Research Report
Agreement T1803, Task 48
Automated CCTV Cameras

Algorithms for Estimating Mean Vehicle Speed Using Uncalibrated Traffic Management Cameras

by

Todd Nelson Schoepflin and Daniel J. Dailey
ITS Research Program
College of Engineering, Box 352500
University of Washington
Seattle, Washington 98195-2500

Washington State Transportation Center (TRAC)
University of Washington, Box 354802
University District Building, Suite 535
1107 N.E. 45th Street
Seattle, Washington 98105-4631

Washington State Department of Transportation
Technical Monitor
Pete Briglia
Manager, Advanced Technology Branch

A report prepared for

Washington State Transportation Commission
Department of Transportation
and in cooperation with
U.S. Department of Transportation
Federal Highway Administration

October 2003

TECHNICAL REPORT STANDARD TITLE PAGE

1. REPORT NO. WA-RD 575.1	2. GOVERNMENT ACCESSION NO.	3. RECIPIENT'S CATALOG NO.

4. TITLE AND SUBTITLE ALGORITHMS FOR ESTIMATING MEAN VEHICLE SPEED USING UNCALIBRATED TRAFFIC MANAGEMENT CAMERAS	5. REPORT DATE October 2003 6. PERFORMING ORGANIZATION CODE
7. AUTHOR(S) Todd Nelson Schoepflin and Daniel J. Dailey,	8. PERFORMING ORGANIZATION REPORT NO.
9. PERFORMING ORGANIZATION NAME AND ADDRESS Washington State Transportation Center (TRAC) University of Washington, Box 354802 University District Building; 1107 NE 45th Street, Suite 535 Seattle, Washington 98105-4631	10. WORK UNIT NO. 11. CONTRACT OR GRANT NO. Agreement T1803, Task 48
12. SPONSORING AGENCY NAME AND ADDRESS Research Office Washington State Department of Transportation Transportation Building, MS 47372 Olympia, Washington 98504-7372 Doug Brodin, Project Manager, 360-705-7972	13. TYPE OF REPORT AND PERIOD COVERED Research Report 14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

This study was conducted in cooperation with the U.S. Department of Transportation, Federal Highway Administration.

16. ABSTRACT

This report documents the second project, in a series of three research projects funded by the Washington State Department of Transportation (WSDOT), that will enable already deployed, un-calibrated CCTV cameras to be used as traffic speed sensors. The principle traffic speed sensors currently deployed by WSDOT are inductance loops; however, in some locations it is impractical or too expensive to install loops. In addition, a large number of un-calibrated cameras are already in place and being used by the traffic management operators to qualitatively assess traffic both on the freeway and on arterials. These projects will leverage the existing cameras to provide a quantitative measurement of traffic speed similar to that which can be obtained using loops and suitable for use in the Traffic Management System (TMS) without installing loops in the roadway. The implementation of this research will culminate with software that creates an automated system compatible with the existing TMS. This system will leverage the existing camera investment to create a new set of speed sensors that increases the geographic extent of the TMS's quantitative surveillance capabilities.

In the second phase, reported on here, roadway features are used to augment the camera calibration. This overcomes the occlusion problem, or apparent blending together of small vehicles as seen in the far field of the camera images, that existed in the first phase. Activity maps, fog lines, and vanishing points are a few of the additional features used, and the details of these algorithms are described in this report. These results have also been peer reviewed and published.

17. KEY WORDS CCTV cameras, video image processing, calibration, speed sensor, vehicle length distribution, Kalman filter	18. DISTRIBUTION STATEMENT No restrictions. This document is available to the public through the National Technical Information Service, Springfield, VA 22616

19. SECURITY CLASSIF. (of this report)	20. SECURITY CLASSIF. (of this page)	21. NO. OF PAGES	22. PRICE
None	None		

DISCLAIMER

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

The following is a list of all algebraic quantities and symbols used in this document. It is alphabetized first by the Greek alphabet, then by the English alphabet. The chapter that first uses the term is included in parentheses.

α:	angle between the v-axis and a line in an image (positive counter-clockwise) (Ch. 3).
α_0, α_1, α_2:	intermediate quantities used in solving for f (Ch. 2).
β:	angle representing the tilt of the road, i.e., whether a marble placed in the middle of the road will roll left ($\beta > 0$) or right ($\beta < 0$) (Ch. 2).
γ:	angle of the gradient direction for a given pixel (Ch. 3).
$\|\nabla I\|$:	magnitude of the gradient for an image defined as $(I_x^2 + I_y^2)^{-1/2}$ (Ch. 3).
$\Delta\Phi$:	quantization stepsize for Φ (Ch. 3).
Δt:	time between image frames (Ch. 5).
ΔY:	distance between two points in the X-Y-Z coordinate system (Ch. 2).
$\Delta Y'$:	distance along the road (Y_c'-axis) between two points in the X_c'-Y_c'-Z_c' coordinate system (Ch. 2).
$\delta Y'$:	smallest increment of Y' represented by a pixel in the image (Ch. 3).
ψ:	angle representing the slope of the road, i.e., whether it is going uphill ($\psi < 0$) or downhill ($\psi > 0$) relative to the earth (Ch. 2).
η:	fraction between 0 and 1 indicating the number of bootstrap samples used to estimate u_1 from the full set of estimates (Ch. 3).
λ:	line parameter indicating the position along line L_x (Ch. 2).
ρ:	parameter for parameterizing a line for the Hough transform. It represents the minimum distance from the origin to the line (Ch. 3).
ρ_{corr}:	correlation coefficient for locating a template in A_{top}' (Ch. 3).
ρ_{min}:	threshold below which we do not accept the hypothesis that the template T matches A_{top}'' at the current offset (Ch. 3).
Φ:	line angle parameter for the Hough transform. It encodes the angle of the line relative to horizontal. The angle itself is measured between the u-axis and a line perpendicular to the line itself (Ch. 3).
Φ_{max}:	threshold for the angles of lines used to estimate u_1. If the median of the distribution exceeds this value we reject the hypothesis that we can estimate u_1 for the current scene (Ch. 3).

ϕ:	down angle of the camera with respect to horizontal (Ch. 2).
ϕ_{max}:	maximum value for ϕ that satisfies noise and line visibility requirements (Ch. 2).
θ:	pan angle of the camera with respect to the road boundaries (Ch. 2).
θ_{max}:	maximum value for θ that satisfies noise and line visibility requirements (Ch. 2).
τ:	generic index into a cross-covariance or autocovariance sequence; see immediate context for clarification (Ch. 3).
τ_1:	offset in the autocovariance function of A_{top} where the first peak is located (Ch. 3).
τ_{12}:	distance between v_1 and v_2 after removing the effects of perspective projection; this quantity is proportional to the physical distance traveled (Ch. 3).
τ_L:	tip-to-tip distance of lane markers after removing the effects of perspective projection. This is a very important quantity for calibrating the camera since it is directly proportional to the known physical distance of 40 feet (Ch. 2).
τ_{lag}:	the location of the first significant peak in the autocovariance function $R(\tau)$ (Ch. 3).
σ:	standard deviation parameter for a gaussian distribution (Ch. 3).
σ_a:	standard deviation for average vehicle acceleration (Ch. 5).
$\sigma_{Atop''}$:	standard deviation for the samples in A_{top}'' (Ch. 3).
σ_{b1}:	standard deviation for b_1 (Ch. 2).
σ_{b1}:	standard deviation for b_1 (Ch. 4).
σ_{b2}:	standard deviation for b_2 (Ch. 2).
σ_{b2}:	standard deviation for b_2 (Ch. 4).
$\sigma_{\tau L}$:	standard deviation for τ_L (Ch. 2).
$\sigma_{\Delta Y'}$:	standard deviation for $\Delta Y'$ (Ch. 2).
σ_d:	standard deviation for d (Ch. 2).
σ_L:	standard deviation for L (Ch. 2).
σ_{m1}:	standard deviation for m_1 (Ch. 4).
σ_{m2}:	standard deviation for m_2 (Ch. 4).
σ_{rel}:	standard deviation of a normally distributed random variable relative to its mean value, i.e., $\sigma_{rel} = \sigma/\mu$ where μ and σ are the mean and standard deviation of the normal distribution, respectively (Ch. 3).
σ_s:	standard deviation for average vehicle speed (Ch. 5).
σ_T:	standard deviation for the samples in T (Ch. 3).
σ_{u0}:	standard deviation for u_0 (Ch. 2).

$\sigma_{u1,\eta}$:	population standard deviation for taking a fraction η of all the samples in the distribution of u_1 estimates (Ch. 3).
σ_{v0}:	standard deviation for v_0 (Ch. 2).
σ_w:	standard deviation for w (Ch. 2).
ϕ_{min}:	minimum value for ϕ that satisfies noise and line visibility requirements (Ch. 2).
θ_{min}:	minimum value for θ that satisfies noise and line visibility requirements (Ch. 2).

$\mathbf{a}$:	direction cosine vector for line L_x (Ch. 2).
A:	generic activity map (Ch. 3).
a:	Kalman filter state representing average vehicle acceleration (Ch. 5).
$\mathbf{a_1}$:	direction cosine vector for line L_1 (Ch. 2).
$\mathbf{a_2}$:	direction cosine vector for line L_2 (Ch. 2).
A_{bot}':	a morphological bottom-hat transform of the $A_{rel}(u)$ signal found by using the same kernel as that used to find A_{top}' (Ch. 3).
A_{max}:	1-D signal in which each sample indicates the maximum of the activity map along one of the lines emanating from (u_0,v_0) and connecting to one of the pixels in the bottom row of the image (Ch. 3).
A_{min}:	1-D signal in which each sample indicates the minimum of the activity map along one of the lines emanating from (u_0,v_0) and connecting to one of the pixels in the bottom row of the image (Ch. 3).
$A_{rel}(u)$:	a 1-D sequence in which each sample is the average value of the activity map along a line connecting (u_0,v_0) and pixel u_j on the bottom row of the image (Ch. 3).
A_{top}:	a morphological top-hat transform of the $A_{rel}(u)$ signal (Ch. 3).
A_{top}':	a morphological top-hat transform of the $A_{rel}(u)$ signal found by using a kernel that may be more narrow than that used to find A_{top} (Ch. 3).
A_{top}'':	subset of A_{top}' in which the template currently overlaps A_{top}' (Ch. 3).
a_x-a_y-a_z:	components of the vector $\mathbf{a}$ (Ch. 2).
$b(j)$:	u-intercept parameter for jth line in the optimization to find (u_0,v_0) (Ch. 3).
B:	binary image generated by thresholding the gradient magnitude of an image (Ch. 3).
b:	intercept parameter for describing a line as $u = mv + b$ (Ch. 3).
$\mathbf{b}$:	vector for an arbitrary 3-D point contained by line L_x (Ch. 2).
$\mathbf{b_1}$:	3-D coordinates of a point contained in line L_1 (Ch. 2).

b_1:	u-axis intercept for the road boundary that is closest to the camera when using slope-intercept form (Ch. 2).
$\mathbf{b_2}$:	3-D coordinates of a point contained in line L_2 (Ch. 2).
b_2:	u-axis intercept for the road boundary that is farthest from the camera when using the slope-intercept form (Ch. 2).
B_{L1}:	1-D signal sampled along the line connecting the vanishing point and $u_{spike,left}$ on the bottom row of the image. The samples are taken from a binary image obtained by thresholding I_{top2} (Ch. 3).
B_{L2}:	1-D signal sampled along the line connecting the vanishing point and $u_{spike,right}$ on the bottom row of the image. The samples are taken from a binary image obtained by thresholding I_{top2} (Ch. 3).
b_x-b_y-b_z:	components of the vector **b** (Ch. 2).
c_1, c_2, c_3, c_4, c_5:	constants representing summation terms in the optimization to solve for (u_0,v_0) (Ch. 3).
C_{max}:	maximum value for a given column of $C_{ss}(u,\tau)$ excluding the main lobe (Ch. 3).
C_{min}:	minimum value for a given column of $C_{ss}(u,\tau)$ (Ch. 3).
$C_{ss}(u,\tau)$:	an image whose columns contain the autocovariance function for each of the columns in $T_{2d}(u,r)$ (Ch. 3).
$C_{t,t+1}(\tau)$:	noncircular cross-covariance function between $S_2(r,t)$ and $S_2(r,t + 1)$ (Ch. 5).
$C_{total}(\tau)$:	IIR-filtered cross-covariance function accumulated over the process of estimating vehicle speeds (Ch. 5).
$C_{xx}(\tau)$:	autocovariance function for a finite multi-cycle square wave (Ch. 3).
d:	on the ground plane, the perpendicular distance between the camera and the nearest road boundary (Ch. 2).
D:	duty cycle of the lane markers, i.e., L_2/L_{tot} (Ch. 3).
D_{img}:	absolute value of the intensity difference between two images (Ch. 3).
D_{max}:	the maximum distance allowed between adjacent peaks in $A_{rel}(u)$ (Ch. 3).
d_{min}:	the smallest distance from u_{max} in $A_{rel}(u)$ where certain criteria are met (Ch. 3).
$\mathrm{E}[\cdot]$:	expectation operator (Ch. 3).
E_1:	the full ensemble of autocovariance functions composed of all the columns of $C_{ss}(u,\tau)$ (Ch. 3).
E_2:	a subset of E_1 meeting certain criteria (Ch. 3).
E_3:	a subset of E_2 meeting certain criteria (Ch. 3).
$EE_2(t)$:	spatial variance of $S_2(r,t)$ (Ch. 5).
E_{rel}:	error in the computer estimate of a parameter relative to a hand-based estimate (Ch. 4).

F:	the distance between the camera lens and the ground plane (Ch. 2).
f:	focal length of the camera (Ch. 2).
F_0:	a non-negative functional containing the sum of the squared differences between points on all the lines and the vanishing point (u_0,v_0) (Ch. 3).
F_1:	a non-negative functional indicating the square of the distance between (u_0,v_0) and (u_c,v_c) (Ch. 3).
h:	camera height (Ch. 2).
H:	image height (Ch. 2).
H_k:	the 2-D perspective transformation between two images (Ch. 1).
i:	index variable (examine context) (Ch. 3).
I_{bg}:	background image for a scene (Ch. 3).
I_I, I_{i+1}:	image frames at two adjacent time steps (Ch. 3).
I_i:	i^{th} image in the image sequence (Ch. 3).
i_{max}:	index of the maximum peak in $A_{rel}(u)$ (Ch. 3).
I_{spike}:	the signal I_{spike}' convolved with a low-pass kernel (Ch. 3).
I_{spike}':	1-D signal in which each sample represents the average value of I_{top} along a line emanating from the vanishing point and connecting with a pixel in the bottom row of the image (Ch. 3).
$I_{top,new}$:	top-hat image after normalization and gamma-correction (Ch. 3).
I_{top}:	top-hat image for a scene containing no vehicles (Ch. 3).
I_{top2}:	the bottom half of the I_{top} image (Ch. 3).
I_x:	horizontal gradient of an image (Ch. 3).
I_y:	vertical gradient of an image (Ch. 3).
j:	index variable (examine context) (Ch. 3).
K_1:	derivative-of-gaussian convolution kernel (Ch. 3).
$\mathbf{K}$:	Kalman gain (Ch. 5).
K_2:	derivative-of-gaussian kernel used to detect horizontal image edges (Ch. 5).
L:	known distance along the road (Y_c'-axis) used to calibrate the camera in Method 3 (Ch. 2).
L_1:	line defined in 3-D that represents the road boundary nearest to the camera (Ch. 2).
L_2:	line defined in 3-D that represents the road boundary away from the camera (Ch. 2).
$L_{1,mark}$:	length of the blank space between lane markers (Ch. 3).
$L_{2,mark}$:	length of the lane marker (Ch. 3).

L_{tot}:	total tip-to-tip distance defining the lane marker period. $L_{tot} = L_{1,mark} + L_{2,mark}$ (Ch. 3).
$m(j)$:	slope parameter for j^{th} line in the optimization to find (u_0,v_0) (Ch. 3).
$\mathbf{M}(k)$:	measurement matrix for the Kalman filter at time step k (Ch. 5).
m:	slope parameter when describing a line as $u = mv + b$ (Ch. 3).
m_1:	slope in the image for the road boundary that is closest to the camera, when using the slope-intercept form (Ch. 2).
m_2:	slope in the image for the road boundary that is farthest from the camera, when using the slope-intercept form (Ch. 2).
m^i_j:	the homogeneous coordinates of the projection of the j^{th} point onto the i^{th} camera (Ch. 1).
$\mathrm{N}(\cdot,\cdot)$:	designation of a normally distributed random variable for which the first parameter indicates the mean and the second indicates the standard deviation of the distribution (Ch. 3).
n:	indices into a kernel sequence (Ch. 3).
N_{all}:	total number of lines found for estimating u_1 (Ch. 3).
N_{cam}:	number of cameras used in a Euclidian reconstruction from multiple images (Ch. 1) (Ch. 3).
N_{dist}:	number of pairs of points clicked by the human observer in hand-calibrating a scene (Ch. 4).
N_{length}:	number of samples for the signals used in a Monte-Carlo simulation of lane markers (Ch. 3).
N_{peak}:	number of peaks found in the $A_{rel}(u)$ signal (Ch. 3).
N_{pulse}:	number of lane marker pulses used in the Monte-Carlo simulation of lane markers (Ch. 3).
$N_{samp,min}$:	minimum number of samples to obtain linear sampling in the Y' domain when finding $T_2(u,r)$ from $T_1(u,v)$ (Ch. 3).
N_{samp}:	number of samples used in a calculation (Ch. 5).
$\mathbf{P}^-(k+1)$:	covariance matrix for $\mathbf{X}^-(k)$ in the Kalman filter at time step k (Ch. 5).
$\mathbf{P}^+(k)$:	covariance matrix for $\mathbf{X}^+(k)$ in the Kalman filter at time step k (Ch. 5).
p_0:	the homogeneous coordinates of a point in the reference image (Ch. 1).
p_1:	arbitrary point on line L_1 after applying rotation matrices for pan and tilt (Ch. 2).
p_1':	generic point on line L_1 (Ch. 2).
p_2:	arbitrary point on line L_2 after applying rotation matrices for pan and tilt (Ch. 2).
p_2':	generic point on line L_2 (Ch. 2).
P^I:	calibration matrix i (Ch. 1).

p_k:	the homogeneous coordinates corresponding to p_0 in image k, calculated from p_0 as $p_k = H_k p_0$ (Ch. 1).
$q_{11}, q_{12}, q_{21}, q_{22}$:	Individual scalar components of **Q** in our two-state Kalman filter (Ch. 5).
$R(\tau)$:	autocovariance sequence for a column of $T_2(u,v)$ (Ch. 3).
s:	Kalman filter state representing average vehicle speed (Ch. 5).
S:	scale factor involving h, f, and ϕ, or w, θ, b_1, and b_2. Once the nonlinear effects of perspective projection have been removed, this value scales the result to complete the conversion of pixels in the image to real-world coordinates (e.g., feet) (Ch. 2).
S':	scale factor that enables the conversion of vertical distances in the image to physical distances along the road. Related to scale factor S by $S' = S \sec(\theta)$ (Ch. 2).
$s^+(k)$:	Kalman-filtered speed estimate at time step k (Ch. 5).
$S_1(v,t)$:	1-D signal at time sample t found by horizontally summing a feature image and normalizing each sample by the number of pixels in each row of the lane mask. The signal has nonlinear spatial sampling (Ch. 5).
$S_2(r,t)$:	$S_1(v,t)$ after linearly resampling in the spatial domain (Ch. 5).
S_{avg}:	nominal average speed measured by the Kalman filter process (Ch. 5).
$s_{u1,\eta}$:	sample standard deviation estimated by repeatedly obtaining a fraction η of all the samples in the distribution of u_1 estimates (Ch. 3).
t:	general variable indicating time (Ch. 5).
T:	template defining the prototypical peak in the A_{top}' signal (Ch. 3).
T:	translation vector between the camera and its nearest road boundary (Ch. 2).
$T_1(u,v)$:	the image formed by sampling I_{top2} along lines emanating from the vanishing point. This image is distorted by perspective projection (Ch. 3).
$T_2(u,r)$:	the image formed by linearly resampling $T_1(u,v)$ to compensate for perspective projection. There are $N_{samp,min}$ samples in the r-direction (Ch. 3).
$T_{2d}(u,r)$:	$T_2(u,v)$ after morphological dilation (Ch. 3).
T_{Cmin}:	threshold for C_{max} below which we do not accept a given column of $C_{ss}(u,\tau)$ as containing lane markers (Ch. 3).
T_{min}:	threshold for $\max(A_{top}'')/\max(T)$ below which we do not record a correlation coefficient because the signal amplitudes are too small (Ch. 3).
$U(\cdot,\cdot)$:	designation of a uniformly distributed random variable for which the two parameters indicate the range of the distribution (Ch. 3).

$\mathbf{u}(k)$:	zero-mean gaussian noise vector having covariance matrix $\mathbf{Q}$; represents the difference between the current state $\mathbf{X}(k)$ and the state prediction $\mathbf{A}(k\text{-}1)\mathbf{X}^{+}(k\text{-}1)$ based on the previous state estimate $\mathbf{X}^{+}(k\text{-}1)$ (Ch. 5).
u:	generic horizontal coordinate for a point in the image plane (Ch. 2).
u_0:	horizontal coordinate in the image for the vanishing point of the lines that are parallel to the road boundaries (Ch. 2).
u_1:	horizontal coordinate in the image for the vanishing point of the lines that are perpendicular to the road boundaries (Ch. 2).
u_{bound}:	lane boundary positions along the bottom row of the image (Ch. 3).
$(u_c(j),v_c(j))$:	the point on line j that is closest to (u_0,v_0) (Ch. 3).
u_{int}:	parameter indicating the horizontal position of the intersection between any line and the line $v = -H/2$ in an image (Ch. 3).
$(u_{int}(j),\alpha(j))$:	pair of specific values for u_{int} and α describing the j^{th} line when finding (u_0,v_0) (Ch. 3).
u_j:	horizontal coordinate in the image in the bottom row used to index the line samples from the activity map and top-hat image (Ch. 3).
$u_{lane}(i)$:	horizontal positions where lane markers are present in $T_1(u,v)$, $T_2(u,r)$, and $T_{2d}(u,r)$ (Ch. 3).
u_{left}:	leftmost coordinate along the bottom row of the image beyond which we do not expect to find any vehicle lanes (Ch. 3).
u_{max}:	location of the maximum value of $A_{rel}(u)$ (Ch. 3).
$u_{peak}(i)$:	location of candidate activity map peaks along the bottom row of the image (Ch. 3).
u_{right}:	rightmost coordinate along the bottom row of the image beyond which we do not expect to find any vehicle lanes (Ch. 3).
$u_{spike,left}$:	location along the bottom row of the image in the left-hand half of I_{spike} where the largest value of I_{spike} is found (Ch. 3).
$u_{spike,right}$:	location along the bottom row of the image in the right-hand half of I_{spike} where the largest value of I_{spike} is found (Ch. 3).
U-V-W:	coordinate system centered at the point where the camera gaze intersects the ground plane. U is contained by the ground plane, V is coincident with the camera gaze, and W is orthogonal to these two axes (Ch. 2).
v:	generic vertical coordinate for a point in the image plane (Ch. 2).
v_0:	vertical vanishing point coordinate for the lines parallel to the road boundaries in the image (Ch. 2).
v_1,v_2:	v-coordinates in the image defining the distance traversed by a vehicle between time samples (Ch. 3).
v_a,v_b:	v-coordinates of the point pair separated by a distance L in 3-D (Ch. 3).

v_{top}:	the uppermost v-coordinate that defines the subwindow of interest within the image (Ch. 3).
$\mathbf{w}(k)$:	zero-mean gaussian noise vector having covariance matrix $\mathbf{R}$; represents the noise in the measurement process of $\mathbf{Y}(k)$ (Ch. 5).
w:	road width (distance between the two road boundaries, i.e., L_1 and L_2) (Ch. 2).
w_j:	world coordinates for point j (Ch. 1).
W_{temp}:	template width for defining the template for locating peaks in $A_{rel}(u)$ (Ch. 3).
$\mathbf{X}^-(k)$:	state prediction for the Kalman filter found by calculating $\mathbf{A}(k\text{-}1)\mathbf{X}^+(k\text{-}1)$ (Ch. 5).
$\mathbf{X}(k)$:	state vector for the Kalman filter at time step k (Ch. 5).
$\mathbf{X}^+(k)$:	state vector estimate for the Kalman filter at time step k (Ch. 5).
X_c'-Y_c'-Z_c':	3-D earth coordinate system in which Z_c' is the height above the road, X_c' is perpendicular to the road, and Y_c' is coincident with the road boundary nearest to the camera (Ch. 2).
X_c-Y_c-Z_c:	3-D camera coordinate system in which X_c is positive to the left, Y_c is positive upwards, and Z_c is the negative of the depth of the object from the viewpoint of the camera (Ch. 2).
X-Y-Z:	3-D earth coordinate system in which Z is the height above the road and X-Y represents the ground plane (Ch. 2).
$\mathbf{Y}(k)$:	measurements obtained by the Kalman filter at time step k (Ch. 5).
Y':	designation of a coordinate domain axis parallel to the road (Ch. 3).

EXECUTIVE SUMMARY

Overview

This report documents the second project, in a series of three research projects funded by the Washington State Department of Transportation (WSDOT), that will enable already deployed, un-calibrated CCTV cameras to be used as traffic speed sensors. The principle traffic speed sensors presently deployed by WSDOT are inductance loops; however, in some locations it is impractical or too expensive to install loops. In addition, a large number of un-calibrated cameras are already in place and being used by traffic management operators to qualitatively assess traffic both on the freeway and on arterials. These projects will leverage the existing cameras to provide a quantitative measurement of traffic speed similar to that which can be obtained using loops and suitable for use in the Traffic Management System (TMS) without installing loops in the roadway. The implementation of this research, which is funded and just beginning, will culminate with software that creates an automated system compatible with the existing TMS. This system will leverage the existing camera investment to create a new set of speed sensors that increases the geographic extent of the TMS's quantitative surveillance capabilities. The usual operating condition of these cameras, focused a long way down the road, and the calibration variability, as a result of the operators moving and zooming the cameras, make this a challenging problem.

In a previously funded and completed first phase of this project, algorithms were developed to perform a simplified camera calibration. This calibration depended on the vehicle length distribution of the vehicles traveling on the roadway and the time between image frames. Vehicle length was used to create a scale factor function that can be used to estimate speed. The algorithms first detect camera motion; then, if necessary, calibrate the camera; and, finally, estimate speed. However, these techniques require that individual vehicles can be automatically identified in the images. In very congested traffic conditions with the cameras in the default position, looking a long way down the roadway, it is difficult to guarantee that individual vehicles can be identified. In this case additional information is needed. Obtaining this additional information is the topic of the report (WA-RD 527.1). The algorithms from the first phase were reviewed and published in both the Transportation

Research Record, operated by the National Academy of Sciences, and the IEEE Transactions on Intelligent Transportation Systems. Publication in these two peer review journals demonstrates that national and international experts have reviewed the work and determined that it was original and significant.

In the second phase, reported here, other external features are used to augment the camera calibration. This overcomes the occlusion problem, or apparent blending together of small vehicles as seen in the far field of the camera images, that existed in the first phase. Activity maps, fog lines, and vanishing points are a few of the additional features used, and the details of these algorithms are described in this report. These results have also been peer reviewed and published.

The third phase will leverage the technical and mathematical results of the first two phases to automate this technology. This automation will take the form of software suitable for deployment into traffic management activities. It is expected that this software will expand the geographic coverage and measurement capabilities of the Traffic Management Systems in Washington State by allowing the quantitative use of the already deployed cameras.

Phase 2 Activity

This report presents a new set of algorithms to calibrate roadside traffic management cameras and process the images to estimate mean vehicle speed. Past work that has used cameras to estimate traffic conditions has generally postulated that either that the focal length or two-dimensional transformation are available *a priori*, or that a human operator can move the camera to an assigned and calibrated position. In the work presented here, *a priori* calibration information is not available. The equipment consists of monocular, single lens, roadside cameras controlled by traffic management center operators who can pan, tilt, and zoom them at will. The algorithm presented is in the role of a passive viewer of the available images, and it must calibrate the camera using only the information available in the scene.

In general, an automated algorithm must analyze the scene by using a model to determine the regions of interest and then track the vehicles and estimate their speed even when the traffic volume, illumination, and weather conditions vary widely. Additional constraints include low-to-medium-quality JPEG compressed images at a low frame rate (three to five frames/second). Thus, this problem presents major obstacles in both camera calibration and tracking.

The report describes a simplified camera model designed to convert distance measurements in the image to estimates of distances along the road. This method is quite accurate even when the road slope or road tilt is not zero. The method requires only a reasonably accurate estimate of the vanishing point (i.e., ~5 pixels) and that the lane markers be uniformly spaced and visible in the image. These estimates from the image sequence are not error-prone and are straightforward to extract. In contrast, higher-order camera models typically depend on accurate estimates of quantities that are difficult to measure in sparse traffic scenes.

We introduced the morphological top-hat operator, a well-known technique in the image-processing community, as an important technique to generate feature images with which to analyze traffic scenes. After manipulating the images appropriately, we were able to estimate the lane marker interval using the ensemble of auto-covariance sequences for the image. This method yielded an estimate of the lane marker interval that was nearly identical to the hand-calibrated result in a typical scene.

The report also describes how to estimate the average spatial shift of the vehicles in a lane of traffic using the cross-covariance function. The estimated spatial shift perfectly complements the mean length parameter in the simplified camera model for estimating distance. Similarly, the cross-covariance method provides a satisfying corollary to the auto-covariance method for estimating these parameters. Both of these methods are ingenious because they automatically incorporate all the information available in each image and do not require the identification of individual lane stripes or vehicles, which is the case for most other tracking approaches. In fact, because more features exist in the image, our algorithm works even better when the vehicles occlude one another than when traffic conditions are sparse. Furthermore, the method is computationally inexpensive and is suitable for implementation at high frame rates (e.g., 30 Hz).

After a Kalman filter has been applied to the output of the spatial shift estimator, the mean speed estimation method presented in this report functions well, even with raindrops on the camera lens, under dark conditions, when traffic is sparse, or under very congested conditions. Specifically, the speed histograms from the computer vision sensor closely matched those of inductance loops in free-flowing conditions for the very difficult scenes involving raindrops and darkness. Under dynamic traffic conditions, the mean speed sensor produced estimates at a rate of 5 Hz that closely matched subjective observations of the video sequence. In addition to providing both coarse and fine resolution estimates, our algorithm estimates space mean speed, which has better theoretical properties than the time mean speed estimated by inductance loops.

We believe that if a human being can hand-calibrate the scene and draw lane masks, then we should be able to design an algorithm to perform the same task. This additional information increases the certainty of the speed estimates and expands the operating region for the algorithm.

This report documents the detailed analysis necessary to implement an automated speed sensor based on un-calibrated cameras. The implementation of the algorithms in software is the goal of phase three. The third phase will use the technical and mathematical results from this report to create an automated speed sensor. It is expected that this software will expand the geographic coverage and measurement capabilities of the Traffic Management Systems in Washington State by allowing the quantitative use of the already deployed cameras. It is also expected that these data will be compatible with and available to the existing TMS in the Northwest Region.

1 BACKGROUND

1.1 Introduction

As the population has increased in urban centers, so has vehicle traffic. According to the U.S. Department of Transportation [1], metropolitan traffic has grown by 30 percent in the last decade, and the number of cars and trucks on the road will increase by another 50 percent in the next ten years. In a recent urban mobility study [2], researchers surveyed 68 urban areas. They found that in 1997, traffic congestion cost travelers 4.3 billion hours of delay (approximately one work-week per person) and 6.6 billion gallons of wasted fuel, for an estimated total cost of $72 billion (p. xvii). The same study found that accidents and breakdowns caused 57 percent of the travel time delay; the remaining delay is due to crowded traffic conditions. A national effort to combat the traffic problems resulted in the Intelligent Transportation Systems initiative, which has explored and implemented a variety of solutions to these problems. One way transportation agencies have responded to freeway traffic congestion is by installing ramp meters, resulting in speed increases ranging from 8 percent to 60 percent ([3], pp. *ix*-*x*).

In Seattle, the Washington State Department of Transportation (WSDOT) has created a traffic management center where operators can respond rapidly to highway incidents and control ramp meters. WSDOT effected this strategy by installing hundreds of cameras throughout the greater Seattle area to monitor key sections of freeway and major arterials. Operators may pan, tilt, and zoom the cameras remotely from the traffic management center to better view traffic incidents or congestion. Because the cameras are expensive to install, WSDOT has tried to leverage them for other uses, e.g., current Seattle traffic conditions are now viewable on the Web [4]. Figure 1 contains some typical images. Recently, researchers [5][6] have started to develop computer vision algorithms and tools that would enable a freeway traffic camera to serve as a speed sensor. In the past, traffic agencies have used two closely spaced inductance loops embedded in the pavement to form a speed sensor. Although the loops produce fairly accurate data, they are expensive and invasive to deploy, and an appreciable fraction requires replacement each year. In contrast, a camera sensor costs less or about the same amount of money, it is easier to deploy and replace, and problems are simple to detect by inspecting the output images.

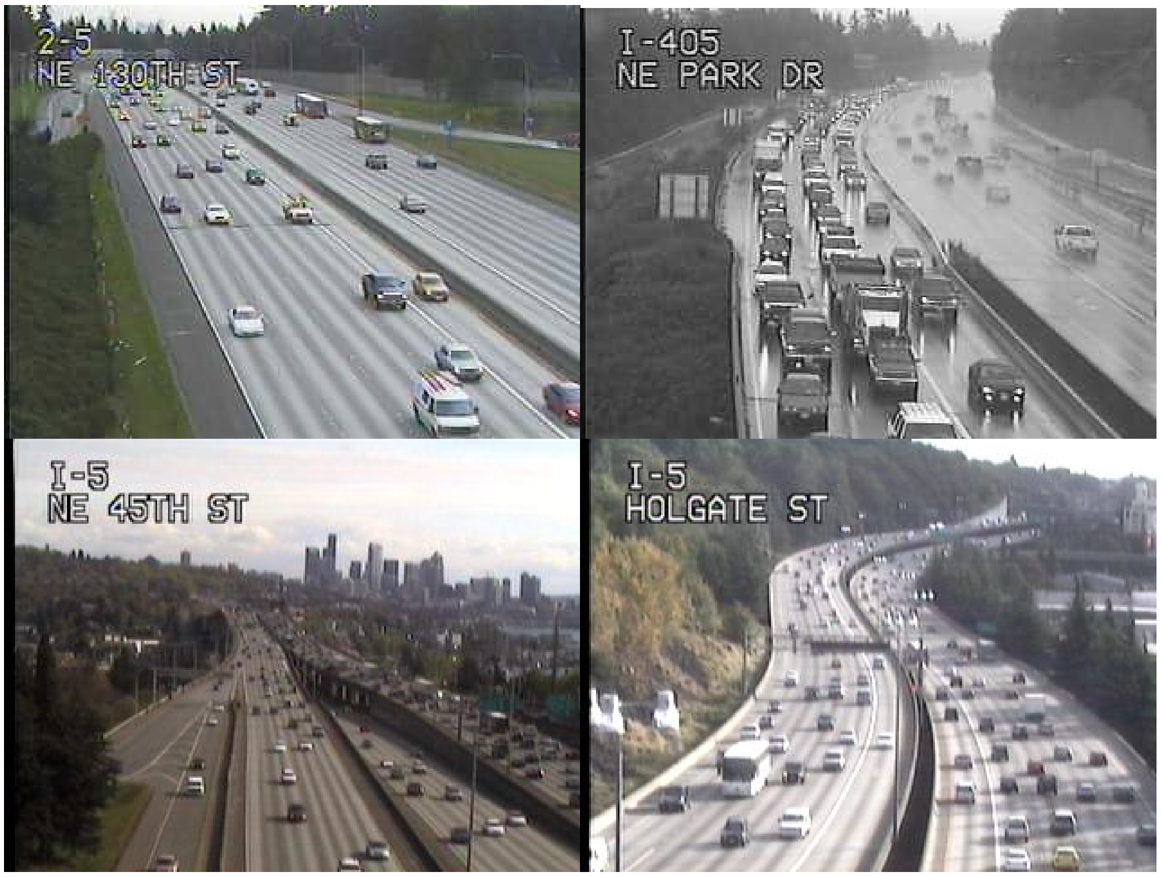


Figure 1. Typical scenes viewed by traffic cameras in Seattle, Washington.

The task of estimating mean speed by using traffic monitoring cameras poses a difficult challenge in several respects. First of all, the computer must calibrate the camera in order to convert pixel distances to real-world measurements. However, the WSDOT operator needs to be able to pan, tilt, and zoom the camera at will. As soon as the camera stops moving, the system must detect this new state and automatically calibrate the camera. Second, the computer must analyze the scene to determine the regions of interest and then track the vehicles even when the traffic volume, illumination, and weather conditions vary widely. In some cases, the scene will not be tractable for computer vision techniques because too many roads are in the scene or the scene is too sparse to calibrate the camera. Technical considerations such as low-to-medium-quality JPEG compressed images at a low frame rate (three to five frames/second) also create constraints. Clearly, this problem presents major obstacles in both camera calibration and tracking.

The challenge, then, is to create a set of algorithms that process video data from a pan-tilt-zoom camera to estimate the speed of traffic on a given section of freeway. The algorithms must first identify the vehicles of interest in the scene and track their position in the image. Next, they must calculate the actual distance traveled using a nonlinear transformation and knowledge about the current parameters of the camera. Since the operator may move or zoom the camera at any time using a joystick, an algorithm must detect when the camera parameters change and recalibrate the camera automatically. To simplify the present research, we made a variety of assumptions, as follows.

1. Vehicles that change lanes may be disregarded for the purposes of mean speed estimation.
2. The road is approximately straight in the bottom one-third of the image.
3. The road is planar with negligible tilt and slope.
4. The parallel road boundary stripes are visible and can be reliably extracted from the image set.
5. The distance between the road boundary stripes is known.
6. The average distance between the tips of successive lane markers on the road is known.
7. The bottom edges of the vehicles are perpendicular to the road boundary stripes.
8. Shadows falsely detected as cars won't noticeably affect the average speed calculations.
9. Any water droplets detected on the camera lens or other barriers between the camera and the road disqualify a scene for processing.
10. Mean speed estimation occurs during daytime or dusk light conditions.
11. High- and low-resolution JPEG-compressed images of the roadway are available at 2 Hz and 5 Hz, respectively.
12. The option exists to have the operator orient and focus the camera one time so that the system can extract detailed structural information from the passing vehicles in order to estimate the camera position.

