Title
Estimating and Validating Models of Microscopic Driver Behavior with Video Data

Author
Skabardonis, Alex

Publication Date
2005-04-01

CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

Estimating and Validating Models of Microscopic Driver Behavior with Video Data

Alex Skabardonis
University of California, Berkeley

California PATH Research Report
UCB-ITS-PRR-2005-14

This work was performed as part of the California PATH Program of the University of California, in cooperation with the State of California Business, Transportation, and Housing Agency, Department of Transportation; and the United States Department of Transportation, Federal Highway Administration.

Final Report for Task Order 4154

April 2005

ISSN 1055-1425

ESTIMATING AND VALIDATING MODELS OF MICROSCOPIC DRIVER BEHAVIOR WITH VIDEO DATA

FINAL REPORT (PATH TO4154)

Research Team:

Principal Investigator:
J. Malik, Computer Science, malik@cs.berkeley.edu

Co-Principal Investigators:
P. Bickel, Statistics, bickel@stat.berkeley.edu
J. Rice, Statistics, rice@stat.berkeley.edu
A. Skabardonis, Institute of Transportation Studies, skabardonis@ce.berkeley.edu

University of California, Berkeley

prepared by

Alexander Skabardonis
Institute of Transportation Studies
e-mail:skabardonis@ce.berkeley.edu

December 2004

ACKNOWLEDGMENTS

This study was performed as part of the California's PATH (Partners for Advanced Highways and Transit) Program (TO 4154) at the Institute of Transportation Studies (ITS) University of California Berkeley.

We appreciate the guidance and support of Joe Palen of the Division of New Technology, Caltrans Headquarters. Chao Chen led the video system upgrade and the pilot study. Zu Whan Kim, Dan Lyddy and Young Cho carried out the research tasks.

Estimating and Validating Models of Microscopic Driver Behavior with Video Data

A. Skabardonis

December 2004

ABSTRACT

This report describes the enhancements to the video data collection of the Berkeley Highway Laboratory (BHL), a unique surveillance system on a section of I-80 freeway in the city of Emeryville. We also present the development of advanced machine vision algorithms to process the video data to generate vehicle trajectories. A pilot application of the BHL system produced trajectories of over 4700 vehicles. This is the largest dataset of vehicle trajectories on extended freeway segments. In addition, algorithms and software were developed for data analysis and visualization.

Keywords:
Freeways, detectors, Machine Vision, Traffic Flow

EXECUTIVE SUMMARY

Traffic simulation models are increasingly being used to analyze alternative operational improvements, and to develop and evaluate ITS systems and strategies. However, there are still gaps in the functionality of traffic simulation tools that require new algorithm development. Furthermore, most of the existing simulators have not been properly calibrated and validated with field data.

Loop detectors provide macroscopic parameters (flow, density, speed), but they do not provide vehicle trajectories that are essential for detailed modeling of vehicle interactions. A large number of real world vehicle trajectories can only be obtained from processing video data. This in turn requires a video surveillance system with sufficient area of coverage and technologies to automatically and robustly obtain trajectories from the video data.

We have developed a unique surveillance system on a section of I-80 freeway in the city of Emeryville. The system, called the Berkeley Highway Laboratory (BHL), consists of detector stations along the freeway section, and video cameras on top of the 30 story building adjacent to the freeway section.

In this study we upgraded and enhanced the BHL video data collection system, and developed improved machine vision algorithms to automatically generate vehicle trajectories. We also developed additional algorithms and software for data processing and visualization. We collected and processed a prototype data set as part of the FHWA Next Generation Simulation (NGSIM) program. The dataset consists of trajectories of over 4700 vehicles over 1 km at 1/15 of a second. This constitutes the largest and most comprehensive dataset on vehicle trajectories available to-date that can be used for developing, calibrating and validating microscopic simulation models.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

Traffic simulation models are increasingly being used to analyze alternative operational improvements, and to develop and evaluate ITS systems and strategies. However, there are still gaps in the functionality of traffic simulation tools that require new algorithm development. Furthermore, most of the existing simulators have not been properly calibrated and validated with field data.

Conventionally, traffic flow parameters and traffic performance data have been generated from loop detectors or a small number of instrumented vehicles. Loop detectors provide macroscopic traffic flow parameters (flow, density, speed). They do not give vehicle trajectories that are essential for detailed modeling of vehicle interactions and for estimating the parameters of car-following and lane changing algorithms. Instrumented vehicles generate long trajectories with detailed maneuvering parameters, but we only get a single trajectory for a single driver. On the contrary, a large number of real world vehicle trajectories can be obtained from video data. The Berkeley Highway Laboratory (BHL) is a unique testbed for collection and processing of video data.

This report describes our efforts to improve the existing BHL video surveillance system and develop improved machine vision algorithms to generate vehicle trajectories on extended freeway sections. The proposed data collection and processing system was applied to produce a prototype dataset of vehicle trajectories.

Section 2 provides background information on BHL and the machine vision algorithms for generating vehicle trajectories. The improvements to the BHL video data collection system and the improved machine vision algorithms are presented in Sections 3 and 4. Section 5 describes the development of data visualization tools. A pilot application of the algorithm is described in Section 6. The final section summarizes the study findings and outlines ongoing and future work. Appendix A includes two papers that provide details on 1) the improved machine vision algorithm, and 2) data analysis and visualization techniques for obtaining traffic parameters with low quality video.

2. BACKGROUND

2.1 Overview of The Berkeley Highway Laboratory (BHL)

BHL is a unique surveillance system on a section of I-80 freeway in the city of Emeryville. It consists of loop detector stations along the 4 mile freeway section, and video cameras on top of the 30 story Pacific Park Plaza apartment building. The coverage of the video surveillance system overlaps with the locations of the loop detectors. The configurations of the cameras and the loop detectors for I-80 north- and south- bound are shown in Figures 1 and 2.

> **Video Surveillance System:** Twelve fixed mount video cameras provide continuous coverage of the freeway, with one camera's surveillance region overlapping the next one. In addition, there are two pan-tilt-zoom (PTZ) cameras connected to Caltrans

District 4 and UC Berkeley. The twelve fixed mount cameras are connected to studio-grade video tape recorders (VTRs), housed on top of the Pacific Park Plaza.

Loop Detectors: There are eight loop detector stations at approximately 1/3 mile apart. The system currently utilizes the I-880 Caltrans software that preserves the 60 Hz event data (4). The data are temporarily stored in a laptop computer inside the controller cabinet and continually transmitted to UC campus via a wireless modem where they stored and processed to provide values of flow, speed, occupancy.

Data have been recorded each weekday during the commute periods. Tapes are collected and replaced daily and then delivered to the PATH Headquarters at the Richmond Field Station on a weekly basis, where they are catalogued and used for off-line data processing and analysis. We have collected video tapes of over 12,000 camera-hours. For some of the video data, the loop detector data is also available (at the time it was collected). We have digitized some of the video data for developing and testing machine vision algorithms to automatically generate vehicle trajectories.

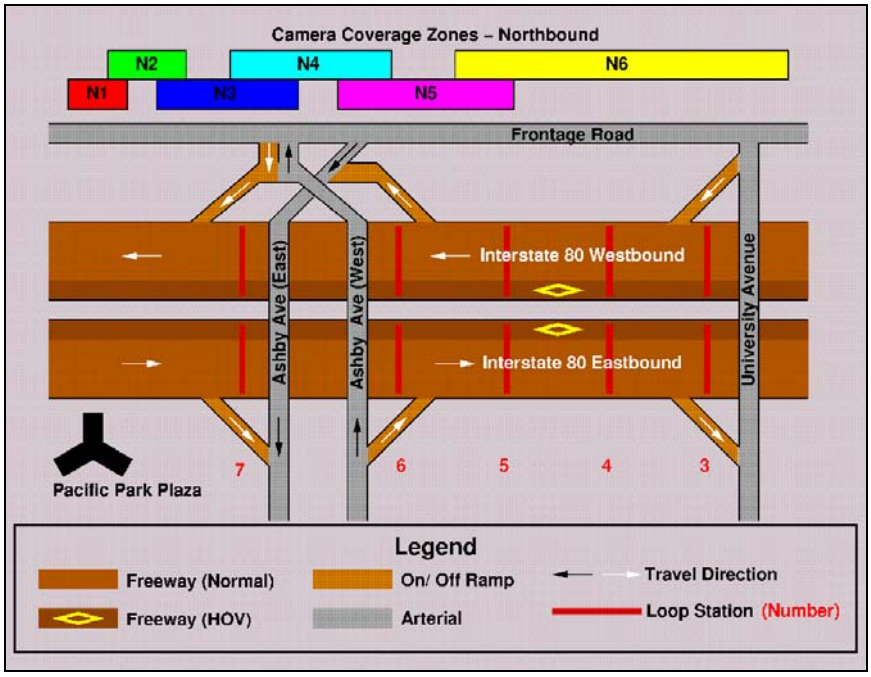


Figure 1. Video Camera Coverage Zones, and the Locations of the Loop Detector Stations
I-80 northbound

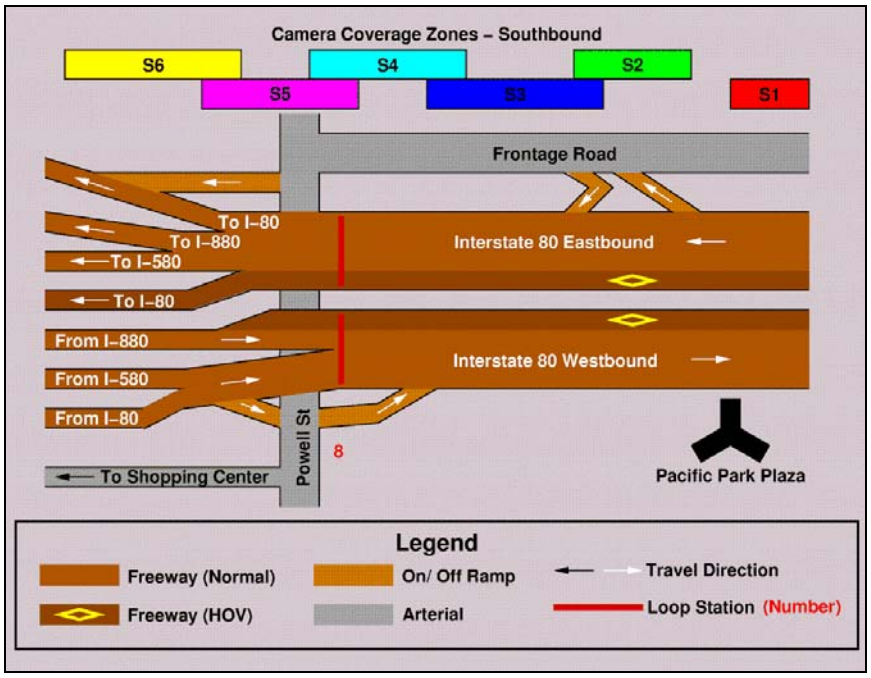


Figure 2. Video Camera Coverage Zones, and the Locations of the Detector Loop Stations
I-80 Southbound

2.2 Extraction of Vehicle Trajectories from Video Data

The generation of vehicle trajectories from raw video data requires accurate and reliable computer vision algorithms. We face several challenges in producing reliable vehicle trajectories:

- The tracking targets (vehicles) vary in sizes, shapes, and colors.
- Our video data includes various times and weather conditions. Thus, the illumination conditions, such as the brightness of image and the direction of the shadows, vary.
- We need to deal with the occlusions among vehicles due to the presence of overpasses.
- Under heavy traffic conditions, where vehicles move slowly, and the distances between vehicles are small, it is hard to separate nearby vehicles, and there are more chances of occlusion.
- The algorithm must work fast. Although there is no real-time constraint, we still want a fast algorithm because we need to process a very large amount of video data.

