A Real-time Computer Vision System for Measuring Traffic Parameters *

David Beymer, Philip McLauchlan, Benn Coifman, and Jitendra Malik
Dept. of Electrical Engineering and Computer Sciences
University of California
Berkeley, California, 94720-1776
E-mail: {beymer, pm, zephyr, malik}@cs.berkeley.edu

Abstract

For the problem of tracking vehicles on freeways using machine vision, existing systems work well in free-flowing traffic. Traffic engineers, however, are more interested in monitoring freeways when there is congestion, and current systems break down for congested traffic due to the problem of partial occlusion. We are developing a feature-based tracking approach for the task of tracking vehicles under congestion. Instead of tracking entire vehicles, vehicle sub-features are tracked to make the system robust to partial occlusion. In order to group together sub-features that come from the same vehicle, the constraint of common motion is used. In this paper we describe the system, a real-time implementation using a network of DSP chips, and experiments of the system on approximately 44 lane hours of video data.

1 Introduction

Traffic management and information systems must rely on a system of sensors for estimating traffic parameters in real-time. Currently, the dominant technology for this purpose is that of magnetic loop detectors, which are buried underneath highways to count vehicles passing over them. Video monitoring systems promise a number of advantages. First, a much larger set of traffic parameters can be estimated in addition to vehicle counts and speeds. These include vehicle classifications, link travel times, lane changes, rapid accelerations or decelerations, queue lengths at urban intersections, etc. Second, cameras are less disruptive and less costly to install than loop detectors, which require digging up the road surface.

For some years, our group has been developing a prototype vision-based traffic surveillance system [11, 12]. The core idea is to have video cameras mounted on poles or other tall structures looking down at the traffic scene. Video is captured, digitized, and processed by onsite computers, and then transmitted in summary form to a Transportation Management Center (TMC) for collation and computation of multi-site statistics such as link travel times. Processing occurs in three stages:

1. Segmentation of the scene into individual vehicles and tracking each individual vehicle to refine and update its position and velocity in 3D world coordinates, until it leaves the tracking zone.
2. Processing the track data to compute local traffic parameters including vehicle counts per lane, average speeds, lane change frequencies, etc. These parameters, together with track information (time stamp, vehicle type, color, shape, position), are communicated to the TMC at regular intervals.
3. At the TMC, local traffic parameters from each site are collated and displayed as desired, and/or used in controlling signals, message displays, and other traffic control devices. Computers at the TMC also process the track information from neighboring camera sites to compute long-distance parameters such as link times and origin–destination counts.

In this paper, we focus on the first two stages, the vehicle segmentation and tracking stage and the computation of traffic parameters from the tracking data.

2 Tracking Approach

Tracking moving objects in video streams has been a popular topic in the field of computer vision in the last few years; earlier contributions to the areas of multi–target tracking and data association were made by control and aerospace engineers. Our application entails several stringent requirements for a proposed scheme:

1. Automatic segmentation of a vehicle from the background and other vehicles so that there can be a unique track associated with each vehicle.
2. Deal with variety of vehicles – motorcycles, passenger cars, buses, construction equipment, trucks, etc.
3. Deal with range of traffic conditions – light midday traffic, rush-hour congestion, varying speeds in different lanes.
4. Deal with variety of lighting conditions – day, evening, night, sunny, overcast, rainy days.
5. Real-time operation of the system.

Even though a number of commercial systems for traffic monitoring have been introduced recently, many of these criteria still cannot be met. In a recent evaluation of a group of these commercial systems [4], problems were reported with congestion, long shadows linking together vehicles, and the transition between night and day.

In the computer vision literature, the different tracking approaches for video data can be classified as follows.

*This research was supported by FHWA through a contract monitored by JPL, and by Caltrans/PATH (MOU 152 and MOU 214).

2.1 3D Model based tracking

Three-dimensional model-based vehicle tracking systems have previously been investigated by several research groups, the most prominent being the groups at Karlsruhe [10] and at the University of Reading[1, 15]. The emphasis is on recovering trajectories and models with high accuracy for a small number of vehicles. The most serious weakness of this approach is the reliance on detailed geometric object models. It is unrealistic to expect to be able to have detailed models for all vehicles that could be found on the roadway.

