that is an estimate for the mean spatial shift \( y_m \). The algorithm of Figure 1 scales the abscissa of Figure 8 to convert the cross-correlation lag into speeds using the known time sampling interval and the geometry of Figure 6

\[
speed = \frac{y}{\cos \theta} \cdot \frac{5 \text{ samples}}{3600 \text{ sec/hour}} = \frac{5280 \text{ feet/mile}}{sec}
\]

(9)

Applying this formula to the point where the maximum of \( R_{\text{mean}}[y_R] \) yields an average speed of 63 MPH. Wire loop speed trap data obtained from a section of freeway nearby at about the same time yielded an average speed of 67 MPH.

This classic cross-correlation method offers several advantages over single-vehicle velocity estimates. First, the ensemble averaging automatically integrates all the information available from all \( N \) frames in a natural manner. Second, the relative value of the peak in \( R_{\text{mean}}[y_R] \) indicates the goodness of the mean-shift estimate. As typical of other cross-correlation results, values above 0.7 are reliable whereas those below 0.5 are questionable. Third, cross-correlating in the vertical direction in the image is useful because the road contains very few horizontal features that are stationary in the scene (i.e., a shift of \( y_R = 0 \)). Fourth, the method is very computationally efficient because it involves at most a separable 2-D convolution in a subset of a 320 X 240 image, the application of an image mask, and two 1-D FFTs for each frame. This makes it suitable for real-time applications.
EXPERIMENTAL RESULTS

The methodology described above is applied to the two rainy scenes of Figure 9 in order to characterize its performance under adverse conditions. Figure 10 illustrates the cross-correlation functions used to obtain the results. The mean speeds estimated over 20 seconds are 55 MPH and 28 MPH, respectively. For reference, data from wire loop speed traps located nearby yielded average speeds of 65 MPH and 34 MPH. This bias is probably due to errors in the camera calibration since errors in the correlation process should be eliminated by the averaging in Eq. (8). However, comparing the tracking results to those obtained by manually tracking the vehicles will be the only way to identify the source of the bias with certainty. Further study into the tracker’s behavior in different weather and traffic conditions is also warranted, though the positive results from the difficult test conditions thus far are very encouraging.

All three of the scenes (i.e., the sunny scene used to describe the algorithm and the two rainy scenes just mentioned) can create difficulties for tracking algorithms. For example, a sunny scene with free-flowing traffic may prove too difficult for single-vehicle trackers if the shadows confuse the tracker or if the vehicle moves too far between frames. Second, congested scenes are particularly troublesome because it is nearly impossible for the tracker to consistently distinguish individual vehicles and track them through the scene. Thirdly, rainy scenes prove challenging for typical background-subtraction methods because they have low contrast between the vehicles and the road. In some cases, raindrops may even fall on the camera lens, distorting a portion of the image for minutes or even hours. Extracting the lane masks from scene (a) with water drops on the camera lens proved to be the most difficult part in estimating the vehicle speeds. However, the cross-correlation method proved quite robust and yielded a reasonable speed estimate as in Figure 10(a). Scene (b) contains a smoothly-moving high-occupancy vehicle lane on the right whose results are presented in Figure 10(b) and two normal lanes that are slowing to a standstill. Estimating the individual peaks for each $R_{i,j,k+1}[y]$ yields the time sequence for the leftmost lane as presented in Figure 11. The cars transition from a stop-and-go state to a near stand-still, consistent with the plot. One can see that even without the power of time-averaging, individual cross-correlation results can give a reasonable indication of the current traffic dynamics. This is particularly true when many vehicle features are present in the lane mask, i.e., congestion is worst, because the overall lag is obtained from the spatial average of all the individual feature lags.

CONCLUSION

The algorithm presented here provides a speed estimate from uncalibrated cameras under a variety of traffic and weather conditions. The algorithm has three basic steps: (1) it extracts masks for each lane of traffic, (2) it uses this mask to create a one-dimensional signal, and (3) it cross-correlates signals in adjacent images to estimate the mean vehicle translation between frames for a given lane. In the experiments presented here, the robustness of this new tracking method is demonstrated with a wide variety of conditions expected in a roadway scene. This robustness is largely due to its basis in the cross-correlation signal processing technique. The results shown here outperform any other vehicle-tracking algorithm to date when traffic is heavily congested. Thus, this approach shows a great deal of promise for vehicle tracking in general, regardless of weather or road geometry.
REFERENCES
FIGURE 1 New algorithm for vehicle tracking and speed estimation.
FIGURE 2 Extracted activity map from 1,000 roadway images.
FIGURE 3 Activity map with superimposed orientation lines for a scene with raindrops on the camera lens.
FIGURE 4 Lane boundaries found for a scene with raindrops on the camera lens.
FIGURE 5 Horizontal edge features found in an uncongested scene.
FIGURE 6 Camera and roadway geometry.
FIGURE 7 Sample one-dimensional signals obtained from horizontal edge features in an uncongested lane of traffic.  a) Signal computed the image.  b) Signal after nonlinear warping according to Eq. (6).
FIGURE 8 Cross-correlation function obtained by averaging the cross-correlation functions of adjacent samples from an ensemble (N = 100) of signals. Scaling according to Eq. (9) easily converts the lag distance to speed. Peak occurs at 63 MPH.
FIGURE 9 Difficult scenes for vehicle tracking. a) Rainy day with free-flowing traffic and water drops on the camera lens. b) Rainy day with two congested lanes and one free-flowing lane.
FIGURE 10 Average of 100 cross-correlation functions (20 sec. interval):  

a) Free-flowing leftmost lane of Figure 9(a); peak occurs at 55 MPH.  
b) Free-flowing rightmost lane of Figure 9(b); peak occurs at 28 MPH.
FIGURE 11 Instantaneous velocity for the middle lane of Figure 9(b) obtained by finding the lag of the maximum for each cross-correlation function.