We have developed three algorithms for vehicle tracking. The first approach known as contour tracking (1), finds the boundaries (closed contours) between vehicles and the background, and dynamically tracks them. The second approach (2) uses the background subtraction algorithm. The background is dynamically estimated from each incoming image, and the difference between the current and the background images is thresholded to form ``blobs" corresponding to vehicles. However, the performance of the background subtraction algorithms significantly degrades in the presence of heavy shadows. It is difficult to separate a shadow from the vehicle because the shadow moves along with the vehicle. Also, often, the shadow cast on nearby vehicles makes the separation between vehicles difficult.

The third approach is based on corner feature tracking (3,4). In this system, vehicle trajectories are generated by grouping individually tracked corner features. Figure 3 shows the flow diagram and some of the intermediate results of the algorithm. This tracking algorithm has performed quite well in a variety of lighting and traffic conditions. The comparisons of the tracker output with human-generated ground truth show that in favorable conditions (light traffic and low shadow), one-to-one detection accuracy (one vehicle hypothesis matches to exactly one ground truth vehicle) exceeds 95%. Under less favorable conditions (heavy traffic, heavy shadow) this one-to-one accuracy is still better than 90% in most cases.

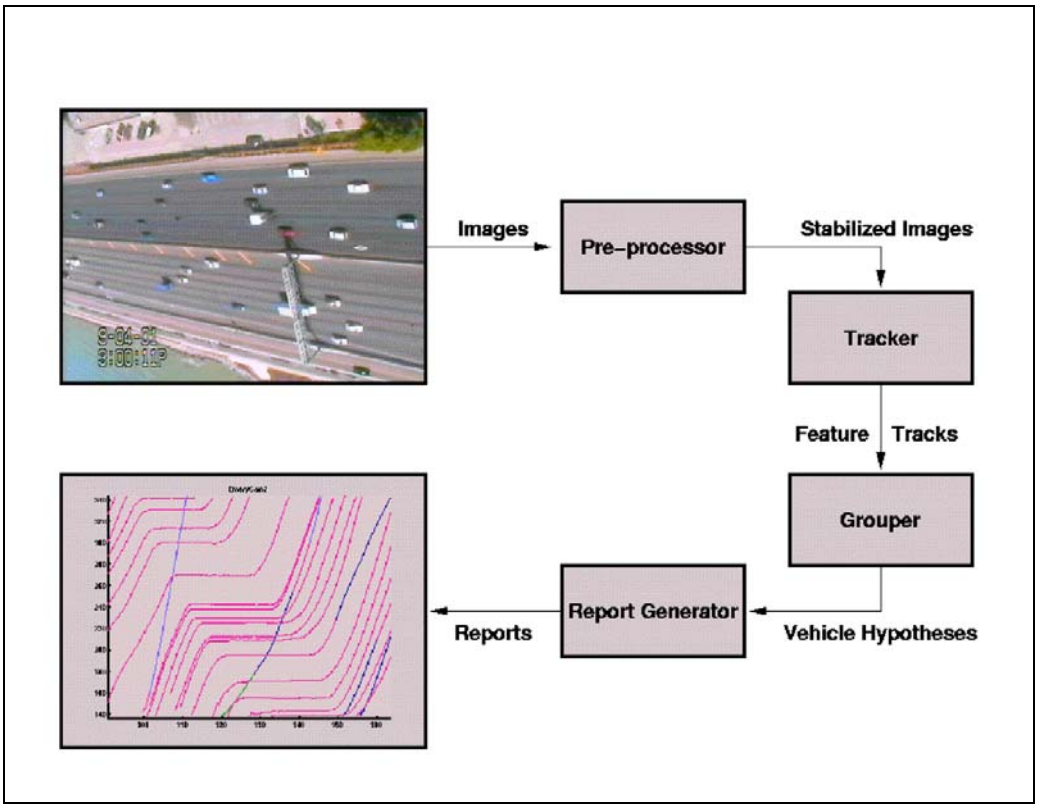


Figure 3. Flow Diagram of the Current System—Feature Tracker

3. ENHANCEMENTS TO THE BHL VIDEO DATA COLLECTION SYSTEM

Experience with the existing BHL video data collection and processing showed several limitations of existing system:

Collection and replacement of video tapes is a labor intensive and expensive process, the quality of video images was not satisfactory for extracting data using machine vision algorithms, the coverage was incomplete. Finally, because the video cameras were mounted on the window washing equipment structure the data collection had to be interrupted periodically.

A new data collection system was designed and implemented (Figure 4). The upgraded data collection system consists of eight digital progressive scan cameras permanently mounted on the roof of the building. Data are continually recorded and stored at computer servers at the roof of PPP. The data are transmitted via wireless 20 Mbps to the CCIT (California Center for Innovative Transportation) in Berkeley were they stored in a database. The raw data are then processed to generate vehicle trajectories. Both raw and processed data are made available.

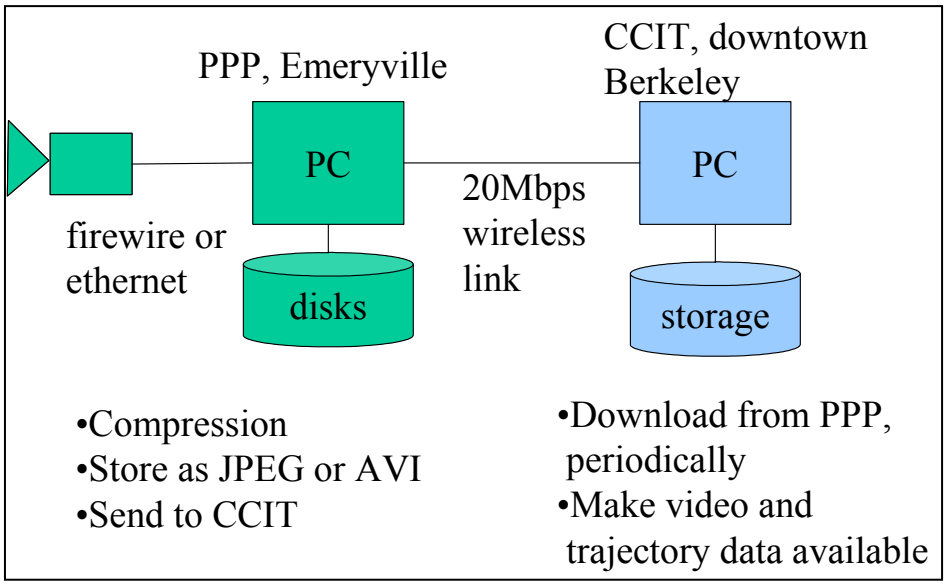


Figure 4. Architecture of the Upgraded BHL Video Data Collection System

4. IMPROVED MACHINE VISION ALGORITHM

The current "corner feature" algorithm provides satisfactory vehicle trajectories reliable enough for the statistical analysis of overall traffic flows. However, more accurate trajectories and precise measurements of the actual sizes of vehicles, are required for calibration and validation of car-following or lane-changing models. Problems we need to address are:

1. A vehicle's position and dimension may not be accurate because they are estimated from the corner features which do not cover the whole vehicle (moreover, some of them may belong to the shadow).

2. The position error caused by missing features (tracking failures) may introduce a significant error in the velocity estimation.

3. Feature grouping is based on only the location and the motion of corner features. Consequently, features of nearby vehicles moving at the same speed may be grouped, and the features of a large vehicle (trailers) may not be grouped.

A new algorithm has been developed based on a three-dimensional vehicle model, i.e., introduce more elaborate features such as edges and lines. Then, the position of a vehicle, as well as its boundary, can be estimated more robustly by the spatial relationships among these features. We first detect vehicles at the entrance area (location where the viewing angle is favorable), and track the detected vehicles based on their intensity profiles. Detailed description of the algorithm can be found in the paper by Kim and Malik (5) included in Appendix A.

5. DEVELOPMENT OF DATA VISUALIZATION TOOLS

The output from the machine vision algorithm, consists essentially of a long list of vehicle trajectories corresponding to a large quantity of video data. It is of very little value to simply watch the tapes or plot the trajectories. To gain some real intuition for the data we have to develop tools that can summarize and visualize it.

We have data processing and visualization software to assist both in the verification of the processed data and perform data analysis. Figure 5 shows trajectories of individual cars overlaid on the density field they create. The plot captures both microscopic and macroscopic behavior and is meant to illustrate the shockwave phenomenon. Figure 6 shows the distribution of spacings between vehicles versus their speed. Such plots can be used in the development and calibration of microscopic car following models. Earlier plots of spacing versus speed, found in the literature, are generally based on a mere handful of specially instrumented probe vehicles.

We also developed an algorithm to estimate velocity fields from low resolution video recordings. This algorithm does not attempt to track individual vehicles, nor does it attempt to estimate derivatives of the field of pixel intensities. Rather, we compress a frame by

obtaining an intensity profile in each lane along the direction of traffic flow. The speed estimate is then computed by searching for a best matching profile in a frame at a later time. Because the algorithm does not need high quality images, it is directly applicable to a compressed format digital video stream, such as mpeg, from conventional traffic video cameras. The proposed algorithm was illustrated using BHL video recordings. Detailed description of the algorithm can be found in the paper by Cho and Rice (6) included in Appendix A.

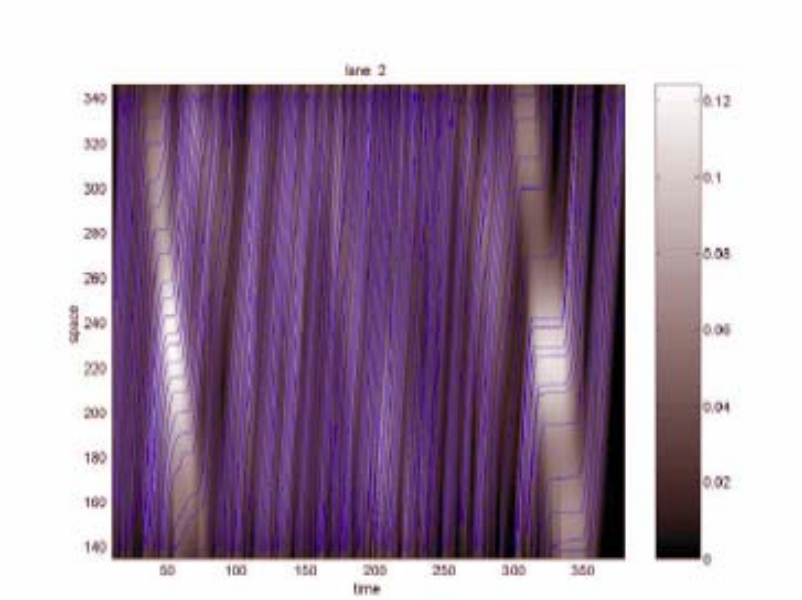


Figure 5. Car Trajectories overlaid on the Density Field

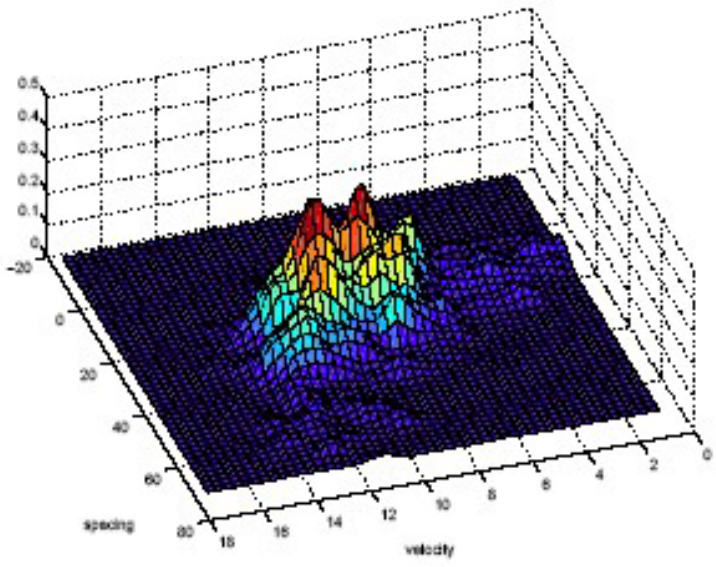


Figure 6. Distribution of Speed-Spacing

6. PILOT APPLICATION

The BHL and the improved machine vision algorithms were used to produce a prototype dataset with vehicle trajectories as part of the Next Generation Simulation (NGSIM) program sponsored by the Federal Highway Administration (FHWA) (7) . The objective of NGSIM is to develop a core of open behavioral algorithms in support of microscopic traffic simulation, with supporting documentation and validation data sets.