2.2 Region based tracking

The idea here is to identify a connected region in the image – a "blob" – associated with each vehicle and then track it over time using a cross-correlation measure. Initialization of the process is most easily done by the background subtraction technique. A Kalman filter-based adaptive background model[8, 9] allows the background estimate to evolve as the weather and time of day affect lighting conditions. Foreground objects (vehicles) are detected by subtracting the incoming image from the current background estimate, looking for pixels where this difference image is above some threshold and then finding connected components.

This approach works fairly well in free-flowing traffic. However, under congested traffic conditions, vehicles partially occlude one another instead of being spatially isolated, which makes the task of segmenting individual vehicles difficult. Such vehicles will become grouped together as one large blob in the foreground image.

2.3 Active contour based tracking

A dual to the region based approach is tracking based on active contour models, or *snakes*. The idea is to have a representation of the bounding contour of the object and keep dynamically updating it. The previous system for vehicle tracking developed in our group [11, 12] was based on this approach. The advantage of having a contour based representation instead of a region based representation is reduced computational complexity.

However, the inability to segment vehicles that are partially occluded remains. If one could initialize a separate contour for each vehicle, then one could track even in the presence of partial occlusion[11]. However, initialization is the difficult part of the problem!

2.4 Feature based tracking

Finally, yet another approach to tracking abandons the idea of tracking objects as a whole but instead tracks sub-features such as distinguishable points or lines on the object. The advantage of this approach is that even in the presence of partial occlusion, some of the sub-features of the moving object remain visible. The technology of tracking points and line features in a Kalman filtering formalism is well developed in the computer vision community. Since a vehicle could have multiple sub-features, the new problem then is that of grouping – what set of features belong to the same object.

3 Motion-Based Grouping

The grouping of vehicle sub-features will be based on a common motion constraint, a concept known to Gestalt psychologists as common fate. Point features that are seen as moving rigidly together will be grouped together into a single vehicle. But since there are many vehicles in traffic scenes, there is also an important segmentation aspect to the problem. One does not want to link together sub-features from neighboring vehicles. The grouping process must be sensitive enough to pick up a motion that distinguishes a vehicle from its neighbors, a motion such as a slight acceleration or lane drift.

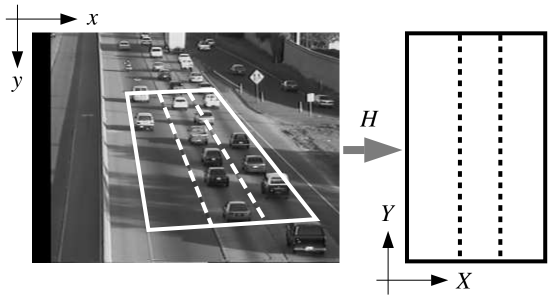

Figure 1: A projective transform H, or homography, is used to map from image coordinates (x, y) to world coordinates (X, Y).

To make the grouping system robust enough to segment different vehicles, the spatial information guiding the grouper will be integrated over a period of time, utilizing as many image frames as possible. Only the sub-features that are tracked from a detection region at the bottom of the image to an exit region near the top will be allowed to participate in the final grouping. Thus, in order to fool the grouper, two vehicles would have to have identical motions during the entire time they were being tracked. In congested traffic, vehicles are constantly changing their velocity to adjust to nearby traffic, thus giving the grouper the information it needs to perform the segmentation. In free-flowing traffic, vehicles may be more likely to maintain constant spatial headways over time, thus making the grouping constraint less useful. But in this scenario, there is more space between vehicles, so a spatial proximity cue is added to aid the grouping/segmentation process.

Since most road surfaces are flat, the grouper exploits an assumption that vehicle motion will be parallel to the road plane. To describe the road plane, the user simply specifies four or more line or point correspondences between the image road and a separate "world" road plane, as shown in Fig. 1. Based on this off-line step, the system can compute a projective transform, or homography, between the image coordinates (x, y) and world coordinates (X, Y). By writing points in homogeneous coordinates, this is a simple linear transform

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \propto H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

The scaling of H is arbitrary, so $H(3, 3)$ is often chosen to be 1.