Regardless of the assumptions, any system designed to estimate vehicle speeds from traffic video must contain several elements: (1) a way to obtain the region of interest in the scene; (2) a vehicle detection and tracking module to locate vehicles and track their position in time; (3) a theoretical model and methodology for transforming 2-D (2-Dimensional) image measurements into 3-D (3-Dimensional) distances; (4) a module to detect when the camera requires recalibration; (5) a means of identifying which scenes can and cannot be used.

In the present research, we analyzed the image time series to extract information about the scene, such as the region of interest, the lane masks, the road boundary stripes, and the interval between lane markers. To obtain 3-D distance measurements, we used a pan-tilt-zoom model of the camera and extracted the position, orientation, and focal length information about the camera using our measurements from the image and our theoretical model. This information was adequate to convert image coordinates into real-world coordinates, using the assumption of road planarity. We used our knowledge about the lane boundaries to constrain the tracking process as the vehicles moved through the image. We estimated the average 3-D shift of the vehicles between image frames and used this to estimate the mean vehicle speed. While it is trivial to identify dark scenes via histogram analysis, we provided a new algorithm to detect conditions in which raindrops or other obstacles obscure the camera's vision of the road.

1.2 Literature review

Algorithms for performing these tasks fall within the discipline of computer vision. This broad area undertakes the challenge of providing the computer with an intelligent way to automatically process images and extract useful information. These images may be computer-generated or captured by cameras from the real world. Although researchers currently pursue a variety of applications, ranging from providing quality control on parts in a manufacturing process to guiding robots, the ambitions of computer vision remain limited only by human imagination. Within this framework, surveillance applications have received much attention because of the computer's potential to perform tasks that are either too large or too tedious for a team of humans to execute. These include the surveillance of banks, buildings, and parking lots for security purposes, as well as traffic at airports, subways, and freeways, the topic of the present research.

1.2.1 Camera calibration

Researchers have made significant progress in a variety of areas that contribute to the solution of the current traffic monitoring problem. For example, in order to make measurements from images, the computer must obtain a calibrated model of the camera.

Over the past ten years, self-calibration of cameras has become a very active topic for researchers in computer vision. Self-calibration falls within the more general problem of reconstruction [7]. That is, consider a set of 3-D points viewed by N_{cam} cameras with calibration matrices P^i, $i = 1, \ldots, N_{cam}$. Let $m^i_j = P^i w_j$ be the homogeneous coordinates of the projection of the jth point onto the ith camera. The reconstruction problem is then to find the set of camera matrices P^i and scene structure (world coordinates) w_j such that $m^i_j = P^i w_j$. Depending on the assumptions and restrictions placed on P^i, the scene may be recovered to varying degrees of reality. For example, an affine reconstruction "provides a good approximation to the perspective projection model when the depth of the object is small compared to the viewing distance" [7]. The affine camera model provides notions of parallelism, "betweenness," and "at infinity," but no notion of rigidity, angle, or absolute length. A Euclidean reconstruction, however, preserves the notion of rigidity, and enough information is present to recover all the information about the 3-D world from the image, up to a scale factor [8]. In the past, researchers satisfied themselves with self-calibration to the point of an affine reconstruction. However, the groundbreaking paper by Maybank and Faugeras in 1992 [9] offered a way to obtain a Euclidean reconstruction, assuming the internal camera parameters (e.g., focal length) were constant. This research paved the way for later work [7], which placed no constraints on the internal camera parameters. Since the present research focuses on the determination of absolute length under conditions of varying focal length and camera orientation, it falls within this realm of Euclidean reconstruction.

The current problem provides more constraints than are typically assumed in the general Euclidean reconstruction problem. In fact, a body of literature [10][11] addresses the self-calibration of motionless cameras that only pan, tilt, and zoom. The skew, aspect ratio, and principal point are assumed to be ideal, which are good approximations in practice. These approaches obtain the absolute focal length and relative pan and tilt angles between images by calculating the 2-D perspective transformation, H_k, between images. That is, suppose p_0 is the homogeneous coordinates of a point in the reference image. Then the coordinates of the matching point p_k in image k are calculated from p_0 as $p_k = H_k p_0$. Given H_k, the desired parameters may be obtained algebraically.

Such an approach is remarkably clever and attractive to the current problem of estimating the camera parameters on the basis of transition images as the user operates the camera. However, reliably calculating the image homography between two images requires the extraction of many matching points. Typical approaches automatically extract corners in both images as features [12], associate the corners together, and obtain the image homography by the Levenberg-Marquardt algorithm [13], a multi-dimensional variant on Newton's method. The strength of this search algorithm is the robust estimation of the homography matrices, H_k. However, despite its attractiveness, traffic images present a large obstacle to the use of pan-tilt-zoom self-calibration methods, i.e., the dearth of stable image features. For example, trees may occupy the non-highway portions of the image, creating large areas where no features exist. In addition, the classical algorithms don't prescribe how to handle objects that move (e.g., vehicles) during the calibration. Widely varying weather and lighting conditions also make point-feature extraction very difficult. Furthermore, small features do not manifest themselves in the images because of the large depth of field. Without numerous and good features, the image homographies cannot be computed consistently nor accurately, and the self-calibration will not succeed.

Work in panoramic image generation stemming from Szeliski [14] offers additional possibilities for camera calibration. In this framework, once a system has extracted the panoramic image describing its environment, it can register any image to the panorama and thereby extract the camera's orientation and focal length. To measure distances properly, the system would require the user to drive a marked car at a known speed in each lane of the freeway, defining the image to a real-world conversion function. Although one could develop a system to automatically generate the panorama and register the current image, the calibration procedure remains untenable given the effort that would be required to calibrate scores of cameras.

Given the difficulties of fully self-calibrating traffic monitoring cameras, researchers have typically assumed some knowledge about the internal and external parameters of the camera. For example, Chausse et al. [15] assumed that the focal length of the camera and width of the road are known and that the user has identified points in the image that define a line perpendicular to the road. Armed with this information, their method can reconstruct a Euclidean model of the roadway. Yang and Ozawa [16] only assumed knowledge of the focal length of the camera and width of the roadway. They then used the parallel and perpendicular structure of the road in the image to reconstruct a Euclidean model. In contrast, a Japanese group [17] assumed knowledge of the camera's position and orientation relative to the road and used this information to obtain absolute distance measurements.

Other researchers have created systems that use information available in the image to calibrate the camera. Jung and Ho [18] greatly simplified the calibration problem by manually placing four poles in the image from which the camera parameters were easily extracted. Zhu et al. [19] calibrated their camera by sending a box-like vehicle of known length, width, and height through the image, which was a step toward self-calibration. Their camera, however, was set up in an orientation and position very favorable to the application: the down-angle was very large (nearly a top-down view), the cars occupied a large portion of the image (small focal length), and the camera was positioned so that the line of sight was aligned with the middle of the road (i.e., zero pan angle).

Of the remaining approaches, most systems (e.g., [18][19]) have made assumptions typical for the state-of-the-art in traffic monitoring research, i.e., cameras with fixed position and orientation. The single line of research that deviated from this norm was done at the University of Washington by Dailey and Pumrin [5][6], who shared some of our same assumptions, i.e., the camera is uncalibrated and can pan, tilt, and zoom at any time. Instead of directly calibrating the camera, they used the average car length to establish a calibration function that can directly convert pixel distances in the front region of the image into real-world distances. Disadvantages inherent to this approach include the need to know the distribution of vehicle lengths and the system's sensitivity to the orientation of the camera, which affects the apparent length of the vehicles when they are projected onto the 2-D image. Because it relies on the vehicles themselves, this approach also encounters problems when traffic is very sparse (long calibration time) or extremely dense (the vehicles occlude one another to become one big blob).

Many researchers in the area of traffic monitoring have obviated the calibration problem by providing the system with a 2-D to 3-D homography created manually from known distances on the roadway. Algorithms developed by Yu et al. [20] and Bouzar et al. [21] that simulate speed traps are examples. The work of Gupte et al. [22] describes a calibration tool operated by the user that enables the vehicle classification system to work properly. Similarly, work at U. C. Berkeley [23] assumes a user-provided camera calibration in order to focus on the tracking aspects of the traffic monitoring problem.

1.2.2 Vehicle tracking

A wide range of methods exists for tracking vehicles to estimate traffic speed. The tracking approaches in the aforementioned research of Yu et al. [20] and Bouzar et al. Bouzar et al. [21] amounts to background subtraction, which fails under occlusion. In contrast to this simple scheme, the state-of-the-art approach by Beymer et al. [23] extracts vehicle corner features from traffic images, tracks the features through the image sequence, and groups the features into vehicle tracks using spatial and velocity constraints. This approach is more robust than tracking the entire object when a portion of the vehicle is occluded. It also seems to offer reasonable performance at night when only the headlights and taillights of vehicles are visible. The main disadvantage to this method is the problem common to all point-feature trackers: features become difficult to find when the object moves a large distance between frames, which is certainly the case in free-flowing traffic at the low frame rates available to our system. Another drawback to the work at U.C. Berkeley is the high computational expense associated with grouping and tracking the features.

As one would expect from the plethora of object tracking methods found in the general computer vision literature, a variety of less successful methods exist for tracking vehicles. Examples include vehicle tracking based on neural network recognition of wavelet features [24] as well as optical flow-based trackers [25]. Several researchers [26][27] have employed active contours, but a comparison of the approaches and results of Byemer et al. [23] and Koller et al. [26] shows that the performance ceiling is usually higher for the feature-based approach. Recently, the snake-based approach has been revived by applying the Kalman filter at various stages in the tracking process [28]. One snake-based approach that employs the Kalman filter also attempts to use regional information to develop the contour [29]. Rigid deformable templates have also been used to detect, classify, and track the position and orientation of vehicles ([30][31][32]). Although deformable templates offer many possibilities for classification, they are not particularly well suited for vehicle tracking because the system must identify the proper template and orient it for every vehicle blob before it can track the vehicle. Region-based trackers (e.g., [33]) or blob detectors with background subtraction [34] and frame-differencing have proved to be the most popular techniques for rapidly developing a working system. All of these approaches have their relative merits and work properly in some situations. However, because each approach attempts to track individual vehicles, they are all fundamentally challenged by vehicle occlusion, i.e., when two vehicles overlap or merge in the image.

Kato et al. [35] went to great lengths to classify all pixels as background, shadow, or vehicle for robust segmentation using Hidden Markov Models. This enabled the researchers to identify vehicle pixels with surprising accuracy. However, such an approach is more appropriate for vehicle counting and classification where pixel-accurate vehicle masks become important. Furthermore, their approach has yet to be proven in realistic traffic situations that involve occlusion.

Perhaps most intriguing are the systems based on extracting information accumulated from a spatiotemporal image ([19][36][37]). Although this approach remains unproven for dense traffic situations, the data are amenable to regression techniques, and hence this work provides a detailed error analysis [19], which is extremely rare in the vehicle tracking literature.

1.2.3 Systems that estimate vehicle speeds

As described inn the literature thus far, only a few researchers have created systems designed to estimate traffic speeds in realistic situations. Garibotte et al. [38] estimated individual vehicle speeds and read license plates for the purpose of automatic traffic tickets. To solve this particular problem, they placed the camera at a specific position and orientation to track and read the license plates as they passed by. They employed a binocular approach and obtained impressively low errors in their speed estimates, i.e., 4 percent or less. However, although their solution may seem attractive, we note that the constraints and assumptions used by our algorithms are vastly different because of the positioning of our cameras.

Another recent attempt at estimating vehicle speeds is the work of Pae et al. [39]. These researchers applied a block-matching algorithm to low-resolution images (320 X 240 pixels) captured at a high frame rate (15 Hz). They assumed that the camera position and orientation were fixed and known. In one sense, their sample images were very unrealistic, i.e., one or two vehicles on the road, but they successfully eliminated a very busy background on the sides of the road. Although it performs in real-time, this system has a long way to go before it can be applied to the wide variety of conditions we expect from our image sequences.

The simple scene analysis of Pai et al. [39] is typical for the state-of-the-art. The tools available for analyzing a complex traffic scene are fairly rudimentary. Most researchers employ a variation on background subtraction, and they typically assume that the traffic scene is relatively simple. To our knowledge, only two sets of researchers have even attacked the problem of extracting the traffic lanes from a scene [40] [41]. The first approach [40] applies morphology to a binary mask that indicates where image pixels are changing in order to extract very rough lane boundaries. Because this approach works only in scenes where the pan angle is nearly zero, it is irrelevant to the current traffic monitoring problem where the pan angle often exceeds 5 degrees. The latter approach [41] relies on static line features on the road, which may vary considerably from road to road. Furthermore, the method will probably produce mixed results when more than one set of traffic lanes are within the image. We also note that an edge-based approach is likely to regularly fail under some of the more extreme lighting and weather conditions in the real world. Thus, although the systems we present below have produced some encouraging results, we recognize that the research in traffic scene analysis must mature before we can expect to automatically process complex scenes involving more than one set of lanes. Given this state of affairs, it is also not surprising that researchers have yet to attempt to calibrate the camera from knowledge about the lines painted on the road.

The VISATRAM system [19] estimates the largest number of traffic parameters of any approach in the literature, ranging from individual vehicle dimensions and speeds to headway, occupancy, volume, and mean vehicle speed. To achieve this objective, the authors highly constrained the camera position and orientation. The system uses high-resolution images (768 X 576) taken at a high frame rate (25 Hz) from a camera mounted on an overpass with the optical axis aligned with the center of the road. The camera's down angle is quite large, and the road below the camera occupies most of the image. The authors calibrated the system by sending a vehicle of known dimensions through the scene. As mentioned previously, the tracker obtains vehicle information with a spatiotemporal image generated from the well-sampled image sequence. Though the tracker cannot handle occlusion or congestion, its measurements are quite accurate (within 5 percent) for the situations studied, and the system is well-characterized. The system can cope with shadows and dim conditions, though it remains unproven in rainy or night-time conditions. The authors claim hours of lane observation with nearly a 100 percent vehicle detection rate, though it appears that the road did not suffer from any congestion problems. In the present context, the greatest shortcomings of VISATRAM include the requirements of a fixed and well-positioned camera, high bandwidth connection, and a camera calibration procedure that involves multiple people.

In contrast to the VISATRAM system, researchers of the RoadWatch program at U.C. Berkeley [23] did not constrain the camera position and orientation to such a high degree. However, they did assume that these parameters were fixed, that the user would provide the camera calibration, and that the user would identify the individual traffic lanes of interest. They also assumed a medium bandwidth connection, using uncompressed low-resolution images (320 X 240) sampled at 10 Hz. The strength of their approach is that the point feature tracker described above can handle partial occlusion, stop-and-go traffic, lane changes, and night-time conditions. However, the tracker performance decreases as the frame rate drops since the Kalman filter-based tracker cannot track position jumps of more than about 15 pixels. The tracker is also very computationally expensive. Although the RoadWatch system requires human calibration and does not allow a roving camera, the tracking results are very promising, particularly over a medium to high bandwidth network connection.

The present research shares many of the same assumptions as other work at the University of Washington [5], since both aim to extract speed estimates with uncalibrated, roving cameras over a low-bandwidth network connection. However, the work begun by Dailey et al. [5] and continued by Pumrin [6] varies from the current research in several respects. First, Pumrin relied on information contained in the length of the vehicles to calibrate the camera. As a consequence, the camera had to be within a fairly narrow range of positions and orientations such that the perspective projection of the vehicle was approximately proportional to the vehicle length. Meeting this requirement had the side effect that 3-D roadway distances at the same depth in different lanes had approximately the same lengths in the image. Our calibration approach does not enforce such stringent requirements of the camera orientation. Second, Pumrin assumed that the computer can isolate the vehicles to perform the calibration, i.e., the roadway was assumed to be uncongested, with favorable weather and lighting conditions. We make no assumptions about the weather or traffic conditions. Thirdly, Pumrin assumed that the distribution of vehicle lengths was known and time-invariant during calibration. The suitability of this approximation will certainly vary with the scene in question and with the time of day. Fourth, Pumrin [6] assumed a known camera height, which required field work or estimation by another algorithm. Lastly, we note that the camera calibration is only as good as the vehicle detection and tracking module; this coupling could prove undesirable at some point. In all, previous University of Washington work made many of the same challenging assumptions described above while adding several more that would be likely to severely limit the applicability of their solution.

To summarize the current state-of-the-art, no system has successfully solved a generic camera geometry similar to the cameras producing the WSDOT images, though Dailey et al. [5] and Pumrin [6] provide a step forward when assumptions about the camera position are met. Though the research of Zhu et al. [19] and Beymer et al. [23] holds some promise for vehicle tracking, these solutions are not directly applicable because neither has been tested at low frame rates or under adverse weather conditions, and that of Zhu et al. Is unproven in highly congested traffic. Despite its success and intriguing possibilities, the license plate tracking approach of Garibotte et al. [38] cannot be applied to the problem at hand. Approaches such as that of Pai et al. [39] still require major development and are mentioned only to offer perspective on the range of maturity and realism that is present in the current literature.

1.3 Overview of our work

To this point, no one has undertaken the challenge of solving the camera calibration, vehicle tracking, and speed estimation problems to the degree required by a real system. Such a system should be able to estimate the mean vehicle speed for a section of freeway under a wide variety of traffic conditions, weather conditions, scene geometries, and camera parameters. We offer our work with the belief that these algorithms can form the basis for a fully functional system during daylight to dusk light conditions when rain is not on the camera lens.

We begin by introducing our camera and scene model in Chapter 2. This chapter also derives and analyzes three methods of calibrating the camera for speed estimation, of which one is shown to be clearly superior in its sensitivity to measurement errors and errors in the camera model. In Chapter 3 we explain the bulk of the image processing algorithms designed to enhance image features and perform measurements on the images. We also detail algorithms that will analyze the scene by obtaining lane masks and detecting rain or obstacles. Chapter 4 contains results for our camera calibration procedures for real scenes. We compare the results to hand-calibrated results and analyze the errors. It is here that the same calibration method emerges as a clear winner for calibrating the camera in real scenes. In Chapter 5 we explain the details of our mean speed estimation algorithm. Not only does it fit the camera calibration method very naturally, but it automatically averages the vehicle speeds during the mean speed estimation process without estimating the actual vehicle positions. We conclude in Chapter 6 by summarizing our work, its significance, and its implementation. We also offer directions for future research.

2 CAMERA AND SCENE MODEL

2.1 Fundamental Model and Assumptions

To provide a context for our vehicle speed estimation algorithm, we present analytical models of the camera and the scene. We work primarily with a set of simplified models that ignore camera roll and road slope and tilt. We develop multiple methods of calibrating the camera in terms of features that are available for measurement in the images, together with information about the scene known *a priori*. We present results from a Monte-Carlo simulation of errors for each calibration method. We then present the operational limits for each calibration method and describe the effects of a sloped or tilted road. Finally, we compare all of the camera calibration methods and recommend how they can be used.

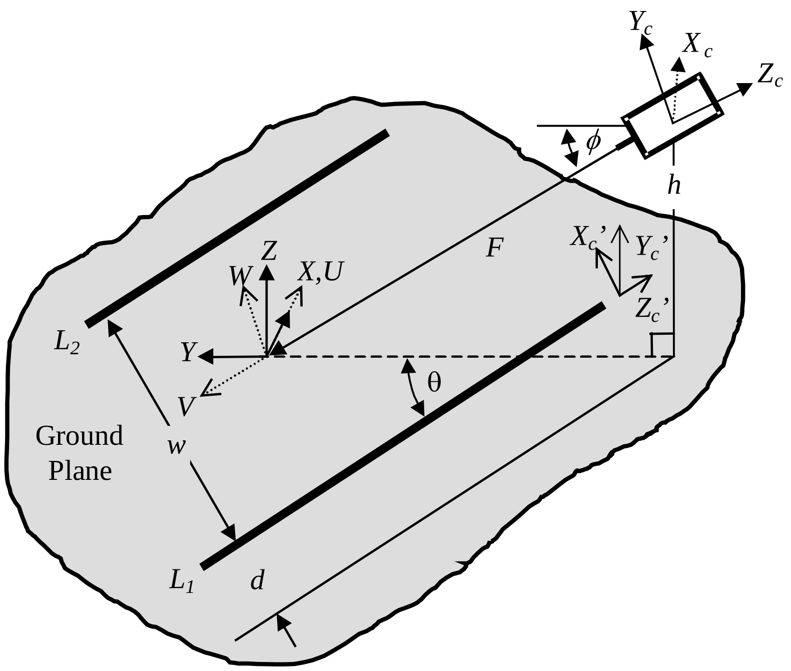


Figure 2. Camera and roadway geometry.

The geometry of the camera and roadway are shown in Figure 2. We model the scene as two parallel lines L_1 and L_2 viewed through a pinhole camera, assuming that the camera is located at a height h above the ground plane and a perpendicular distance d from the edge of the roadway. The camera is oriented at a pan angle θ with respect to the road and tilt (down) angle ϕ such that a point in the earth system (X, Y, Z) is transformed into the camera coordinate system (X_c, Y_c, Z_c) by only a translation and a rotation. The camera is oriented along the negative Z_c axis (which requires us to associate a negative sign with the focal length), and its line of sight intersects the ground plane a distance $F = h \cdot \csc(\phi)$ away. We also offer side views of the scene in figures 3 and 4 to describe the orientation of the road relative to the camera in real scenes.

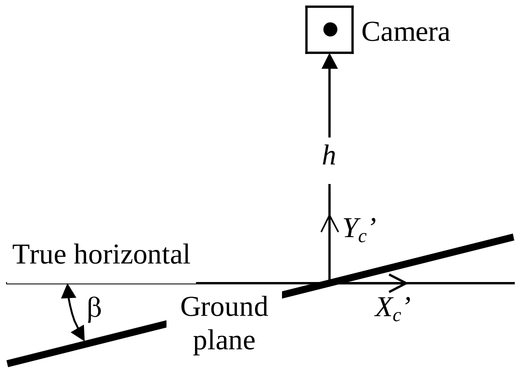


Figure 3. Head-on view of camera and roadway geometry emphasizing the nonzero road tilt β.

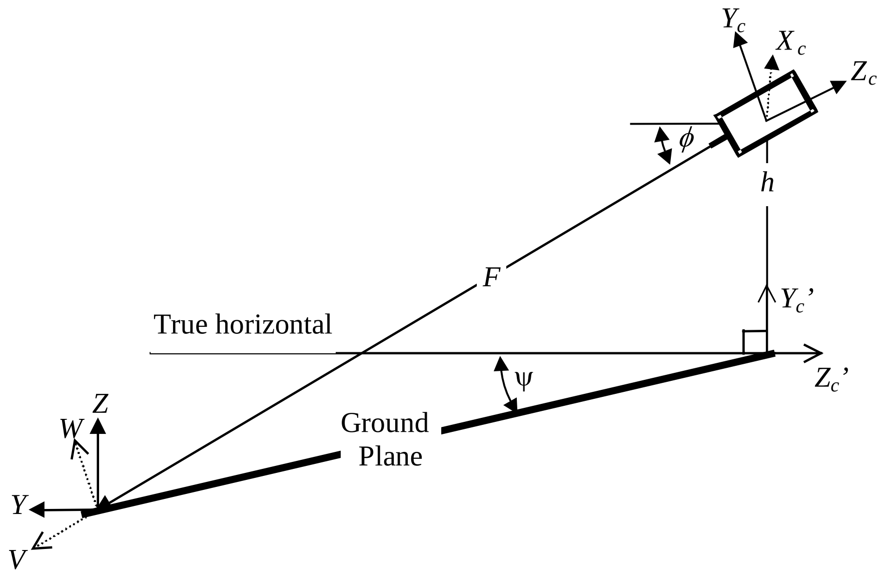


Figure 4. Side view of camera and roadway geometry emphasizing the nonzero road slope ψ.

2.1.1 *Perspective projection of the scene geometry*

The following derivations assume that the angles ψ (road slope) and ` (road tilt) are zero and that the camera roll angle about the Z_c-axis is also zero. In this case, the following equations describe the perspective projection of points in the pinhole camera's coordinate system onto the image, given a focal length $f > 0$:

$$\begin{aligned} u &= -f\,\frac{X_c}{Z_c} \\ v &= -f\,\frac{Y_c}{Z_c} \end{aligned} \tag{1}$$

Starting with the *X*-*Y*-*Z* coordinate system, we develop expressions for points on the ground plane in terms of the X_c-Y_c-Z_c coordinate system. First, we obtain the *U*-*V*-*W* system by rotating an angle ϕ around the *X*-axis:

$$\begin{aligned} U &= X \\ V &= Y\cos(\phi) - Z\sin(\phi) \\ W &= Y\sin(\phi) + Z\cos(\phi) \end{aligned} \tag{2}$$

These expressions are further simplified because objects are assumed to lie on the ground plane where $Z = 0$. Note that θ does not enter the derivation because we assume ψ = 0, and because we have chosen not to align the *U*-axis perpendicular to L_1 and L_2. Next, we apply a displacement $F = h \cdot \csc(\phi)$ to obtain the camera-centered coordinates X_c-Y_c-Z_c.

$$\begin{aligned} X_c &= U = X \\ Y_c &= W = Y\sin(\phi) \\ Z_c &= -V - F = -Y\cos(\phi) - F \end{aligned} \tag{3}$$

Applying Equation 1 yields

$$u = -f\,\frac{X_c}{Z_c} = -f\,\frac{X}{-Y\cos(\phi) - F} \tag{4}$$

$$v = -f\,\frac{Y_c}{Z_c} = -f\,\frac{Y\sin(\phi)}{-Y\cos(\phi) - F} \tag{5}$$

Because we assumed the vehicles lie on a flat plane when developing our camera model, we can transform the image coordinates into their 3-D coordinates (*X,Y,0*) using the camera calibration, similar to the result of Lai and Yung [41]. Referring to Equations (4) and (5), we solve for *X* and *Y*, using the fact that $F = h \cdot \csc(\phi)$, yielding

$$X = S \frac{u \cdot v_0}{v_0 - v} \tag{6}$$

$$Y = \frac{S}{\sin(\phi)} \frac{v \cdot v_0}{v_0 - v} \tag{7}$$

where $S = \frac{F}{f} = \frac{h}{f}\csc(\phi)$ meters/pixel is a scale factor that extends Lai and Yung's result for real-world coordinates, and $v_0 = f \cdot \tan(\phi)$ is the vertical vanishing point coordinate (see Figure 5 below).

We now analyze the transformation of lines L_1 and L_2 on the ground plane into lines in the image in the slope-intercept form $u = m_1v+b_1$ and $u = m_2v+b_2$. Choosing this form, rather than *v* in terms of *u*, has important advantages later in the derivation. Suppose a 3-D line L_x contains a point $\mathbf{b} = [b_x\ b_y\ b_z]^T$ and has a direction cosine vector $\mathbf{a} = [a_x\ a_y\ a_z]^T$, i.e., points on L_x satisfy $\lambda\mathbf{a} + \mathbf{b}$, where λ is arbitrary. If a point $\lambda\mathbf{a} + \mathbf{b}$ on the line is projected from 3-D to 2-D according to Equation (1), we obtain

$$u = -f\frac{X}{Z} = -\frac{f(\lambda a_x + b_x)}{\lambda a_z + b_z} \tag{8}$$

$$v = -f\frac{Y}{Z} = -\frac{f(\lambda a_y + b_y)}{\lambda a_z + b_z} \tag{9}$$

Using Equation (9), we obtain an expression for λ, which enables us to find *u* in terms of *v*

$$\lambda = -\left(\frac{b_y f + b_z v}{a_y f + a_z v}\right) \tag{10}$$

$$u = \left(\frac{a_z b_x - a_x b_z}{a_z b_y - a_y b_z}\right)v + \left(\frac{f(a_y b_x - a_x b_y)}{a_z b_y - a_y b_z}\right) = mv + b \tag{11}$$

where the coefficient of *v* is the slope of the line, and the constant term is the *u*-intercept.

Returning to the model in Figure 2, we derive the necessary transformations to express lines L_1 and L_2 in terms of the X_c-Y_c-Z_c coordinate system. We start from the X_c'-Y_c'-Z_c' system located near the base of the camera ,with X_c' perpendicular to the road lines, *Y_c'* coincident with L_1, and Z_c' perpendicular to the ground plane. The following derivation translates and rotates L_1 and L_2 from the perspective of the camera, which is held still. In the X_c'-Y_c'-Z_c' system, the line parameters for L_1 and L_2 would be **a = a₁** = **a₂** = [0 0 1]^T and **b₁** = [0 0 0]^T and **b₂** = [*w* 0 0]^T, respectively. First, we translate L_1 and L_2 below and to the right of the camera by adding **T** = [*d* –*h* 0]^T to points **b₁** and **b₂**. We then tilt the road by an angle θ about the Y_c'-axis and by an angle ϕ about the X_c'-axis, yielding the camera pose of Figure 2. The following matrices encode this rotation:

$$R_{\phi\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{12}$$

Applying these transformations to arbitrary points $p_1' = \lambda\mathbf{a} + \mathbf{b_1}$ lying on L_1 and $p_2' = \lambda\mathbf{a} + \mathbf{b_2}$ lying on L_2 in the X_c'-Y_c'-Z_c' coordinate system yields the new vectors describing the lines in the current camera reference frame X_c-Y_c-Z_c:

$$\begin{aligned} p_1 &= R_{\phi\theta}\left(\lambda\,\mathbf{a} + \mathbf{b}_1 + \mathbf{T}\right) \\ &= \lambda\begin{bmatrix} \sin\theta \\ -\sin\phi\cos\theta \\ \cos\phi\cos\theta \end{bmatrix} + \begin{bmatrix} d\cos\theta \\ -h\cos\phi + d\sin\phi\sin\theta \\ -h\sin\phi - d\cos\phi\sin\theta \end{bmatrix} \\ p_2 &= \lambda\begin{bmatrix} \sin\theta \\ -\sin\phi\cos\theta \\ \cos\phi\cos\theta \end{bmatrix} + \begin{bmatrix} (d+w)\cos\theta \\ -h\cos\phi + (d+w)\sin\phi\sin\theta \\ -h\sin\phi - (d+w)\cos\phi\sin\theta \end{bmatrix} \end{aligned} \tag{13}$$

Applying Equation (11) to p_1 and p_2, we obtain

$$\begin{aligned} m_1 &= -\frac{d}{h}\cos(\phi)\sec(\theta) - \sin(\phi)\tan(\theta) \\ b_1 &= \frac{df}{h}\sin(\phi)\sec(\theta) - f\cos(\phi)\tan(\theta) \\ m_2 &= -\frac{d+w}{h}\cos(\phi)\sec(\theta) - \sin(\phi)\tan(\theta) \\ b_2 &= \frac{(d+w)f}{h}\sin(\phi)\sec(\theta) - f\cos(\phi)\tan(\theta) \end{aligned} \tag{14}$$

Manipulating these four nonlinear equations to isolate the camera parameters is nontrivial, and we leave it for a later derivation. However, we note that we can solve for the coordinates of the intersection of L_1 and L_2. In general, under the perspective projection geometry defined by Equation (1), any set of lines that are parallel in 3-D will converge to a single point in the image as they extend toward infinity. This is shown pictorially in Figure 5, where the road lines are dark and the lines perpendicular to them are lighter. We can solve the following set of equations:

$$\begin{aligned} u_0 &= m_1 v_0 + b_1 \\ u_0 &= m_2 v_0 + b_2 \end{aligned} \tag{15}$$

and obtain u_0 and v_0

$$\begin{aligned} v_0 &= \frac{b_2 - b_1}{m_1 - m_2} \\ u_0 &= \frac{b_2 m_1 - b_1 m_2}{m_1 - m_2} \end{aligned} \tag{16}$$

Substituting Equation (14), we obtain

$$u_0 = -f\ \tan(\theta)\sec(\phi) \tag{17}$$

$$v_0 = f\ \tan(\phi) \tag{18}$$

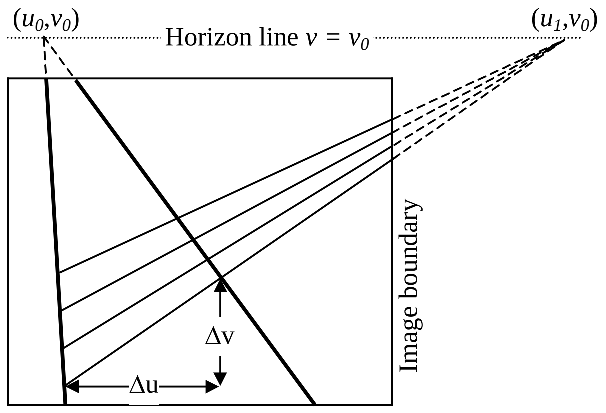


Figure 5. Road geometry in the image showing the vanishing points for lines parallel and perpendicular to the road.

However, this approach leaves us without a value for u_1, the coordinate for the vanishing point of the lines perpendicular to the road. In general, we can obtain all vanishing point coordinates by taking the limits of Equations (4)-(5) as X and Y on the ground plane tend toward infinity away from the camera

$$u_0 = \lim_{Y\to\infty} \frac{fX}{Y\cos\phi + F} = \lim_{Y\to\infty} \frac{f\,\frac{X}{Y}}{\cos\phi + \frac{F}{Y}} = -f\ \tan(\theta)\sec(\phi) \tag{19}$$

$$v_0 = \lim_{Y\to\infty} \frac{fY \sin\phi}{Y\cos\phi + F} = \lim_{Y\to\infty} \frac{f \sin\phi}{\cos\phi + \frac{F}{Y}} = f\ \tan(\phi) \tag{20}$$

$$u_1 = \lim_{Y\to\infty} \frac{fX}{Y\cos\phi + F} = \lim_{Y\to\infty} \frac{f\,\frac{X}{Y}}{\cos\phi + \frac{F}{Y}} = f\ \cot(\theta)\sec(\phi) \tag{21}$$

To obtain the final result, we use the fact that $X/Y = -\tan(\theta)$ for lines parallel to the road, as shown in Figure 7.

The 2-D-to-3-D relationships of Equations (6) and (7), the expressions for the road boundary line parameters in terms of the camera parameters and scene geometry, and the vanishing point coordinates in terms of the camera parameters will prove very useful in the remainder of our analysis.

2.2 Methods of camera calibration

We now derive three methods of calibrating the camera with different tradeoffs between known information about the scene and quantities estimated from the image sequence. As described above, we anticipate that we can estimate m_1, b_1, m_2, and b_2 (the parameters for L_1 and L_2 in the image), and u_0, v_0, and u_1 (the vanishing point coordinates in the image). We also note that the lanes of all freeways in the United States are required to have lane separators longitudinally spaced at a fixed distance L [42], as shown in Figure 6. If we assume knowledge of L and can estimate the corresponding interval, τ_L, in the image, then this will also prove useful in calibrating the cameras. Our final assumption is that the width, w, of the road is known. Table 1 describes how the three methods convert assumptions about the camera geometry and measurements from the images into estimates of the camera calibration parameters. The output parameter S' is a scale factor encoding the fact that we have sufficient knowledge to measure distances along the road. The appropriate method for a given situation is determined by what information is available. For example, Method 1 is appropriate only when we can measure u_1. Similarly, Method 3 is only appropriate if lane markers are visible on the road. Our sensitivity analysis in a future section will provide additional guidance for choosing the best method for a given situation.

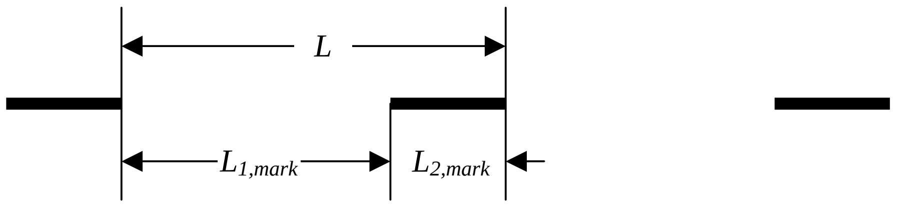


Figure 6. Periodic lane markers with distance definitions.

Table 1. Assumptions and outputs of various camera calibration methods (* denotes optional parameters)

	Method 1 (vanishing points)	Method 2 (known camera position)	Method 3 (known distance)
Camera parameters known *a priori*	None	None	None
Assumed values of scene geometry	w	d, w	L, w
Quantities estimated from the image sequence	u_0, v_0, u_1, b_1, b_2	u_0, v_0, b_1, b_2	u_0, v_0, b_1*, b_2* τ_L
Parameters of the scene geometry estimated by the method	d	None	d*
Camera parameters estimated	f, ϕ, θ S'	f, ϕ, θ S'	f*, ϕ*, θ* S'

2.2.1 Method 1 (vanishing points)

In this camera calibration method, we assume that we can estimate the coordinates for the vanishing point of the lines parallel to the road and the vanishing point of lines perpendicular to the road. Using these quantities, knowledge of the road width w, and measurement of b_1 and b_2 (see Equation (14)), we can estimate all of the camera and scene parameters in Figure 2. Applying the trigonometric identity $\tan^2(\phi) = \sec^2(\phi) - 1$ to Equations (19)-(21) we obtain

$$\begin{aligned} u_0 u_1 &= f^2 \sec^2(\phi) \\ v_0^{\,2} &= f^2 \tan^2(\phi) = f^2\left(\sec^2(\phi) - 1\right) = u_0 u_1 - f^2 \\ f &= \sqrt{u_0 u_1 - v_o^{\,2}} \end{aligned} \tag{22}$$

We obtain ϕ and θ from Equations (20) and (19) by substituting the values for f and v_0. Of the parameters necessary for the transformation of Equations (6)-(7), only h is yet to be found. Using the measured horizontal distance $b_2 - b_1$ in the image, we get the following result from Equation (14):

$$h = \frac{fw}{b_2 - b_1}\sin(\phi)\sec(\theta) \tag{23}$$

Interestingly, we can arrange Equation (23) to develop two expressions for the scale factor S used in the transformation of Equations (6)-(7).

$$S \equiv \frac{h}{f}\csc(\phi) = \frac{w\sec(\theta)}{b_2 - b_1} \tag{24}$$

Thus, we see that for the purposes of calculating S, knowledge of h, f, and ϕ is equivalent to knowledge of w, θ, b_1, and b_2.