The goal of the prototype data collection effort was to assess the viability of emerging vehicle trajectory data collection systems and provide data for the prototype validation process. For the prototype study, both vehicle trajectory video and detector data were collected for two half-hour, peak-period time segments on December 3, 2004.

The machine vision system was used to produce the trajectories of all vehicles traveling in the northbound direction. The system automatically detected and tracked approximately 75 percent of the vehicles. Supplementary analysis tools were developed and used to correct false-negatives (vehicles that were not detected) and false-positives (vehicles that were erroneously detected). Also, the system allows to manually detect vehicles (that are not automatically detected) and then they are automatically tracked. Figure 7 shows a screenshot of vehicles tracked by the algorithm. Figure 8 shows lane changes of tracked vehicles.

The prototype dataset consists of trajectories of 4,733 vehicles over 2,952 ft (approximately 1 km) at 1/15 of a second, for a total of 2.8M data points. This is the largest and most comprehensive dataset on vehicle trajectories ever produced.

7. CONCLUSIONS

BHL is a unique testbed for collecting data to study traffic dynamics. In this study we upgraded and enhanced the BHL video data collection system, and developed improved machine vision algorithms to automatically generate vehicle trajectories. We also developed algorithms and software for data processing and visualization.

We collected and processed a prototype data set as part of the NGSIM program. The dataset consists of trajectories of over 4700 vehicles over 1 km at 1/15 of a second. This constitutes the largest and most comprehensive dataset on vehicle trajectories and can be used for calibration and validation of microscopic simulation models. The dataset and processing algorithms are distributed free to the transportation research community.

Currently, as part of an effort sponsored by the FHWA we improve the vehicle detection and tracking technology and we will collect and process data on other freeway sites.

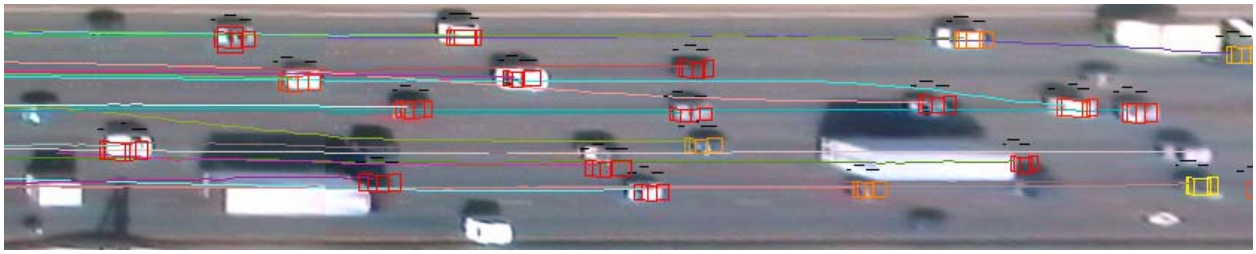


Figure 7. Automated Tracking of Vehicle Trajectories

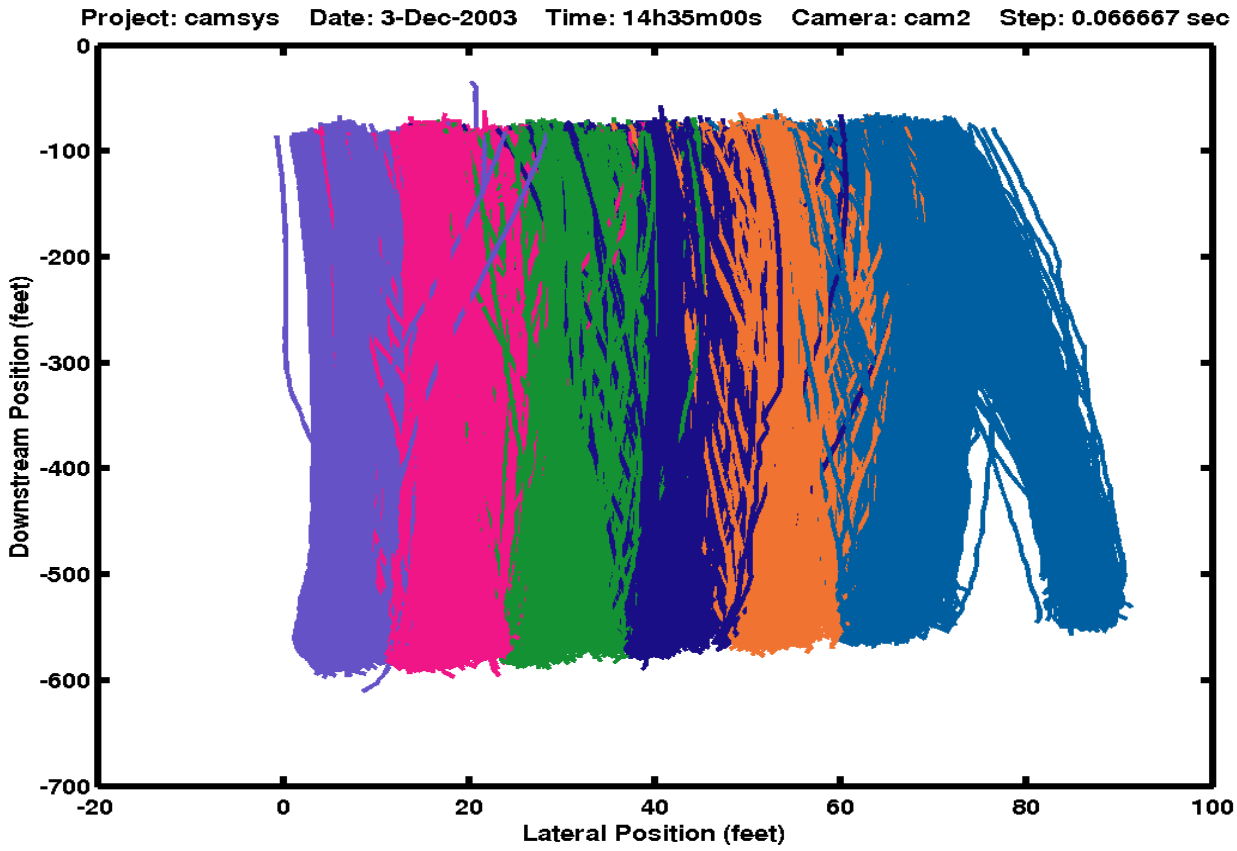


Figure 8. Lane-Changing Vehicle Trajectories

REFERENCES

1. D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, and S. Russell, “Towards robust automatic traffic scene analysis in real-time,” *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, Jerusalem, Israel, October 1994.
2. D. Koller, J. Webber, and J. Malik, “Robust Multiple Car Tracking with Occlusion Reasoning,” *Proceedings of European Conference on Computer Vision*, vol. A, pp. 189-196, 1994.
3. D. Beymer, P. McLauchlan, B. Coifman, and J. Malik, “A Real-time Computer Vision System for Measuring Traffic Parameters,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 495-501, 1997.
4. Coifman, et al, 2000, "The Berkeley Highway Laboratory-Building on the I-880 Field Experiment," *Proceedings 3rd IEEE Conference on Intelligent Transportation Systems*, Detroit, October 2000.
5. Kim Z. and J. Malik, (2003), "High-Quality Vehicle Trajectory Generation from Video Data Based on Vehicle Detection and Description," *Proc. IEEE Intelligent Transportation Systems Conference*, pp. 176-182.
6. Cho Y., and J. Rice (2003), "Estimating Velocity Fields on a Freeway from Low Resolution Video,” Department of Statistics Technical Report #665 available at <http://www.stat.berkeley.edu/tech-reports/index.html> (under review IEEE Transactions on Intelligent Transportation Systems)
7. Alexiadis V., J. Colyar, J. Halkias, R. Hranac, and G. McHale “The Next Generation Simulation Program, ITE Journal, 2004.

APPENDIX A.

1. Kim Z. and J. Malik, (2003), "High-Quality Vehicle Trajectory Generation from Video Data Based on Vehicle Detection and Description," *Proc. IEEE Intelligent Transportation Systems Conference*, pp. 176-182.

2. Cho Y., and J. Rice (2003), "Estimating Velocity Fields on a Freeway from Low Resolution Video," Department of Statistics Technical Report #665 available at <http://www.stat.berkeley.edu/tech-reports/index.html> (under review IEEE Transactions on Intelligent Transportation Systems)

High-Quality Vehicle Trajectory Generation from Video Data Based on Vehicle Detection and Description

ZuWhan Kim and Jitendra Malik
Computer Science Division
University of Berkeley, CA, USA
{zuwhan,malik}@cs.berkeley.edu

Abstract

Vehicle trajectories contain rich information on microscopic phenomena such as car following and lane changing. Despite many efforts to retrieve reliable trajectories from video images, previous approaches do not give high enough quality of trajectories that can be used in microscopic analysis. We introduce a new vehicle tracking approach based on a model-based 3-D vehicle detection and description algorithm. The proposed algorithm uses a probabilistic line feature grouping method to detect vehicles with little computation. A dynamic programming algorithm is proposed for fast reasoning. We present the system implementation and the vehicle detection and tracking results.

1. Introduction

Vehicle trajectories contain rich information about traffic flow. We can study microscopic behaviors, such as car-following and lane-changing, with vehicle trajectories. For example we can evaluate or tune the parameters of the car-following models for the simulation tools, or study the relationship between lane changing and traffic congestion. We cannot obtain such information from traditional loop detectors because they cannot detect lane changes or continuously measure the car following distances over time.

One way of obtaining vehicle trajectories is to use instrumented vehicles. Instrumented vehicles generate long trajectories with detailed maneuvering parameters including the lane changes and the car following distances. However, given a travel, we only get a single trajectory of a single driver (and some additional information on leading or near-by vehicles). Therefore, only trajectories from a small number of drivers (who are aware of the experiments) are available, which may bring in a bias to the result. Moreover, in-vehicle sensors have limited coverage, and we cannot observe, for example, shockwave created by a lane change.

Video data is a good complementary data source. We can get a large number of trajectories (much less biased, but short) from video data. Example images from our video data are shown in Figure 1. The images were obtained from 3 cameras installed on the roof of a 30-story building alongside a freeway. There are small overlapping regions between the cameras covering nearby regions. The nearest image, Figure 1c, is close to a nadir (top) view while the farthest one, Figure 1a, is very oblique.

(a) (b) (c)

Figure 1. Example images from three cameras taken at the same time.

It is difficult to generate reliable vehicle trajectories from such video data because:

- The tracking targets (vehicles) vary in sizes, shapes, and colors.
- The video data includes various times and weather conditions. Thus, the illumination conditions, such as the direction of the shadows, vary.
- We need to deal with the occlusions among vehicles or by other structures.
- Traffic conditions vary, and many of the tracking algorithms degrade with heavy traffic congestion, where vehicle moves slowly, and the distances between vehicles are small. In this case, motion-based vehicle detection and background extraction are difficult and it is also hard to separate nearby vehicles.

To model drivers behaviors well it is important to have accurate information about inter-vehicle spacing and vehicle trajectory. Previous video-based approaches, [6], [2],

were suitable enough for counting the number of vehicles or finding general traffic flows. However, they were either restricted to being used under favorable lighting and traffic conditions [6] or did not give accurate location and dimension of vehicles [2].