The grouper considers sub-feature points in pairs. That is, the basic grouper computation is whether or not to

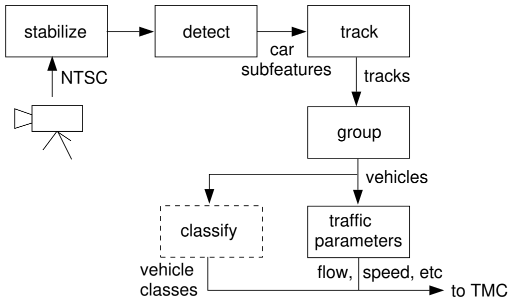

Figure 2: Block diagram of our vehicle tracking and grouping system.

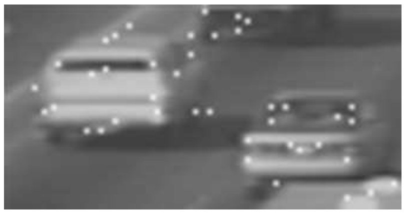

Figure 3: Example corner features located by the system.

group together the 2D point features $\mathbf{p}_a(t)$ and $\mathbf{p}_b(t)$. The dependence on time t is written to emphasize that the grouper is working with sub-feature tracks, and hence has access to the time history of points. The 3D coordinates of these points in the real world will be written in upper case $\mathbf{P}_a(t)$ and $\mathbf{P}_b(t)$.

Consider the simple case where $\mathbf{P}_a$ and $\mathbf{P}_b$ are at the same distance to the camera (e.g. both on the back face of a truck). In this scenario, the grouper only needs to look at a simple function of the displacement vector $\mathbf{p}_a(t) - \mathbf{p}_b(t)$. Since $\mathbf{P}_a$ and $\mathbf{P}_b$ are both at the same distance from the camera d, $\mathbf{p}_a(t)$ and $\mathbf{p}_b(t)$ are both scaled by the same scale factor $1/d$. Thus, for points on the same vehicle, $\mathbf{p}_a(t) - \mathbf{p}_b(t)$ will be constant over time if we can simply compensate for the $1/d$ scaling. Fortunately, the homography can be used for this compensation. Given a point (x, y) in the image, we can estimate the scale factor s that transforms the region around that point to world coordinates. The difference vector $\mathbf{p}_a(t) - \mathbf{p}_b(t)$ can then simply be scaled by s.

We have also considered the more general case where $\mathbf{P}_a$ and $\mathbf{P}_b$ are not at the same distance from the camera. Space considerations in these proceedings prevent a discussion of this case; please see [3] for the details.

4 Algorithm

4.1 Off-line camera definition

Before running the tracking and grouping system, the user specifies some camera-specific parameters off-line. These parameters include:

1. line correspondences for the homography (Fig. 1),
2. a detection region near the image bottom and an exit region near the image top, and
3. a fiducial point for camera stabilization.

4.2 On-line tracking and grouping

A block diagram for our vehicle tracking and grouping system is shown in Fig. 2. First, the raw camera video is stabilized by tracking a manually chosen fiducial point to subpixel accuracy. Next, the stabilized video is sent to a detection module, which locates corner features in a detection zone near the bottom of the image. These corner features are then tracked over time in the tracking module. Next, sub-feature tracks are grouped into vehicle hypotheses in the grouping module. Finally, traffic parameters such as flow rate, average speed, and average spatial headway are computed from the vehicle tracks. In the future, we intend to add a vehicle classification module that will identify vehicles as automobiles, motorcycles, trucks, buses, etc. In this section, we describe the detection, tracking, and grouping modules.

4.2.1 Feature Detection and Tracking

Vehicle sub-features are detected and tracked in order to be insensitive to partial occlusion. Even if part of the vehicle is obscured due to congested traffic, some of the vehicle's sub-features should still remain visible.

Corner features are the chosen sub-features since they can be reliably tracked. Our corner detector computes the windowed second moment matrix by averaging in a spatial window the 2x2 matrix, $\nabla I \nabla I^T$, where ∇I is the image gradient [6]. Corners are declared where the numerical rank of this matrix is 2 (smaller eigenvalue above threshold). Fig. 3 shows some example corner features detected by the system. When a corner sub-feature is detected, a small 9x9 template of the grey level image is extracted and used for correlation in the tracking module. Also, while there are some undesirable corners present near the vehicle boundaries and background, these corners will be pruned away by the feature tests employed by the tracker.

The tracking module tracks corner sub-features from the detection region at the bottom of the image to the exit region near the top. To address the problem of noisy measurements, we employ Kalman filtering[7] to provide most likely estimates of the state of a vehicle sub-feature based on accumulated observations. In our system, the state vector contains sub-feature positions and velocities $(X, Y, \dot{X}, \dot{Y})$ in the world coordinate system; vehicle acceleration is captured in the system dynamics noise process.