We introduce an overhead view of the roadway scene in Figure 7 to assist in the calculation of d from the parameters already estimated. Inserting the measurement $u = b_1$ into Equation (6) gives X_1. Then the intercept coordinate $Y_1 = X_1/\tan(\theta)$. Finally, we have

$$d = (Y_1 - (-h\cot(\phi))\sin(\theta) \tag{25}$$

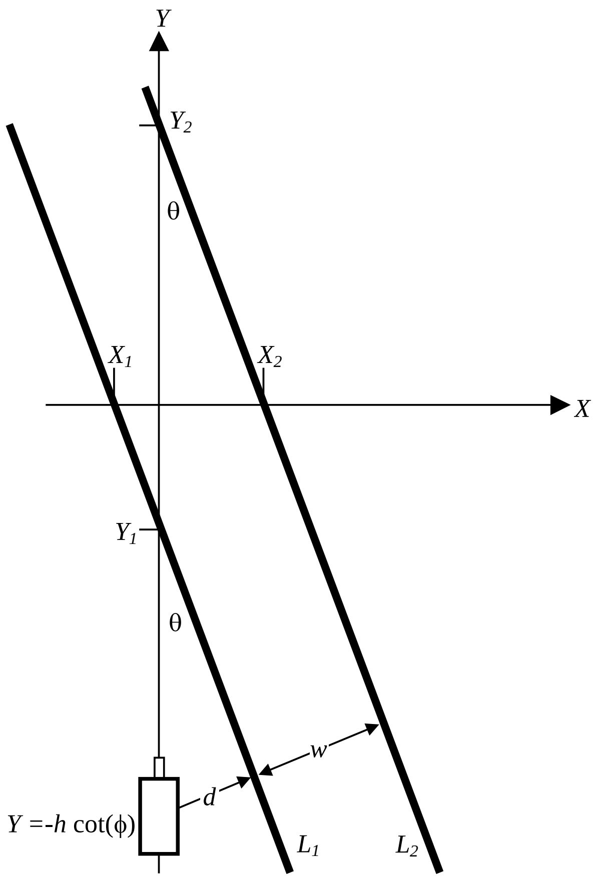


Figure 7. Road geometry from a bird's-eye view with relevant *X*- and *Y*-axis intercepts.

2.2.2 Method 2 (known camera position)

If we know the scene geometry, i.e., parameters *d* and *w*, we can estimate the camera parameters by obtaining estimates of the line parameters b_1 and b_2 and vanishing point coordinates u_0 and v_0 from the digital images. Using Equation (14),

$$b_1(d+w)-b_2 d = -wf\cos(\phi)\tan(\theta) \tag{26}$$

Dividing both sides by u_0 from Equation (19) yields

$$\begin{gathered}\frac{b_2 d - b_1(d+w)}{u_0} = \frac{wf\cos(\phi)\tan(\theta)}{-f\sec(\phi)\tan(\theta)} = -w\cos^2(\phi)\\ \phi = \arccos\left(\left(\frac{b_1(d+w)-b_2 d}{u_0 w}\right)^{1/2}\right)\end{gathered} \tag{27}$$

Equations (19)-(20) may be used to estimate *f* and θ. It is interesting to note that *a priori* knowledge of *h* is not needed to solve for the three camera parameters. As pointed out in Equation (24), knowledge of *w*, θ, b_1, and b_2 is sufficient for calculating the scale factor without *h*.

2.2.3 Method 3 (known distance)

One way of viewing the camera calibration problem is that we need to measure two types of scaling information: parallel to the road and perpendicular to it, i.e., the *Y'* and *X'* axes in Figure 7, respectively. As described in Section 2.2.2, we can measure the distance along the *X*-axis (which isn't necessarily perpendicular to the road) to within a scale factor using Equation (6) when $v = 0$, i.e., $X\big|_{v=0} = S \cdot u$. If at least one additional distance is known that is somewhat independent from $b_2 - b_1$, then we can derive the remaining scene and calibration information that we lack.

In the image, we traverse from v_a to v_b, which corresponds to a known distance *L* along the road. Transforming these points into 3-D using Equation (7), this distance is

$$\Delta Y = \frac{Sv_0}{\sin(\phi)}\left(\frac{v_b}{v_0 - v_b} - \frac{v_a}{v_0 - v_a}\right) \tag{28}$$

Referring to Figure 7, we can see that

$$L = \Delta Y \sec(\theta) = \frac{Sv_0 \sec(\theta)}{\sin(\phi)}\left(\frac{v_0(v_b - v_a)}{(v_0 - v_b)(v_0 - v_a)}\right) \tag{29}$$

For ease of derivation, we define

$$\tau_L \equiv \left(\frac{v_0(v_b - v_a)}{(v_0 - v_b)(v_0 - v_a)}\right) \tag{30}$$

Squaring both sides and performing some algebra, we have

$$L^2 = \frac{\tau_L{}^2 S^2 v_0{}^2 \sec^2(\theta)}{\sin^2(\phi)} = \frac{\tau_L{}^2 S^2 f^2 \tan^2(\phi)\sec^2(\theta)}{\sin^2(\phi)} = \tau_L{}^2 S^2 f^2 \sec^2(\phi)\sec^2(\theta) \tag{31}$$

As an aside, we use Equation (23) and the fact that $S \equiv h/f \csc(\phi)$ to obtain

$$S = \frac{w\sec(\theta)}{(b_2 - b_1)} \tag{32}$$

Inserting Equation (32) into Equation (31), we have

$$L^2 = \tau_L{}^2 f^2 \sec^2(\phi)\sec^2(\theta)\left(\frac{w}{b_2 - b_1}\right)^2 \sec^2(\theta) \tag{33}$$

Applying the trigonometric identity sec^2(·) = 1 + tan^2(·) several times yields

$$\begin{aligned}
L^2 &= \tau_L{}^2 f^2 \left(1+\tan^2(\phi)\right)\left(1+\tan^2(\theta)\right)\left(\frac{w}{b_2 - b_1}\right)^2 \sec^2(\theta) \\
L^2 &= \tau_L{}^2 \left(f^2 + f^2\tan^2(\phi) + f^2\tan^2(\phi)\tan^2(\theta) + f^2\tan^2(\theta)\right)\left(\frac{w}{b_2 - b_1}\right)^2 \sec^2(\theta) \\
L^2 &= \tau_L{}^2 \left(f^2 + f^2\tan^2(\phi) + f^2\tan^2(\phi)\sec^2(\theta)\right)\left(\frac{w}{b_2 - b_1}\right)^2 \sec^2(\theta)
\end{aligned} \tag{34}$$

Applying the identities in Equation (19)-(20) for *u₀* and *v₀* gives us

$$L^2 = \tau_L{}^2 \left(f^2 + v_0{}^2 + u_0{}^2\right)\left(\frac{w}{b_2 - b_1}\right)^2 \sec^2(\theta) \tag{35}$$

along with a new identity

$$f^2 + v_0{}^2 + u_0{}^2 = f^2 \sec^2(\phi)\sec^2(\theta) \tag{36}$$

Noting that

$$\sec^2(\theta) = \frac{f^2 + v_0{}^2 + u_0{}^2}{f^2\sec^2(\phi)} = \frac{f^2 + v_0{}^2 + u_0{}^2}{f^2\left(1+\tan^2(\phi)\right)} = \frac{f^2 + v_0{}^2 + u_0{}^2}{f^2 + v_0{}^2} \tag{37}$$

we finally obtain

$$L^2 = \tau_L{}^2 \left(\frac{w}{b_2 - b_1}\right)^2 \frac{\left(f^2 + v_0{}^2 + u_0{}^2\right)^2}{f^2 + v_0{}^2} \tag{38}$$

Equation (38) represents an expression involving only a single unknown parameter (*f*), the known road width *w*, and the measured quantities u_0, v_0, v_1, v_2, b_1, and b_2 from the image. To solve the polynomial for *f*, we first define

$$\alpha_0 \equiv \left(\frac{L(b_2 - b_1)}{w\tau_L}\right)^2 \tag{39}$$

Expanding Equation (38) results in

$$\begin{aligned} \alpha_0\left(f^2+v_0{}^2\right) &= f^4+2\left(u_0{}^2+v_0{}^2\right)+\left(u_0{}^2+v_0{}^2\right)^2 \\ 0 &= f^4+f^2\left(2\left(u_0{}^2+v_0{}^2\right)-\alpha_0\right)+\left(u_0{}^2+v_0{}^2\right)^2-\alpha_0 v_0{}^2 \end{aligned} \tag{40}$$

We define

$$\begin{aligned} \alpha_1 &\equiv 2\left(u_0{}^2+v_0{}^2\right)-\alpha_0 \\ \alpha_2 &\equiv \left(u_0{}^2+v_0{}^2\right)^2-\alpha_0 v_0{}^2 \end{aligned} \tag{41}$$

We can now solve for *f*, taking the positive root for a real-valued solution

$$\begin{aligned} f^2 &= -\frac{\alpha_1}{2}\pm\sqrt{\frac{\alpha_1{}^2}{4}-\alpha_2} \\ f &= \sqrt{-\frac{\alpha_1}{2}+\sqrt{\frac{\alpha_1{}^2}{4}-\alpha_2}} \end{aligned} \tag{42}$$

The remaining parameters θ, ϕ, *h*, and *d* can now be found, as described previously in Equations (19), (20), (23), and (25).

Because we assumed that the road has no slope or tilt to it, we note that the 2-D to 3-D transformation in Equation (7) only involves the vanishing point coordinate v_0 and a scaling factor $h\csc^2(\phi)/f$ that is constant for a given scene. Referring to Figure 7, we see that multiplying the 3-D position *Y* by sec(θ) yields the position *Y'* along the road itself. Thus, we can express the position of a vehicle along the road in terms of a single scale factor

$$Y' = \frac{h}{f}\csc^2(\phi)\sec(\theta)v_0\left(\frac{v}{v_0-v}\right) = S'\left(\frac{v}{v_0-v}\right) \tag{43}$$

Thus, if we know that two points (u_a,v_a) and (u_b,v_b) in the image are separated by a known distance *L* (as projected onto the Z_c'-axis parallel to the road) and we have an estimate of v_0, then we can use this information to obtain the scaling factor *S'* as follows by manipulating Equation (29).

$$S' \equiv \frac{h}{f}\csc(\phi)\frac{\sec(\theta)}{\sin(\phi)}v_0 = \frac{Sv_0\sec(\theta)}{\sin(\phi)} = L\frac{(v_0-v_a)(v_0-v_b)}{v_0(v_b-v_a)} = \frac{L}{\tau_L} \tag{44}$$

In this way, we can transform the vertical position of a vehicle in the image into its position in 3-D with minimal knowledge of the scene geometry and no knowledge of the camera parameters. In other words, as far as measuring distances along the road is concerned, we need not calculate the camera parameters as above unless we wish to estimate *d* or *h* (see Table 1).

In conclusion, it is interesting to note that, regardless of the camera calibration method, we are simply estimating *S'* in order to estimate distances along the road. Equation (44) shows that our various approaches are just ways of manipulating the multiple equalities between *S'* and the various camera, image, and scene parameters. Thus, it is very appropriate to use *S'* as a means of comparing the accuracy of the three approaches to calibration, since the only other unknown in Equation (43) is v_0, to which all three have equal access via the same algorithm.

2.3 Sensitivity analysis via Monte-Carlo simulation

An algorithm that processes images from traffic monitoring cameras in Seattle usually encounters three types of scenes: 1) a side view where the camera is a moderate distance from the road and looks out at the road with moderate values for ϕ and θ; 2) a view where the camera looks down on the road with large values for ϕ and θ; 3) a view where the camera is located on an overpass above the road and looks straight down the road. Figure 8 contains three images representing these types of scenes.

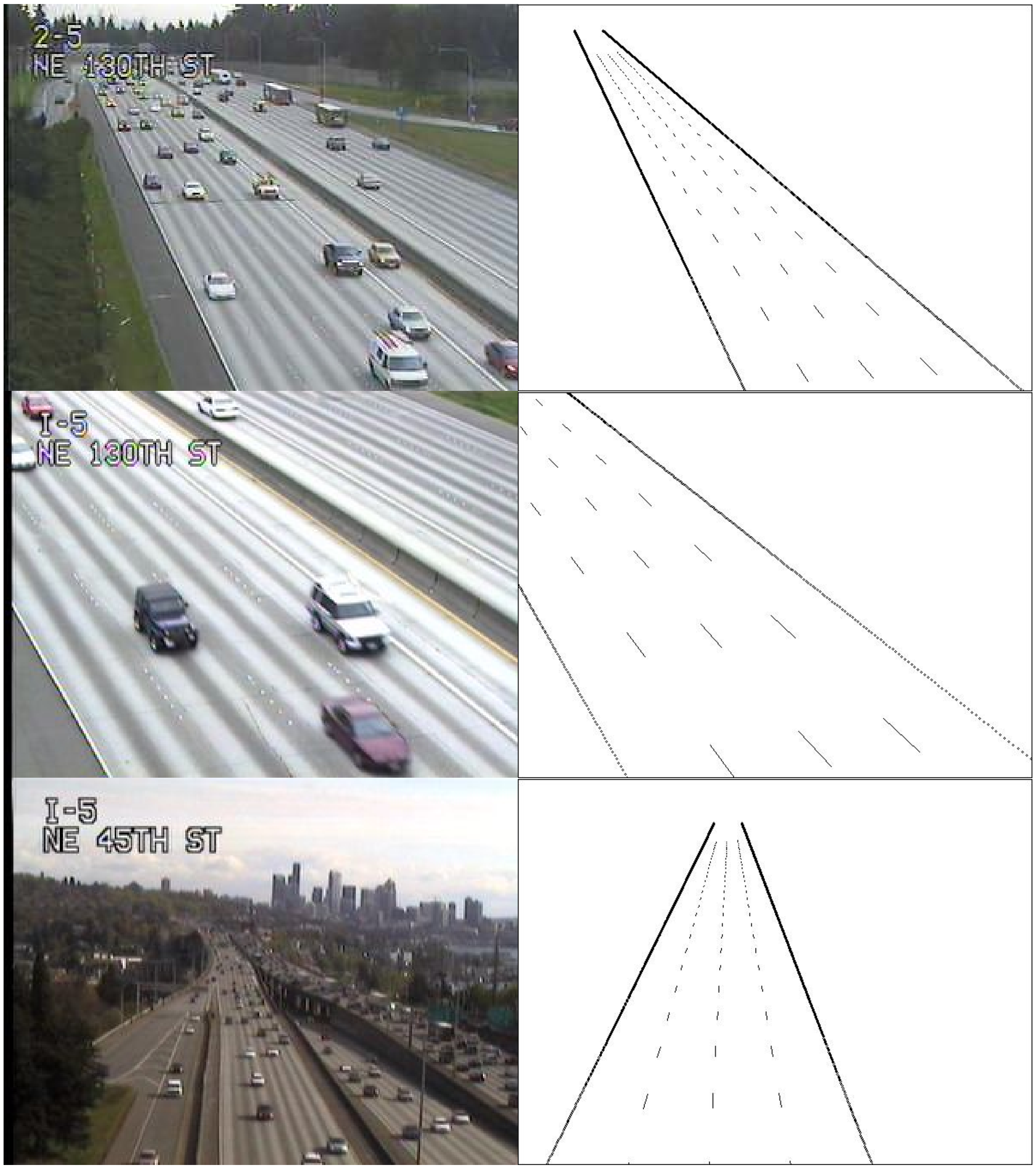


Figure 8. Typical scenes viewed by a traffic camera in Seattle, Washington and their computer simulation (from top-to-bottom). a) Camera on the side of the road (moderate values for ϕ and θ). b) Camera looking down on the traffic (large values for ϕ and θ). c) Camera located on an overpass (small θ).

We designed a Monte-Carlo simulation to determine the sensitivity of the various camera calibration methods when viewing the different scenes. Table 2 contains the geometrical and camera parameters used. Note that the traffic image and simulated versions of scene 3 are mirror images of one another because the simulated version has a much more negative value for *d* than the traffic image.

Table 2. Exact geometrical and camera parameters used in the simulation.

	Scene 1	Scene 2	Scene 3
Width of road *w* (feet)	44	44	44
Lane stripe distance (feet)	40	40	40
Camera-to-road distance *d* (feet)	28.3	28.3	-25
Camera height *h* (feet)	63.5	63.5	50
Lane stripe distance (feet)	40	40	40
Focal length *f* (pixels)	1600	1600	1600
Down angle ϕ (deg.)	9.2	18	8
Pan angle θ (deg.)	9.6	20	2

On the basis of these parameters, we generated nominal measurements for u_0, v_0, u_1, b_1, and b_2 for each scene using Equations (14), (19), (20), and(21). These values are detailed in Table 3.

Table 3. Image measurements calculated from the parameters of Table 2.

Measurement	Scene 1	Scene 2	Scene 3
u_0 (pixels)	-274.1463	-612.3215	-56.4223
v_0 (pixels)	259.1435	519.8715	224.8653
u_1 (pixels)	9583.04	4622.19	46268.3
b_1 (pixels)	-151.5127	-319.3574	-166.7358
b_2 (pixels)	28.2589	45.2248	29.3393

We designed the Monte-Carlo simulation to test the sensitivity of each of the outputs for each camera calibration method to each of its inputs. Table 4 contains the details of the inputs and outputs for each method. We emphasize that $\Delta Y'$ denotes the primary quantity of interest, i.e., the distance measured along the road. In addition to determining the sensitivity of individual output variables to variations in individual input variables (one-to-one), we also tested the cumulative effects of variation in all the inputs on each of the individual outputs (many-to-one) for each calibration method. Furthermore, we exhaustively performed these two types of sensitivity analyses for each of the three scenes in combination with each of the three calibration methods. Table 5 contains a matrix summarizing the complete results.

Table 4. Inputs and outputs for each calibration method.

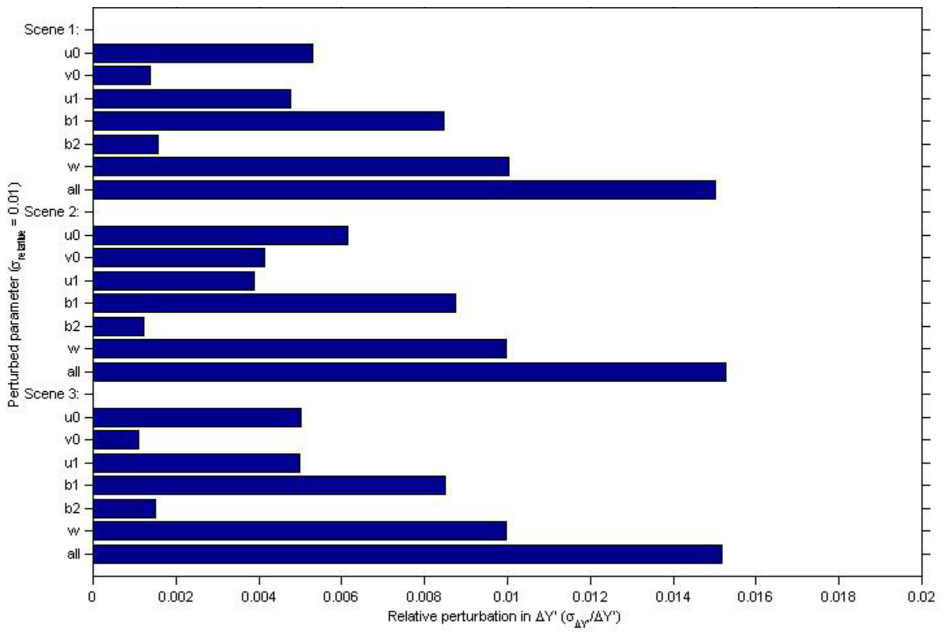
	Method 1	Method 2	Method 3
Inputs Perturbed	w u_0, v_0, u_1 b_1, b_2	w, d u_0, v_0 b_1, b_2	w, L u_0, v_0 b_1, b_2 τ_L
Outputs Perturbed	d f, ϕ, θ, $\Delta Y'$	f, ϕ, θ, $\Delta Y'$	d f, ϕ, θ, $\Delta Y'$

Table 5. Sensitivity of the calibration methods to input errors when applied to different types of scenes.

	Method 1	Method 2	Method 3
Scene 1	Low	High	Low
Scene 2	Medium	Medium	Medium
Scene 3	Low	High	Low

To determine the approximate sensitivity of each calibration method, we systematically added gaussian noise to each of the input parameters appropriate for each method, as outlined in Table 4. The standard deviation σ of the additive noise was chosen to be 1 percent of the value of the parameter itself, e.g., σ_{u0} = |-274.1463| / 100 = 2.741463 for Scene 1. Of course, the actual noise levels for each parameter will probably exceed this level, but these will indicate how much each of the methods magnifies noise in its inputs.

We obtained 10^5 samples of each input for each scene (see Table 2 and Table 3) with the appropriate noise level. For each combination of the three calibration methods and three scenes, we calculated 10^5 samples of each of the tested outputs listed in Table 4. We generated multiple sets of 10^5 samples for each outputs, by varying the inputs one at a time, and then all together. For example, we applied Method 1 to Scene 1 and calculated 10^5 samples of *d* for seven cases: six in which all inputs were perfect except for one changed parameter, and one case in which all the inputs were varied by adding gaussian noise with a standard deviation of 1 percent relative to the nominal value. Method 1 yielded 315 sets, Method 2 totaled 252 sets, and Method 3 gave 360 sets of 10^5 samples. After generating all the sets of data, we calculated the standard deviation of each set of samples and normalized each value by the nominal value from Table 2 and Table 3. Figures 9 through 14 contain bar graphs illustrating the results for all the combinations.



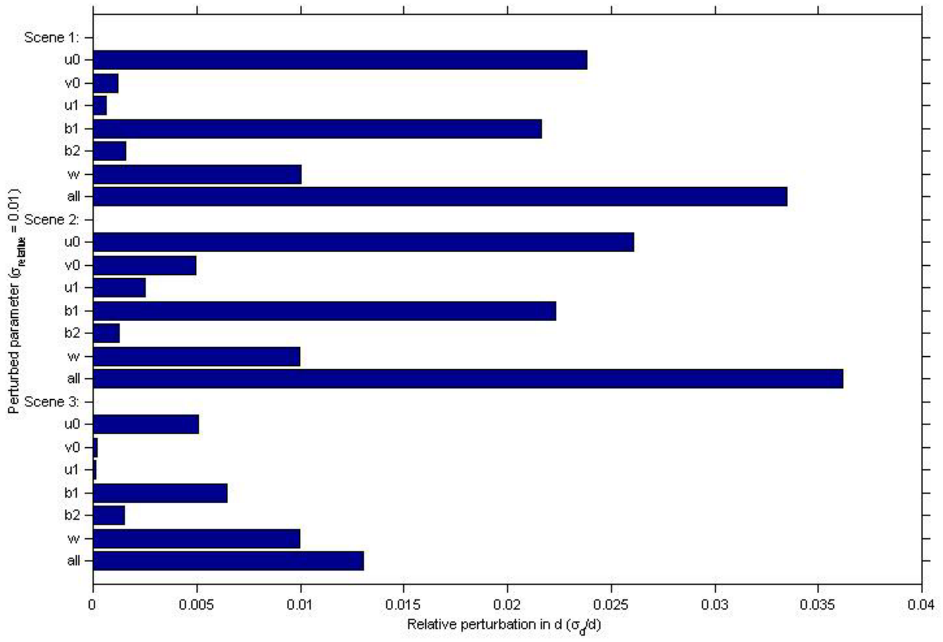


Figure 9. Relative contributions to the errors in road distance $\Delta Y'$ and d measured via Method 1 for individual input errors, where each input i has value Q_i and noise $\sigma_i/Q_i = 0.01$.

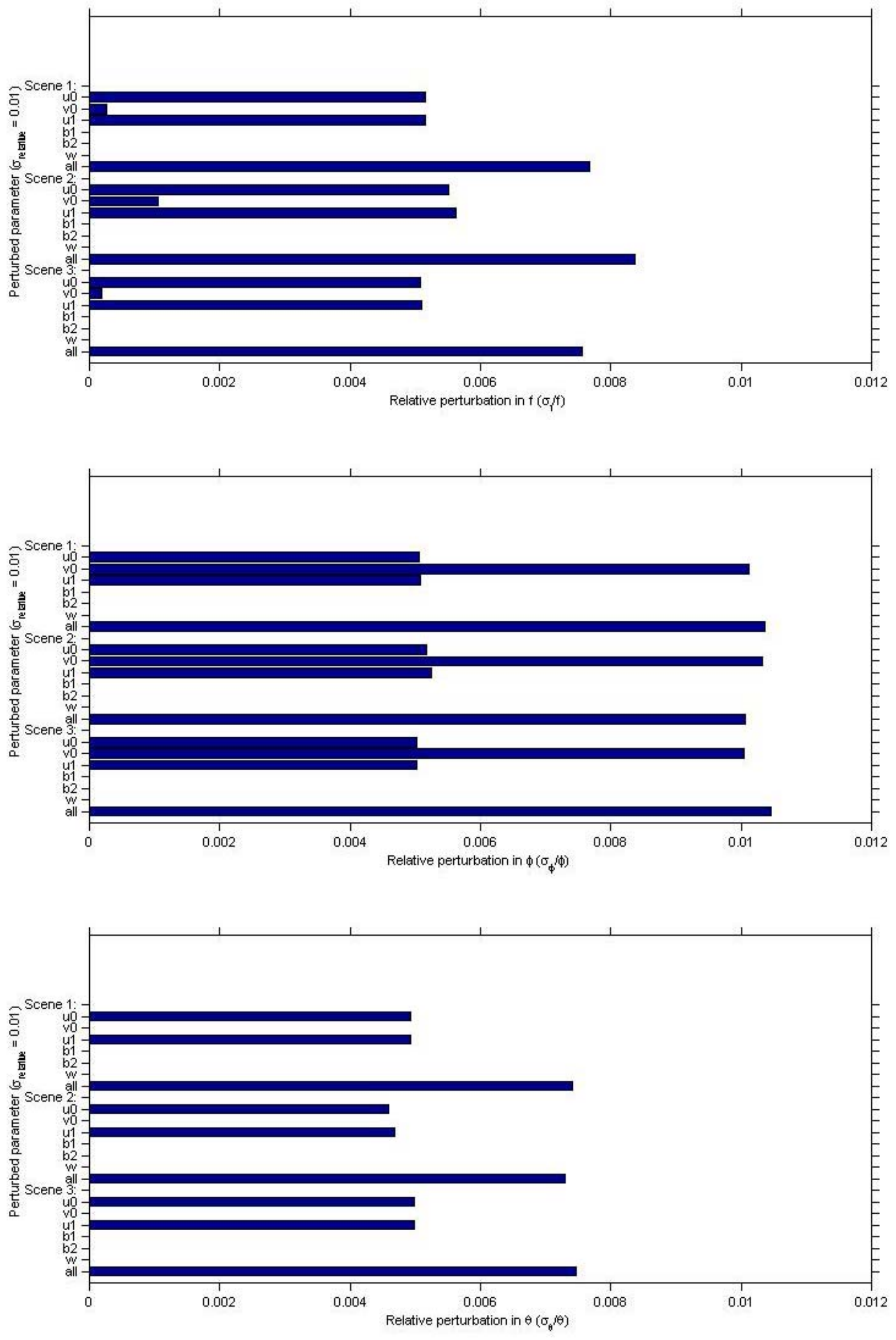


Figure 10. Relative contributions to the errors in camera parameters *f*, ϕ, and θ measured via Method 1 for individual input errors, where each input *i* has value Q_i and noise $\sigma_i/Q_i = 0.01$.

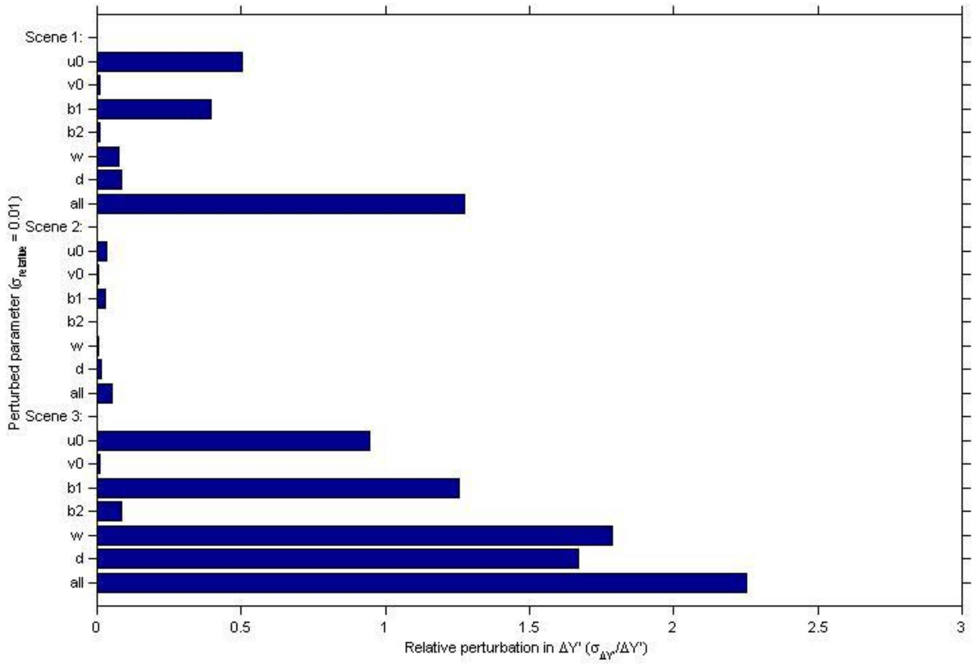
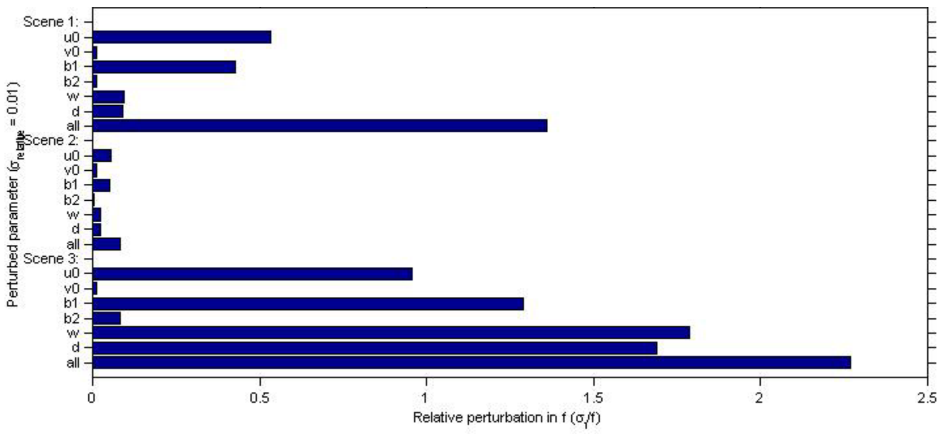
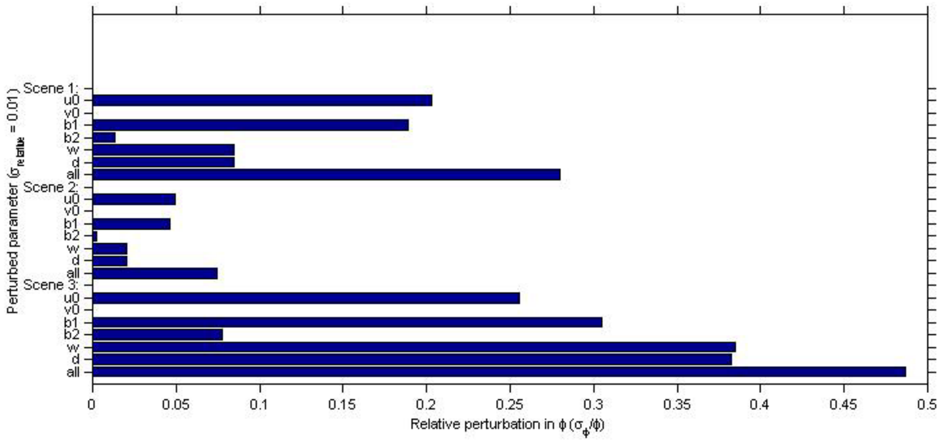


Figure 11. Relative contributions to the errors in road distance $\Delta Y'$ measured via Method 2 for individual input errors, where each input i has value Q_i and noise $\sigma_i/Q_i = 0.01$.





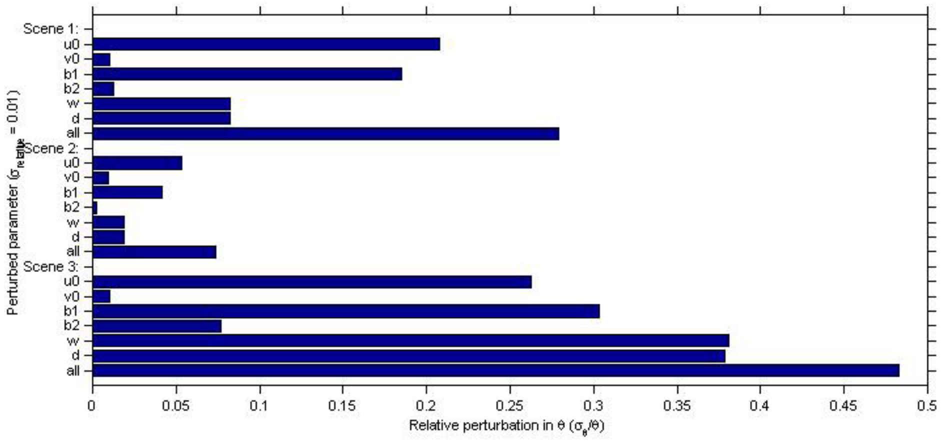
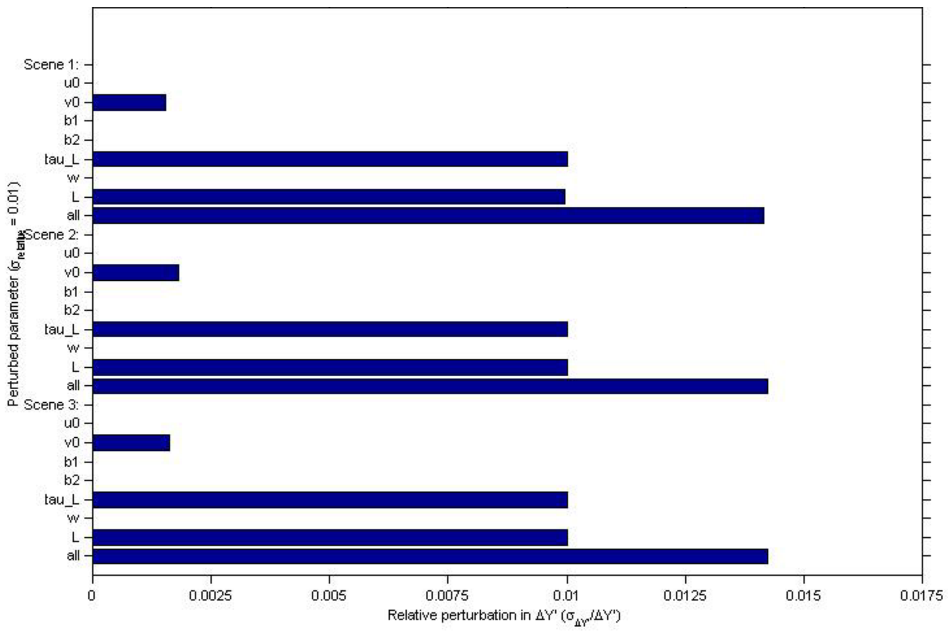


Figure 12. Relative contributions to the errors in camera parameters *f*, ϕ, and θ measured via Method 2 for individual input errors, where each input *i* has value Q_i and noise $\sigma_i/Q_i = 0.01$.



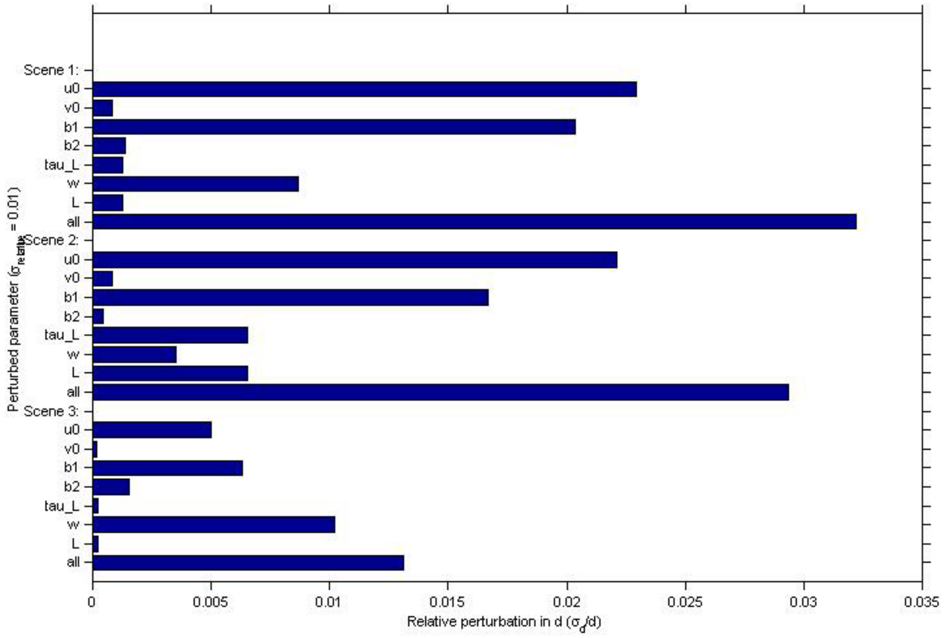
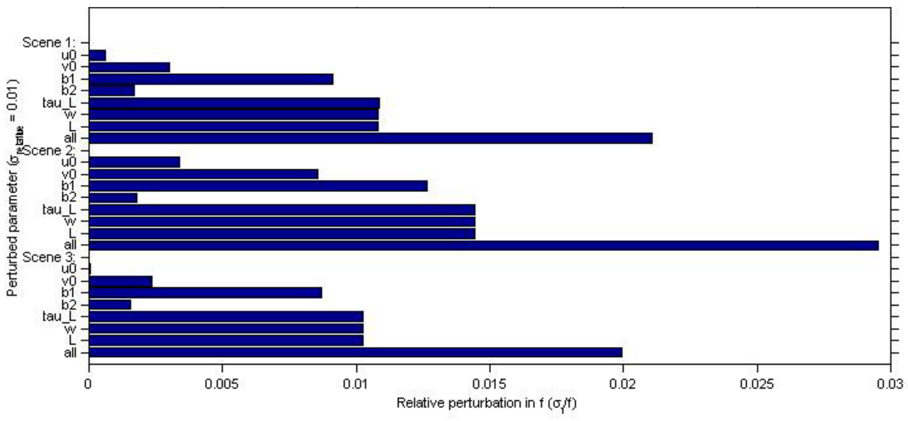
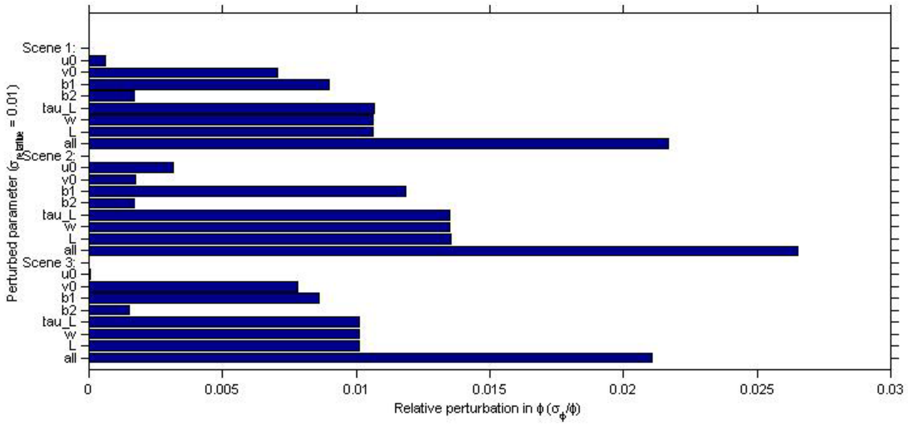


Figure 13. Relative contributions to the errors in road distance $\Delta Y'$ and d measured via Method 3 for individual input errors, where each input i has value Q_i and noise $\sigma_i/Q_i = 0.01$.