We present an approach which works well under various lighting conditions and provides high quality trajectories. Our approach is based on a 3D model-based vehicle detection and description algorithm. In Section 2, we present related work and overview our approach. The vehicle detection and description algorithm is shown in Section 3, and other implementation issues including the tracking algorithm is presented in Section 4. In Section 5, we show detection and tracking results. Finally, we present the conclusion and future work in Section 6.

2. Related Work and Our Approach

There are two well-known vehicle tracking approaches. The first one, [6], [4], is the *background subtraction algorithm*. In this approach, the background is dynamically estimated from incoming images, and the difference between the current and the background images is thresholded to form "blobs" corresponding to vehicles. This algorithm gives reliable vehicle detection given a favorable illumination condition and a camera angle.

However, the performance of the background subtraction algorithms significantly degrades in the presence of heavy shadows. It is difficult to separate a shadow from the vehicle because the shadow moves along with the vehicle. Also, often, an occlusion or a shadow cast on nearby vehicles makes the separation between vehicles difficult. In addition, the background estimation performance is degraded when the traffic is heavy because the movements of the vehicles are small and a significant part of the background is not observable.

The other approach, [2], uses corner features. In this approach, individually extracted and tracked corner features are grouped based on the proximity of their positions and the similarity of the motion. This approach gives good detection even with less favorable illumination conditions. However, it still has several limitations:

- The location and the dimension of a detected vehicle may not be accurate because they are estimated from the corner features which do not cover the whole vehicle (moreover, some of them may belong to the shadow).
- The position error caused by missing features (tracking failures) may introduce a significant error in the velocity estimation.
- The feature grouping is based on only the locations and the motions of corner features. Thus, there are times that features of nearby vehicles (of the same speed) are grouped together, or the features of a large vehicle (for example, trailer trucks) are not grouped together.

Therefore, the quality of the trajectories are not fine enough to be used for microscopic analysis.

We introduce a new approach based on a 3D vehicle detection and description algorithm. We first detect vehicles at the entrance area (where the viewing angle is favorable), and track the detected vehicles based on their intensity profiles. Although our application does not require realtime processing, fast computation is still important because we need to process a huge volume of video data. Most of the previous vehicle detection algorithms (whether they are based on template matching or not) work on the intensity image pixels directly, and requires significant processing time. In the next section, we present a fast algorithm which uses the line features. For robust detection, we apply probabilistic reasoning. We also present a dynamic programming algorithm for fast reasoning.

3. 3D Vehicle Detection and Description with Probabilistic Feature Grouping

In this section, we introduce a model-based car detection and description algorithm. Vehicle detection, description, and/or recognition have been an active research area [11], [8], [9], [1], [5], [12]. Early model-based approaches, [11], have been focused on generic pose estimation with pre-defined shape. Other approaches have been focused on high resolution ground views, [1], or uses the background subtraction algorithm, [5]. Most of these approaches require a large amount of computation, except the background subtraction algorithms which suffer from shadow and traffic congestion.

In [8], Rajagopalan *et al.* presented vehicle detection algorithm based on higher order image statistics. However, it requires too much computation while the detection rate (73% with 14% false alarm) was not satisfactory (although the experiment was performed on relatively complex scenes). In addition, they have focused on the detection algorithm and the localization performance may not be as good. In [12], Zhao and Nevatia presented an algorithm of detecting cars from aerial images by examining their rectangular shape, front and rear windshields, and shadow. It showed good detection rate (about 90% with 5% false alarm) with less computation (about 30 secs plus preprocessing for 1000×870 image on a PII 400 MHZ), but it still requires image-based comparison and the amount of the computation is still large for our application. In addition, the detection is based on the rectangular boundaries, which is only applied for aerial images.

In this section, we present a faster and more flexible al-

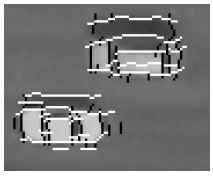


Figure 2. The horizontal and vertical line features used in the algorithm.

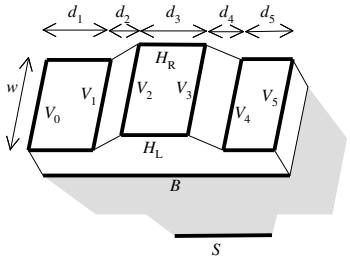


Figure 3. The car model. The distances are estimated in the world coordinates. H_L and H_R are the left and right lines, $V_0, \ldots, V_1$ are the vertical lines, B is the base line, and S is the shadow line.

gorithm, which gives 3D structures (descriptions) of the detected cars at the same time. Our algorithm uses line features. Example line features (horizontal and vertical) are shown in Figure 2. Given the orientation of the cars (which is known) the line feature extraction algorithm is as follows: 1) apply a 2-D oriented edge detectors (horizontal and vertical, separately), 2) apply the *non-maxima suppression algorithm* [3] on the given orientation, and 3) perform the *connected-component analysis* for line grouping.

Figure 3 shows our car model. Once line features are extracted, we fit the car model to them. We assume that the front line (V_0), the rear line (V_5), the left line (H_L), and the right line (H_R) of a car are always detected. Our algorithm is applied for each vertical line feature, v_0, which we assume to be V_0. We then gather near-by horizontal line features assuming maximum car width and length. For each pair of horizontal line features, l, and r (which we assume to be H_L and H_R), we gather vertical line features, $v_1, \ldots, v_n$ (ordered from front to rear), and apply a dynamic programming algorithm to find the configuration among them.

In the following equations, we will use $w_{i,j}$ as a shorthand for $V_i = v_j$, $w_{0\neq 0}$ for $V_0 \neq v_0$, and $P(l, r)$ for $P(l = H_L, r = H_R)$. Our goals are:

Detection: for each v_0, estimate $P(w_{0,0}|\mathbf{E})$, where $\mathbf{E}$ is all the evidence we gather, and

Description: find l, r, m_0, ..., m_5 which maximize $P(l, r, w_{1,m_1}, \ldots, w_{5,m_5}|w_{0,0}, \mathbf{E})$.

We estimate $P(w_{0,0}|\mathbf{E})$ by summing up all the possible configurations of H_L, H_R, and V_i because our model requires evidence derived from an unknown assignment of these variables, such as the width of the vehicle and the distances between the vertical lines.

$$P(w_{0,0}|\mathbf{E}) = \sum P(w_{0,0}, l, r, w_{1,m_1}, \ldots, w_{5,m_5}|\mathbf{E})$$

where

$$P(w_{0,0}, l, r, w_{1,m_1}, \ldots, w_{5,m_5}|\mathbf{E}) = \frac{P(w_{1,m_1}, \ldots, w_{5,m_5}|w_{0,0}, l, r, \mathbf{E})P(\mathbf{E}|l, r, w_{0,0})P(l, r|w_{0,0})P(w_{0,0})}{P(\mathbf{E})}$$

We assume that $P(w_{0,0})$ is uniform (over all the possible v_0) as well as $P(l, r|w_{0,0})$ (which is $1/|H_L \times H_R|$). Unfortunately, $P(\mathbf{E})$ is difficult to estimate (unless we know $P(w_{0\neq 0}|\mathbf{E})$). With limited choices, we assum that $P(\mathbf{E})$ is uniform over v_0. In the following subsections, we describe how we estimate $P(\mathbf{E}|l, r, w_{0,0})$ and $P(w_{1,m_1}, \ldots, w_{5,m_5}|w_{0,0}, l, r, \mathbf{E})$.

3.1 $P(\mathbf{E}|l, r, w_{0,0})$

We use the distance (in the world coordinates) between l and r and the existence of the shadow and the base lines (S and B of Figure 3) to estimate $P(\mathbf{E}|l, r, w_{0,0})$. In addition, we use a gradient sign of l or r according to the shadow location. For example, when the shadow is cast on the left side of the vehicle, the left side of l will be darker than its right side.

In principle, the sun angle can be estimated from the approximate position of the vehicle (on the earth) and the time and the date. When such information is not available, we can estimate the angle of the Sun from the direction and the length of shadow cast. We have implemented the second approach which is good enough to analyze a video sequence of 10 minutes or shorter.

3.2 $\sum P(w_{1,m_1}, \ldots, w_{5,m_5}|w_{0,0}, l, r, \mathbf{E})$

For simplicity, we define $\Phi \equiv \{w_{0,0}, l, r\} \cup \mathbf{E}$. It is much harder to estimate $\sum P(w_{1,m_1}, \ldots, w_{5,m_5}|\Phi)$ because of its complexity. Although there are only 5 lines that we need to configure, we still have $P_{5,n}$ different assignments (in addition, multiplied by $|H_L| \times |H_R| \times |V_0|$) when n is the number of the vertical line features. In addition, we need to deal with at least six dimensional joint probabilities which is huge enough to result serious overfitting. In this section, we present a dynamic programming algorithm to estimate this value efficiently.

The evidence features we use are 1) the distances between lines, 2) the gradient changes at the lines, 3) the sampled intensity levels between lines, and 4) the length (coverage) of the lines, The distances are estimated in the world

coordinate. For this, we assume that the height of the car is fixed (1.4 meter for the passenger cars) and the heights of the front hood and the trunk of the car are 1 meters each. Then we back-project the (center) positions of the line features to 3-D coordinate.

The gradient changes on the edge of the windshields are useful cues to detect cars [12]. As in [12], we assume that, for most of the bright cars, the windshields are usually darker than the car frame, and for most of the dark cars, the windshields are brighter than the car. In other words, with a high probability p, the gradient directions of V_1 and V_2 (or V_3 and V_4) are opposite to each other while those of V_1 and V_3 (or V_2 and V_4) are the same.

To reduce the computational complexity of the problem, we apply a dynamic programming algorithm based on following Markov-style assumptions:

$$\begin{aligned}
&P(w_{i,j}|w_{i-1,k}, v_{i-2}, \ldots, v_1, \Phi) = P(w_{i,j}|w_{i-1,k}, \Phi), \\
&P(w_{i,j}|w_{i-1,\phi}, w_{i-2,k}, v_{i-3}, \ldots, v_1, \Phi) \\
&\quad = P(w_{i,j}|w_{i-1,\phi}, w_{i-2,k}, \Phi), \text{ and} \\
&P(w_{i,j}|\Phi) = \sum P(w_{i,j}, v_{i-1}, \ldots, v_1|\Phi),
\end{aligned} \tag{1}$$

where $w_{i,\phi}$ is the probability of V_i being missing. The first and the third assumptions are typical Markov assumptions, and the second one extends them to a case of missing features.

Then,

$$\begin{aligned}
&\sum P(w_{i,j}, w_{i-1,m_{i-1}}, \ldots, w_{1,m_1}|\Phi) \\
&= \sum_{k<j} P(w_{i,j}|w_{i-1,k}, \Phi) P(w_{i-1,k}|\Phi) \\
&+ \sum_{k<j} P(w_{i,j}|w_{i-1,\phi}, w_{i-2,k}, \Phi) P(w_{i-1,\phi}|w_{i-2,k}, \Phi) P(w_{i-2,k}|\Phi) \\
&+ \ldots,
\end{aligned} \tag{2}$$

where the parameters can be obtained as follows:

$$\begin{aligned}
&P(w_{i,j}|w_{i-1,k}, \Phi) = P(w_{i,j}|w_{i-1,k}, l, r, \mathbf{E}) = \\
&\frac{P(\mathbf{E}|w_{i,j}, w_{i-1,k}, l, r)(1 - P(w_{i,\phi}|w_{i-1,k}, l, r, \mathbf{E}))}{\sum_{j'>k} P(\mathbf{E}|w_{i,j'}, w_{i-1,k}, l, r)} \\
&P(w_{i,j}|w_{i-1,\phi}, w_{i-2,k}, l, r, \mathbf{E}) \\
&= \frac{P(\mathbf{E}|w_{i,j}, w_{i-1,\phi}, w_{i-2,k}, l, r)(1 - P(w_{i,\phi}|w_{i-1,\phi}, w_{i-2,k}, \mathbf{E}, l, r))}{\sum_{j'>k} P(\mathbf{E}|w_{i,j'}, w_{i-1,\phi}, w_{i-2,k}, l, r)} \\
&\ldots
\end{aligned} \tag{3}$$

Our implementation allows at most two consecutive missing features (*i.e.* $P(w_{i,j}|w_{i-1,\phi}, w_{i-2,\phi}, w_{i-3,\phi}) = 0$).