The measurement process in the Kalman filter is based on normalized correlation. At each time frame, the Kalman filter predicts where to search for each corner feature. This prediction is mapped back to the image plane, and then the template extracted when the corner was originally detected is correlated in a window around the prediction. The template is scaled down over time to reflect the fact that vehicles are getting smaller as they move down the road surface. We can use the position in world coordinates to predict the proper scale of the template. Once we have located the correlation peak, this measurement is mapped back onto the road plane. Finally, the standard Kalman filter equations for updating the state and error variance are employed.

Two tests are used to eliminate bad sub-feature tracks:

1. *Kalman filter innovations.* The distance between the Kalman filter prediction and the current measurement is computed and the track is rejected if the

Figure 4: Example tracks of corner features.

Figure 5: Example groups of corner features.

distance is above a threshold.

2. *Imprecise measurement test.* If the correlation values form a broad, undefined peak around the correlation maximum, then the measurement process is probably not localizing the sub-feature within the needed precision. To measure the peak's curvature, we compute the number of pixels in the correlation peak that are within a certain fraction of the peak. The track is rejected if the count is over a threshold.

Fig. 4 shows the time evolution of some example tracks, plotted as position over time. The image shown is the frame when the corners were originally detected.

4.2.2 Grouping

The purpose of the grouping module is to group together sub-features that come from the same vehicle. The central cue used by the grouper – common motion – was described already in section 3. In this section, we discuss the details of how the common motion constraint is applied to the sub-feature tracks.

The grouper organizes its task by constructing a graph over time. The vertices are sub-feature tracks, edges are grouping relationships between tracks, and connected components correspond to vehicle hypotheses. When a new sub-feature is detected and is added to the grouping graph, it is initially connected to all neighboring tracks within a certain radius in the image plane. The attitude of the grouper is that nearby tracks are compatible until they prove otherwise through relative motion. For all pairs of tracks $\mathbf{p}_a(t)$ and $\mathbf{p}_b(t)$ joined by an edge, the grouper keeps track of the relative displacement $\mathbf{d}(t) = \mathbf{p}_a(t) - \mathbf{p}_b(t)$ as scaled by the depth-compensating factor computed from the homography. Upon each time frame, another $\mathbf{d}$ value is computed for each edge, and the edge is broken if either

$$\begin{aligned} \max_t d_x(t) - \min_t d_x(t) &> x \text{ threshold, or} \qquad (1) \\ \max_t d_y(t) - \min_t d_y(t) &> y \text{ threshold.} \end{aligned}$$

This breaks the link between two tracks if there is enough relative motion between the two.

In the normal evolution of the graph, vehicles are overgrouped near the detection region since the graph is liberally connected at first. But as vehicles move down the road, they are segmented as they perform a distinguishing motion such as lane drift or an acceleration. When the last track of a connected component enters the exit region, a new vehicle hypothesis is generated and the component is removed from the grouping graph.

Fig. 5 shows the final groups computed for the vehicles in the tracking region (which is the middle part of the image). Corner features are indicated by circles, and there is an edge drawn between grouped corners.

How are the grouping thresholds in equation (1) determined? Consider how the median vehicle size changes as a function of the grouping threshold (Fig. 6). Here, we assume that the same threshold is used for x and y, and vehicle size is measured as the maximum distance between any two points in the group. Empirically, one notices that the plot of median vehicle size versus threshold exhibits two linear regimes:

1. **Oversegmentation.** Below optimum threshold. Vehicle size increases rapidly as one raises the threshold, as correct groups are still being constructed out of vehicle fragments.
2. **Overgrouping.** Above optimum threshold. This part of the graph has a lower slope, as it is harder to group together different vehicles than it is to group a single vehicle's sub-features.

Given this relationship, our goal is to detect the breakpoint between the two regimes. In an off-line step, we sample the graph by running the grouper at different thresholds and computing the median vehicle size. Next, two line segments are fit to the graph by minimizing the sum of squared error, which locates the breakpoint. We performed this procedure for all 7 video sequences in section 6.2. The thresholds computed led to vehicle recognition rates that were very close to the optimum thresholds (optimum thresholds were computed via exhaustive search). In the worst case, the computed thresholds led to a decline of only 3.6% in the recognition rate.