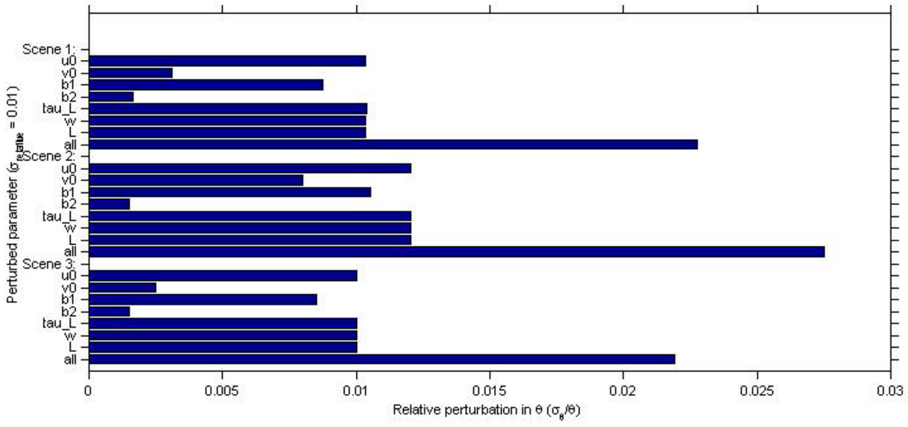


Figure 14. Relative contributions to the errors in camera parameters f, ϕ, and θ measured via Method 3 for individual input errors, where each input i has value Q_i and noise $\sigma_i/Q_i = 0.01$.

As one would expect, the case in which all inputs are varied results in the greatest effect on the output. Even in this case, the outputs of Method 1 typically have fairly tight distributions about their nominal values, as illustrated in the results for Scene 1 in Figure 15 (the nominal value is indicated by a vertical line). The histograms for Method 3 have a similar shape. This indicates that these methods are fairly insensitive to small errors in the input parameters. However, Method 2 is quite sensitive to variations in its inputs, and the output distributions are typically broad and skewed, as shown in Figure 16.

We now examine figures 9 through 14 in detail to identify the parameters to which each method is most sensitive. The results of Method 1 in figures 9 and 10 show that both d and $\Delta Y'$ depend linearly on w because the output deviation is 0.01 when w deviates by 0.01. This is consistent with the proportionality between w and the scale factor S in Equation (24). Overall, the parameters b_1 and w are the largest contributors of error to $\Delta Y'$. Examining the results for the camera parameters f, ϕ, and θ in Figure 10 shows that they depend only on the measured vanishing point coordinates, as expected from Equations (19)-(20). As a corollary, we observe that the patterns of relative error are consistent for each of the five output parameters, regardless of the scene, but that each output is uniquely sensitive to the different input variations.

Figures 11 and 12 contain the results for Method 2. Unlike Method 1, Method 2 responds very differently to each of the three scenes. Each of the four output parameter variation patterns is consistent for a given scene, but very different between scenes. Based on the relative error values, Scene 2 is the only scene that is amenable to calibration via Method 2. Scene 1 is too sensitive to small errors in u_0 and b_1 to be useful. Similarly, Scene 3 is hypersensitive to u_0, b_1, d and w. All four of these parameters appear in the solution for ϕ in Equation (27) and contribute different levels of error, depending on the camera parameters.

As shown in Figure 13, the distance $\Delta Y'$ computed by Method 3 depends only on v_0, τ_L, and L, as predicted by Equations (43)-(44). Of these, L and τ_L are equally quite significant. We note from Figure 13 that the variation of $\Delta Y'$ is completely independent of the type of scene. The calculations for d and the camera parameters f, ϕ, and θ involve all of the inputs to some degree, as shown in figures 13 and 14. The parameter d depends most on u_0 and b_1, which is not surprising since all three have to do with scaling perpendicular to the road, whereas v_0, τ_L, and L all serve to provide scaling along the road in the 2-D-to-3-D coordinate conversion. The camera parameters f, ϕ, and θ are generally most sensitive to L and w, though they are also sensitive to b_1 (and u_0 in the case of θ).

When comparing the values on the abscissa for the figures associated with the three methods, we can see that the estimates of f, ϕ, and θ by Method 1 are least sensitive to input variations. On the other hand, Method 3 generally owns the same distinction for the outputs $\Delta Y'$ and d. Method 2 is clearly hypersensitive because it typically amplifies the input variations in the output rather than attenuating them. Of course, the actual relative noise levels for the input parameters will exceed 0.01, but these simulation results help us understand how much each method magnifies the noise in its inputs.

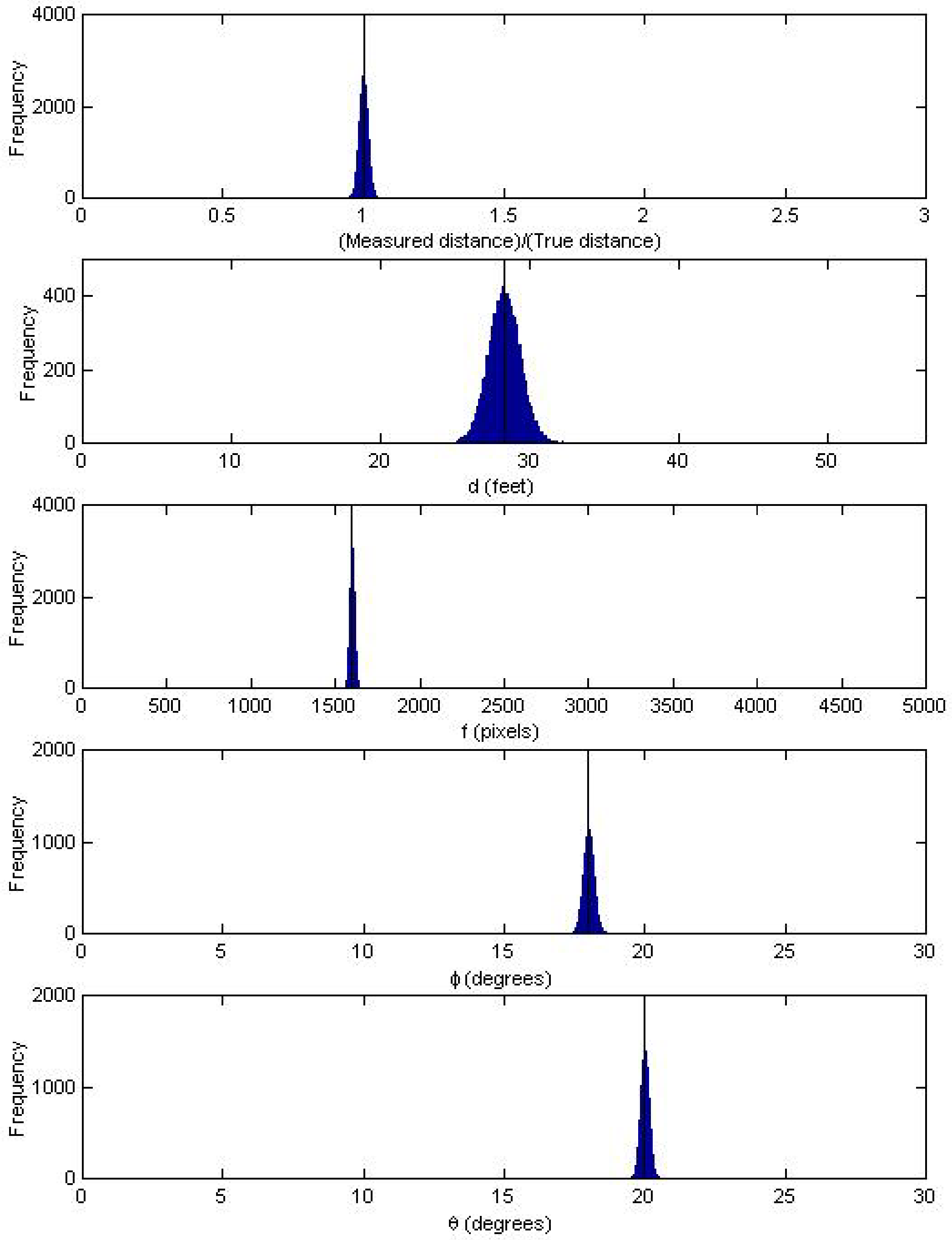


Figure 15. Distribution of output parameters for Method 1 (Scene 2), where each input *i* has value Q_i and additive gaussian noise $\sigma_i/Q_i = 0.01$.

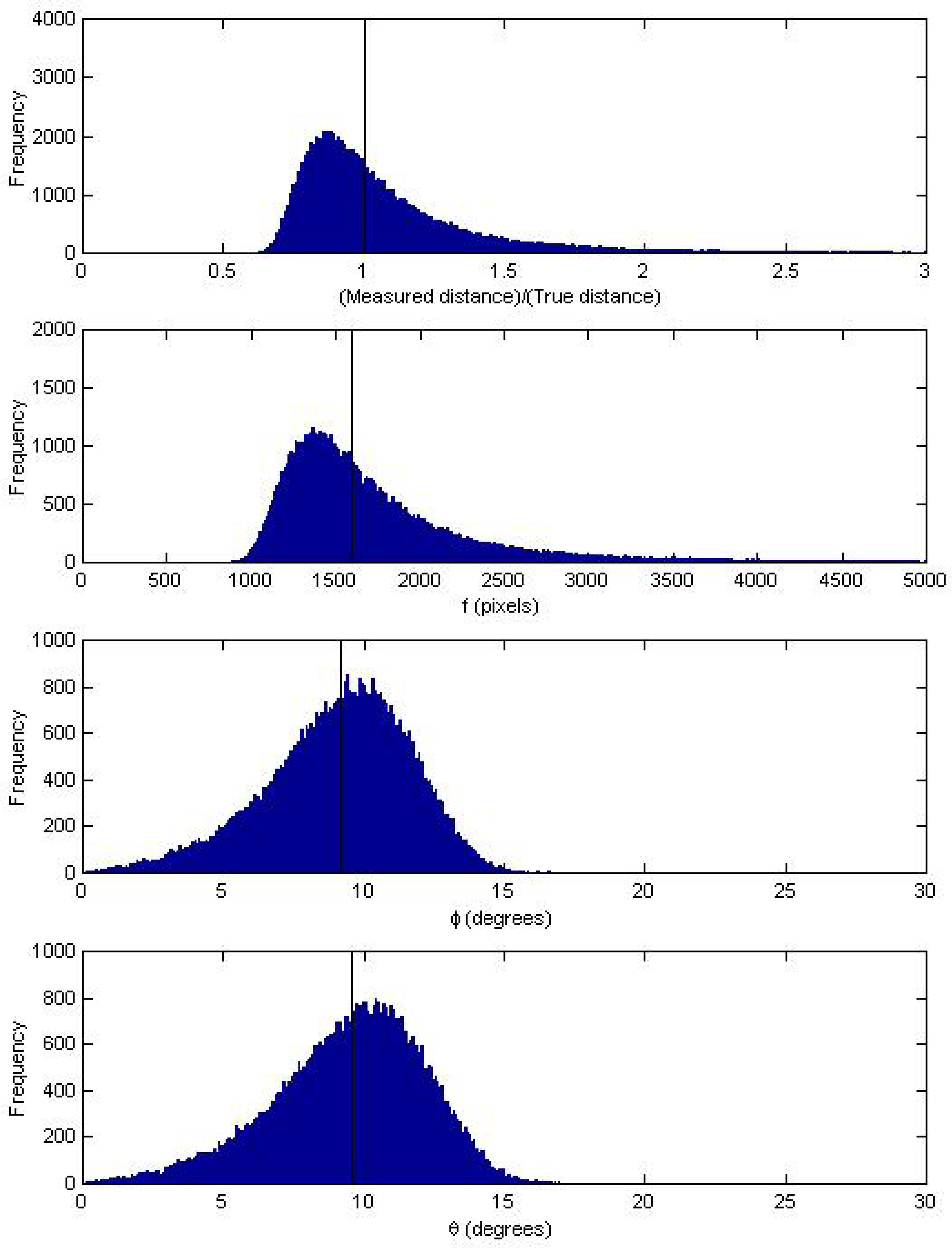


Figure 16. Distribution of output parameters for Method 2 (Scene 1), where each input *i* has value Q_i and noise $\sigma_i/Q_i = 0.01$.

2.4 Range of operation

As seen in Section 2.3, the sensitivity of the camera calibration method can limit its usefulness in certain situations. For example, for a relative noise level of about 1 percent in its input parameters, Method 2 should only be used for situations similar to Scene 2, where the tilt angle ϕ is large. We now offer a more well-defined description of the range of operation in which we can expect the different calibration methods to work properly under reasonable levels of noise in their inputs. In essence, we are attempting to define a region or regions in a multivariable space involving *w, d, h, f,* ϕ, and θ. Because the space has six dimensions, defining and visualizing it to determine the behavior of each method is a daunting task.

We begin by acknowledging our assumption that the road boundary lines do not enter or exit the bottom half of the image, i.e., they are fully present in this subimage. This restriction ensures that our image processing algorithms have the best chance of success, though they will likely function properly even if it is relaxed slightly. Specifically, suppose we fix the parameters *w, d, h,* and *f*, and find the valid range of ϕ and θ. We do this by requiring the road lines in the bottom half of the image to enter through the bottom of the image at $v = -H/2$ and remain within the image at least up to the middle of the image $v = 0$. For example, in the simulated scenes in Figure 8, we see that the bottom halves of the window of scenes 1 and 3 completely contain the road boundary lines. However, the right boundary of Scene 2 exits the side of the image rather than the bottom, and the position at which the left boundary exits the side of the image may be too low.

We use the following inequalities involving the slopes m_1 and m_2 and *u*-intercepts b_1 and b_2 from the road boundary lines described by Equation (14). The inequalities are applied at the top and bottom of the half-image, i.e., $v = 0$ and $v = -H/2$:

$$\begin{aligned} -\frac{W}{2} &< b_1 < \frac{W}{2} \\ -\frac{W}{2} &< b_2 < \frac{W}{2} \\ -\frac{W}{2} &< m_1\left(-\frac{H}{2}\right) + b_1 \\ m_2\left(-\frac{H}{2}\right) + b_2 &< \frac{W}{2} \end{aligned} \tag{45}$$

In order for us to apply either Method 1 or Method 2 to calibrate the camera, we must be able to estimate the parameters of the road boundary lines. Thus, we can examine the ϕ-θ plane to determine which values of ϕ and θ satisfy Equation (45). We illustrate the accumulation of the constraints of Equation (45) for the geometry of Scene 1 in Figure 17. The gray areas indicate values of ϕ and θ that satisfy one or more of the inequalities in Equation (45), and the dark area indicates the values of ϕ and θ that satisfy all the constraints in Equation (45). In this particular example, we see that the constraints on b_1 at $v = 0$ and one of the constraints at $v = -H/2$ determine the region. We note that the requirement of keeping the road in full view of the camera is surprisingly restrictive, i.e., the shaded area is relatively small. From Equation (14), we note that f is directly proportional to b_1 and b_2. Thus, using a smaller value for f will enlarge the useable area, whereas raising f will shrink the shaded regions. In other words, zooming the camera (increasing f) will eventually make it impossible for the bottom half of the image to fully contain both road boundary lines.

When determining the useable region in the six-dimensional space, the constraints of Equation (45) give us a maximum size for the ϕ-θ subspace, assuming w, d, h, and f are already fixed. If we add noise to the inputs and require the output noise levels to remain below a threshold, we expect the acceptable region to shrink even further. Our strategy, then, is to vary w, d, h, and f in a systematic fashion and examine the limits of the camera angles ϕ and θ.

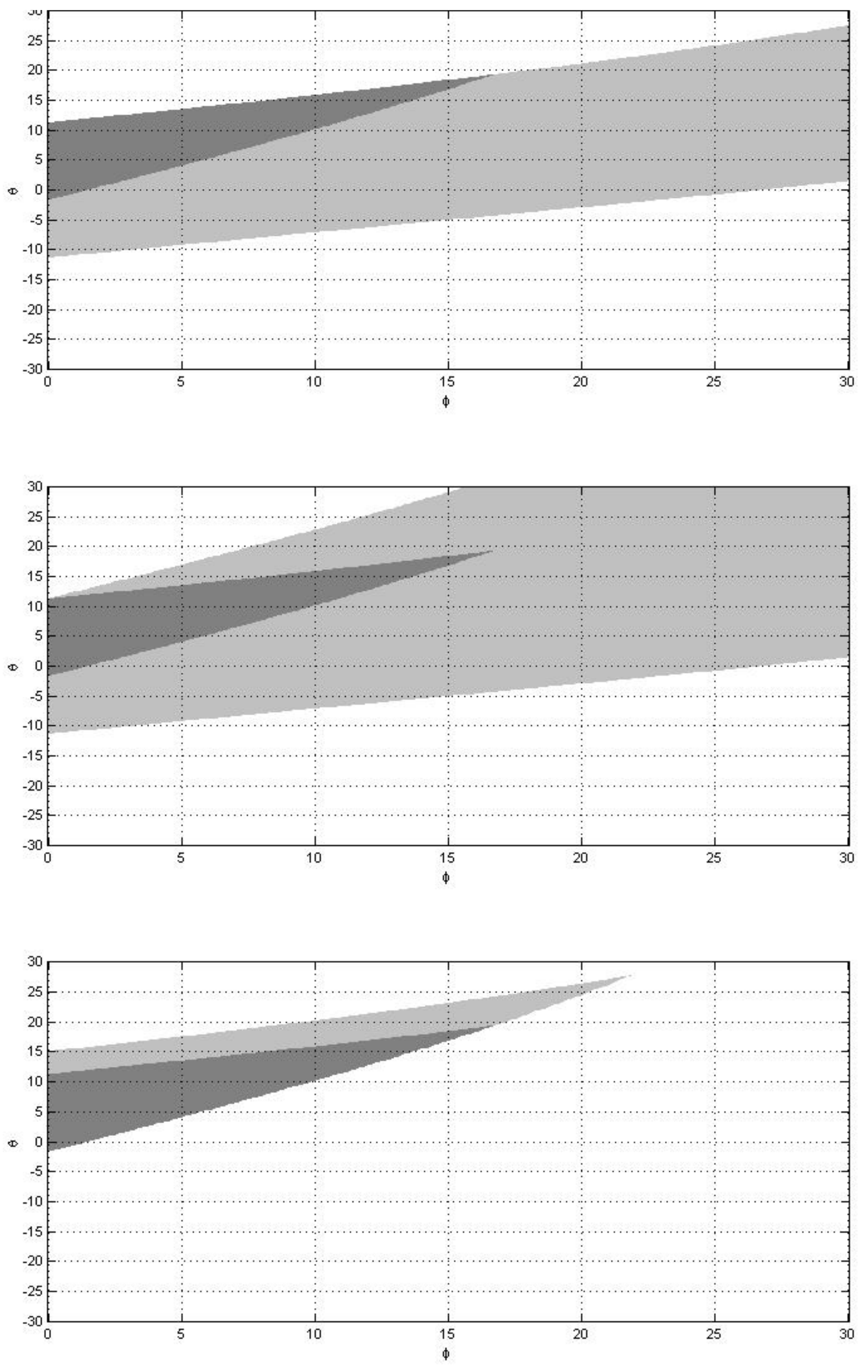


Figure 17. Valid range of pan and tilt angles to keep both road lines in view in scenes 1 and 2. The dark region indicates the intersection of all the constraints. a) Constraints from b_1. b) Constraints from b_2. c) Constraints from the horizontal road line coordinates on the bottom row of the image.

2.4.1 Limits of operation for Method 1

In addition to the road line constraints above, Method 1 requires that the lines perpendicular to the road have a sufficiently large slope ($\Delta v/\Delta u$ in Figure 5) so they can be measured in the image. We assume that the slope of the line must exceed 1/15 in order to be measured in the digital image. Vehicle lanes are typically at least 45 pixels wide, so this ensures that we have a reasonable amount of angular resolution. In order to collect an adequate sample size, we require that (at a minimum) the lower third of the image contain lines with a slope of greater than 1/15. We note from Figure 5 that as the perpendicular line moves toward the bottom of the image, its slope $|\Delta v/\Delta u|$ increases. Thus, its behavior at the top of the image window is our restriction, which we state as follows:

$$\left|\frac{v_0 - \left(\frac{-H}{6}\right)}{u_1}\right| > \frac{1}{15} \tag{46}$$

If we add this restriction to the example of Scene 1 in Figure 17, it noticeably decreases the allowable range of camera pan and tilt, as shown in Figure 18 below.

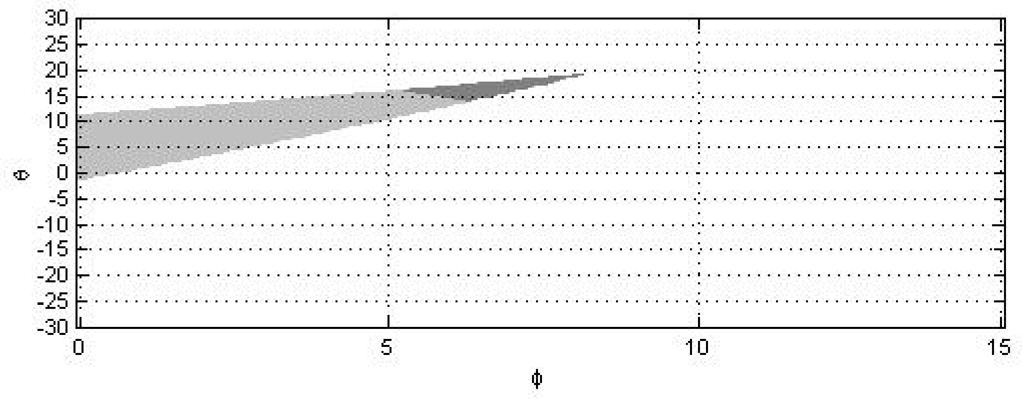


Figure 18. Valid range of pan and tilt angles to keep both road lines in view in Scene 1, where the perpendicular lines have a slope of greater than 1/15. The dark region indicates the intersection of all the constraints, and the lighter region shows the constraints found in Figure 17 as specified by Equation (45).

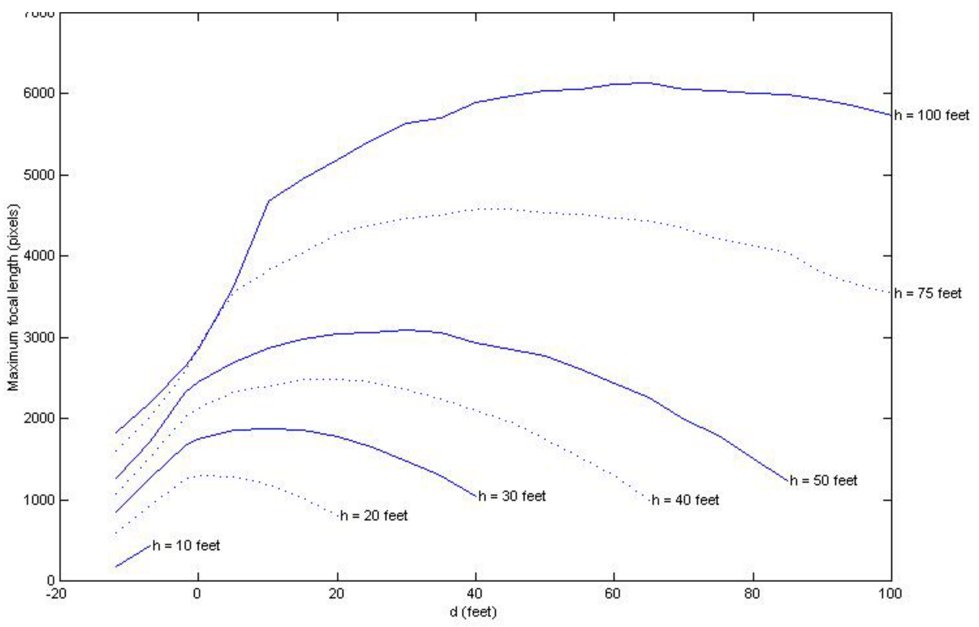
Note that the results of figures 17 and 18 were obtained for single, fixed values of *f*, *h*, *d*, and *w*. We built on this framework and developed a simulation to determine the effects of varying the scene geometry on the allowable ranges for *f*, ϕ, and θ, which are under operator control. Our goal is to identify the regions where Method 1 is usable, assuming ideal measurements. Such results will enable engineers to select appropriate lenses for a given scene geometry. In our simulation, we varied *h* and *d* in increments of 5 feet and varied *w* to simulate two to five lanes of traffic. For the ranges of ϕ and θ like the plot of Figure 18, we determined the minimum and maximum focal length for which there were any values of ϕ and θ that met the requirements of Equations (45) and (46). We did not record the actual range of ϕ and θ, since we assumed the operator can guide the camera to an appropriate orientation for a given focal length.

We found that the minimum focal length was (at the most) 800 pixels for a 640 X 480 image when $h < 75$ feet and $d < 80$ feet. Values of *h* and *d* outside this range had a much larger minimum focal length when the road was very narrow. The cameras used by WSDOT have a minimum focal length ranging between 1000 and 1600 pixels for a 640 X 480 image, so the minimum focal length requirements of Method 1 should not prohibit its use.

Figures 19 and 20 provide output curves for the maximum allowable focal length. WSDOT cameras are equipped with a 10x zoom lens, so they can easily achieve focal lengths in excess of the largest values in figures 19 and 20. The curves in these figures should enable an engineer to design a system that would use Method 1 to calibrate the camera. For example, suppose the road is known to be 44 feet wide (i.e., four lanes), and the engineer has the choice of placing the camera atop a 40-foot pole on an overpass above the freeway or to the side of the road. Using Figure 20(a) as an approximation, the engineer can weigh the relative benefits of two options. First, if the camera is on top of the overpass over the traffic ($h \approx 70$ feet, $d < 0$), it can view both sides of the freeway. However, the slope of the perpendicular lines is generally small because the camera is close to the center of the road. In the second option, the camera can be placed to the side of the freeway (i.e., $d > 0$) at a shorter height ($h \approx 40$ feet), where it can view only one direction because of the obstruction of the overpass. Somewhat surprisingly, the greater value of *h* actually allows more flexibility in *d*, with $f \in (0,1500)$ for a wide range of *d*. In contrast, $h = 40$ implies that *f* must probably range somewhere between 600 and 1000 pixels, and the camera must be placed somewhere fairly near the left road boundary (while maintaining $d > 0$).

The white boundary lines for typical Seattle freeways contain four or fewer lanes between them. From Figure 20(a), we can see that a camera with the focal length minimum stated above must be at least 40 feet above the traffic in order to maintain these lines within the image. Fortunately, the WSDOT cameras are specified to be placed atop 40-foot poles. We also note that for smaller values of *h*, the camera must be positioned fairly close to the left boundary of the freeway. Thus, Method 1 is theoretically applicable to reasonably wide freeways using the standard WSDOT specifications, as long as the camera is positioned near one of the road boundary lines.

Even though these plots provide ranges for *f*, they do not identify the specific values of ϕ and θ necessary to satisfy the constraints of Equations (45) and (46). A general procedure to satisfy these constraints is the following: tilt the camera as far down as possible (maximize ϕ) while adjusting θ to keep the road lines within the bottom half of the image. This corresponds to locating the tip of the dark regions of the examples in figures 17 and 18. The simulation results in the figures below guarantee that the camera operator can find at least one orientation for the camera to satisfy the constraints of Equations (45) and (46), assuming that the focal length is in the appropriate range specified by the figures below for the chosen values of *w*, *h*, and *d*.



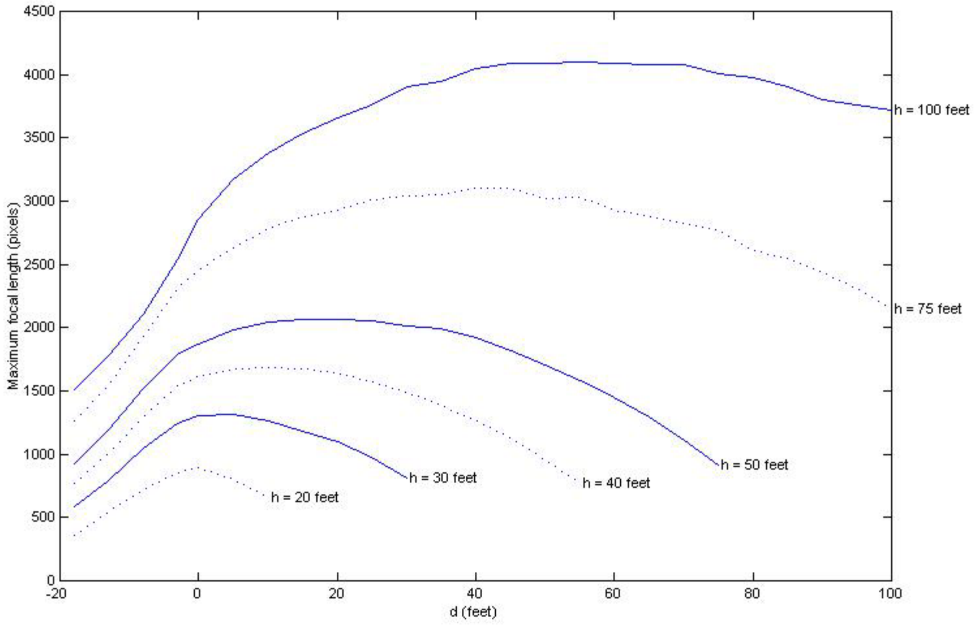
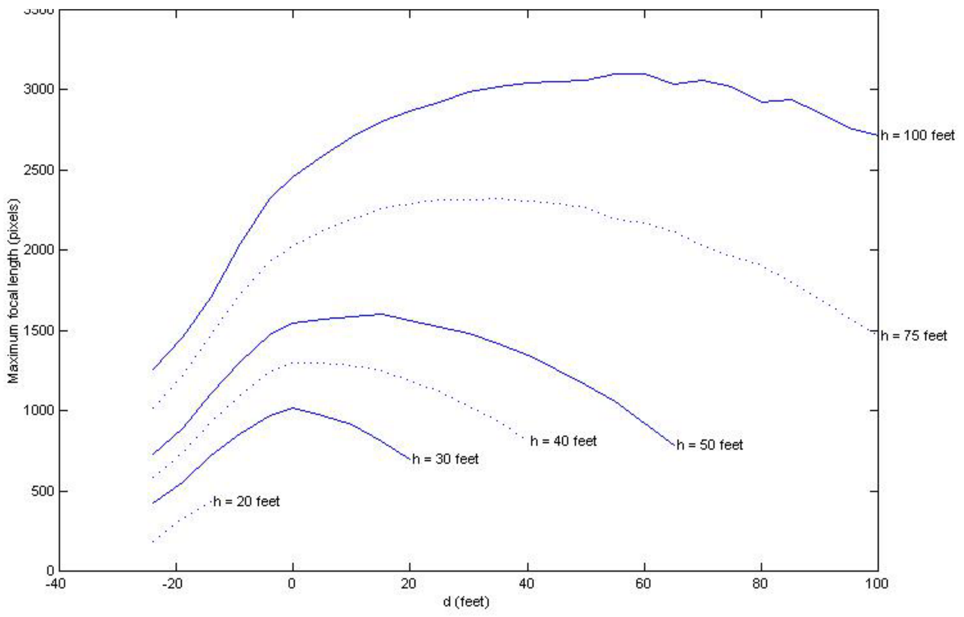


Figure 19. Maximum focal length allowed for different camera geometries in order to utilize camera calibration Method 1. a) Two lanes of traffic. b) Three lanes of traffic.



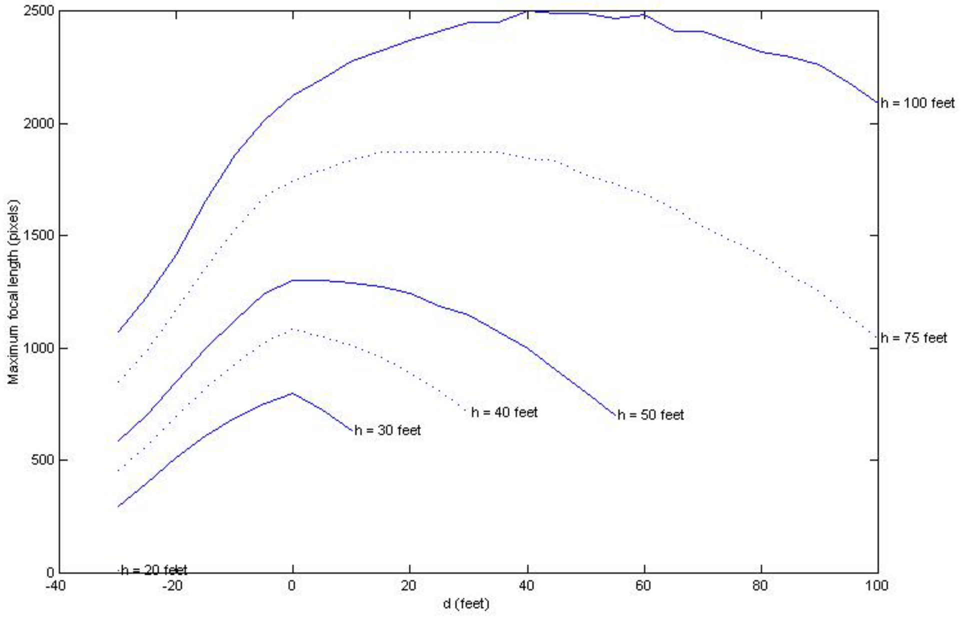


Figure 20. Maximum focal length allowed for different camera geometries in order to utilize camera calibration Method 1. a) Four lanes of traffic. b) Five lanes of traffic.

Once we specified the theoretical limits for f given h, d, and w, we fixed f at a nominal value of 1000 pixels. We developed a Monte-Carlo simulation to test the sensitivity of Method 1 to variations in its inputs. Our goal was to determine the limits of ϕ and θ that had an acceptable level of output variation, given fixed values for f, h, d, and w. We allowed h and d to vary in increments of 5 feet over a reasonable range. The road width w was tested for two lanes and four lanes, where each lane was 12 feet wide. For every combination of f, w, h, and d, we determined the valid values of ϕ and θ (each in increments of 0.5 degrees) for which the theoretical constraints of Equations (45) and (46) hold. The ideal range for each parameter is described graphically in figures 21and 22 for the case of two lanes, and in figures 23 and 24 for the case of four lanes. We limited the parameters to $\phi \in [0,30°]$ and $\theta \in [-30°,30°]$, and many of the curves reached these extrema.

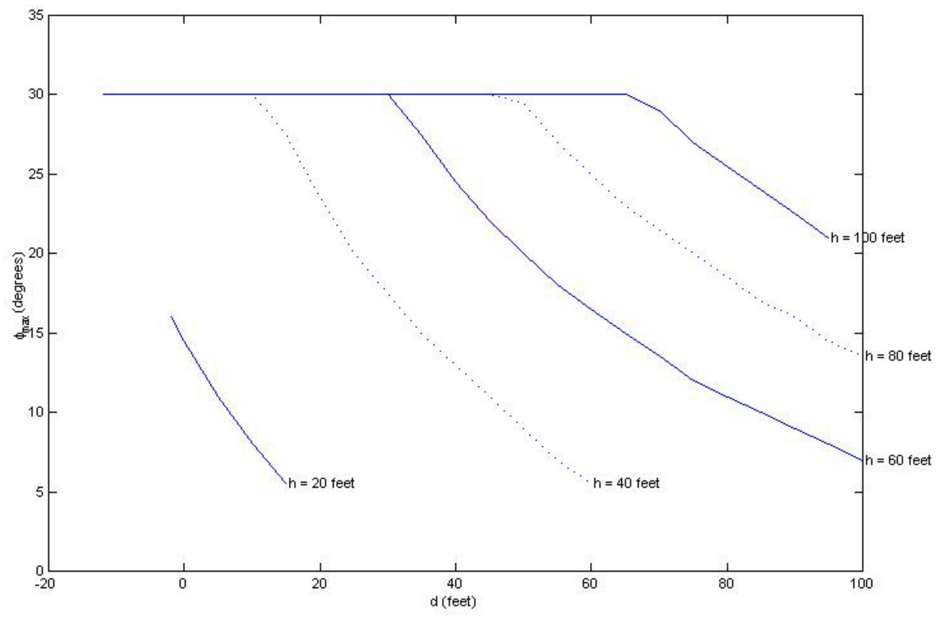
For each of these valid (ϕ, θ) pairs, we conducted a separate Monte-Carlo simulation of 1,000 runs and examined the relative variability in the measurements of $\Delta Y'$, the distance along the road. These measurements of $\Delta Y'$ were randomly selected throughout the lower one-third of the image. Gaussian noise was simultaneously added to all input parameters at an appropriate level, given the anticipated measurement accuracy of the system, i.e., σ_{u0} = 2 pixels, σ_{v0} = 2 pixels, σ_{b1} = 2 pixels, σ_{b2} = 2 pixels, σ_w = 0.5 feet, and $\sigma_d = 0.03d$ feet. Finally, on the basis of the Monte-Carlo simulation, we obtained maximum and minimum values for ϕ and θ where the relative length variability in the output was reasonable, i.e., $\sigma_{\Delta Y'}/\Delta Y' < 0.05$. Although this did not completely specify the shape of the valid region in the ϕ-θ plane for the given f, w, h, and d, it provided a rough idea of the utility of Method 1 under realistic conditions. Interestingly enough, we found that our calibration method produced curves under noisy conditions that were very close, if not equal to, the ideal ones in figures 21 through 24. Hence, we omit the plots of these results that are essentially identical to theory.