Assuming that the evidence features are independent to each other given two vertical lines,

$$\begin{aligned}
&P(\mathbf{E}|w_{i,j}, w_{i-1,k}, l, r) \\
&\quad = D(d_{j,k}|i, i-1) G(g_{j,k}|i, i-1) I(i_{j,k}|i, i-1) C(c_j|i),
\end{aligned}$$

where $D(d_{j,k}|i, i-1)$ is a PDF of distance between V_i and V_{i-1} being $d_{j,k}$ (the distance between v_j and v_k), $G(g_{j,k}|i, i-1)$ is a probability of the gradient difference between V_i and V_{i-1} being that of v_j and v_k ($g_{j,k}$), $I(i_{j,k}|i, i-1)$ is a probability of the intensity samples between V_i and V_{i-1} being similar to that of vehicle frame intensity, and $C(c_j|i)$ is a PDF of the length (coverage) of the extracted line feature. The vehicle frame intensity is sampled near the front line. We assume that $D(d_{j,k}|i, i-1)$ is Gaussian, and $G(g_{j,k}|i, i-1)$ is binary (whether the signs of the gradients are the same or the opposite). For example, the sign of the gradient of V_1 is opposite to that of the V_2 with the probability $G(\text{opposite}|2, 1)$. The parameters of $D(d|i, i-1)$ and $G(g|i, i-1)$ can be obtained by observing learning examples.

Similarly, $P(\mathbf{E}|w_{i,j}, w_{i-1,\phi}, w_{i-2,k}, l, r) = D(d_{j,k}|i, i-2) G(g_{j,k}|i, i-2) I(i_{j,k}|i, i-2) C(c_j|i)$. In fact, $D(d|i, i-2)$ can be obtained from $D(d|i, i-1)$ and $D(d|i-1, i-2)$ when we assume that they are all Gaussian:

$$\begin{aligned}
&E[D(d|i, i-2)] = E[D(d|i, i-1)] + E[D(d|i-1, i-2)] \\
&V[D(d|i, i-2)] = \sqrt{V[D(d|i, i-1)]^2 + V[D(d|i-1, i-2)]^2}.
\end{aligned}$$

We assume that $P(w_{i,\phi}|w_{i-1,k}, \Phi) = P(w_{i,\phi})$, where $P(w_{i,\phi})$ can also be obtained from learning examples.

In summary, our dynamic programming algorithm follows the steps below:

1. Given v_0, l, and r, gather n vertical line candidates (on the right side of v_0) for $V_1, \ldots, V_5$.
2. Make a $6 \times n$ table of $P(w_{i,j}|\Phi)$.
3. $P(w_{0,0}|\Phi) = 1$ and $P(w_{0,i}|\Phi) = 0$ for all $i \neq 0$.
4. Fill in the table using Eq. 2 and Eq. 3.
5. Sum up the last row of the table: $\sum P(w_{5,m_5}|\Phi) = \sum P(w_{5,m_5}, \ldots, w_{1,m_1}|\Phi)$ (Eq. 1).

3.3. Description Algorithm

Finding $P(w_{5,m_5}|\Phi)$ is sufficient for the detection purpose but the description can also be given. The description of the car can be obtained by applying our *backtracking algorithm*: given that $V_i = v_j$, find v_k which maximizes $P(w_{i-1,k}|w_{i,j}, \Phi)$. We find

$$P(w_{i-1,k}|w_{i,j}, \Phi) = \frac{P(w_{i,j}|w_{i-1,k}, \Phi) P(w_{i-1,k}|\Phi)}{P(w_{i,j}|\Phi)}.$$

Note that $P(w_{i,j}|w_{i-1,k}, \Phi)$ is calculated when we calculate $P(w_{i,j}|\Phi)$ (see the previous section). Therefore, when we fill in the table of $P(w_{i,j}|\Phi)$, we make another table, $\arg\max_k P(w_{i-1,k}|w_{i,j}, \Phi)$, for the backtracking.

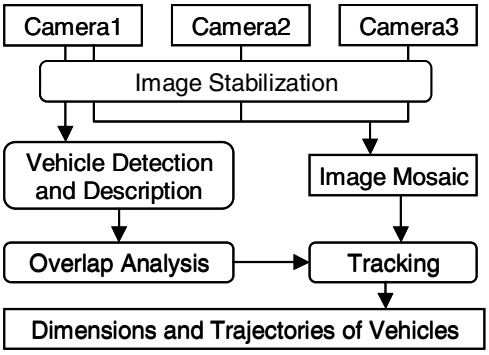


Figure 4. Flow diagram of the system

Where we regard the missing features, the algorithm is slightly modified: given that $V_i = l_j$, find l_k and r which maximize $P(w_{i-1,\phi}, w_{i-2,\phi}, \ldots, w_{i-r,k}|w_{i,j}, \Phi)$. Therefore, we calculate $P(w_{i-1,\phi}, w_{i-2,\phi}, \ldots, w_{i-r,k}|w_{i,j}, \Phi)$ for all the possible values of r ($r \in \{1, 2, 3\}$, in our implementation). For example, when $r = 2$,

$$P(w_{i-1,\phi}, w_{i-2,k}|w_{i,j}, \Phi)$$
$$= \frac{P(w_{i,j}|w_{i-1,\phi}, w_{i-2,k}, \Phi)P(w_{i-1,\phi}, w_{i-2,k}|\Phi)}{P(w_{i,j}|\Phi)}$$
$$= \frac{P(w_{i,j}|w_{i-1,\phi}, w_{i-2,k}, \Phi)P(w_{i-1,\phi}|w_{i-2,k}, \Phi)P(w_{i-2,k}|\Phi)}{\text{constant}}.$$

3.4. Learning Data Collection

Parameters, such as $D(d|i, i-1)$, $G(g|i, i-1)$, $I(i|i, i-1)$, and $P(w_{i,\phi})$, can be learned from examples. We implemented a user-interface to collect such learning examples. A learning example is made by clicking 8 points for $V_0, \ldots, V_5, H_0$, and H_1 (one point for each). We can infer a car structure from given points. Once we have a car model, we match models to the image line features, and estimate the parameters from those line features. Note that we only need a relatively small number of learning examples because all of the above parameters are independent to each other.

4. Tracking and System Implementation

The flow diagram of our system is shown in Figure 4. The cameras are installed on the roof of a tall building and, because of the wind, we sometimes get unstable video streams. Therefore, we first apply a stabilization algorithm to all images. For each image, we manually assign several "static" (background) areas for the stabilization. Then, for each frame, we find corners from these areas, find their matches in the previous frame, estimate camera transformation (affine), and generate new images using the transformation matrix.

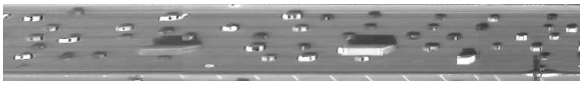


Figure 5. The mosaic image of Figure 1.

Then, we apply the vehicle detection and description algorithm on a small entrance area of the first image. Since the algorithm is applied to each and every line feature, we may have many overlapping hypotheses for a single car. We choose the best hypothesis by comparing $P(w_{0,0}, \mathbf{E})$. The overlap analysis also includes the vehicle hypotheses detected in the past frames. It results redundant detection in all the frames where the vehicle is visible, which increases the detection rate.

The tracking is performed on the mosaic of all three images. We manually calibrated all three cameras. We rectify images using the calibration parameters and attach them to generate the mosaic image. Note that the brightness and contrast levels of all three cameras are different from each other (see Figure 1). Therefore, we adjust the brightness and contrast levels by examining those of overlapping areas: we estimate mean and standard deviation of the intensity pixels, and (linear) transform all the intensity levels of the images so that the brightness and contrast levels of all three images be the same. We only allow small changes (w.r.t. frames) on the parameters for modifying the brightness and contrast levels because a radical change on such parameters degrades the tracking performance. Figure 5 shows the resulting mosaic image.

The tracking is performed based on the *zero-mean cross-correlation matching* [10]. To reduce computation, we perform the search on a two-level image pyramid. For this, we make two mosaic images of different resolution, where the resolution is automatically determined with respect to the resolution of the original images. The search is performed with 9×9 RGB image patches (15×15 for the fine-level image) on 11×11 search windows (5×5 for the fine-level image).

The track may be lost for several consecutive frames due to occlusions or other accidental alignments. For example, in Figure 1c, a part of the first (lower, in the picture) two lanes are occluded by the shadow of a traffic sign structure. To deal with such a case, when a search is failed in one frame, the system continues to search in the following frames based on the previously estimated vehicle speed. A trajectory is discarded when it loses the track for more than a certain number of frames (3 in our implementation). We do not try to refine the tracking result, such as by applying the Kalman filter, because it is better not introduce any bias on the resulting trajectories than produce smooth ones.

Figure 6. An example detection result.

5. Experimental Results

The system works close to realtime (about 2 frame/sec on Pentium 4). The detection algorithm spend less than 100ms for the 200×200, but the image retrieval (from hard disk, 3 MB of uncompressed images per frame) and tracking require more computation.

Figure 6 shows an example detection result in a single frame. We see that the detection and description quality is very good. The quality of the image is poor (smoothed interlaced video) and one car was not detected because important lines were not detected including the front line. Currently, we are in a process of changing the data collection procedure into digital, and we expect the detection rate be significantly increased.

An example tracking result is shown in Figure 7. We find that most of the vehicles are correctly localized and tracked. We observe slight location errors for the vehicles on the left side. This is not the detection error but due to the image-based tracking algorithm which does not handle perspective changes. Our future work includes model-based tracking (Section 6). Our detection algorithm does not handle large trucks but a different algorithm will be applied for the truck detection (also see Section 6).

A comparison with a manual count is shown in Table 1. The detection rate was 85% (116 out of 137) and the false alarm rate was less than 1% (only one). The localization performance was very satisfactory, and only two vehicles were detected with a significant position error (an error bigger than 1/3 of the size of the vehicle). The false alarm was generated from a carpool lane mark combined by the shadow of a dark vehicle (which was counted as misdetection). However, no valid trajectories were generated from it.

Table 1. A detection result on 137 vehicles. Large trucks were not included.

total # of vehicles	137
# of correctly detected (passenger cars)	97
# of correctly detected (other vehicles)	19
detected with wrong position	2
# of missed detection	19
# of false alarms	1

Tracking failures occurred on several vehicles because of 1) occlusion by a large truck (or their shadow casts) for more than 10 frames, and 2) specular highlights. For example, the specular highlight of vehicle on the top of Figure 6 (in the lower right corner) gradually disappeared after several frames.

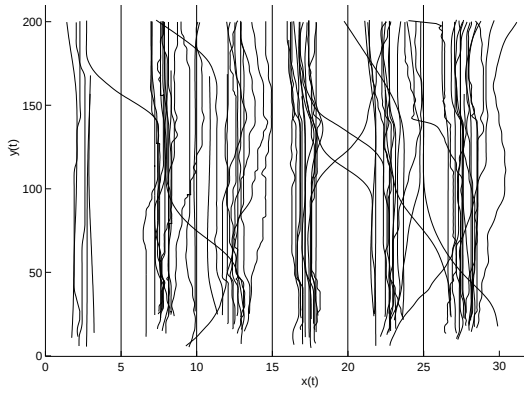


Figure 8. Resulting trajectories.