5 Real-time System

We have implemented the tracker on a network of 13 Texas Instruments C40 DSPs, connected together as shown in Fig. 7. The computationally heavy operations in the tracking algorithm – convolution in the feature detector and correlation in feature tracker – are placed on the C40 network, while the grouper is run on the host PC. Running the grouper on the PC is necessitated by memory

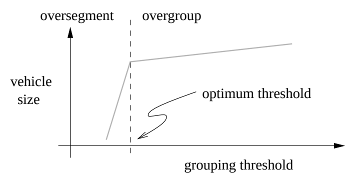

Figure 6: The plot of vehicle size versus grouping threshold exhibits two regimes: an oversegmentation regime and overgrouping regime.

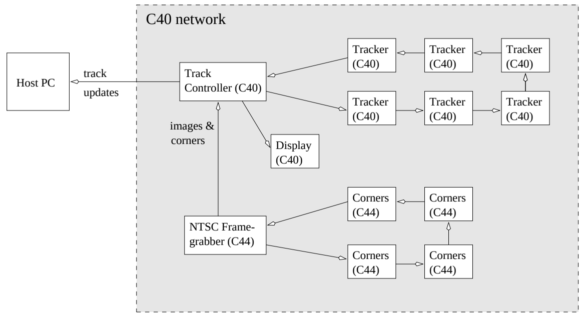

Figure 7: The C40 network used for feature tracking, consisting of an NCSC frame-grabber C44 module (C44s are cut-down C40s with only four communication links); a quad C44 module (four processors) for corner feature detection; a large memory module (8M RAM) for maintaining the state of current feature tracks (track controller); six fast SRAM modules for feature tracking; and a VGA graphics module for display.

requirements. The grouper needs to store track trajectories, which would quickly exhaust the limited memory available on the C40 modules. But keeping the grouper on the PC is also beneficial from a load balancing perspective, as the PC is a 150MHz Pentium and thus equivalent to 3 to 4 C40s.

The processors are arranged in two loops, each of which is operated as a pipeline feeding back to its source. These two pipelines are controlled by the frame grabber and the track controller; they compute the corner features and the track updates respectively. Four C44 processors are assigned to corner detection, each processing one quarter of the user-defined detection region. The corners are fed back to the frame-grabber, which passes them along with the original image to the track controller. A simple efficiency gain is achieved by sending the image first, since the track controller can then update the existing tracks while the corners are computed.

The job of the track controller is to maintain the state of the complete list of current tracks. It does this by receiving updates for existing tracks from its pipeline of six C40s, and creating new tracks at positions indicated by the corner detector. The tracker C40s each update one sixth of the tracks. Since track updates are fairly homogeneous tasks, this achieves good load balancing.

The performance of the tracker is 7.5Hz in uncongested traffic, dropping to 2Hz in congested traffic, where many more tracks are in progress at any given time. This reduction in speed does not of itself lead to a reduction in performance of the tracker, since vehicle speeds in congested traffic are reduced, and so the requirement for tracking rate is naturally reduced.

Sequence	Description	Length	N	G
Highway 55	heavy congestion	2:46	238	203
Florin Rd 1	free flow & congestion	2:46	244	260
Mac Rd 1	free flow & congestion	1:00	69	68
Florin Rd 2	night	1:20	58	64
San Jose	urban intersection	1:45	34	34
Mac Rd 2	free flow & congestion	3:36	249	265
Florin Rd 3	free flow & congestion	3:36	317	342

Table 1: Video sequences for laboratory testing. Length is in min:sec, N is the number of actual vehicles (counted by a human) and G is the number of reported vehicle groups.

6 Results

Our tracking and grouping system has gone through two major phases of testing. First, we tested a software-only, off-line version of the system in terms of its ability to detect vehicles. This testing gave us a "microscopic" view of the system, allowing us to analyze errors such as false detections, false negatives, and overgroupings. Second, the real-time system was tested on a substantial amount of data – 44 lane hours worth – to see if the system could accurately measure the aggregate parameters of flow, velocity, vehicle density, and average spacing.