The simulation results in figures 21 through 24 show several trends that characterize Method 1. The curve families have roughly the same shape regardless of the number of lanes. The case of two lanes is somewhat different than the other cases in that the range of ϕ is noticeable restricted, even for greater values of h. However, for the four-lane case, we note that ϕ_{min} is approximately the same for all values of h. ϕ_{max} is approximately linear in d.

At a certain point, $\phi_{max} = \phi_{min}$, and there are no valid (ϕ,θ) pairs beyond this value of *d*, e.g., *d* must be less than 60 feet when $h = 40$ feet and $w = 24$ feet, as shown in Figure 21. This break-off phenomenon is also evident in the plots for θ_{min} and θ_{max} in Figure 22.

We note a range in the curves of θ_{min} in Figure 22, where θ_{min} is constant at zero. This is due to the constraint that the slope of the perpendicular lines exceed 0.15. This requires that θ must be at least 7° or more. This restriction will cause the down angle to be quite large to achieve this goal when *d* is near zero. Thus, we arrive at the same conclusion as our analysis of the range for *f*, i.e., the operator should use the largest down angle possible and adjust the pan angle to get the lines within an appropriate range in the image.

The maximum angle to which the operator can physically tilt the camera is about 20°, so we require that $\phi_{min} < 20°$. We can see from the ϕ_{min} curves in figures 21 and 23 that achieving a down angle of 20° is generally adequate to place the camera in an orientation where Method 1 can be used. The exceptions occur when $h > 90$ feet and the road is narrow, e.g., Figure 21. In this case, it becomes impossible to get sufficiently sloped perpendicular lines without increasing the focal length (camera zoom) or increasing the down angle further.



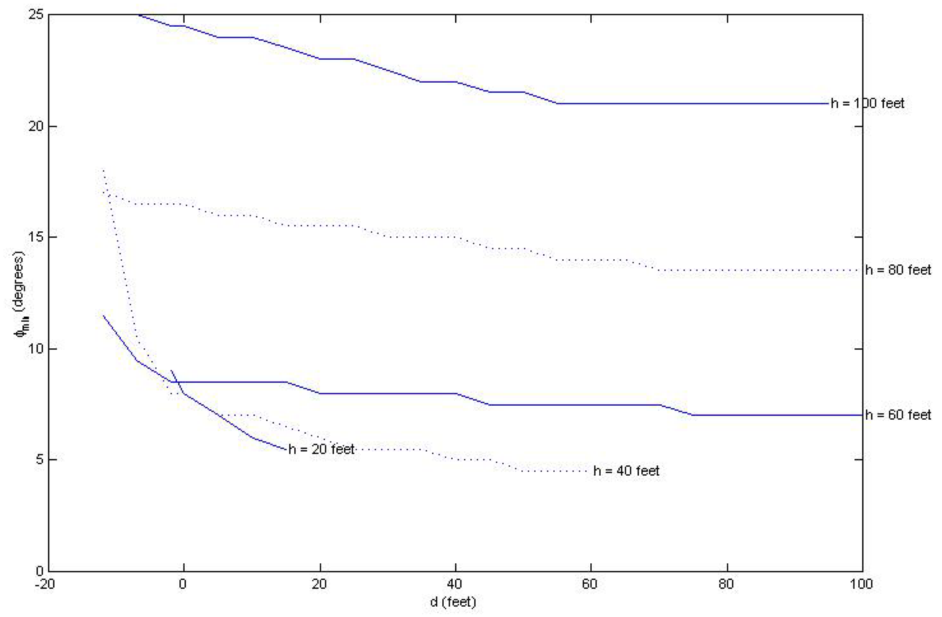
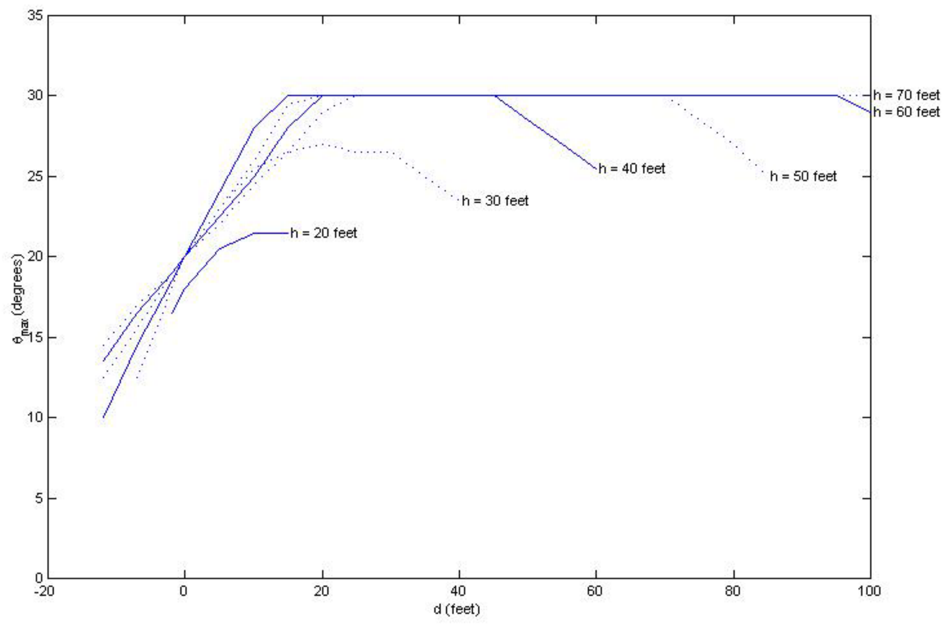


Figure 21. Theoretical range for the tilt angle ϕ for a two-lane road (*w* = 24 feet) when *f* = 1000 pixels to keep the road lines within the image and ensure sufficiently sloped perpendicular lines.



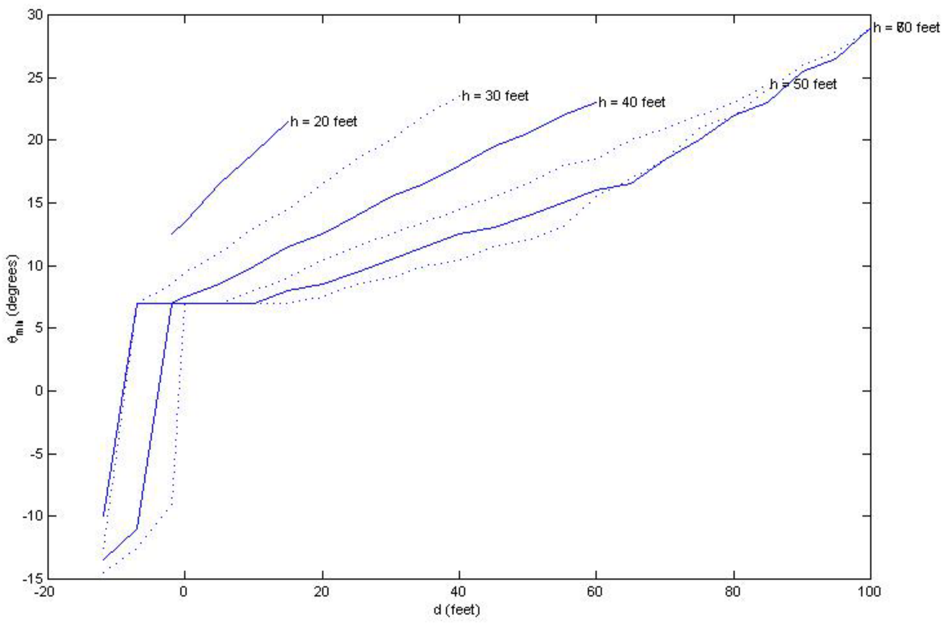
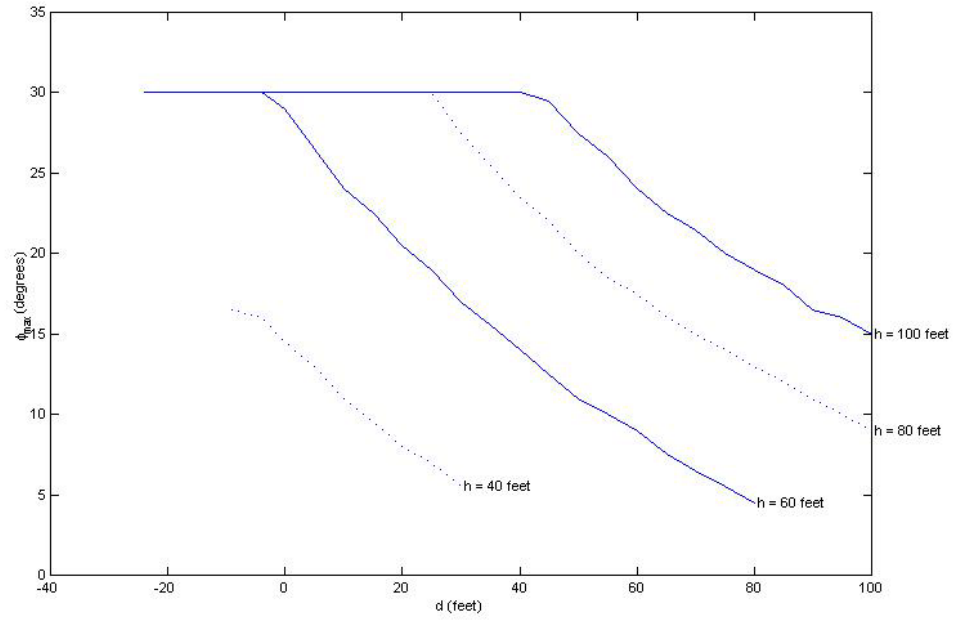


Figure 22. Theoretical range for the pan angle θ for a two-lane road (w = 24 feet) when f = 1000 pixels to keep the road lines within the image and ensure sufficiently sloped perpendicular lines.



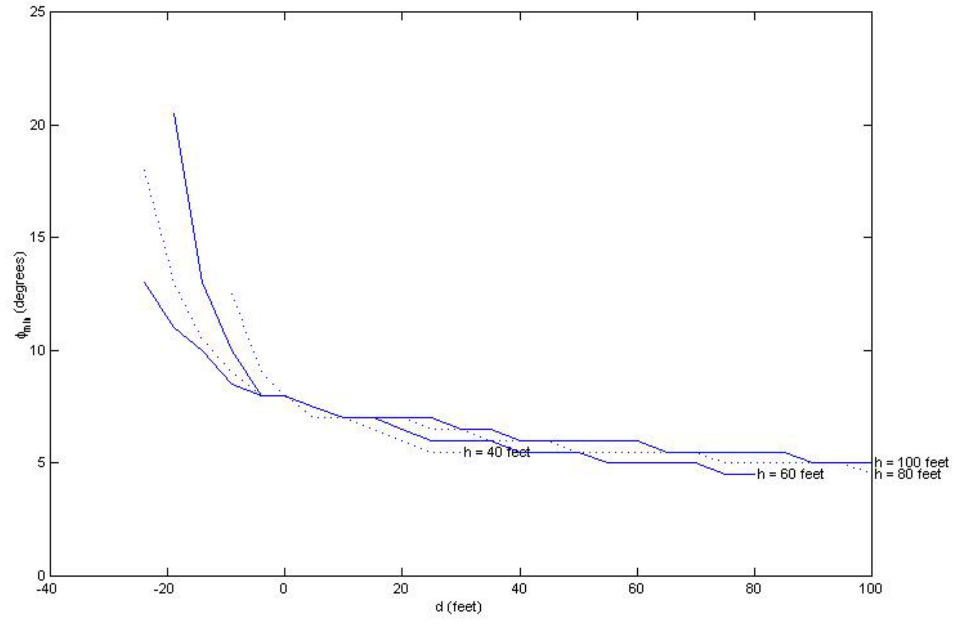
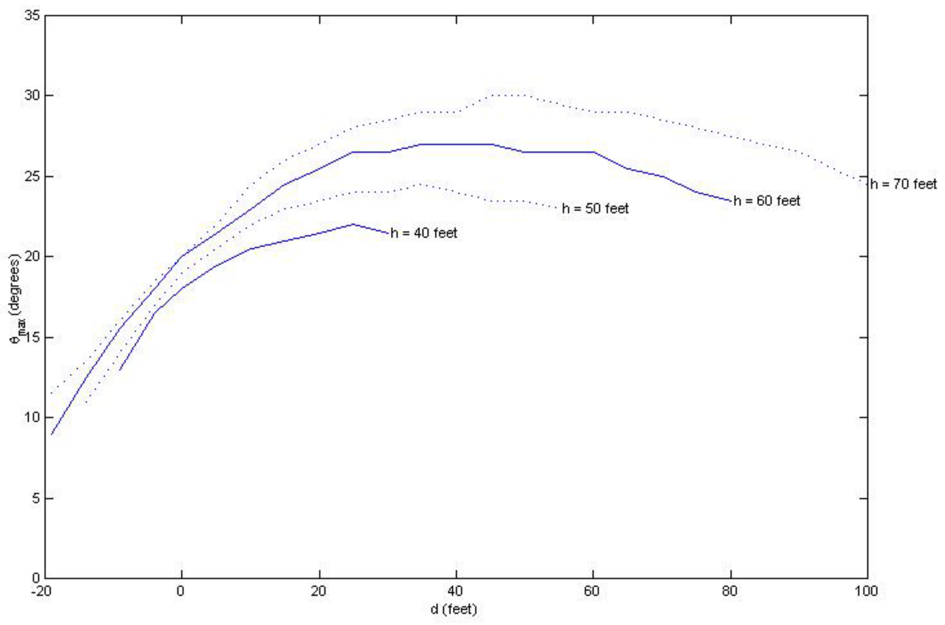


Figure 23. Theoretical range for the tilt angle ϕ for a four-lane road (*w* = 48 feet) when *f* = 1000 pixels to keep the road lines within the image and ensure sufficiently sloped perpendicular lines.



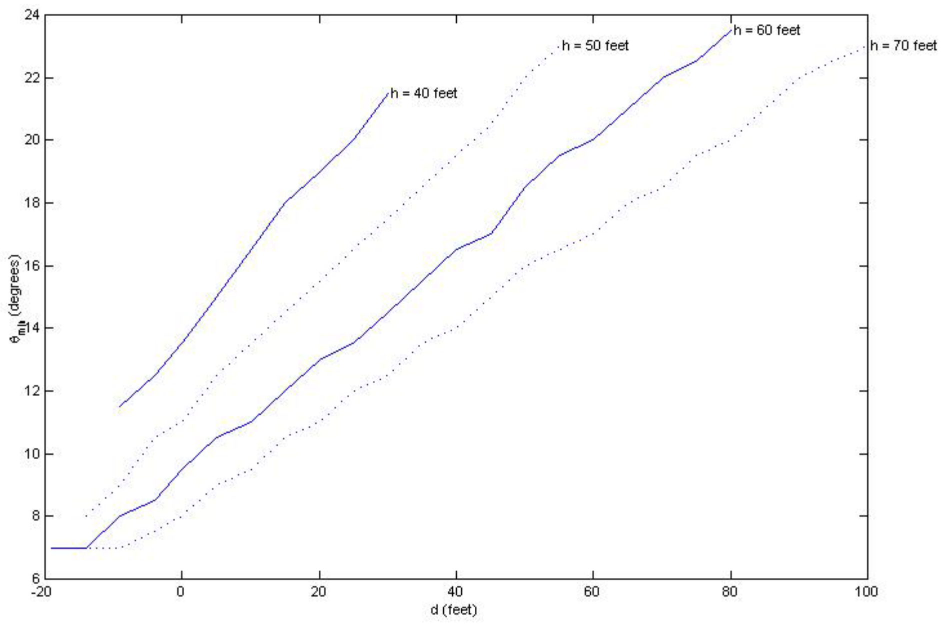


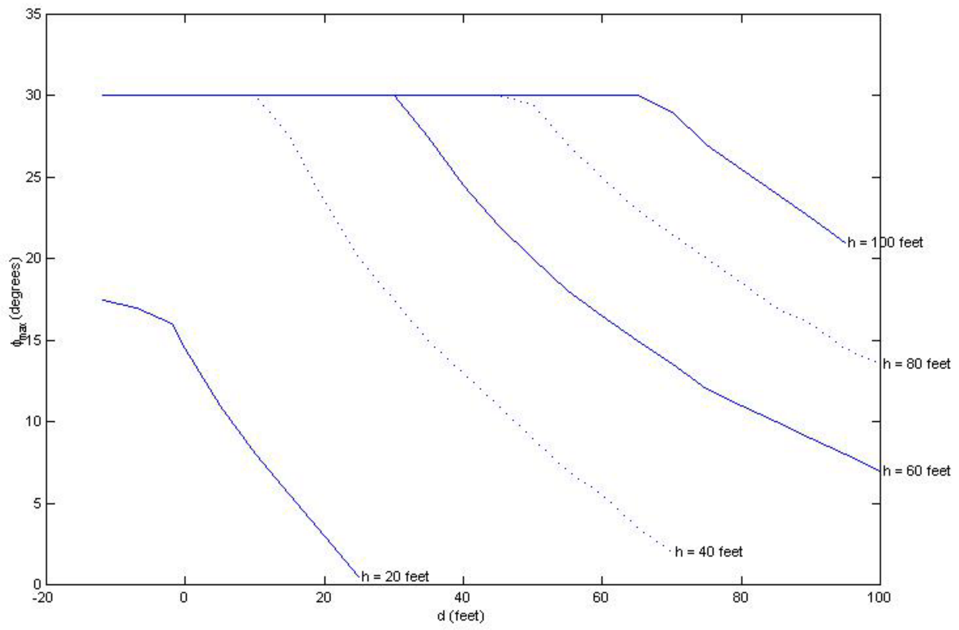
Figure 24. Theoretical range for the pan angle θ for a four-lane road (*w* = 48 feet) when *f* = 1000 pixels to keep the road lines within the image and ensure sufficiently sloped perpendicular lines.

2.4.2 Limits of operation for Method 2

As we found in Section 2.3 in the discussion of figures 11 and12, Method 2 is very sensitive to variations in its input parameters. We developed a Monte-Carlo simulation to model the behavior of this calibration method under noisy conditions across a wide variety of scene geometries and for a fixed focal length. Specifically, we allowed h and d to vary in increments of 5 feet over a reasonable range. The road width w was tested for the cases of two and four lanes, where each lane was 12 feet wide. The focal length f was fixed at 1,000 pixels, which is a lower bound anticipated for the real system. For every combination of f, w, h, and d, we determined the valid values of ϕ and θ (each in increments of 0.5 degrees) for which the constraints of Equation (45) held. These best-case theoretical limits are described graphically in figures 25 and 26 for the case of two lanes and in figures 27 and 28 for the case of four lanes. We limited the parameters to $\phi \in [0,30°]$ and $\theta \in [-30°,30°]$, and some of the curves reached these extrema.

For each of the (ϕ, θ) pairs satisfying the constraints of Equation (45), we performed 1,000 runs using noisy inputs and examined the relative variability in the length measurements. These measurements were randomly selected throughout the lower one-third of the image. Gaussian noise was simultaneously added to all input parameters at an appropriate level, given the anticipated measurement accuracy of the system, i.e., σ_{u0} = 2 pixels, σ_{v0} = 2 pixels, σ_{b1} = 2 pixels, σ_{b2} = 2 pixels, σ_w = 0.5 feet, and $\sigma_d = 0.03d$ feet. These values assumed that we were 95 percent confident that the vanishing point could be located in an 8 X 8 pixel block, that b_1 and b_2 had an error of two pixels or less, that the road width w was within 1 foot of its true value, and that our estimate of d was within 6 percent of its true value. Finally, on the basis of the Monte-Carlo simulation, we obtained maximum and minimum values for ϕ and θ, where the relative length variability was reasonable, i.e., $\sigma_{\Delta Y'}/\Delta Y' < 0.05$. Although this did not completely specify the shape of the valid region in the ϕ-θ plane for the given f, w, h, and d, it provided a rough idea of the utility of Method 2 under realistic conditions.

The simulation results in figures 29 through 32 reveal several trends that characterize Method 2 under the realistic error conditions in comparison to the theoretical results of figures 25 through 28. First of all, the minimum down angles ϕ_{min} from the noisy data in figures 25 and 27 are much larger than the values from theory in figures 29 and 31. This is particularly true as the camera moves farther from the left road boundary (i.e., d increases). The oscillatory behavior of some of the curves is due to certain areas in the ϕ-θ plane being cut off, depending on the outcome of the particular experiment of 1,000 runs. Secondly, comparing the curves for θ_{min} from the noisy inputs in figures 30 and 32 with the theoretical curves in figures 26 and 28 shows that the curves from the noisy inputs end prematurely, i.e., input noise reduces the range for which Method 2 is applicable. For example, in Figure 30, the curve for h = 40 feet ends at d = 30 feet, whereas the theoretical curve doesn't end until d = 70 feet. Thus, it is conceivable that under certain real-world conditions, some cameras might be located too far from the road or that the camera height might be too small for us to be able to use Method 2. Thirdly, we note from the ϕ_{min} curves in Figure 31 that the limits for the wider road are noticeably closer to the theoretical value than those in Figure 29. The values of ϕ_{min} for Method 2 for the narrow road in Figure 29 are much too high for day-to-day use where the camera needs to view the far-field, i.e., ϕ is less than about 10°. Lastly, the θ_{min} curves in Figure 32 generally do not reach below 0°, with a few exceptions. It is quite possible that the operator could orient the camera with $\theta < 0°$ and then Method 2 would not be able to calibrate the camera under real-world conditions. In sum, there are noticeable differences between the theoretical limits for ϕ and θ and their limits under realistic conditions. Method 2 is probably not appropriate for two-lane roads when real-world levels of noise are present, but it could be used in fairly a limited fashion, e.g., for four-lane roads when $50 < h < 70$ feet.



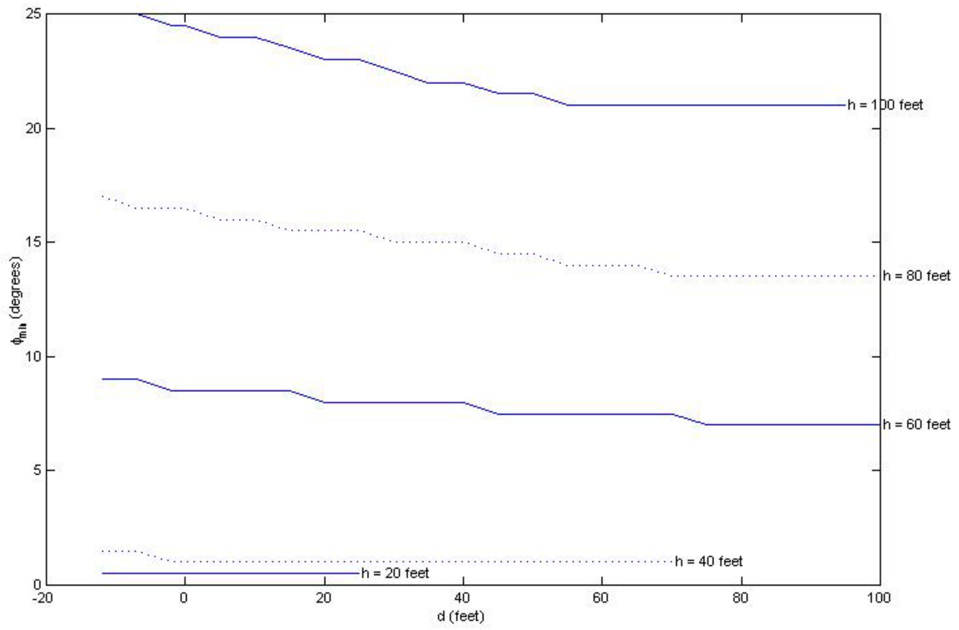
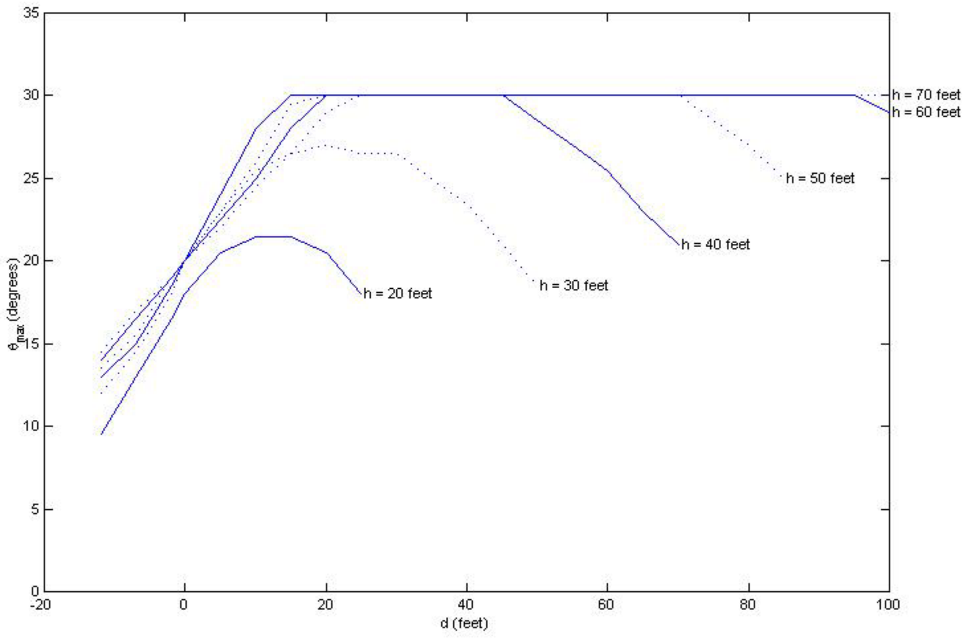


Figure 25. Theoretical range for the tilt angle ϕ for a two-lane road (*w* = 24 feet) when *f* = 1000 pixels to keep the road lines within the image.



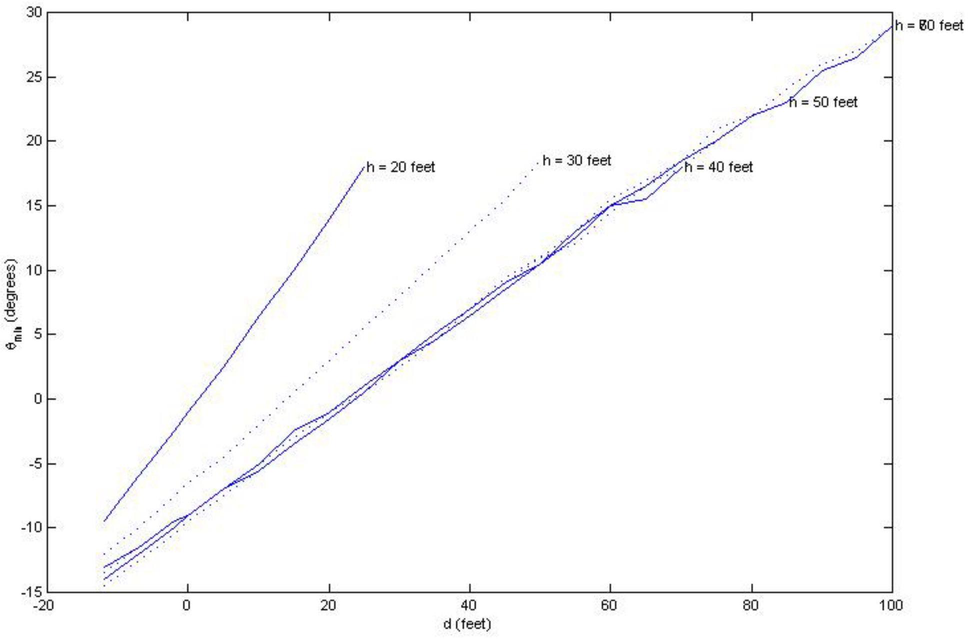
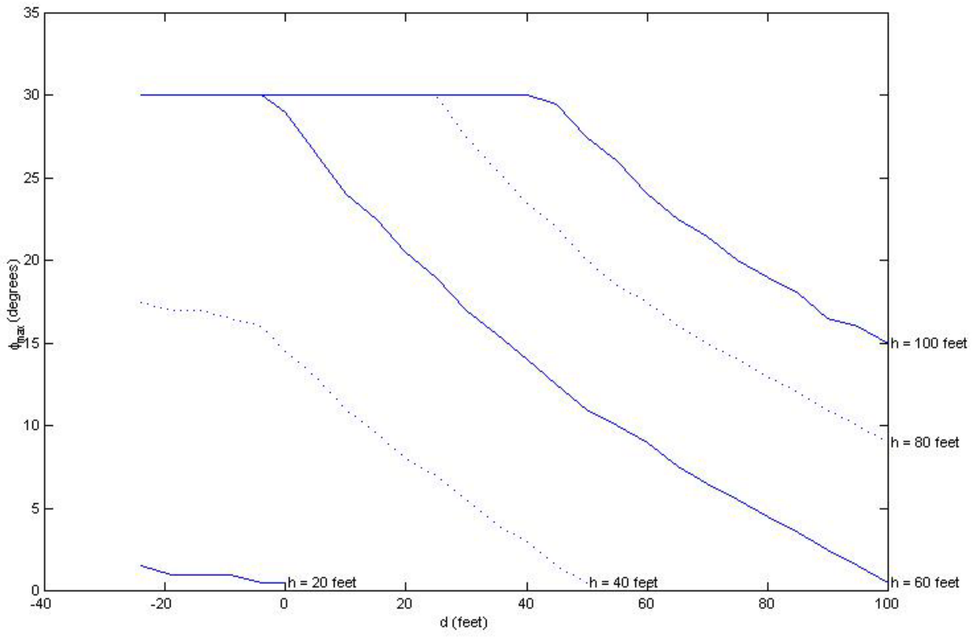


Figure 26. Theoretical range for the pan angle θ for a two-lane road (w = 24 feet) when f = 1000 pixels to keep the road lines within the image.



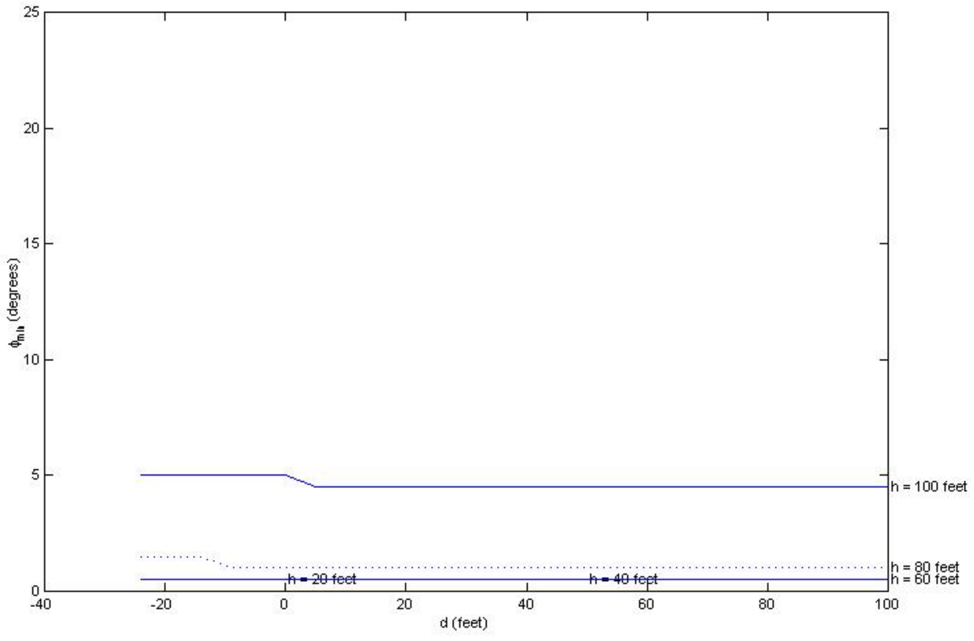
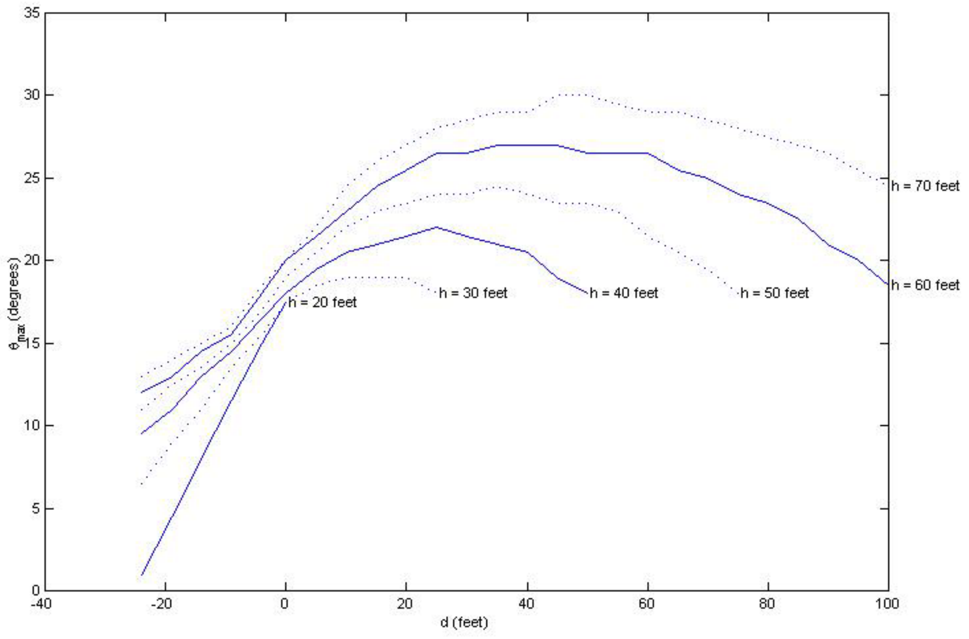


Figure 27. Theoretical range for the tilt angle ϕ for a four-lane road (w = 48 feet) when f = 1000 pixels to keep the road lines within the image.



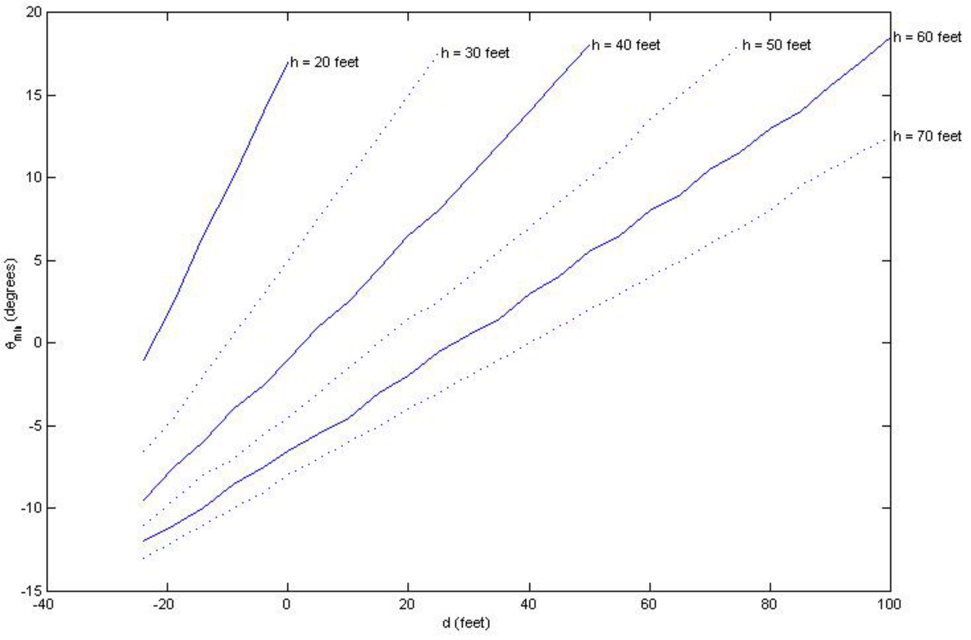
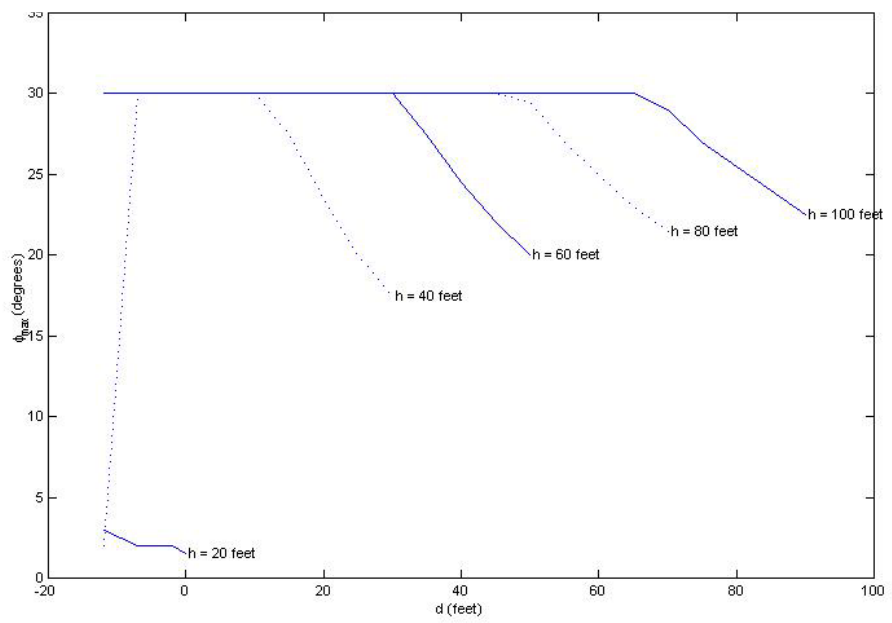


Figure 28. Theoretical range for the pan angle θ for a four-lane road (w = 48 feet) when f = 1000 pixels to keep the road lines within the image.