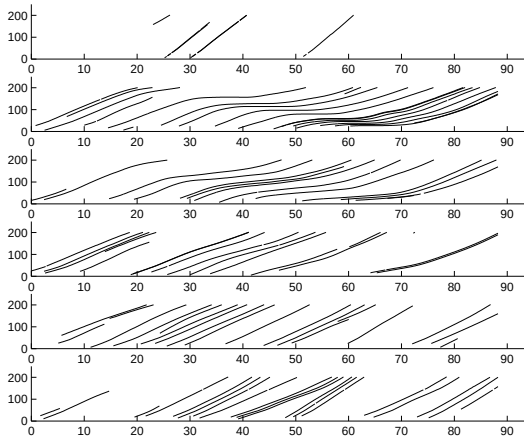


Figure 9. Resulting trajectories plotted lane by lane.

The resulting trajectories are shown in Figure 8 and Figure 9. We find that the quality of trajectories is superior (less noisy) than that of other vision-based data. We observe a good quality of shockwaves (delayed deceleration) on the second and the third lane, which will serve as useful data to generate the parameters of car-following models.

6. Conclusion and Future Work

We presented a vehicle detection and tracking system based on a model-based 3-D vehicle description algorithm. It provides high quality vehicle trajectories with accurate localization, which will bring a significant improvement in traffic flow analysis. We also introduced a new 3-D vehicle detection and description algorithm based on line features.

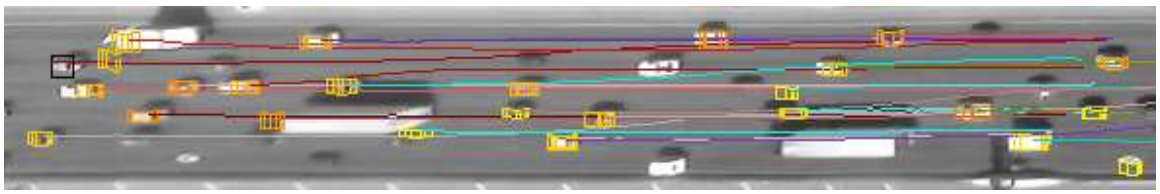


Figure 7. An example tracking result.

Our feature-based algorithm has significant advantages over image-based algorithms (such as [12]). It is fast enough that it can be applied to many other applications which require fast (or even realtime) processing. It is also flexible. It is much more free from the scale problem, and allows detection from more oblique views. The performance does not depend on the small change of the view points.

There are still many challenges and future work left:

- It is difficult to generate a large number of trajectories (trajectory database) due to the current analog data collection system. The digitization process requires significant efforts because we need 1) minimal loss of image quality and 2) perfect synchronization. In addition, the quality and/or the resolution of the images is not satisfactory due to the interlaced format of analog video. We are currently in a process of converting the data collection system into digital (with progressive scan cameras).

- We need to develop a separate algorithm for trailer truck detection. In fact, it is to find a long rectangular structure, and techniques from building detection (for example, [7]) can easily be applied.

- The tracking performance is not satisfactory due to occlusions and specular highlights. One possible solution is to apply a model-based tracking algorithm which uses line features. It can bring better tracking performance because the line features more invariant than image profiles.

- Improving the description performance can enable fine-level vehicle classification. Current transportation research on vehicle classification only discriminate small cars (such as passenger cars and SUV's) from large trucks. Our vehicle detection algorithm gives 3-D model which can be used for discriminating SUV's and pickup trucks from passenger cars. With such information, we can estimate, for example, relative effects on highway capacities among different class of vehicles, which is unprecedented in traffic analysis.

Acknowledgment

This work was supported by CalTrans/PATH (TO4154).

References

[1] A. Bensrhair, M. Bertozzi, A. Broggi, P. Miché, S. Mousset, and G. Toulminet. A cooperative approach to vision-based vehicle detection. In *Proc. IEEE Intl. Conf. on Intelligent Transportation Systems*, pages 209–214, 2001.

[2] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. A real time computer vision system for measuring traffi c parameters. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 495–501, 1997.

[3] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8:679–698, 1986.

[4] R. Ervin, C. MacAdam, J. Walker, S. Bogard, M. Hagan, A. Vayda, and E. Anderson. System for assessment of the vehicle motion environment (savme): volume i, ii, 2000.

[5] S. Gupte, O. Masoud, R. F. K. Martin, and N. P. Papanikolopoulos. Detection and classifi cation of vehicles. *IEEE Trans. Intelligent Transportation Systems*, 3(1):37–47, 2002.

[6] D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. In *Proc. European Conf. on Computer Vision*, pages A:189–196, 1994.

[7] C. Lin and R. Nevatia. Building detection and description from a single intensity image. *Computer Vision and Image Understanding*, 72(2):101–121, 1998.

[8] A. Rajagopalan, P. Burlina, and R. Chellappa. Higher order statistical learning for vehicle detection in images. In *Proc. IEEE Intl. Conf. Computer Vision*, volume 2, pages 1204–1209, 1999.

[9] H. Schneiderman and T. Kanade. A statistical model for 3d object detection applied to faces and cars. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, June 2000.

[10] P. Smith, D. Sinclair, R. Cipolla, and K. Wood. Effective corner matching. In *Proc. 9th British Machine Vision Conf.*, 1998.

[11] A. Worrall, G. Sullivan, and K. Baker. Pose refi nement of active models using forces in 3d. In *Proc. 3rd European Conf. Computer Vision*, pages 341–352, 1994.

[12] T. Zhao and R. Nevatia. Car detection in low resolution aerial image. In *Proc. IEEE Intl. Conf. Computer Vision*, 2001.

Estimating Velocity Fields on a Freeway from Low Resolution Video

Young Cho
Department of Statistics
University of California, Berkeley
Berkeley, CA 94720-3860
Email: young@stat.berkeley.edu

John Rice
Department of Statistics
University of California, Berkeley
Berkeley, CA 94720-3860
Email: rice@stat.berkeley.edu

***Abstract*— We present an algorithm to estimate velocity fields from low resolution video recordings. The algorithm does not attempt to identify and track individual vehicles, nor does it attempt to estimate derivatives of the field of pixel intensities. Rather, we compress a frame by obtaining an intensity profile in each lane along the direction of traffic flow. The speed estimate is then computed by searching for a best matching profile in a frame at a later time. Because the algorithm does not need high quality images, it is directly applicable to a compressed format digital video stream, such as mpeg, from conventional traffic video cameras. We illustrate the procedure using a 15 minute long VHS recording to obtain speed estimates on a one mile stretch of highway I-80 in Berkeley, California.**

I. Introduction

Traffic cameras offer the potential to complement or substitute for loop detectors. Because they can provide finer spatial and temporal resolution, they have many advantages over loop detectors. In principle, video from cameras can also be used to detect lane-changing, accidents, and queuing patterns and to extract macroscopic traffic parameters, such as flow, speed, and density. Also, cameras are becoming less expensive to purchase and maintain.

However, in order to use images from cameras to study traffic, a large amount of video must be processed and an efficient and practical system to extract traffic parameters is thus essential. After being digitized, an hour video can be up to a several gigabytes.

The objective of this paper is to present a simple algorithm to estimate a velocity field, localized in space and time, from video data covering a wide area with limited spatial resolution. The localization is fine enough to reveal the temporal and spatial formation and dissipation of shockwaves. To demonstrate practicality of the algorithm, we present results from a 15 minute long video filmed by a Berkeley Highway Laboratory camera. Figure 1(a) shows the layout of the Berkeley Highway Laboratory. Figure 1(b) shows a single frame covering about one mile of freeway.

Kastrinaki *et al.* [1] provide an extensive survey of state of the art traffic applications of video processing, including road traffic monitoring. Our methodology falls into the general category of optical flow, techniques of which are reviewed in Beauchemin and Barron [2]. Applications of optical flow to traffic monitoring have been based on detecting and tracking individual vehicles to estimate speed, density, and flow. The following are representative examples. Autoscope [3] detects and tracks vehicles within a detection zone (roughly a rectangle the size of a vehicle) and integrates their spatial and temporal signatures to measure their speeds. The ACTIONS system [4], detects and tracks moving objects by estimating optical flow vectors which are then clustered to create candidate moving objects. The MORIO system [5] infers polyhedral models for objects moving relative to a stationary camera. The TITAN system [6] uses mathematical morphology to extract individual vehicle features, aggregates them into individual vehicles, and tracks them. It is capable of monitoring stretches of the motorway of up to about 1000 feet, depending on the height of the camera. The images of individual cars need to be separated. In Fathy and Siyal [7], a morphological edge detector and background differencing are used to identify and track vehicles and calculate traffic parameters. Coifman *et al.* [8] developed a feature-tracking algorithm to extract individual vehicle trajectories from video data by detecting predefined features from images, grouping them, and tracking the groups of features to produce trajectories. Dailey *et al.* [9] developed a method to estimate mean traffic speed, using an edge-detecting algorithm to find centroids and estimating mean speed from centroid movement in successive images.

However, algorithms that rely on identifying and tracking individual vehicles are not feasible for use with images of poor quality and over a wide area. If the spatial resolution is poor, vehicles in the frame do not show distinctive lines or features throughout the whole span of view and thus can be neither clearly identified nor tracked. If a vehicle only occupies a small number of pixels, its features may be hard to identify and furthermore can change as the precise position of the vehicle within the pixel grid changes. Vibration due to wind causes further difficulties, as do shadows and occlusions in congested traffic. Grant *et al.* [10] report on an extensive test of Autoscope on freeways in Atlanta, Georgia, showing that counts degraded in accuracy as the distance of the count location to the camera increased out to a maximum of about 400 ft. Although they did not directly measure the quality of speed estimation as a function of distance, they conjecture that it is similar to that of the volume counts.

In contrast, the method set forth in this paper does not

depend on explicitly identifying and tracking individual vehicles. We demonstrate that it is robust to occlusion and shadows, which can be seen in Figure 1(b), and to the camera motion that is evident in the videos. We compare the estimates obtained from the video to those from high frequency loop detectors.

II. Data

The data used in this study were generated from a 15-minute video of about a one mile stretch of highway I-80 in Berkeley, California. A video test bed, the Berkeley Highway Laboratory, consists of 12 cameras on the roof of Pacific Park Plaza, a 30-story building beside the highway. The analog cameras have S-VHS video recorders attached. Figure 1(a) illustrates the setup and the coverage of each camera.

Among six cameras looking north, the field of view of camera N6 is furthest down I-80E. It covers the longest stretch (about one mile long), from the Ashby Avenue on-ramp to University Avenue off-ramp. But it produces relatively poor quality images due to the poor angle and resolution. Figure 1(b) is a still frame from camera N6. The spatial resolution is such that a pixel in the near field of view is about 5 feet, whereas those furthest from the camera are about 15 feet. The images also suffer from occlusions by vehicles and their shadows and from camera motion.

Despite the poor quality of the images, the data from camera N6 is potentially informative, for example for studying the effect of on-ramp flow on highway performance. Also, it covers the longest stretch, about one mile, and in principle, information extracted from this camera can be combined with that from the higher resolution cameras, which have smaller fields of view. Algorithms developed to analyze images from camera N6 should be applicable to other conventional traffic cameras.

For the analysis, a 15 minute long tape was digitized at the rate of 10 frames per second and the results saved in ppm file format. Each frame, like Figure 1(b), has 800x640 pixels and each pixel has intensities for red, green, and blue channels.