6.1 Off-line testing of vehicle detection

In order to analyze the behavior of the system at the vehicle level, we tested the system's vehicle detection rate for a set of videotapes covering a range of scene conditions: congestion, free-flow, night, and an urban intersection (see Table 1). Since we wanted to measure errors such as vehicle oversegmentation and overgrouping, vehicle ground truth was manually defined for each sequence. For a particular vehicle, ground truth is a binary mask outlining the vehicle in one or two frames. The number of ground truths is denoted as N in Table 1, and the number of reported groups is G.

Table 2 shows the performance of our system using automatically computed grouping thresholds, as well as the distribution of errors. A separate automatic evaluation program compares the vehicle ground truths against the groups reported by the tracker/grouper and tallies the following events:

1. *True match.* A one-to-one matching between a ground truth and a group.
2. *False negative.* An unmatched ground truth.
3. *Oversegmentation.* A ground truth that matches more than one group.
4. *False positive.* An unmatched group.
5. *Overgrouping.* A group that matches more than one ground truth.

In analyzing the results, it should be said that the Highway 55 sequence is a difficult one because of a poor camera position and a number of large trucks that sometimes completely occlude automobiles. In terms of trading off the different error conditions, we have noticed that oversegmentation and overgrouping can be traded off by adjusting the grouping thresholds.

Sequence	true match	false neg.	over-seg	false pos.	over-group
Highway 55	73.9%	18.5%	6.7%	4.9%	0.4%
Florin Rd 1	88.5%	1.6%	6.9%	1.9%	1.5%
Mac Rd 1	94.2%	1.5%	1.5%	2.8%	1.4%
Florin Rd 2	89.6%	6.9%	3.4%	20.0%	0.0%
San Jose	85.3%	2.9%	5.9%	0.0%	2.9%
Mac Rd 2	80.3%	6.0%	10.4%	2.3%	1.5%
Florin Rd 3	84.5%	2.2%	10.1%	0.3%	1.8%

Table 2: Performance of the tracking/grouping system on the off-line test sequences. When computing rates, the first three columns divide the number of true matches, false negatives, etc., by N; the final two columns divide by G.

As the first three sequences have long shadows, the experimental results show that the system can handle shadows – shadow sub-features tend to be unstable over time, especially in congestion.

6.2 On-line testing of traffic parameters

Our second phase of testing evaluated the on-line system's ability to estimate aggregate traffic parameters. The parameters typically used by traffic engineers to monitor the freeways include:

1. *Flow.* Number of vehicles per hour.
2. *Velocity.* Average vehicle velocity.
3. *Density.* Number of vehicles per unit distance.
4. *Headway.* Average spacing between vehicles.

These parameters are computed separately for each lane of traffic and are averaged over a period of time (taken to be 5 minutes in our experiments). Also, it should be apparent that these are not independent variables; we use the methodology from Edie[5] to compute these parameters from the vehicle track data.

Ground truth is provided from inductive loop data that was collected concurrently with the video data. Each lane of traffic has two loops separated by 20 feet, giving us an effective speed trap for measuring velocity.

Our system was tested on approximately 44 lane hours of video from the Florin Road interchange along Highway 99 in Sacramento, Calif. (see Fig. 9 for an example shot). The data includes all observed operating conditions: day, night, twilight, long shadows and rain; congestion and free flow. Lane 1, on the left, is carpool (HOV) lane and exhibited little if any congestion. Lane 3, on the right, exhibited some degree of congestion for approximately 20% of the time. Finally, the loops in lane 2 were bad so it was excluded from the final analysis. The video data was divided into 5 minute aggregation periods, yielding 514 samples for the traffic parameters. Overall, there were roughly 40,000 vehicles in the final video data set.

The vehicle track data from the real-time system can then be compared with the loop data over the 20 foot region of overlap between the tracks and loop data. Fig. 8 shows scatter plots of the flow and velocity estimates provided by the loop and vision data, and Table 3 summarizes the error distribution for velocity, flow, density, and headway. As one would expect from a feature based tracker, the measured velocity is very accurate. Even if the tracker overgroups or oversegments vehicles, the erroneous blobs