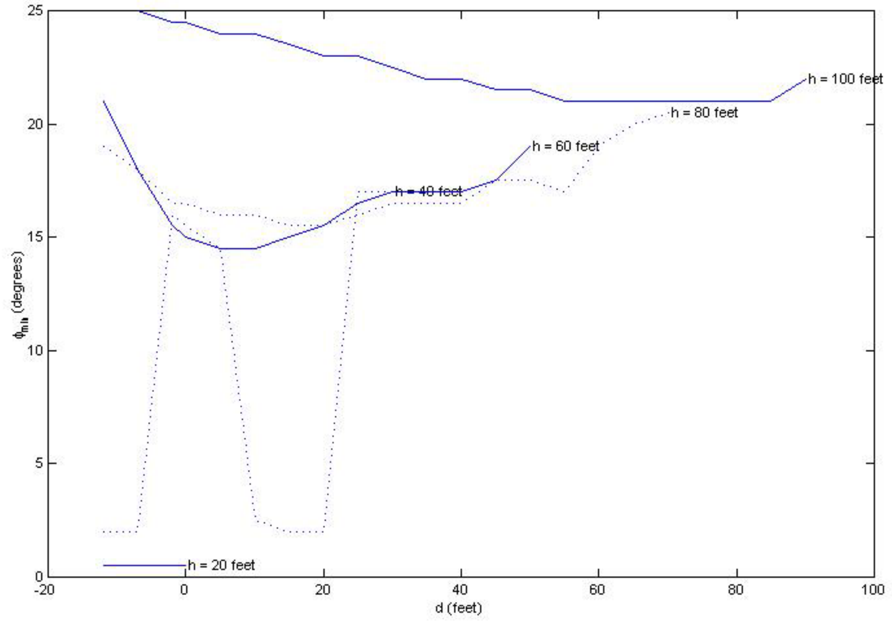
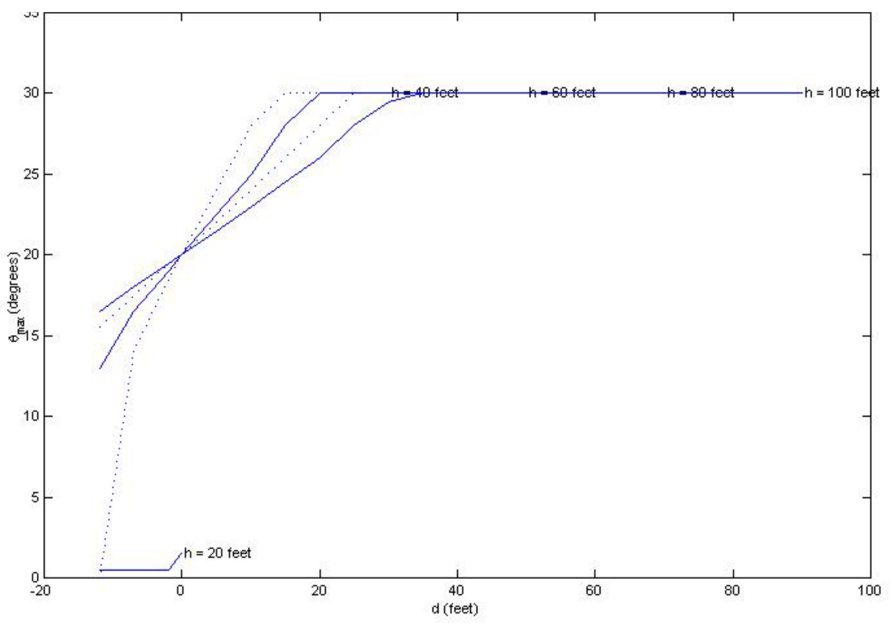


Figure 29. Valid range for the tilt angle ϕ for a two-lane road (*w* = 24 feet) when using Method 2 to calibrate the camera when *f* = 1000 pixels and to maintain $\sigma_{\Delta Y'}/\Delta Y' < 0.05$.



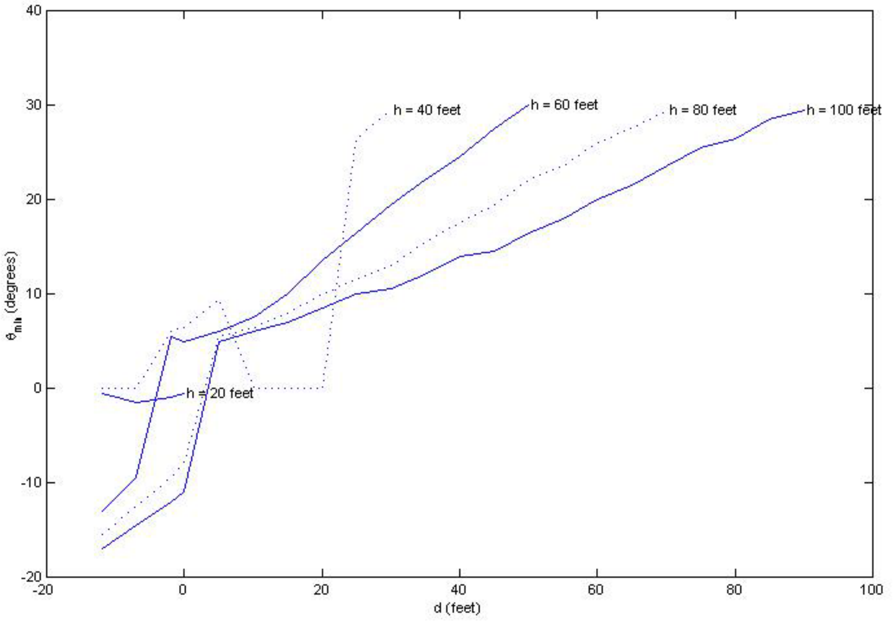
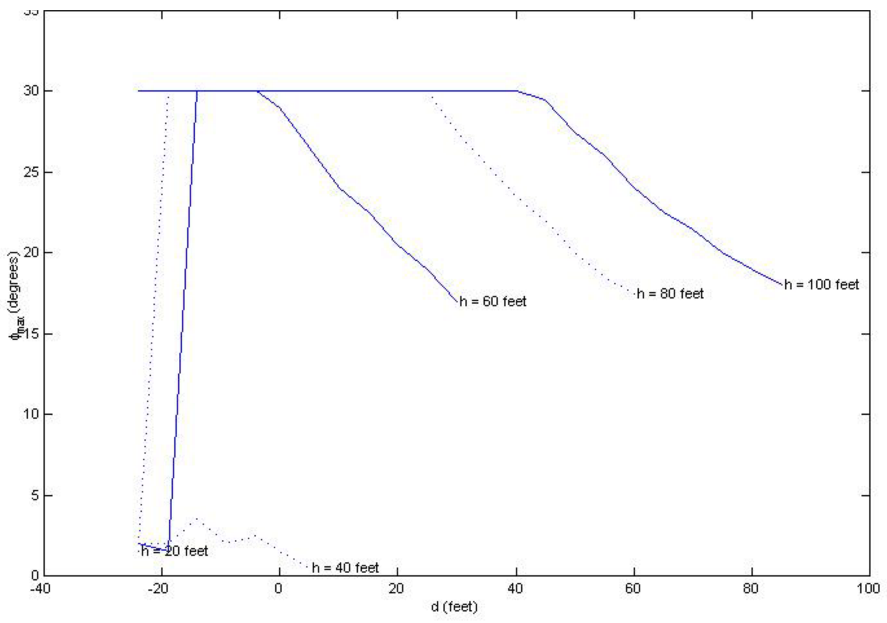


Figure 30. Valid range for the pan angle θ for a two-lane road (*w* = 24 feet) when using Method 2 to calibrate the camera when *f* = 1000 pixels and to maintain $\sigma_{\Delta Y'}/\Delta Y' < 0.05$.



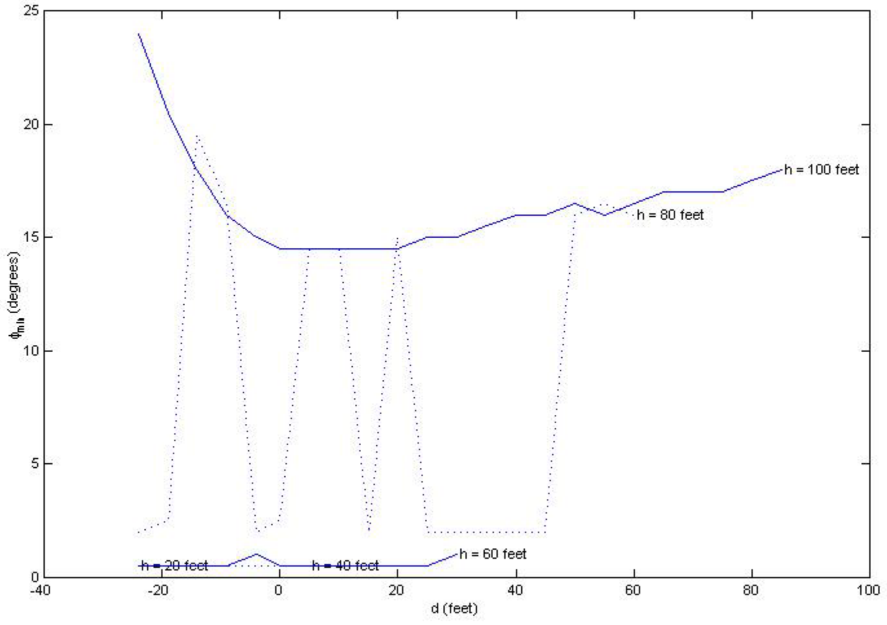
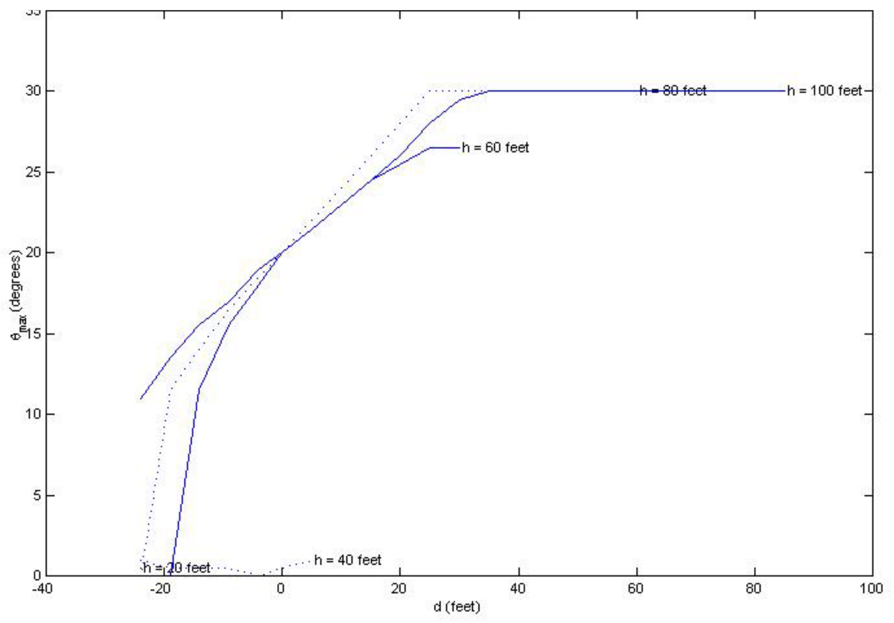


Figure 31. Valid range for the tilt angle ϕ for a four-lane road (*w* = 48 feet) when using Method 2 to calibrate the camera when *f* = 1000 pixels and to maintain $\sigma_{\Delta Y'}/\Delta Y' < 0.05$.



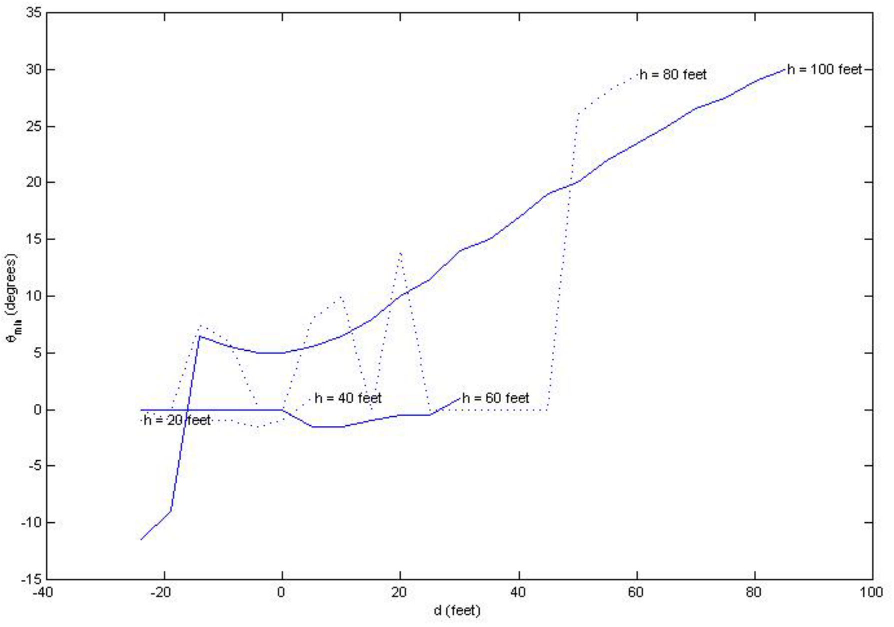


Figure 32. Valid range for the pan angle θ for a four-lane road (*w* = 48 feet) when using Method 2 to calibrate the camera when *f* = 1000 pixels and to maintain $\sigma_{\Delta Y'}/\Delta Y' < 0.05$.

2.4.3 Limits of operation for Method 3

In determining the limits of operation for Method 3, we recall that the calibration for the distance along the road $\Delta Y'$ is only affected by knowledge or measurements of L, v_0, and τ_L. The method is therefore independent of the road geometry, i.e, w, so we anticipate that we may apply it to a wide variety of situations. Despite the simplicity of the formulation for Method 3, it is difficult to devise a realistic Monte-Carlo simulation for it. As Chapter 3 will show, the computer can estimate τ_L very accurately ($\sigma_{\tau L}/\tau_L \leq 0.005$) under noisy image conditions. Chapter 3 also demonstrates that the measurement process for τ_L in the image actually mitigates the effects of the variation in L, reducing the output variability by 50 percent or more relative to the input variability of L. Thus, even though σ_L/L may reach 0.1, the relative variability in the distance measurements in the output, i.e., $\sigma_{\Delta Y'}/\Delta Y'$, is about 0.05 or even smaller, regardless of the camera angle. Thus, the ranges for ϕ and θ are identical to the theoretical ranges in figures 25 through 28. However, the actual range for which we may apply Method 3 to speed estimation exceeds these limits because Method 3 does not enforce any constraints on the road boundary lines. Instead, it merely requires that at least two consecutive road markers be visible in the scene.

2.5 Limitations of the reduced parameter camera model

As noted above in Section 2.1, we assumed that the road slope angle $\psi = 0$ and the tilt angle $\beta = 0$. However, as illustrated by figures 3 and 4, we can easily imagine that this is not necessarily the case for most scenes. Thus, we study the effects of nonzero road slope or road tilt on the three proposed methods of camera calibration that were originally derived under the assumptions that $\psi = 0$ and $\beta = 0$. Parenthetically, we note that studying the effect of camera roll is unnecessary because it can be completely modeled using the two rotations of the road.

We designed a computer simulation for the three scenes whose parameters are contained in Table 2. We created vectors of points in the X_c'-Y_c'-Z_c' coordinate system of Figure 2. We then applied rotations in the following order to fully simulate the effects of road slope and road tilt: β, ψ, θ, and ϕ. After we projected the points into the image according to Equation (1), we obtained measurements for u_0, v_0, b_1, b_2, and u_1. We also measured τ_L by averaging the values of τ for five pairs of points known to be separated by 40 feet in the bottom half of the image (scenes 1 and 3) or the full image (Scene 2). All of these measurements were performed with double floating-point accuracy, i.e., they were perfect for all practical purposes. Because the distance $\Delta Y'$ measured along the road using the reduced-order models will vary with the position of the points in the image, we projected 1000 pairs of randomly generated points, each separated by 10 feet, into the lower half of the image. We then calculated the average distortion for all the pairs of points to obtain the average bias in $\Delta Y'$ for a given level of road tilt or road slope.

We performed 12 different simulations as outlined in Table 6; six examined the effect of nonzero road slope and six examined nonzero road tilt. We only analyzed scenes for which the calibration method was appropriate. Table 6 also contains a list of the output parameters that were calculated for each simulation. The table below summarizes our general findings for the effects of nonzero road slope and tilt on the different methods.

Table 6. Matrix of calibration methods, scenes, and output parameters analyzed for the effects of nonzero road tilt and road slope.

	Method 1	Method 2	Method 3
Output parameters	d $f, \phi, \theta, \Delta Y'$	$f, \phi, \theta, \Delta Y'$	d $f, \phi, \theta, \Delta Y'$
Scenes analyzed	2	1,3	1,2,3
Effect of nonzero road slope on d	Moderate	n/a	Moderate
Effect of nonzero road slope on $\Delta Y'$	Moderate	Moderate to High	Low
Effect of nonzero road tilt on d	Moderate	n/a	Moderate
Effect of nonzero road tilt on $\Delta Y'$	High	High	Low

2.5.1 Effects of nonzero road slope

Figures 33 through 38 contain the curves indicating the relative errors in f, ϕ, θ, d, and $\Delta Y'$ due to the effect of nonzero road slope. We only studied Scene 2 for Method 1 because a real-world system would probably only use Method 1 on close-up scenes to estimate d as an input to Method 2. In Figure 33, we note that the relative error in d due to road slope does not exceed 5 percent, even though the errors in the other parameters are much larger, relatively speaking. However, we recall from Section 2.3 that Method 2 is very sensitive to errors in d of even 1 percent. In fact, from Figure 11 we can see that calibrating Scene 3 using Method 2 is impossible with even small errors in d. Method 2 is less sensitive to errors in d when Scene 1 is calibrated. At the very least, we must conclude that a small amount of road slope could noticeably constrain our error budget for applying the combination of methods 1 and 2.

As far as the direct effects of nonzero road slope on Method 2 are concerned, we note from figures 34 and 35 that the relative errors in $\Delta Y'$ track the relative error in f exactly. This error remains below 5 percent for $\psi \in [-1.6°,2°]$ and below 10 percent for $\psi \in [-3°,4.2°]$. These are road slopes of 2.8 percent at the 1.6° level and 7.3 percent at the 4.2° level. Thus, if the bias due to other input parameters was fairly low, the error budget of Method 2 could withstand a reasonable road slope. However, this is unlikely, given its sensitivity to errors in d. Again, we see that road slope could sabotage our efforts to use Method 2 to calibrate a scene.

Figures 36, 37, and 38 contain the results for simulating the effects of nonzero road slope on calibration Method 3. From Figure 37, we see that Method 3 provides approximately the same or slightly worse level of accuracy as Method 1 for estimating d. On the basis of the previous discussion, we would therefore not recommend Method 3 as a means of estimating d for use by Method 2 in the presence of nonzero road slope.

The errors due to nonzero road slope in the remaining parameters f, ϕ, θ, and $\Delta Y'$ are also below 4 percent, and often lower in some of the scenes. We note that because we only obtained five different estimates of τ_L sampled randomly throughout the appropriate portion of the image, the relative error in $\Delta Y'$ has more random character to it (due to the sampling process of τ_L) than it does a pattern. The plots of the error in $\Delta Y'$ show that it is unbiased by ψ with greater noise amplification for larger magnitudes of ψ. We note that in all three scenes, the relative error in $\Delta Y'$ due to road slope is essentially negligible, i.e., less than 0.5 percent. On the other hand, we note that the relative error in ϕ is quite large, and we can see that the relationship between v_0, ϕ, and f in Equation (20) must not hold when the road slope is nonzero, since the relative error in f is small and we measured v_0 exactly. It is fortuitous that the output parameter that matters most for speed estimation, i.e., $\Delta Y'$, is relatively unaffected by road slope. We conclude that in the presence of nonzero road slope without regard for other factors, Method 3 is clearly preferable to the other methods for the purposes of measuring road distances, though it is very slightly inferior to Method 2 when estimating d.

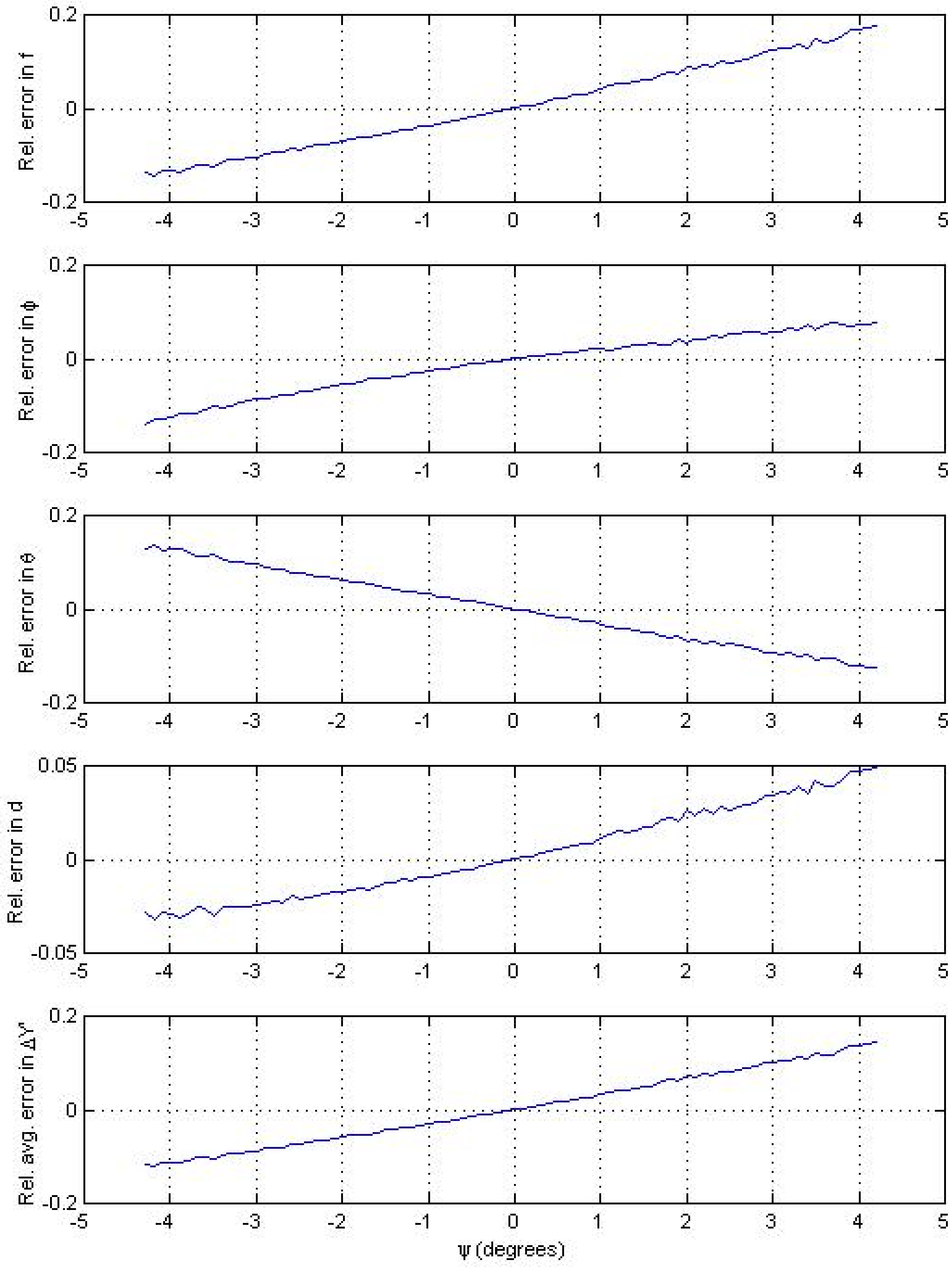


Figure 33. Effects of nonzero road slope angle ψ on camera calibration Method 1 and its output parameters when processing Scene 2.

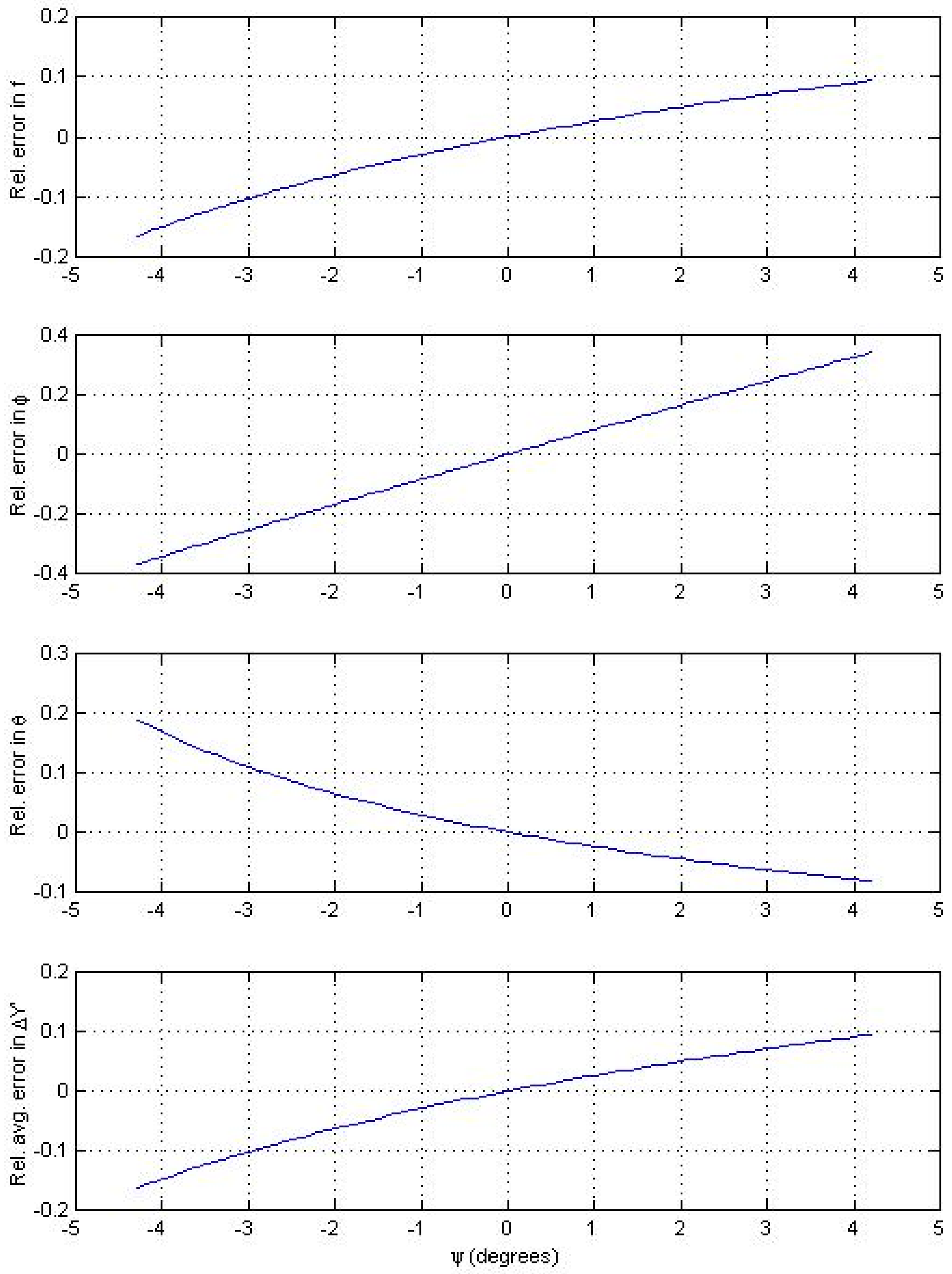


Figure 34. Effects of nonzero road slope angle ψ on camera calibration Method 2 and its output parameters when processing Scene 1.

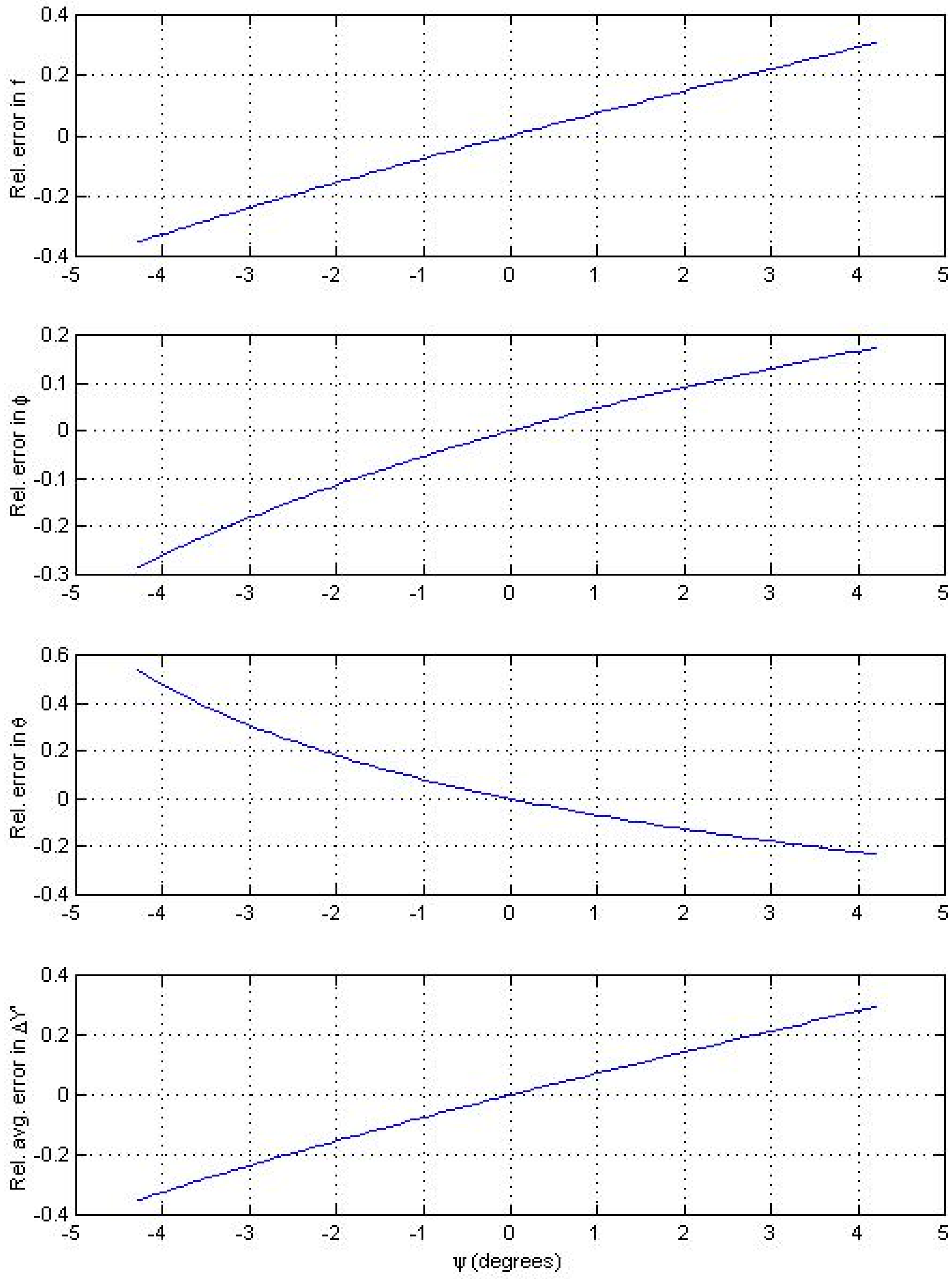


Figure 35. Effects of nonzero road slope angle ψ on camera calibration Method 2 and its output parameters when processing Scene 3.

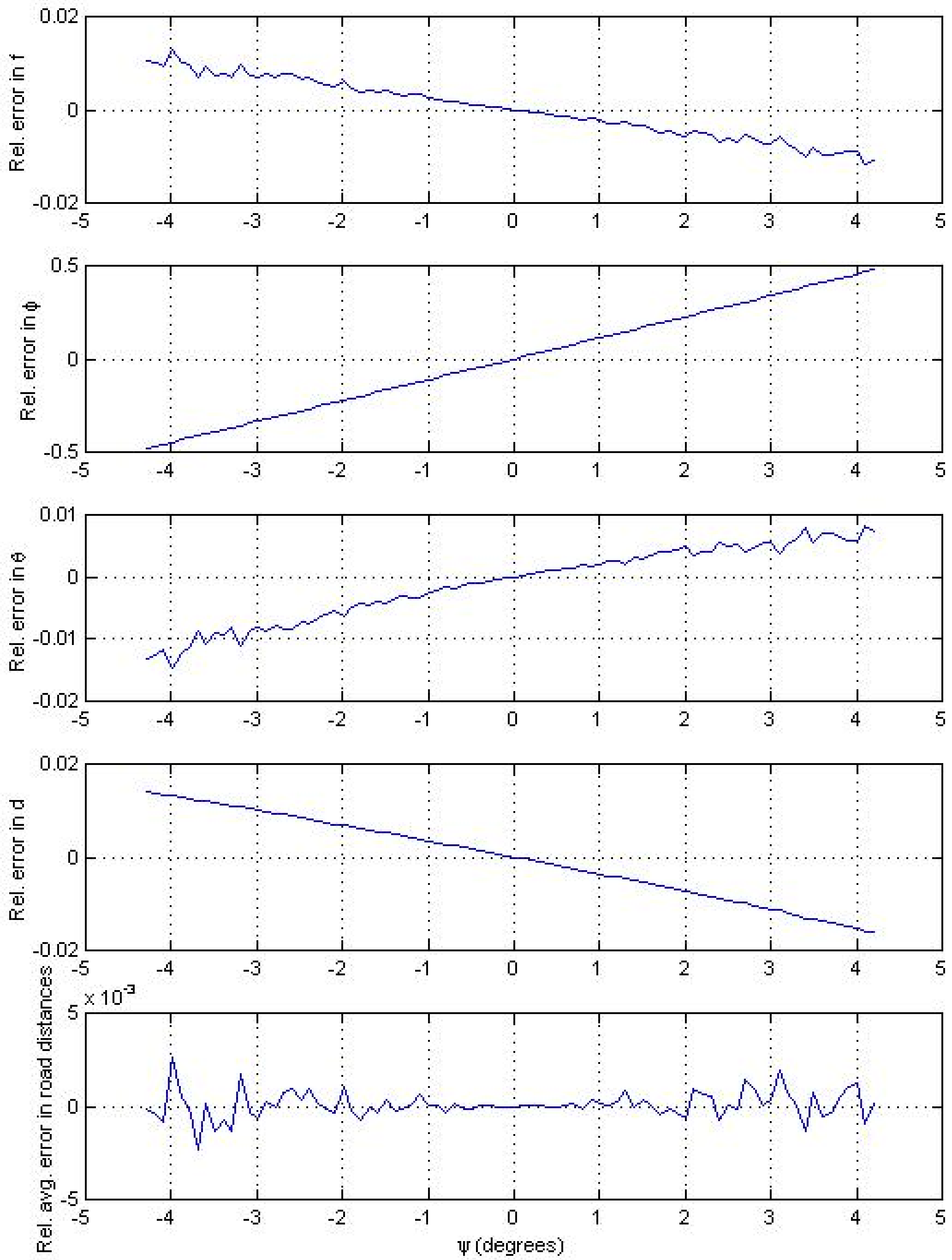


Figure 36. Effects of nonzero road slope angle ψ on camera calibration Method 3 and its output parameters when processing Scene 1.

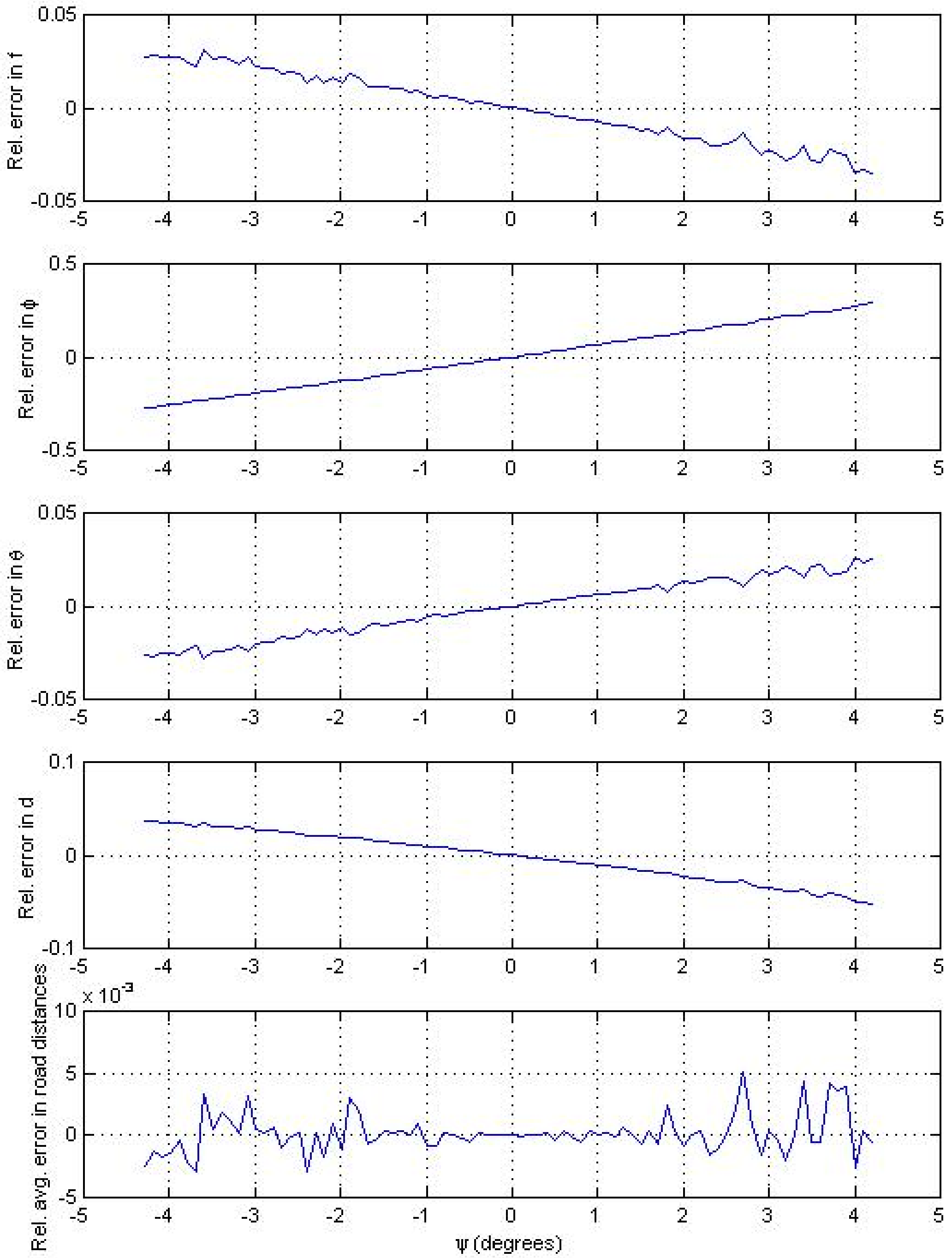


Figure 37. Effects of nonzero road slope angle ψ on camera calibration Method 3 and its output parameters when processing Scene 2.