III. Methodology

Because of the camera resolution, it would be at best extremely difficult to apply a feature detecting and tracking algorithm. Hence, it is necessary to develop a new way to extract information from the images. We developed an algorithm for this purpose. The algorithm proceeds as follows: First, an "*intensity profile*" of each lane in the direction of traffic flow is extracted from each frame. Arranging the profiles in time order gives intensity flow in the time-space domain, in which vehicles appear as stripes, or moving peaks. Second, the speed estimate at (t, x) in the time-space domain is computed via searching a pair of best matching patterns at time $t + \tau$ and $t - \tau$ in terms of the L_1 norm (sum of absolute values of differences) and estimated as the slope of the line connecting the two centers of the pair. For discussion and references to such correlation-based matching methods, see Beauchemin and Barron [2].

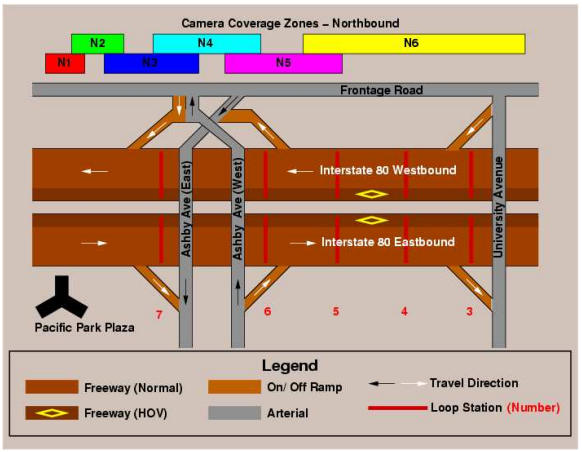
The algorithm has a number of advantages. First, it does not involve vehicle tracking, which can be computationally very expensive, and hence makes it more efficient to process a large amount of data. Secondly, in estimating local speed it does not compute a gradient, or a weighted average of speeds of moving features around the location of interest. It uses the L_1 norm in finding the best matching pattern and hence is robust to noise. Later in this section, we show that under certain conditions the algorithm is equivalent to finding a weighted median of the speeds of moving peaks.

A. Intensity Profiles

To generate the intensity profile for a lane, a mask (M) is created on a particular frame, a so-called reference frame. The mask passes intensities of pixels in a region of interest. A masked image is presented at the top of Figure 2. Once masked, a frame contains intensities of pixels in the region. The pixels have three integers between 0 and 255 for red, green, and blue channels. Because these three intensities are strongly correlated at the pixel level, we take the average of red, green, and blue intensities. Then, we scan across the lane (orthogonal to the direction of traffic) and calculate the maximum average intensity along the direction of traffic, which we refer to as a maximum intensity profile. The maximum intensity profile corresponding to the top image of Figure 2 is presented in the bottom of Figure 2.

The mask should be created for each frame, because the camera may be constantly shaking due to strong wind. To create masks automatically, we find mappings (Ψ_t) from the reference frame to every frame during the time period and use the new transformed masks ($\Psi_t(M)$). First, we choose four fixed objects (reference objects) on the reference frame and place square windows centered at them. Then, in each frame we search for the best matching patterns corresponding to the squared areas centered at the reference objects. Once the patterns are found, we use the coordinates of the center pixels of the square windows to compute the projection matrix. For more information, refer to chapter 5 of Hartley and Zisserman [11].

We repeat this process for each frame and stack the maximum intensity profiles in time order to obtain an intensity flow on the time-space domain. The intensity fades as the vehicle moves away from camera, and the road surface intensity varies depending on the location. To correct for this, at each location of the highway we determine the maximum and minimum intensity during the 15 minutes and form the ratio of the difference between the intensity and minimum to the difference between maximum and minimum. After this background correction, the intensity is standardized between 0 and 1. To change units from pixels to feet, we computed the projection matrix from image to the real world, using the real dimension of I-80 (For a detailed computation, also refer to chapter 5 of Hartley and Zisserman [11]). As a final step, we interpolate the intensities on a finer grid, which are equally spaced by about 5 feet. Two examples are presented in Figure 3.



(a) Berkeley Highway Laboratory Layout

(b) A frame from Camera N6

Fig. 1. Camera Setup

The resulting intensity flow is similar to a trajectory plot in the time-space domain in that the stripes, or moving peaks, contain information about the traffic flow on the highway. But it differs in that one curve does not necessarily correspond to one vehicle. Rather one vehicle can be shown as two lines, or two vehicles traveling closely together can appear as one moving peak. Small or dark vehicles may be barely perceptible. Fine discrimination is not needed for subsequent speed estimation.

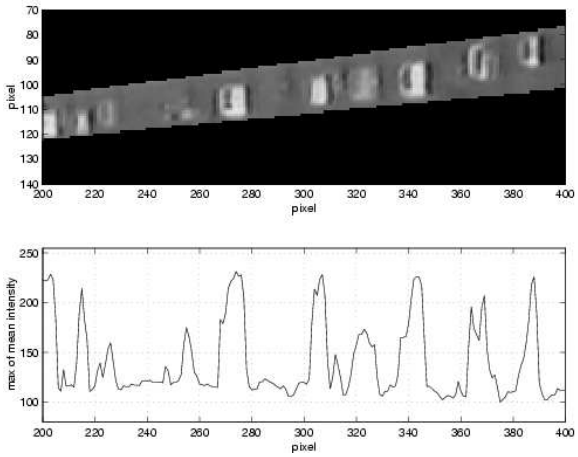


Fig. 2. A filtered image and the corresponding intensity profile

B. Speed Estimation

The idea behind the estimate is that a locally constant intensity pattern represents travel at a locally constant speed. Hence, after a short period of time, the same pattern will show up again at the travel distance above the previous location in the time-space domain.

To estimate the local speed at (t, x) in the time space domain, we choose two rectangular windows of size w_t by w_x and center one at $(t + \tau, x)$ and the other at $(t - \tau, x)$. We then move the window centered at $t + \tau$ up (increase x) and the other one at $t - \tau$ down (decrease x), computing the L_1 norms for the pair. The norm will be minimized at the travel distance. By the nature of the image, the shifts are integer-valued. To deal with this discretization, we fit the L_1 norms near the minimizer to a quadratic function to find the interpolated distance which minimizes the norm. The speed is estimated as the ratio of the distance to the shifting parameter, τ, which is equivalent to the slope between the two centers of the pair of the best matching patterns.

In the examples presented here, we use the L_1 norm criteria and set the window size to 30 seconds by 90 feet, τ as 3 second, and the searching area as 0 to 80 MPH. The step by step outline of the algorithm is as follows.

Let $I(t, x)$ denote the intensity at location (t, x) in the time-space domain.

1) Fix the local window size, $w_t \times w_x$, and the shift parameter τ,
2) For $d = 0, \ldots, m$, compute
$$D(d) = \sum_{t=t_0-w_t}^{t_0+w_t} \sum_{x=x_0-w_x}^{x_0+w_x} |I(t+\tau, x+d) - I(t-\tau, x-d)|.$$
3) Fit $D(d)$ near the minimum to a quadratic function of d.
4) Find d_0, which minimizes D.
5) The speed is estimated as d_0/τ.

Note that the algorithm does not involve gradient computation or slopes for individual lines on the intensity flow. After running the algorithm, we run a 2-dimensional median filter to remove noise, of size 40 second by 370 feet.

Also note that it simultaneously shifts two square windows (*centered shifting*), instead of fixing one window centered at (t, x) and shifting the other window at either $t + \tau$ (*forward shifting*) or $t - \tau$ (*backward shifting*) in time. As can be seen

from a Taylor series expansion, derivatives are estimated more accurately by central differences than by un-centered ones. The derivative of a quadratic function is estimated exactly by central differencing, but not by one-sided differencing.

Because the algorithm involves computing the sums of absolute difference over 2 square windows, it can be quite slow. To speed up, we use a subgrid within the square window instead of using all the intensities. Based on the empirical comparisons, we find even a coarse grid of resolution one second by about 10 feet (using 5% of data points) is sufficient to produce an estimate equally good as using all the data. Also, we need not evaluate the estimate at every location. Instead, we estimate the speed on a sparse grid and then interpolate.

To gain some insight into the nature of the estimate produced by the algorithm we now consider an idealized continuous version. Suppose features, $A_j(t,x)$, have disjoint supports I_j, are parabolic on I_j (3rd and higher order derivatives are zeros), and travel at the speed of v_j. The features correspond to the contributions of individual vehicles to the intensity profile, which we write as

$$f(t,x) = \textstyle\sum_j A_j(t,x) = \sum_j A_j(x - v_j t).$$

Now consider the minimizer of

$$\tfrac{1}{2\tau} \int_{t=t_0-w_t}^{t_0+w_t} \int_{x=x_0-w_x}^{x_0+w_x} |f(t+\tau, x+d) - f(t-\tau, x-d)|\, dxdt.$$

For small d and τ

$$\begin{aligned}
\hat{v} &= \arg\min_d \frac{1}{2\tau} \int_t \int_x |f(t+\tau, x+d) \\
&\quad - f(t-\tau, x-d)| dxdt \\
&= \arg\min_d \frac{1}{2\tau} \int_t \sum_j \int_x |A_j(t+\tau, x+d) \\
&\quad - A_j(t-\tau, x-d)| dxdt \\
&= \arg\min_d \frac{1}{2\tau} \int_t \sum_j \int_x \left|2(d - v_j\tau) A_j'(x - v_j t)\right| dxdt \\
&= \arg\min_d \sum_j \left|\frac{d}{\tau} - v_j\right| \int_t \int_x \left|A_j'(x - v_j t)\right| dxdt \\
&= \arg\min_d \sum_j \frac{S_j}{S} \left|\frac{d}{\tau} - v_j\right|
\end{aligned}$$

where $S_j = \int_{t_0-w_t}^{t_0+w_t} \int_{x_0-w_x}^{x_0+w_x} |A_j'(x - v_j t)| dxdt$ and $S = \sum_j S_j$. The minimizer of the final expression above is a weighted median of the individual velocities v_j in which the weights are S_j/S. That is, it is the median of a discrete probability distribution which has masses S_j/S on the values v_j . Vehicles with large derivatives of their individual intensity profiles thus contribute most heavily to the estimate. The median, however, is insensitive to extreme velocities. By contrast, if we were to use the sum of squared deviations rather than the sum of absolute deviations, the argument above shows that the estimate would be a weighted mean, and less robust to extreme v_j . This argument formalizes the notion that the shifting and matching algorithm estimates a weighted median velocity over a region of space and time.

IV. Results

Two intensity flows during 15 minutes from 3:00pm on 17th of December 2001 to 3:15pm on the same day are presented in Figure 3. We picked two lanes; the right-most (5th) lane of I-80E and the 3rd lane of I-80W. We chose the two lanes for the following reasons. The 5th lane merges with the Ashby on-ramp at the near field of the frame (at around 500 feet) and the inflow creates congestion. The 3rd lane of I-80W experienced the worst stop-and-go traffic and had more trucks than any other lanes during the 15 minutes.

During the 15 minutes, the east bound traffic experienced moderate congestion, shown in the intensity flow as changes in slopes of lines. Examining the figure carefully, one can see some lines disappear and appear, caused by lane-changing and occlusions from the shadows of vehicles traveling in the next lane. The west bound lanes experienced very heavy traffic. Also recall that in the 3rd lane there were the most trucks. In the intensity flow, trucks appear as broad stripes. In the intensity flow, we observe flat patterns lining up, which shows shockwaves propagating against the traffic. The lane also experienced the most frequent occlusion from vehicles and their shadows, due to the stop and go traffic in the next lane. In this lane, there are marks on the road to signify the off-ramp and they create horizontal stripes around 30, 60, 460 pixels even after background correction. However, the speed estimate is robust to these artifacts.

From the speed estimate of I-80E, we observe congestion due to the inflow from the Ashby on-ramp and corresponding shockwaves. We suspect that a traffic signal on Ashby Avenue caused periodic fluctuations in inflow and hence the pulsating series of shockwaves. Also note a pronounced shockwave originating at around 360 second and 0.5 mile from the University exit and travelling against the traffic at about 10 MPH.