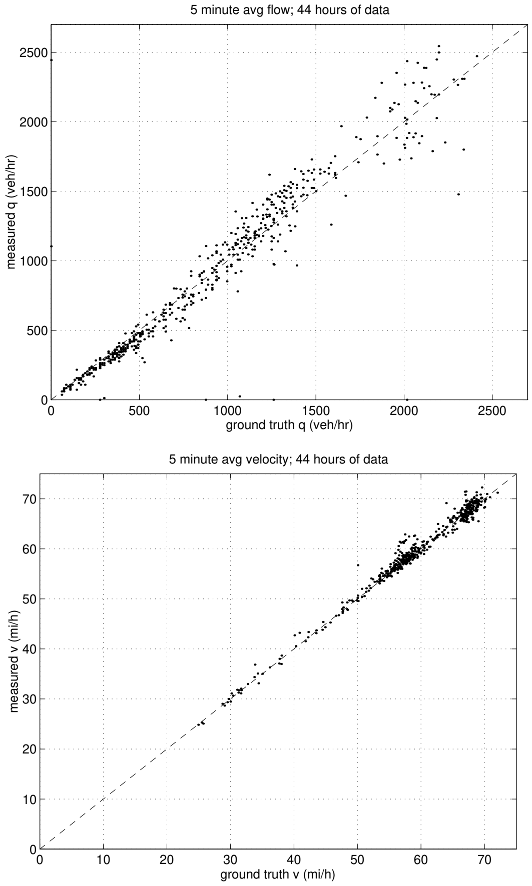

Figure 8: Scatter plots comparing flow q and velocity v to ground truth for the 44 lane hours of data used to test the real-time system.

still move at the prevailing speed. The errors in flow, density and spacing are due to missed or over counted vehicles. Often, an error of two or three vehicles in one sample can be very significant. For example, one missed vehicle in a five minute sample at 1,000 veh/hr results in a 2% error. At the mean flow for the data, 910 veh/hr, the error per missed vehicle is slightly higher, at 2.2%.

Another way to examine estimated traffic parameters is as a time series. To demonstrate the performance of our system during a dramatic change in lighting conditions from night to day, in Fig. 10 we show flow q and velocity v for a two hour stretch of continuous video. The video starts at night (5:30 AM, see Fig. 9, left), progresses

% error less than	% vel samples	% flow samples	% dens samples	% headway samples
2.5%	86%	18%	19%	19%
5%	95%	31%	33%	34%
10%	100%	60%	59%	60%
15%	100%	79%	79%	81%
20%	100%	91%	90%	89%
25%	100%	96%	96%	94%

Table 3: Error distribution for velocity, flow, density, and headway.

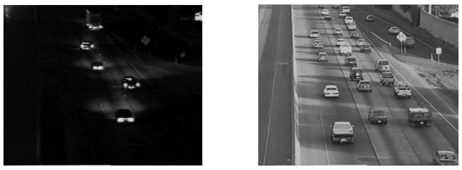

Figure 9: Two images from the start and end of a two hour run of the real-time system.

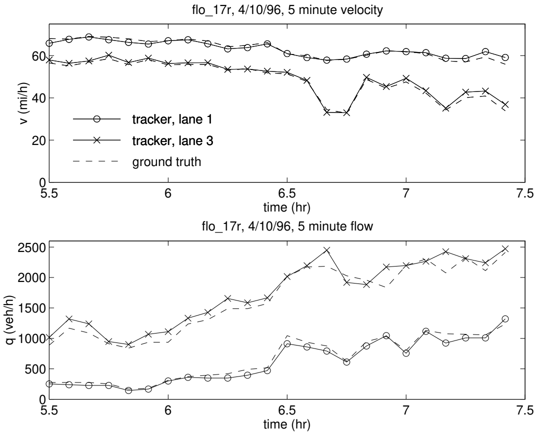

Figure 10: Flow and velocity as a function of time. The sequence begins at 5:30 AM (night lighting conditions) and finishes at daytime.

through sunrise and long shadows, and ends with day (7:30 AM, see Fig. 9, right). In the plot of flow and velocity, there are 48 samples of 5 minute periods and roughly 4,600 vehicles. Note that the morning rush hour peak starts during the sequence and approximately 30 minutes of data from lane three are under light congestion, and thus, frequent occlusions.

In addition, since the primary design goal in developing our system was to deal with congestion, we close the results section with an example of a "shockwave". Fig. 11 plots vehicle tracks as the distance along the lane as a function of time. In this case, ground truth was entered manually at a number of points along each vehicle's trajectory. In the regions of the graph where the slope goes to zero, one notices that vehicles continue to be tracked even when traffic has come to a complete stop.