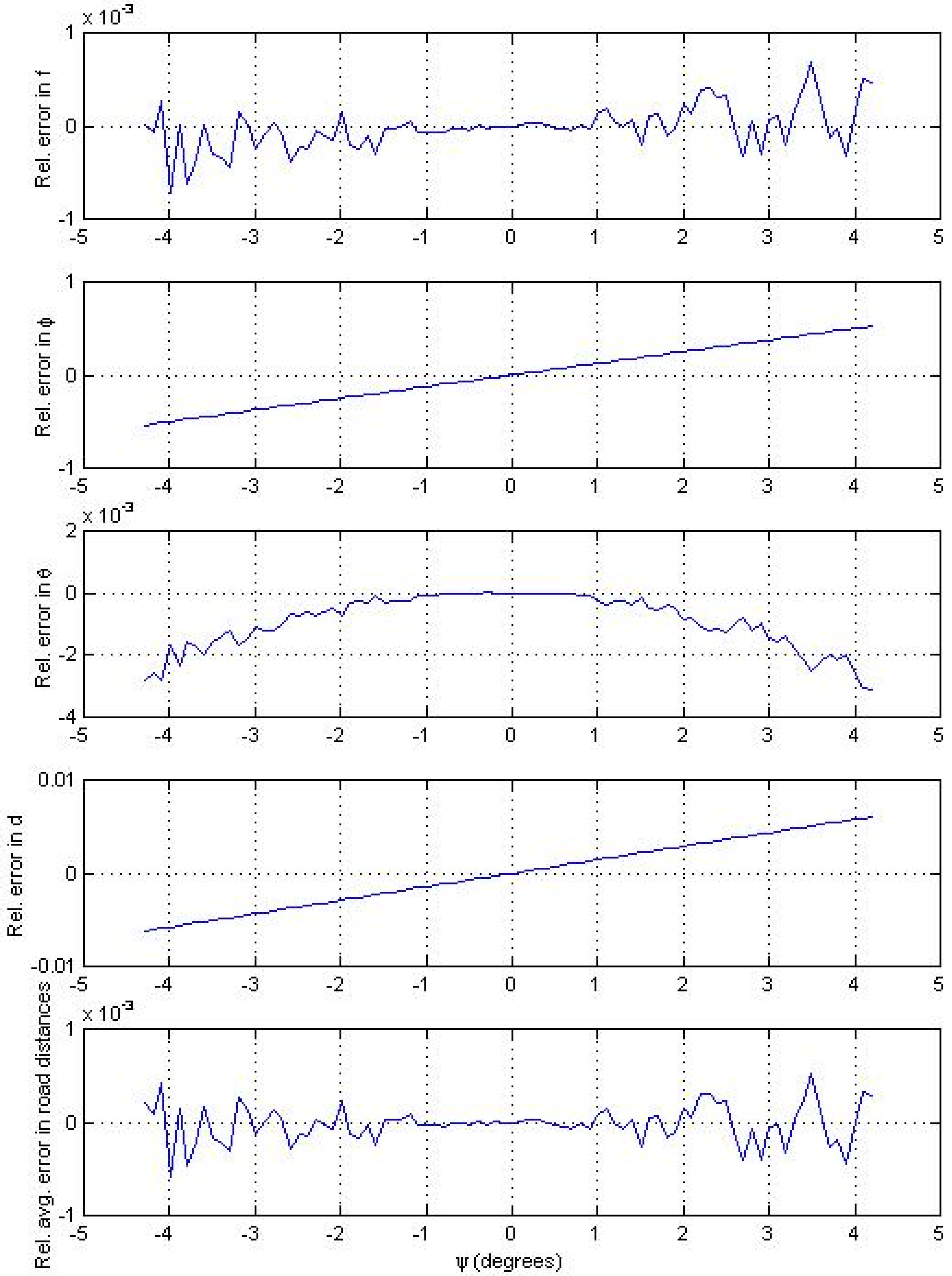


Figure 38. Effects of nonzero road slope angle ψ on camera calibration Method 3 and its output parameters when processing Scene 3.

2.5.2 Effects of nonzero road tilt

We also examined the effects of the road tilt angle β on the different calibration methods. Starting with Method 1, Figure 39 shows that a nonzero road tilt has about twice as much of an effect on estimates of *d* as does a nonzero road slope, i.e., less than 10 percent for $|\beta| < 4°$. Although we expect the road tilt to be less than the road slope, the same arguments from the previous section apply again to methods 1 and 2. Thus, we advise caution when applying the combination of methods 1 and 2 in the presence of nonzero road tilt because of the sensitivity of Method 2 to variations in *d* when it is estimated by Method 1.

Method 2 is much more heavily affected by nonzero road tilt than Method 1. Maintaining a relative error for $\Delta Y'$ of less than 5 percent requires a range of less than ±0.3° for Scene 1, and ±0.03° for Scene 3, which are simply intolerably small values. For side shots like Scene 1, we could imagine the possibility of using Method 2, but we can see that the least amount of road tilt in straight-on shots like Scene 3 will get magnified into large errors in $\Delta Y'$. Thus, in order to use Method 2 to accurately calibrate the camera, we must ensure that the road is not tilted.

Coming to Method 3, we see from figures 42, 43, and 44 that the errors in $\Delta Y'$ are pleasingly small, with a maximum relative error of about 1 percent for even the most extreme road tilt values. As in the case for nonzero road slope, Method 1 is slightly more accurate in estimating an unbiased value for *d* in Scene 2 when the road tilt is nonzero. Given the sensitivity of Method 2 to errors in *d*, we would therefore not recommend Method 3 as a means to estimate *d* in the presence of nonzero road tilt. We also note that the output parameters *f*, φ, and θ for Method 3 are approximately equally affected by nonzero road tilt, unlike the case of nonzero road slope, in which the bias was primarily bound up in φ. As in the case of nonzero road slope, we also conclude that Method 3 is the only realistic option for calibrating the camera in the presence of non-negligible amounts of road tilt.

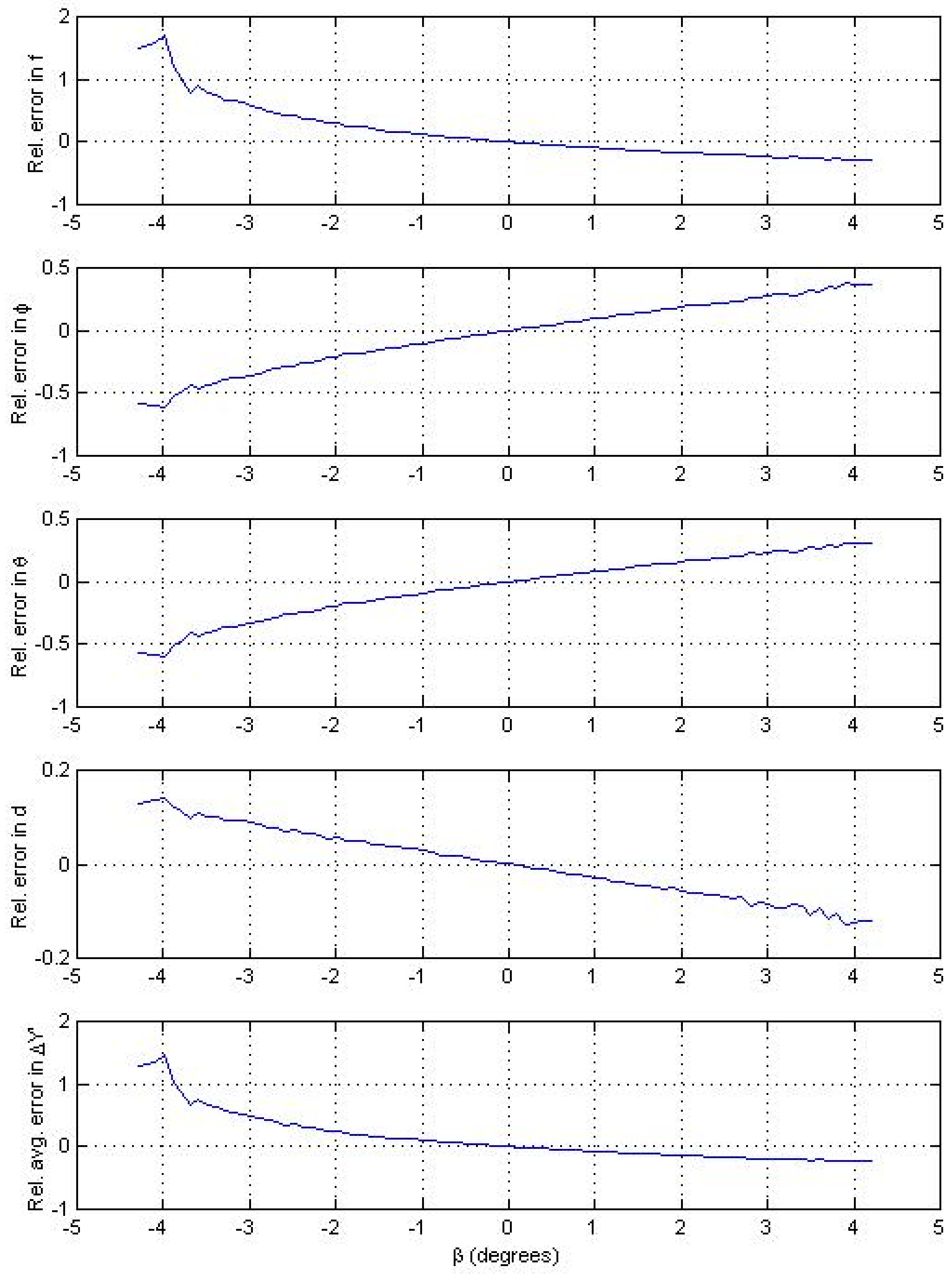


Figure 39. Effects of nonzero road slope angle β on camera calibration Method 1 and its output parameters when processing Scene 2.

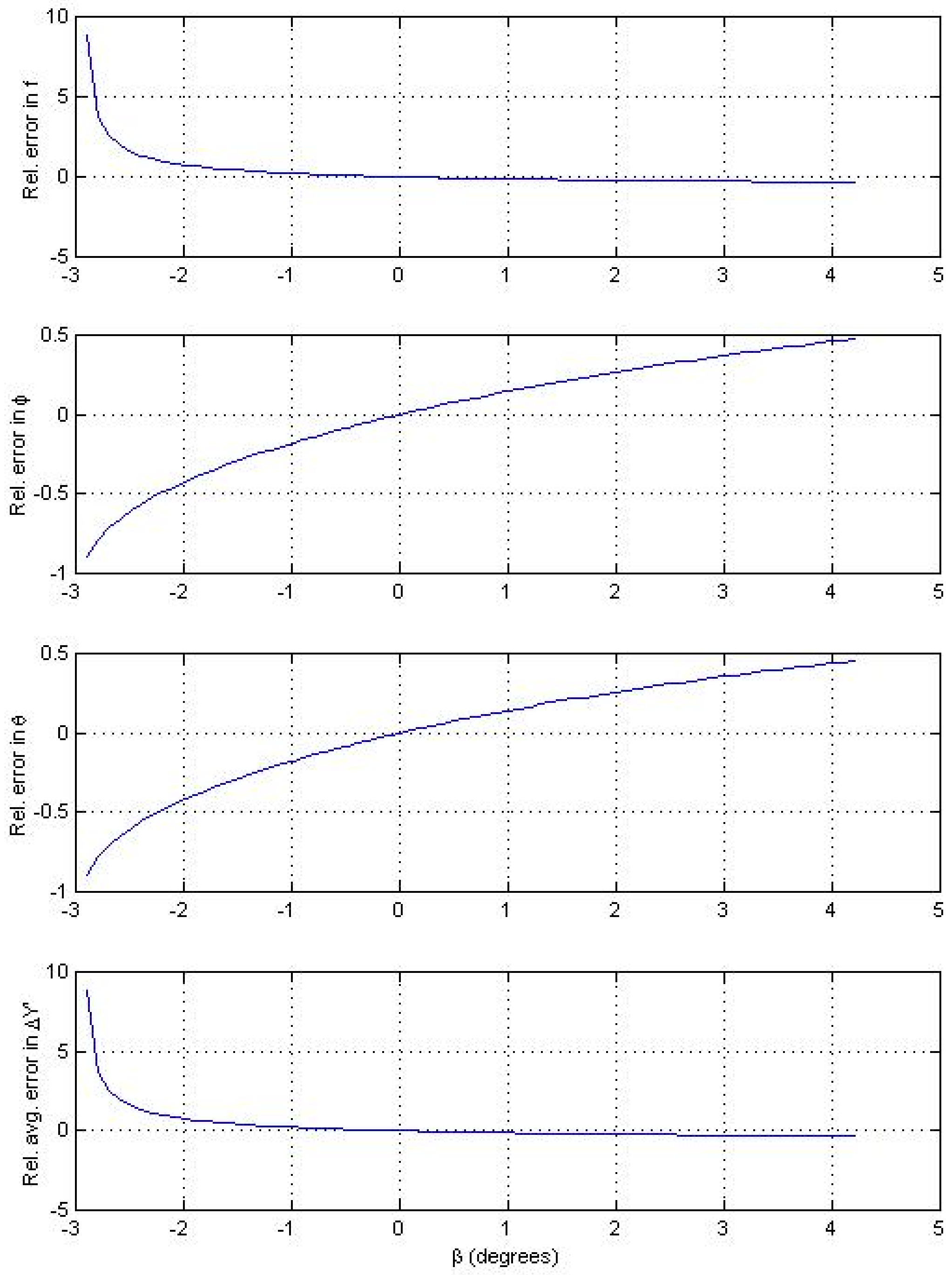


Figure 40. Effects of nonzero road slope angle β on camera calibration Method 2 and its output parameters when processing Scene 1.

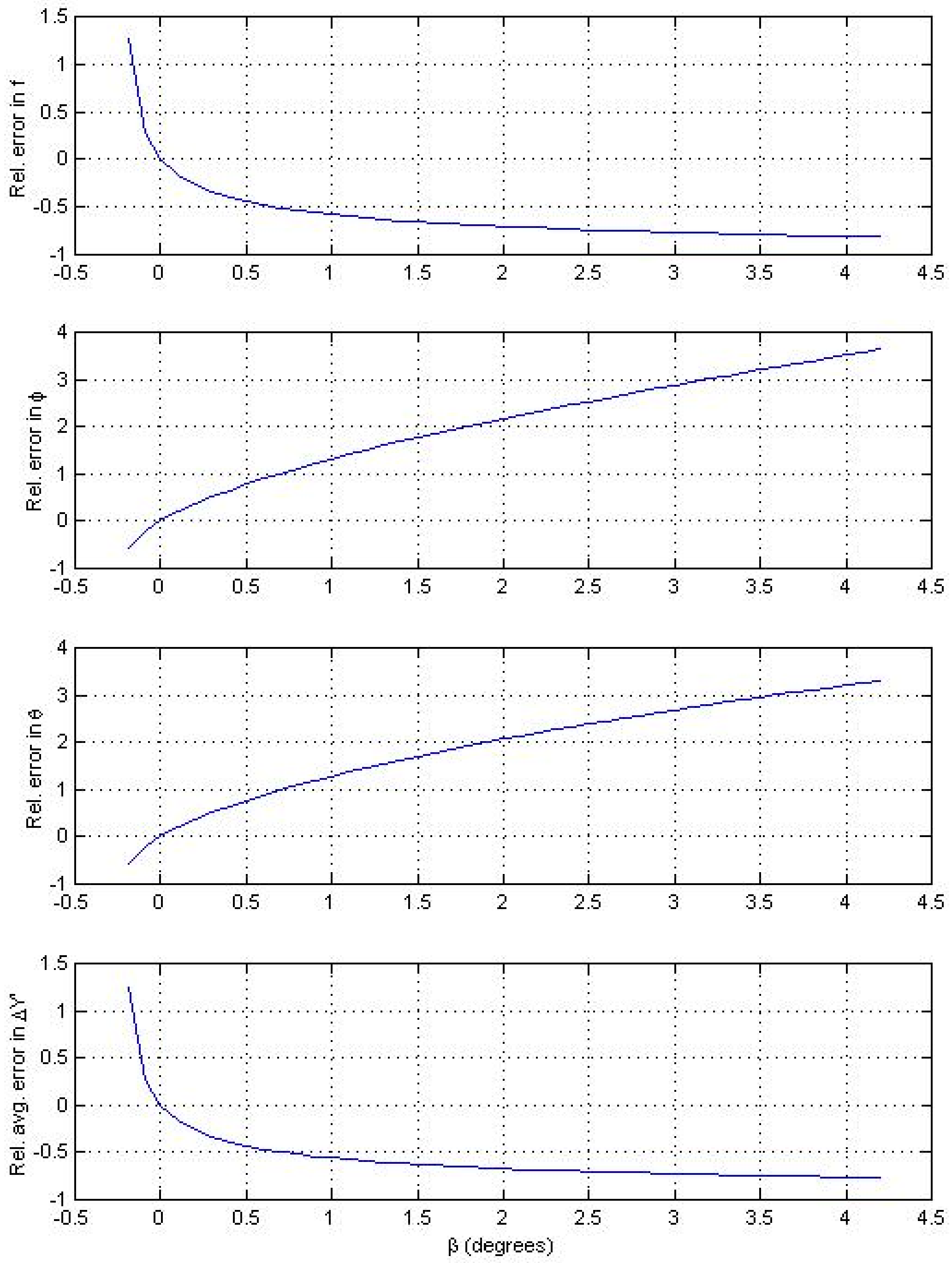


Figure 41. Effects of nonzero road slope angle β on camera calibration Method 2 and its output parameters when processing Scene 3.

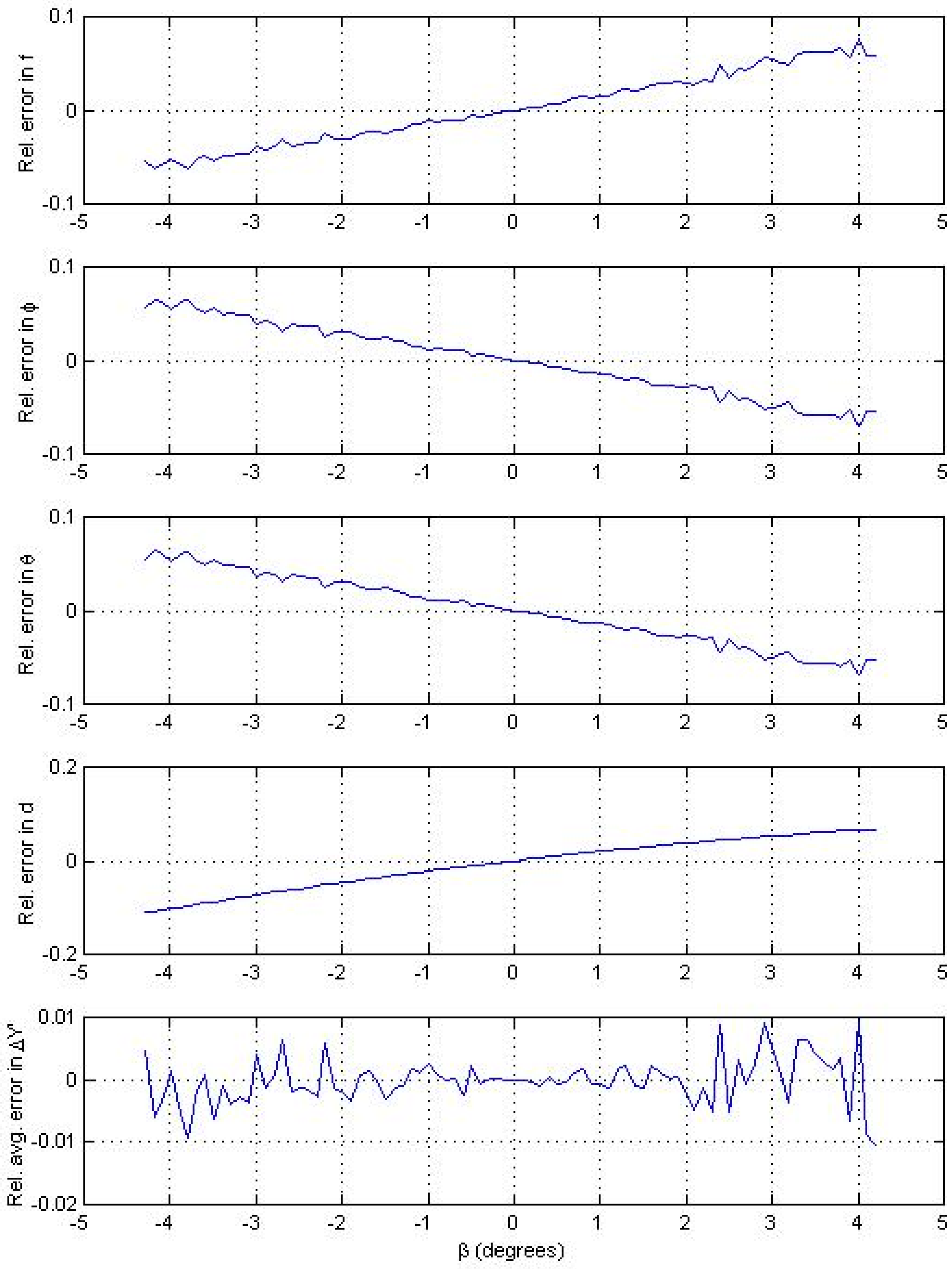


Figure 42. Effects of nonzero road slope angle β on camera calibration Method 3 and its output parameters when processing Scene 1.

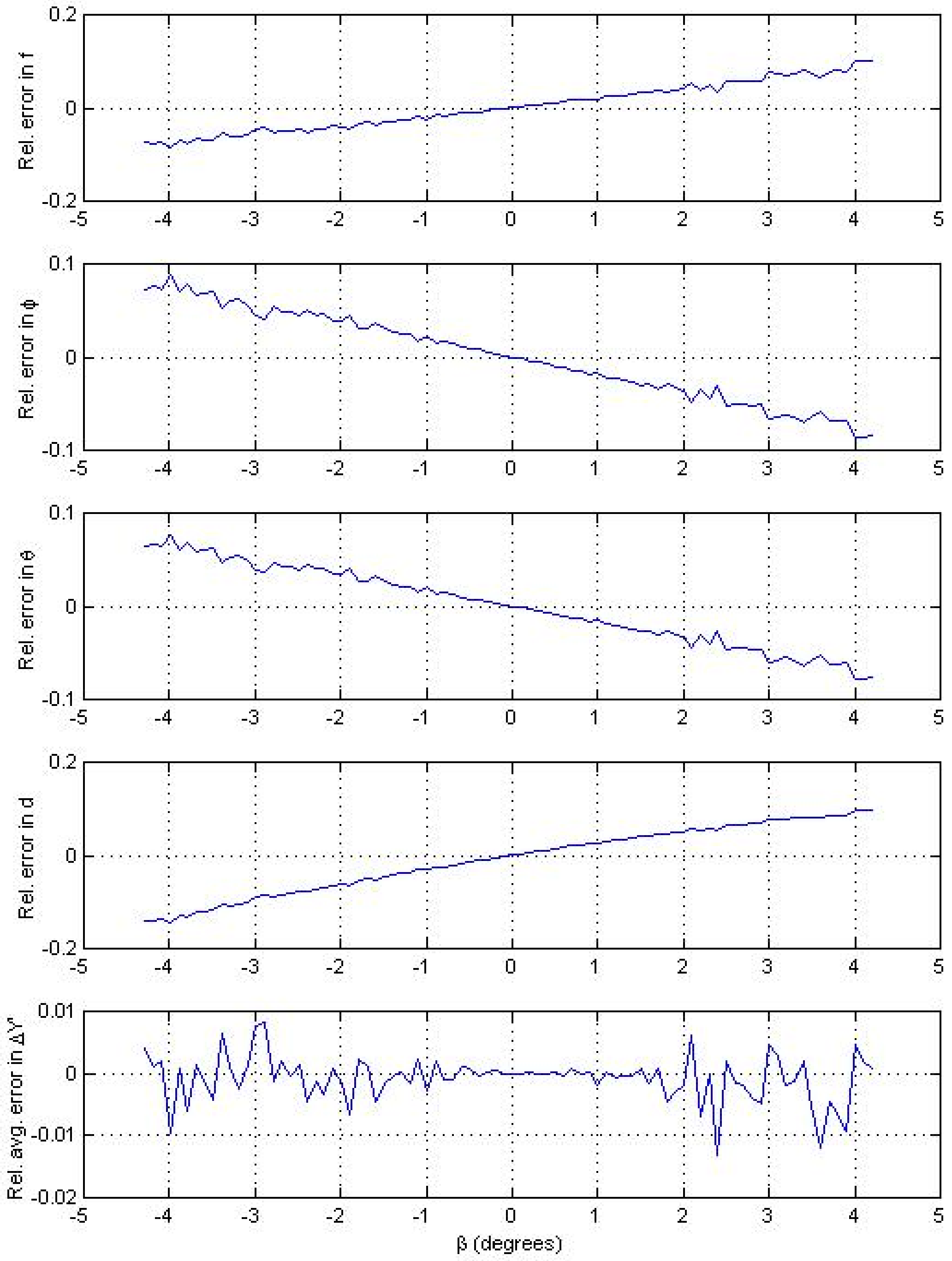


Figure 43. Effects of nonzero road slope angle β on camera calibration Method 3 and its output parameters when processing Scene 2.

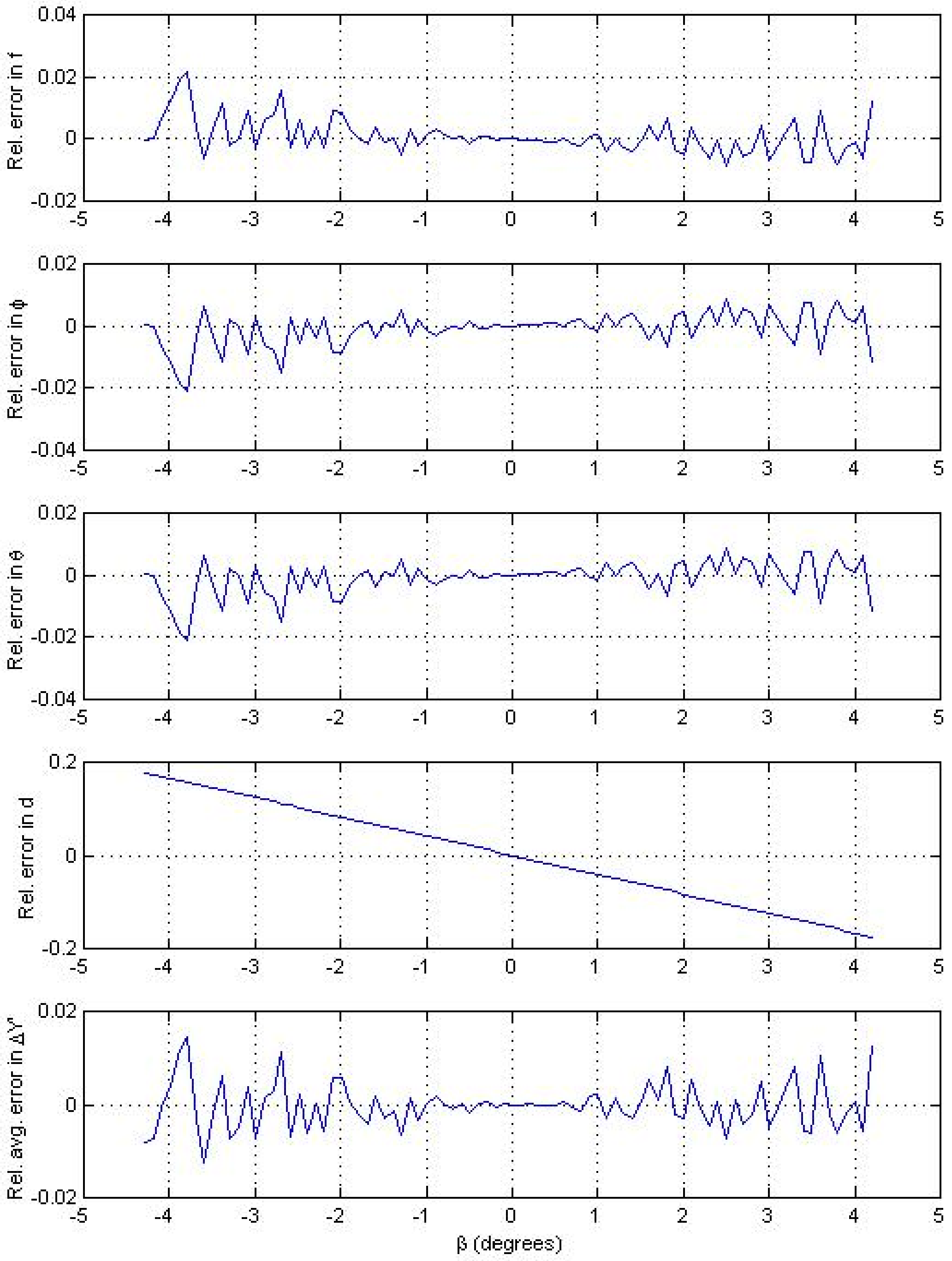


Figure 44. Effects of nonzero road slope angle β on camera calibration Method 3 and its output parameters when processing Scene 3.

2.6 Comparison of the camera calibration methods

The camera calibration methods described above share only two major assumptions in common: 1) the computer can use the image sequence to estimate v_0, the coordinate of the vanishing line, and 2) the focal length and camera angles are all unknown. The first two methods assume that the road is fully visible, whereas the third method only requires that it be able to measure the longitudinal spacing between features parallel to the road. The first two methods also require the computer to estimate the slope-intercept parameters (in image coordinates) for the road boundaries in order to estimate 3-D distances $\Delta Y'$ (assuming the camera height is unknown). The third method only requires these measurements when estimating the camera parameters f, ϕ, and θ. Each method makes different assumptions about the availability of *a priori* knowledge about parameters of the scene geometry, and these assumptions determine the type of information that must be extracted from the image sequence.

Our analysis of the limits of operation and sensitivity of Method 1 to input errors, nonzero road slope, and nonzero road tilt reveals that the following criteria must be met for it to serve as a stand-alone means of calibrating the camera for speed estimation: 1) No road tilt. 2) Road slope of less than 1°. 3) Camera down angle ϕ of at least 7°. 4) Camera pan angle θ of at least 10°. 5) Road boundaries that are fully visible in the lower half of the image with a known road width w. Alternately, if the camera height h is known, the image need not contain the road boundaries. All of these criteria require the camera operator to take special care in orientating and focusing the camera. In general, Method 1 is not directly applicable to the problem posed in Chapter 1, in which the WSDOT cameras typically focus on the far field. Improving the camera resolution by several multiples could lessen the restrictions of criteria 3 and 4, but this would not change the sensitivity of Method 1 to road tilt or slope. However, when the road slope and road tilt are truly negligible and when the camera is properly oriented, Method 1 can provide a good estimate of d, which is necessary for Method 2 to succeed.

Method 2 has multiple areas of weakness that reduce its usefulness in the real world to a very narrow range. First of all, this method is very sensitive to errors in d, which is assumed to be known. For example, Figure 11 shows that in situations where the camera is focused on the far-field (e.g., scenes 1 and 3), a 1 percent error in *d* resulted in at least a 10 percent error in the road distance estimate $\Delta Y'$. Although methods 1 and 3 can both estimate *d*, any nonzero road slope or tilt combined with the other estimation errors could easily add up to errors in the 3-5 percent range, requiring us to look elsewhere for a more accurate estimate of *d*. In a related issue, if Method 2 is to be used, the road must be perfectly straight because any road curvature will result in a slightly different estimate of *d*, depending on which portion of the road the camera is viewing. In addition, even a small amount of road slope or tilt can easily impose a bias of 5 percent or more on measurements of road distance, as seen in figures 34, 35, 40, and 41. Lastly, figures 32 and 32 show that for typical four-lane roads, Method 2 is reasonably accurate only when $50 < h < 70$ feet and $-20 < d < 40$ feet.

Thus, although it is correct under our original assumptions, we can recommend Method 2 for calibrating the camera under real world conditions only if the following requirements are met. 1) Perfectly straight road. 2) Road slope of less than 0.5°. 3) No road tilt. 4) Estimate for *d* with better than 0.5 percent accuracy. 5) Camera positioned such that $50 < h < 70$ feet and $-20 < d < 30$ feet.

Method 3 offers several unique and desirable advantages over the other calibration approaches. First of all, it does not require that the road lie entirely within the bottom half of the image. Instead, it only requires the availability of features parallel to the road separated by a known distance. This expands the range of camera views well beyond the requirements of Equations (45) and (46). Second, we found in Section 2.4.3 that Method 3 can estimate road distances under realistically noisy conditions over a wide range of ϕ and θ. Furthermore, the distance estimates provided by the camera calibration of Method 3 deviate less than 1 percent from the true value even when the road orientation clearly violates our assumptions of zero road slope and tilt. On the basis of our sensitivity analysis, assuming that lane marker features are present in the image, we are able to recommend Method 3 as a sound approach for calibrating the camera under a wide variety of focal lengths and orientations, including those typically applied to the WSDOT traffic cameras.

3 IMAGE FEATURE EXTRACTION FOR AUTOMATIC SCENE ANALYSIS

Chapter 2 provided a fundamental model for the camera and road scene. It also described how to use measurements of the line parameters from the road boundary to calibrate the camera for measuring distances along the road. However, these measurements must be extracted automatically from digital images by a computer when the camera may move at any time. To make the task even more difficult, the algorithm must perform this task under the wide variety of lighting and weather conditions described in Chapter 1.

Figure 45 contains a high-level block diagram that describes our framework for achieving the tasks and obtaining the measurements required by a system for estimating mean vehicle speed from an uncalibrated traffic monitoring camera. Working at the highest resolution possible, we generate two feature images using 1,000 images from the image sequence that enable us to characterize the scene: the activity map and the top-hat image. From these feature images alone, we can estimate u_0, v_0, b_1, b_2, and τ_{ab}. We can also extract lane masks using this information. By re-processing the 1,000 images (or a new set of images) together with the feature images, we are also able to estimate u_1. The following sections describe how to generate the feature images. They also expand each of the blocks in Figure 45 to provide the exact details of how to perform the tasks described using image and signal processing techniques.

We note that when discussing image limits in this chapter, we will often alternate between image-centered and non-image centered coordinates; the convention should be clear from the context. We afford ourselves this convenience because the images are represented in the computer with non-image centered coordinates.

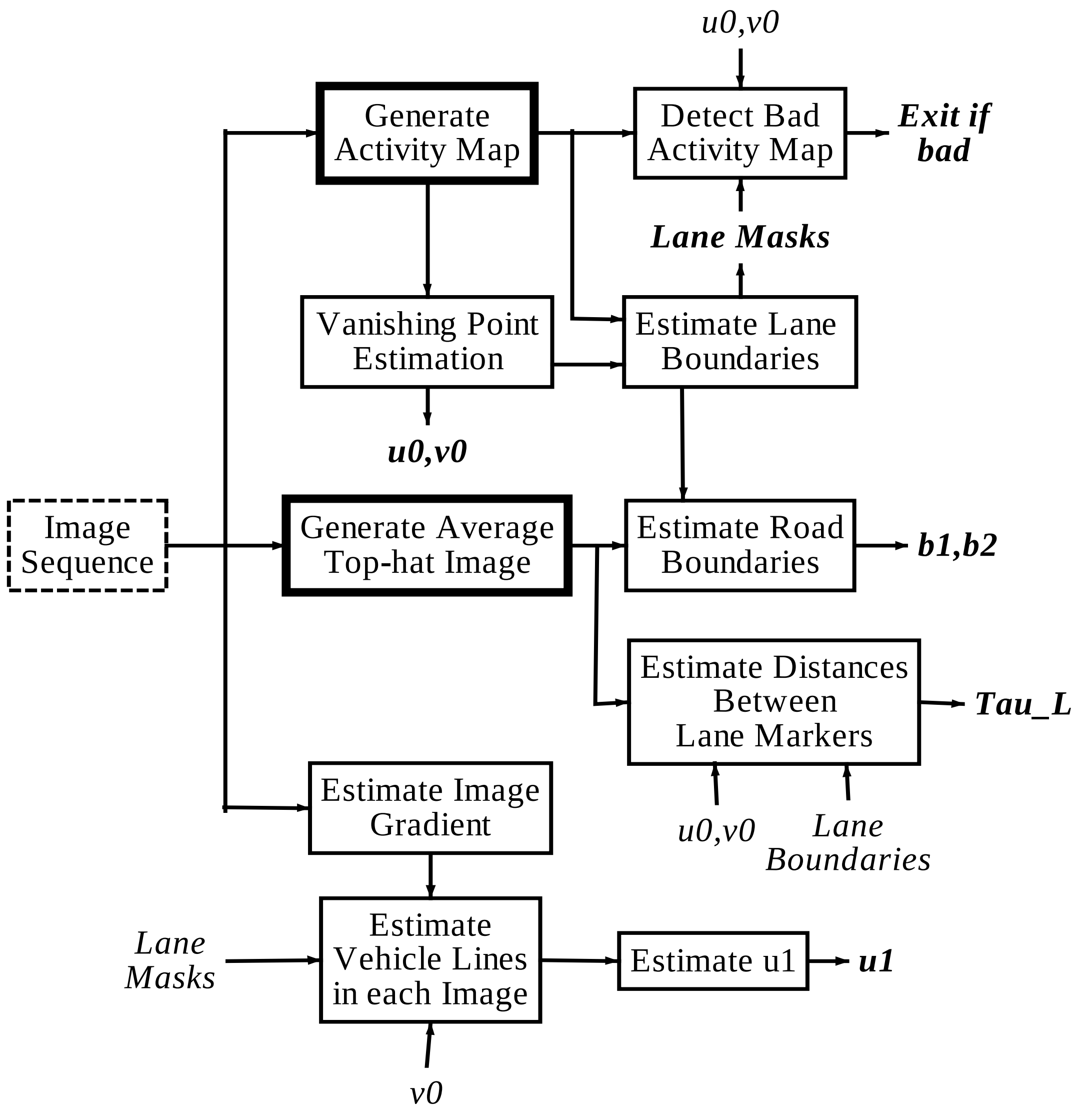


Figure 45. Framework for performing the computer vision tasks necessary to calibrate the camera and prepare for the tracking algorithm.