The I-80W speed estimate shows even stronger oscillations shockwave evolution, and some variation in their velocities of propagation. The figure shows that the shockwaves typically travel at about 10 MPH. Because we do not observe where they originated and dissipated, we cannot verify how long the shockwaves traveled before dissipating, based solely on camera # 6. For now, we conjecture that the shockwaves were created further downstream on I-80W, about 1.3 miles south of the Ashby off-ramp, at the notorious split of I-80W into I-580S, I-880, and I-80W. Further investigation using tapes from cameras # 1-5 would reveal more information.

To check our estimates, we compared them to loop detector data. Loop detectors are located at stations 3, 4, and 5 in the order of distance from the University exit; refer to Figure 1(a). Unfortunately, the stretch had been paved recently and we could not locate precisely where the loops were. So, we approximated the loop locations by those of the cabinets and pull-boxes of the loop counter stations, which are located at the side of the wall of I-80E. In Figure 5, the dots are the point

estimates(vehicle by vehicle) from the loop data. The speed estimates corresponding to the cabinet(pull-box) locations are shown as the solid lines.

TABLE I
MEANS AND STANDARD DEVIATIONS (MPH)

	East bound	West bound
Station 3	-3.7 (2.1)	-2.0 (3.8)
Station 4	0.3 (1.6)	0.0 (1.8)
Station 5	4.3 (1.8)	1.6 (3.1)

The figures show that the estimates are very close to the loop data during the 15 minutes and pick up most of the oscillations. There are some systematic differences, which may be attributable to the imprecision of loop detector locations. Note that the speed ranges and traffic conditions for the west and east bound lanes are very different, yet the estimates are very consistent in both cases. The means and standard deviations of the errors between the estimates and the loop data are reported in Table I.

V. CONCLUSIONS AND DISCUSSION

The results above demonstrate the potential of our algorithm for processing a video recording from a traffic camera, providing a useful tool to study numerous traffic issues, such as the effect of an on-ramp, the evolution and dissipation of queuing and congestion, and for monitoring highway performance. Despite its poor quality image, camera N6 provides very useful information in these regards. For some purposes, simple functionals of the estimated velocity field may be sufficient. For example, travel times can be estimated by tracing through the field, or the average velocity over space at a given time can be computed.

Although the results we have shown are quite reasonable, we will study several issues in more detail in the future. One is the choice of the region on which to base shifting and matching. In principle, the rectangle could be as small as one pixel in time and several pixels in space, or vice-versa. The computing time is faster for smaller rectangles, but the results are noisier (a defect which can be ameliorated, however, by smoothing the estimates). Smaller rectangles yield a finer resolution in space and time, but again at the cost of noise. Larger rectangles localize less and are computationally more expensive, but produce less noisy estimates. In principle, the regions need not be rectangular and weight functions, such as Gaussian kernels, can be used instead of uniform weighting. Initial experimentation indicates that the final results are quite insensitive to these choices, but further study is necessary to optimize the algorithm for speed and accuracy.

In addition to further improving speed estimation, we are developing algorithms to extract the other macroscopic parameters, flow and density, from the intensity profiles. This is more difficult than velocity estimation. Counting is more feasible in the near field of view, and the results can be propagated through the estimated velocity field to obtain estimates of density and flow in the far field of view. We will also investigate the still more challenging problem of detecting lane changing.

Finally we mention that we have used our method on MPEG and AVI compressed video, with little degradation of the results. This may be useful if data are to be transmitted prior to analysis.

ACKNOWLEDGEMENTS

This research was supported by Partners in Advanced Highways and Transportation and by a grant from the National Science Foundation. We wish to thank Dan Lyddy for assistance with the data, Ryan Lovett and Phil Spector for assistance with computation and Peter Bickel, Z Kim, Jaimyoung Kwon, and Erik van Zwet for helpful discussions.

REFERENCES

[1] V. Kastrinaki, M. Zervakis, and K. Kalaitzakis, "A survey of video processing techniques for traffic applications," *Image and Vision Computing 21*, pp. 359–381, 2003.

[2] S. Beauchemin and J. Barron, "The computation of optical flow," *ACM Computing Surveys*, vol. 27, no. 3, pp. 433–467, Nov. 1995.

[3] P. Michalopoulos, "Vehicle detection video through image processing: the autoscope system," *IEEE Trans. Veh. Technol.*, vol. 40, pp. 21–29.

[4] W. Enkelmann, "Interpretation of traffic scenes by evaluation of optical flow fields from image sequences," *IFAC Control, Computers, Communications in Transportation*, 1989.

[5] L. Dreschler and H.-H. Nagel, "Volumetric model and 3d-trajectory of a moving car derived from monocular tv-frame sequences of a street scene," *Computer Vision, Graphics, and Image Processing*, vol. 20, pp. 199–228, 1982.

[6] J. Blosserville, C. Krafft, F. Lenoir, V. Motvka, and S. T. Beucher, "New traffic measurements by image processing," *IFAC Control, Computers, Communications in Transportation*, pp. 35–42, 1989.

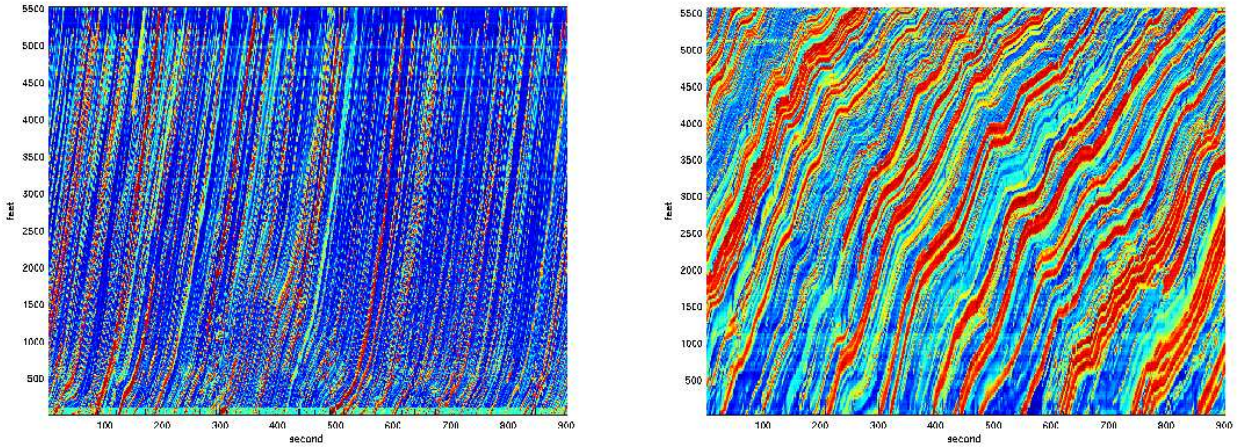
[7] M. Fathy and M. Siyal, "An image detection technique based on morphological edge detection and background differencing for real-time traffic analysis," *Pattern Recognition Letters*, vol. 16, pp. 1321–1330, 1995.

[8] 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. 6C(4), pp. 271–288, Aug. 1998.

[9] D. Dailey, F. Cathey, and S. Pumrin, "An algorithm to estimate mean traffic speed using uncalibrated cameras," *IEEE Trans. Intell. Transport. Syst.*, vol. 1, no. 2, pp. 98–107, June 2000.

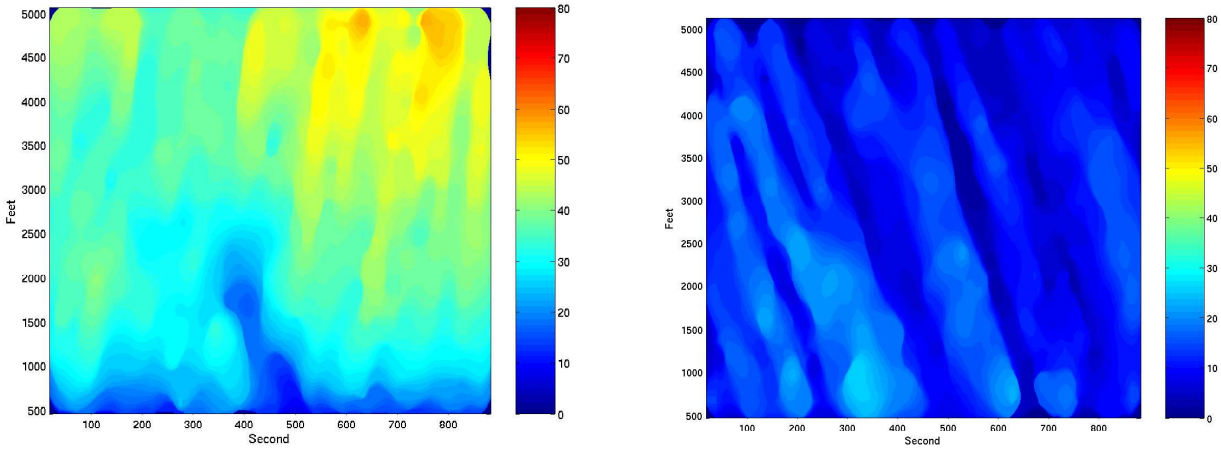
[10] C. Grant, B. Gillis, and R. Guensler, "Collection of vehicle activity data by video detection for use in transportation planning," *ITS Journal 5*, pp. 342–361, 2000.

[11] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, UK: Cambridge University Press, 2000.



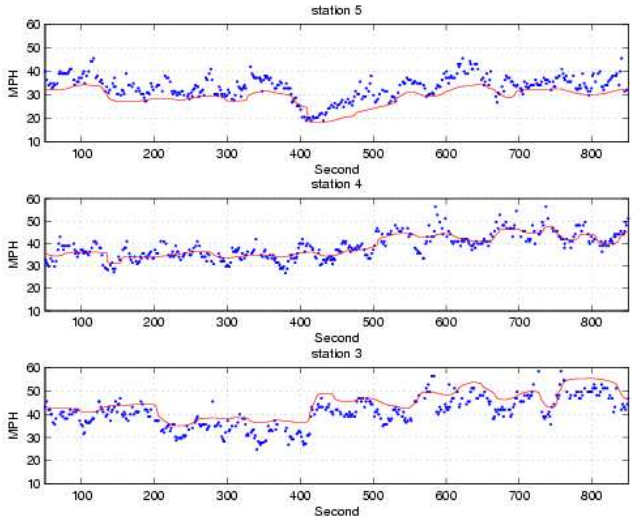
(a) I-80 East Bound (b) I-80 West Bound

Fig. 3. Intensity flows

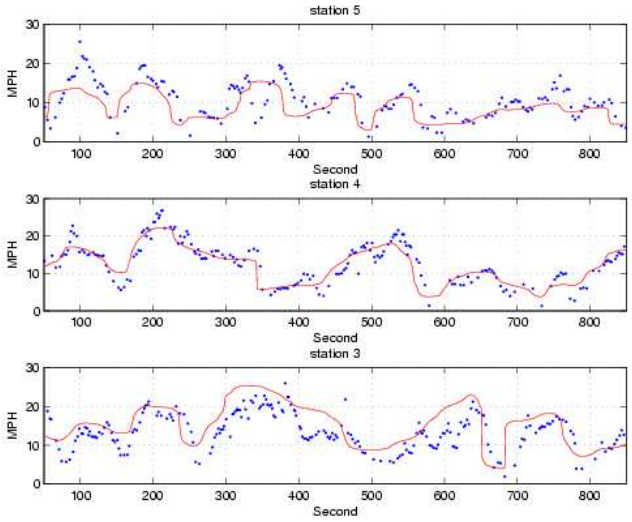


(a) I-80 East Bound (b) I-80 West Bound

Fig. 4. Estimated velocity fields



(a) I-80 East Bound



(b) I-80 West Bound

Fig. 5. Comparsion between the estimate and the loop data