7 Summary

We have presented a vehicle detection and tracking system that is designed to operate in congested traffic. Instead of tracking entire vehicles, vehicle sub-features are tracked, which makes the system less sensitive to the problem of partial occlusion. In order to group sub-features that come from the same vehicle, the constraint of common motion over trajectory lifetimes is used. A real-time version of the system has been implemented using a network of C40 DSP chips connected to a host PC. The system has been tested on approximately 44 lane hours of data and has demonstrated good performance not only in congested traffic, but also on free-flowing, nighttime, and urban intersection traffic.

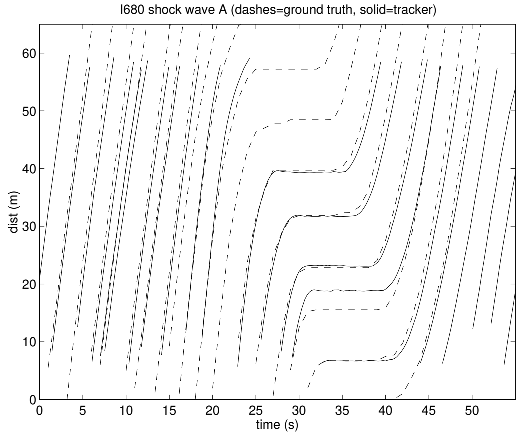

Figure 11: Vehicle tracks for the real-time system during a shockwave. When the slope is zero, the vehicle is completely stopped.

References

[1] K.D. Baker and G.D. Sullivan. Performance assessment of model-based tracking. In *Proc. of the IEEE Workshop on Applications of Computer Vision*, pp. 28-35, Palm Springs, CA, 1992.

[2] Yaakov Bar-Shalom and Thomas E. Fortmann. *Tracking and Data Association*. Academic Press, New York, 1988.

[3] David Beymer and Jitendra Malik. Tracking vehicles in congested traffic. In *SPIE Vol. 2902, Transportation Sensors and Controls: Collision Avoidance, Traffic Management, and ITS*, pages 8–18, Boston, MA, November 1996.

[4] A. Chatziioanou, S. Hockaday, L. Ponce, S. Kaighn and C. Staley. *Video Image Processing Systems Applications in Transportation, Phase II.* Final Report, Calif. Poly. State Univ., San Luis Obispo, Calif., Dec. 30, 1994.

[5] L.C. Edie. Discussion of traffic stream measurements and definitions. In *Proc. Second International Symposium on the Theory of Traffic Flow, OECD*, pages 139–154, Paris, France, 1963.

[6] W. Forstner and E. Gulch. A fast operator for detection and precise location of distinct points, corners, and centers of circular features. In *Proc. of the Intercommission Conf. on Fast Processing of Photogrammetric Data*, pages 281–305, 1987.

[7] Arthur Gelb, editor. *Applied Optimal Estimation.* The MIT Press, Cambridge, MA., 1974.

[8] Klaus-Peter Karmann and Achim von Brandt. Moving object recognition using an adaptive background memory. In V Cappellini, editor, *Time-Varying Image Processing and Moving Object Recognition, 2.* Elsevier, Amsterdam, The Netherlands, 1990.

[9] Michael Kilger. A shadow handler in a video-based real-time traffic monitoring system. In *IEEE Workshop on Applications of Computer Vision*, pages 1060–1066, Palm Springs, CA, 1992.

[10] D. Koller, K. Daniilidis, and H.-H. Nagel. Model-based Object Tracking in Monocular Image Sequences of Road Traffic Scenes. *International Journal of Computer Vision*, 10: 257-281,1993.

[11] D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. In *ECCV*, pp. 189–196, Stockholm, Sweden, May 2-6, 1994.

[12] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, and S. Russell. Towards robust automatic traffic scene analysis in real-time. In *ICPR*, Israel, November 1994.

[13] H.A. Mallot, H.H. Bulthoff, J.J. Little, and S. Bohrer. Inverse perspective mapping simplifies optical flow computation and obstacle detection. *Biological Cybernetics*, 64(3):177–185, 1991.

[14] P. Michalopoulos. Vehicle detection video through image processing: the Autoscope system. *IEEE Trans. on Vehicular Technology*, 40:21-29, 1991.

[15] G.D.Sullivan. Visual interpretation of known objects in constrained scenes. In *Phil. Trans. Roy. Soc (B)*, 337: 361-370, 1992.