3.1 The activity map

From only a single image, it is generally difficult for the computer to distinguish the roadway from the background. Color clustering might be effective in some cases, but the large variety in roadway scenes would probably make such an approach ineffective in general. Attempting to locate road boundary lines in the image is another possibility, but many distracters exist with no good way to distinguish the correct lines from the incorrect ones. However, the computer can easily recognize the roadway as the portion of the image where the pixels change over time as the vehicles travel through the scene. We now describe how to generate an activity map to identify both the location and intensity of vehicular motion [40]. Inactive lanes are thus excluded from the scene analysis and, therefore, ultimately, from the speed estimation.

We assume that pixel values change over time as vehicles pass through the image and that the only significant motion in the image is due to the vehicles. We generate the activity map, A, by calculating the expected value of the absolute intensity difference, D_{img}, between two frames, I_i and I_{i+1}. Each image frame is smoothed with a 3 X 3 boxcar kernel to remove JPEG artifacts prior to the calculation of E[D_{img}].

$$A = E\left[D_{img}\right] = \frac{1}{N}\sum_{i=1}^{N}\left|I_i - I_{i+1}\right| \tag{47}$$

Such a summation of an image sequence resembles a concept presented by Stewart et al. [40] for generating an activity map with binary images obtained by thresholding. They use the map to obtain very rough lane masks from morphological operators. Our method does not require us to choose a threshold to binarize the image and is more appropriate for the task of extracting the traffic lanes. For example, the sample activity map for a normal traffic scene in Figure 46 shows horizontal gray-scale variations, indicating the middle of the lanes and their boundaries. The small circular imperfections on the right and top of the image are due to raindrops on the camera lens.

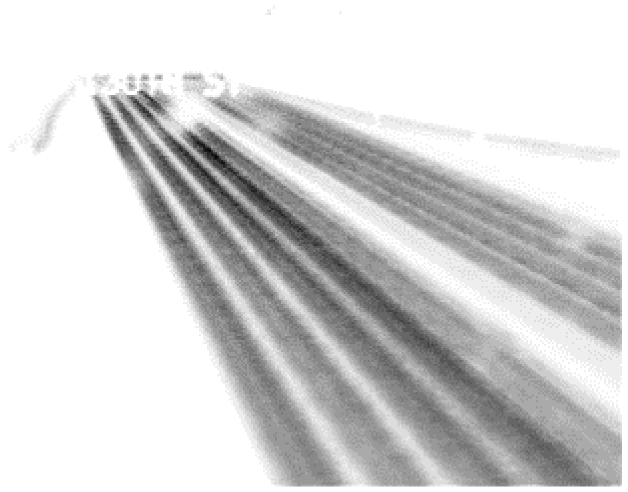


Figure 46. Extracted activity map from 1,000 roadway images

The following sections will demonstrate the great utility of the activity map in estimating the vanishing point (u_0,v_0) and the lane boundaries, which form the basis for performing other key tasks in the process of Figure 45.

3.2 Estimating (u_0, v_0) from the activity map

The vanishing point (u_0,v_0) is essential for all three calibration methods derived in Chapter 2, as well as several of the blocks in Figure 45. We use the line structure present in the activity map to estimate the vanishing point for the lines parallel to the roadway using the Hough transform. The next section contains our theoretical justification for doing so, even though our original assumption of zero vehicle height has been violated. We obtain the line structure from the activity map using Canny's edge detector [43] with $\sigma = 3$ (for 320 X 240 images) and an edge sensitivity threshold of 0.01. The wide kernel and small edge sensitivity ensure that all possible edges are detected. We use only the bottom one-third of the image as a compromise between distortion due to road curvature and vanishing point accuracy.

Once a binary edge image is available, we employ the Hough transform [44] to detect the lines that intersect at the vanishing point (u_0,v_0). Because we require the road to exit the bottom of the image, we parameterize the lines by their angle α to the v-axis (positive counter-clockwise) and their intersection u_{int} with the line $v = -H/2$ in the image as follows:

$$v = (u - u_{\text{int}})\cot(\alpha) - \frac{H}{2} \tag{48}$$

This formulation assumes image-centered coordinates. We quantize u_{int} in single-pixel intervals and α such that the transform can distinguish lines at the extremes of the bottom half of the image:

$$\Delta\alpha = \tan^{-1}\left(\frac{W-1}{H/2}\right) - \tan^{-1}\left(\frac{W}{H/2}\right) \tag{49}$$

After generating the Hough transform, we accept lines that occupy at least 40 percent of the bottom window, i.e., the Hough accumulators that exceed $0.4 \cdot H/2$. Figure 47 contains a sample Hough transform image in which the dark spots indicate the (u_{int},α) pairs corresponding to lines found in the activity map.

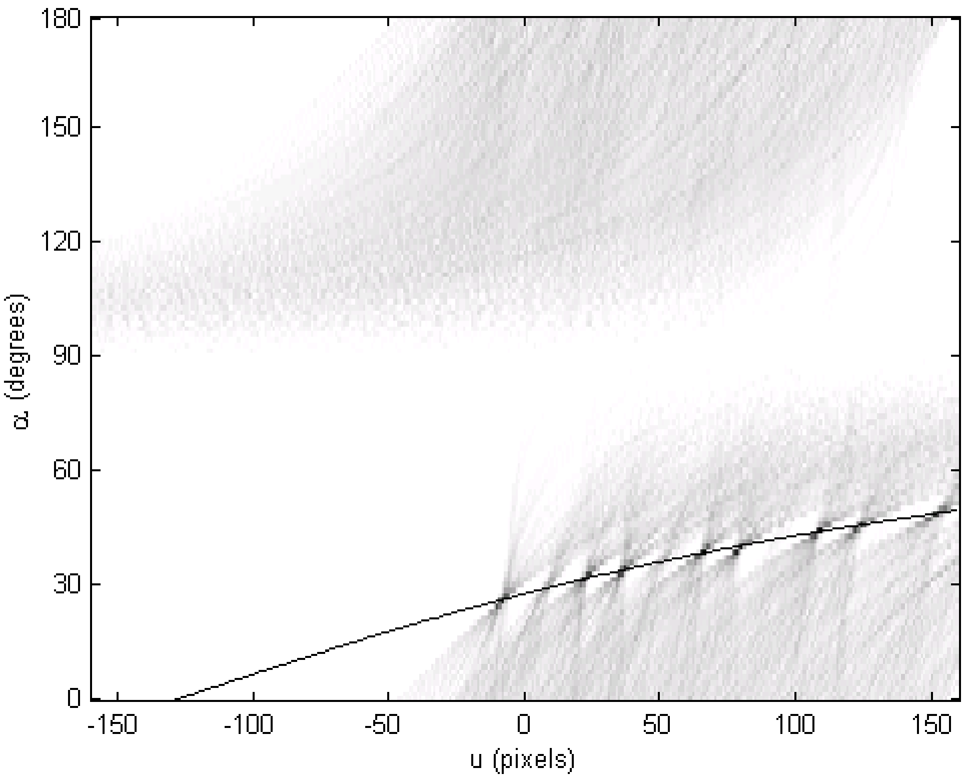


Figure 47. Hough transform in u_{int}-α space with superimposed best-fit curve for Equation (48).

Once we have detected the lines, we find the point (u_0,v_0) that minimizes the distance between the point and each line defined by the $(u_{int}(j),\alpha(j))$ pairs. Suppose that $(u_c(j),v_c(j))$ is the point on line j that is closest to (u_0,v_0). Then (u_0,v_0) is the point that minimizes the functional F_0 as follows:

$$F_0 \equiv \sum_j \left(\left(u_c(j) - u_0\right)^2 + \left(v_c(j) - v_0\right)^2 \right) \tag{50}$$

The least-squares solution is available analytically. First, however, we must solve for $u_c(j)$ and $v_c(j)$ in terms of u_0, v_0, $u_{int}(j)$, and $\alpha(j)$. We will then substitute these expressions into Eq. (50), differentiate F_0, and solve for u_0 and v_0.

Suppose we use the traditional slope-intercept form in terms of the slope m and u-intercept b to describe the line of Equation (48):

$$\begin{aligned} v + \frac{H}{2} &= \left(u - u_{\text{int}}\right)\cot(\alpha) \\ u &= \left(v + \frac{H}{2}\right)\tan(\alpha) + u_{\text{int}} = \left(\tan(\alpha)\right)v + \left(u_{\text{int}} + \frac{H}{2}\tan(\alpha)\right) \equiv mv + b \end{aligned} \tag{51}$$

For convenience, we drop the index notation for the following portion of the derivation. Next, suppose that u_0 and v_0 are known and we wish to solve for the unknown values of u_c and v_c that satisfy Equation (51) and minimize the functional

$$F_1 \equiv (u_c - u_0)^2 + (v_c - v_0)^2 \tag{52}$$

That is, (u_c,v_c) is the closest point on the line $u = mv + b$ to (u_0,v_0). Substituting Equation (51) into Equation (52) with $u = u_c$ and $v = v_c$ yields F_1 with only v_c unknown.

$$F_1 \equiv (mv_c + b - u_0)^2 + (v_c - v_0)^2 \tag{53}$$

We then differentiate F_1 with respect to v_c and set the expression equal to zero to determine the value of v_c that minimizes F_1. This yields

$$\begin{aligned} 0 &= 2m(mv_c + b - u_0) + 2(v_c - v_0) \\ v_c &= \frac{v_0 + mu_0 - bm}{1+m^2} \\ u_c &= mv_c + b = \frac{mv_0 + m^2u_0 + b}{1+m^2} \end{aligned} \tag{54}$$

We now have the values for the point $(u_c(j),v_c(j))$, the closest point on line j to the vanishing point (u_0,v_0).

Next, we add the index notation and substitute the values of $u_c(j)$ and $v_c(j)$ in Equation (54) into the original functional F_0 in Equation (50) and simplify:

$$\begin{aligned} F_0 &\equiv \sum_j \left(\left(\frac{m(j)v_0 + m(j)^2u_0 + b(j)}{1+m(j)^2} - u_0 \right)^2 + \left(\frac{v_0 + m(j)u_0 - b(j)m(j)}{1+m(j)^2} - v_0 \right)^2 \right) \\ &= \sum_j \left(\frac{\left(m(j)v_0 + m(j)^2u_0 + b(j) - u_0 - m(j)^2u_0\right)^2 + \left(v_0 + m(j)u_0 - b(j)m(j) - v_0 - v_0m(j)^2\right)^2}{\left(1+m(j)^2\right)^2} \right) \\ &= \sum_j \left(\frac{(m(j)v_0 + b(j) - u_0)^2 + m(j)^2(u_0 - b(j) - v_0m(j))^2}{\left(1+m(j)^2\right)^2} \right) \\ &= \sum_j \left(\frac{(u_0 - b(j) - v_0m(j))^2}{1+m(j)^2} \right) \end{aligned} \tag{55}$$

Differentiating F_0 with respect to u_0 yields the following equation:

$$\begin{gathered}0=\sum_j\left(\frac{2(u_0-b(j)-v_0m(j))}{1+m(j)^2}\right)\\ 0=u_0\sum_j\left(\frac{1}{1+m(j)^2}\right)-\sum_j\left(\frac{b(j)}{1+m(j)^2}\right)-v_0\sum_j\left(\frac{m(j)}{1+m(j)^2}\right)\equiv u_0c_1-c_2-v_0c_3\end{gathered}\tag{56}$$

For convenience, we define c_1, c_2, and c_3 as the coefficients of the u_0, constant, and v_0 terms of the equation. Differentiating F_0 with respect to u_0 yields the following equation:

$$\begin{gathered}0=\sum_j\left(\frac{-2m(j)(u_0-b(j)-v_0m(j))}{1+m(j)^2}\right)\\ 0=u_0\sum_j\left(\frac{m(j)}{1+m(j)^2}\right)-\sum_j\left(\frac{m(j)b(j)}{1+m(j)^2}\right)-v_0\sum_j\left(\frac{m(j)^2}{1+m(j)^2}\right)\equiv u_0c_3-c_4-v_0c_5\end{gathered}\tag{57}$$

For convenience, we define c_4 and c_5 as the coefficients of the constant and v_0 terms of the equation. From here we have

$$\begin{gathered}u_0=\frac{c_3c_4-c_2c_5}{c_3{}^2-c_1c_5}\\ v_0=\frac{c_1c_4-c_2c_3}{c_3{}^2-c_1c_5}\end{gathered}\tag{58}$$

where the constants are defined in terms of $m(j)$ and $b(j)$, which in turn are calculated from $u_{int}(j)$ and $\alpha(j)$. Substituting these values of u_0 and v_0 for u and v in Equation (48) completely specifies the best-fit relationship between the set of lines in terms of a curve in the α-u_{int} plane (see Figure 47).

If there are only a few lines, e.g., less than five, then the algorithm should either display a warning or failure message. Otherwise, we can estimate the standard deviation of u_0 and v_0 simply by estimating the standard deviation of the u_c and v_c data. Figure 48 illustrates one distribution of the $(u_c(j),v_c(j))$ pairs and their associated lines surrounding the estimate of (u_0,v_0). We can see that u_c and v_c are strongly correlated. Figure 49 contains histograms of the u_c and v_c data, which, though not necessarily normal, are centroidal enough that the standard deviation is a reasonable way to estimate their error and the spread of the data.

After obtaining an initial estimate of (u_0,v_0), we eliminate outliers and re-estimate the vanishing point. Specifically, we remove lines whose closest point $(u_c(j),v_c(j))$ lies more than twice the sample standard deviation (estimated above) away from the estimate of (u_0,v_0). We re-estimate the vanishing point and repeat this procedure until no more points are eliminated. This rough estimate for (u_0,v_0) provides us with a critical component necessary for several of the important steps in Figure 45, as well as the three camera calibration methods themselves.

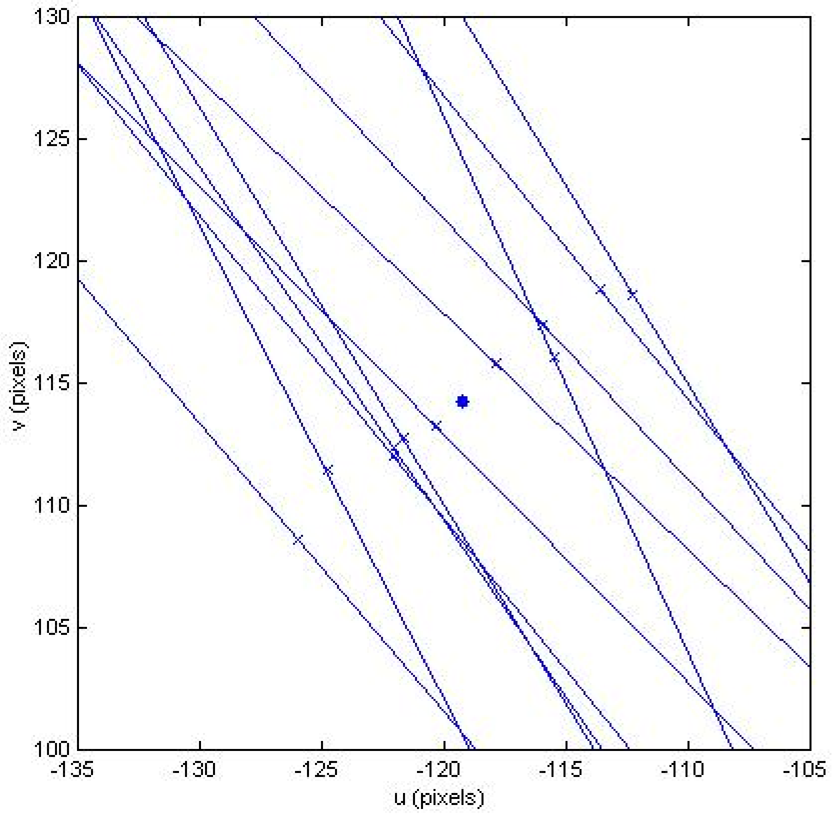


Figure 48. Close-up view of a vanishing point estimate, one-third of the lines used to estimate it, and the ($u_c(j)$,$v_c(j)$) pairs closest to (u_0,v_0) on each line.

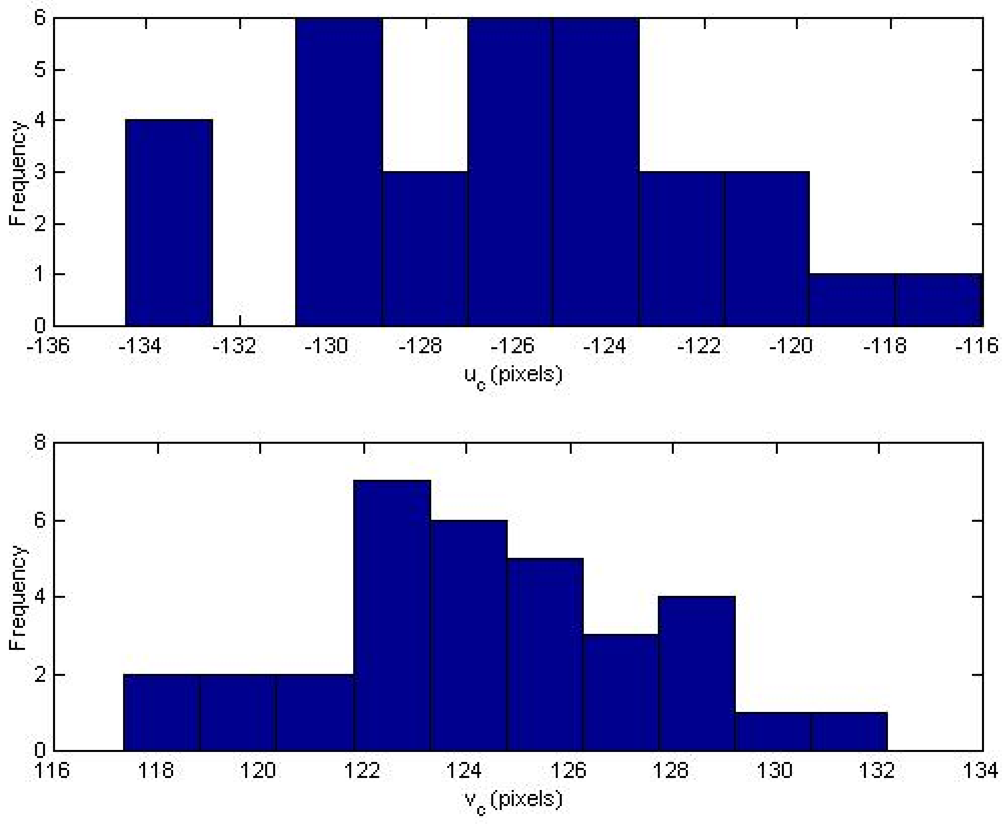


Figure 49. Histograms of the values of u_c and v_c used in estimating u_0 and v_0.

3.2.1 *Theoretical effects of nonzero vehicle height on* (u_0,v_0)

In our derivation of the simplified camera model presented in Chapter 2, we assumed a flat plane on which the vehicles traveled, as illustrated in Figure 2. Furthermore, we tacitly assumed that all moving vehicle features were positioned completely within the plane, i.e., $Z = 0$. Clearly, this is not the case in reality. Semi-trucks often appear to completely occupy two lanes within the image, e.g., Figure 51. We now expand the derivation and examine what happens when we do not assume that $Z = 0$.

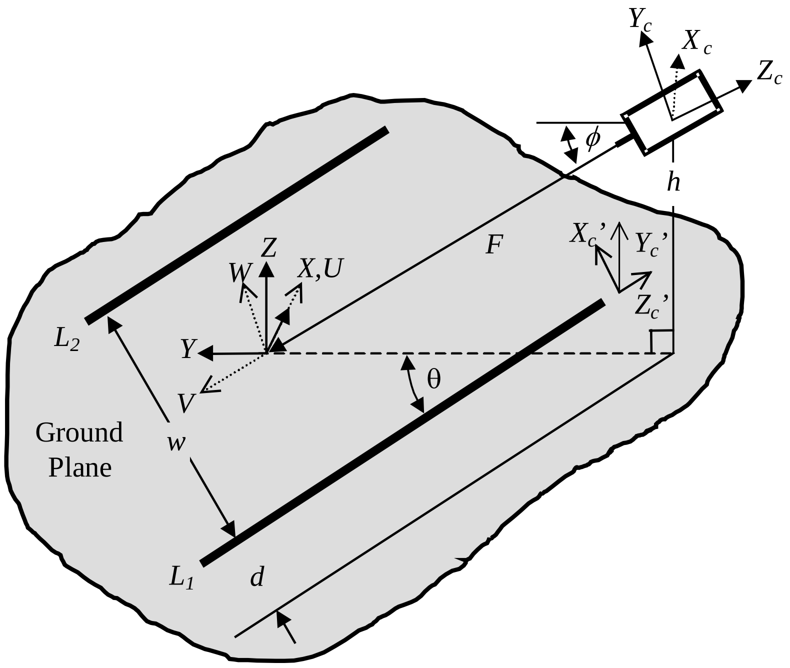


Figure 50. Camera and roadway geometry.

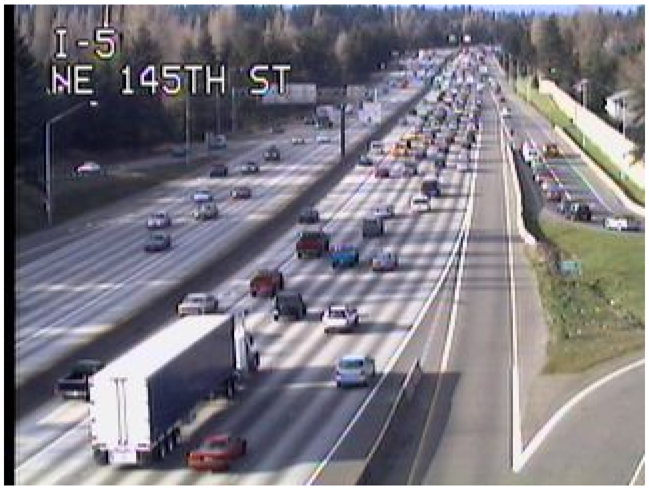


Figure 51. Traffic image that illustrates the effects of nonzero vehicle height.

We begin by finding the *U*-*V*-*W* system by rotating an angle ϕ around the *X*-axis:

$$\begin{aligned} U &= X \\ V &= Y\cos(\phi) - Z\sin(\phi) \\ W &= Y\sin(\phi) + Z\cos(\phi) \end{aligned} \tag{59}$$

Here is where the current derivation departs from the previous approach, i.e., we include Z in the remainder of our derivation rather than assuming $Z = 0$ in Equation (2). We now apply a displacement $F = h \cdot \csc(\phi)$ to obtain the camera-centered coordinates X_c-Y_c-Z_c.

$$\begin{aligned} X_c &= U = X \\ Y_c &= W = Y\sin(\phi) + Z\cos(\phi) \\ Z_c &= -V - F = -Y\cos(\phi) + Z\sin(\phi) - F \end{aligned} \tag{60}$$

Taking the perspective projection yields

$$u = -f\,\frac{X_c}{Z_c} = -f\,\frac{X}{-Y\cos(\phi) + Z\sin(\phi) - F} \tag{61}$$

$$v = -f\,\frac{Y_c}{Z_c} = -f\,\frac{Y\sin(\phi) + Z\cos(\phi)}{-Y\cos(\phi) + Z\sin(\phi) - F} \tag{62}$$

Next, we determine the coordinates of the vanishing point by calculating the limit as $Y \rightarrow \infty$. We evaluate the terms of the limit by obtaining constant terms in the ratio as well as terms that go to zero as $Y \rightarrow \infty$, since Y is in the denominator.

$$u_0 = \lim_{Y\to\infty} \frac{fX}{Y\cos\phi - Z\sin(\phi) + F} = \lim_{Y\to\infty} \frac{f\,\frac{X}{Y}}{\cos\phi - \frac{Z}{Y}\sin(\phi) + \frac{F}{Y}} = -f\,\tan(\theta)\sec(\phi) \tag{63}$$

$$v_0 = \lim_{Y\to\infty} \frac{fY\sin\phi + Z\cos(\phi)}{Y\cos\phi - Z\sin(\phi) + F} = \lim_{Y\to\infty} \frac{f\sin\phi + \frac{Z}{Y}\cos(\phi)}{\cos\phi - \frac{Z}{Y}\sin(\phi) + \frac{F}{Y}} = f\,\tan(\phi) \tag{64}$$

Thus, we obtain the same result as in Chapter 2, demonstrating that the vanishing point coordinates are not biased by nonzero vehicle heights. This fully justifies our use of the activity map to estimate (u_0,v_0).

3.3 Estimating the lane boundaries and lane masks

As shown in Figure 45, once we have obtained the activity map and estimated the vanishing point (u_0,v_0), we can extract the position of the boundaries for the vehicle lanes. Obtaining these positions enables us to greatly simplify the task of tracking vehicles since their motion is constrained by the known boundaries. We begin the lane boundary estimation process by sampling the activity map along the lines connecting the vanishing point (u_0,v_0) and pixel u_j in the bottom row of the image. Averaging the activity value along each line creates a one-dimensional signal $A_{rel}(u)$, as shown in Figure 52. The peaks indicate strong traffic activity (middle of the lane), and the valleys indicate the absence of vehicle activity (lane boundaries). We smooth the signal with a low-pass filter with a cutoff frequency of 1/(20 pixels) based on the assumption that each lane is at least 20 pixels wide on the bottom of the image. In most of the examples presented, we have chosen to sample $u_j \in [-0.15W, 1.15W]$ to detect road lanes that are generally within the image but lie outside the image on the bottom row (using non-image centered coordinates).

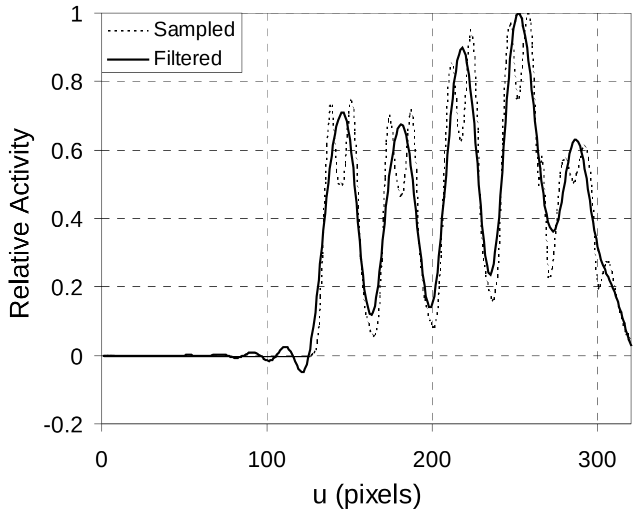
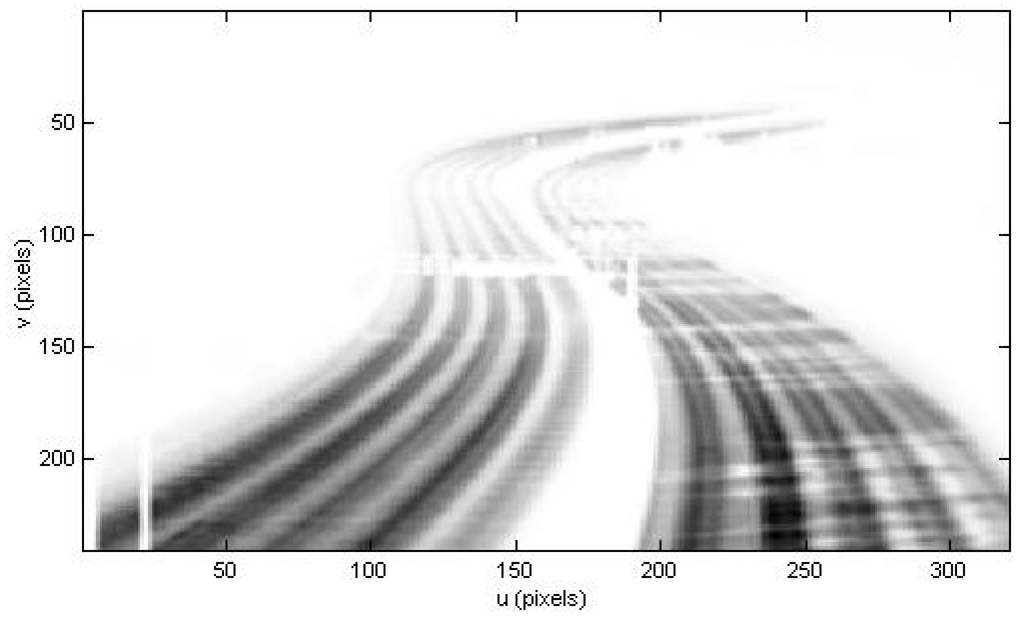


Figure 52. Average relative activity across the lower third of the activity map in Fgiure 46 at the orientations specified by the vanishing point.

The data in figures 46 and 52 actually represent a nearly ideal case. All of the freeway lane peaks are obvious, and there is no ambiguity about which group of lanes to identify. However, we desire an algorithm that can robustly identify the lane boundaries even when $A_{rel}(u)$ remains noisy after the low-pass filtering or when two or more sets of lanes are present. Figures 53 and 54 contain activity maps and $A_{rel}(u)$ signals that illustrate these problems.



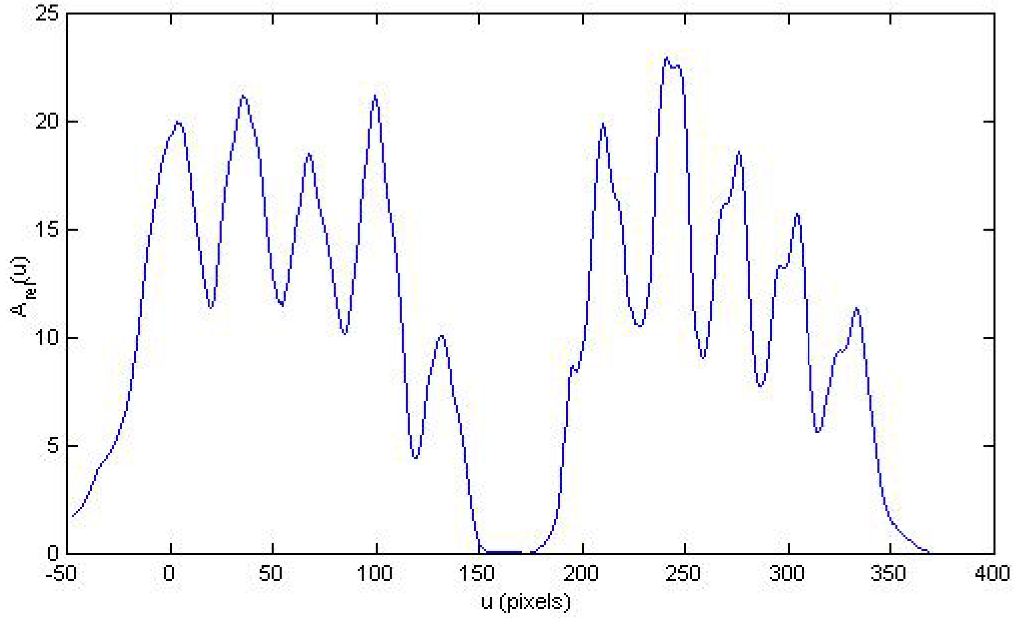
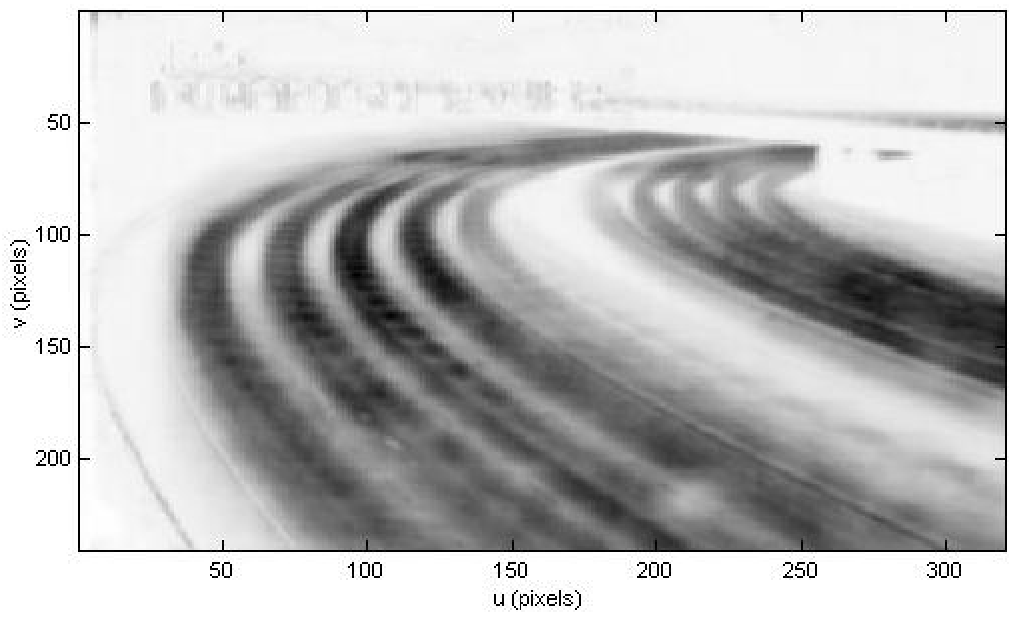


Figure 53. a) Activity map containing multiple sets of lanes for a scene with shadows on the road. b) $A_{rel}(u)$ signal for the activity map.



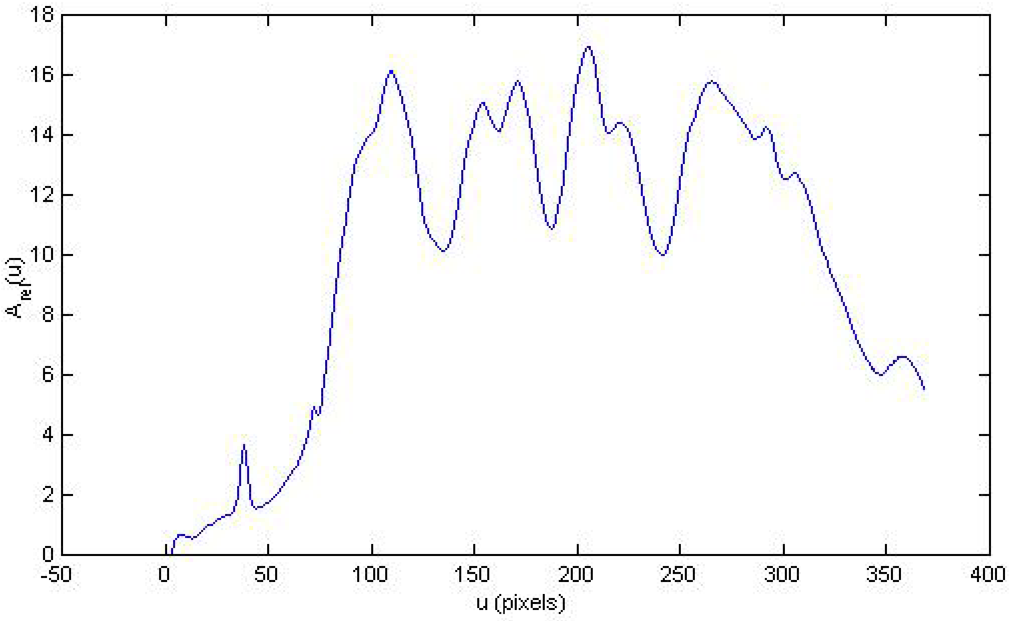


Figure 54. a) Noisy activity map due to wide lanes and raindrops.
b) $A_{rel}(u)$ signal for the activity map.

Our process for extracting the lane boundaries is quite complex, but it can be summarized in three major steps: 1) find a template T of the best peak in $A_{rel}(u)$; 2) locate all other instances of T in $A_{rel}(u)$; 3) remove the bad peaks. On the basis of the peak locations, we interpolate to find the lane boundaries on either side of each peak. For N_{peak} peaks we expect to locate $N_{peak}+1$ lane boundaries.

3.3.1 Finding the peak template

We start our search for the exemplary peak template, T, by taking the morphological top-hat transform A_{top} of the $A_{rel}(u)$ signal [45]. We use a kernel width equal to one-third the length of the signal. This size is narrow enough to expose the peaks without shortening them too much, which could happen if the kernel was too narrow. Figure 55 contains the top-hat transform of the signal in Figure 54(b), normalized by the peak value of $A_{rel}(u)$. The image width is 320 pixels.

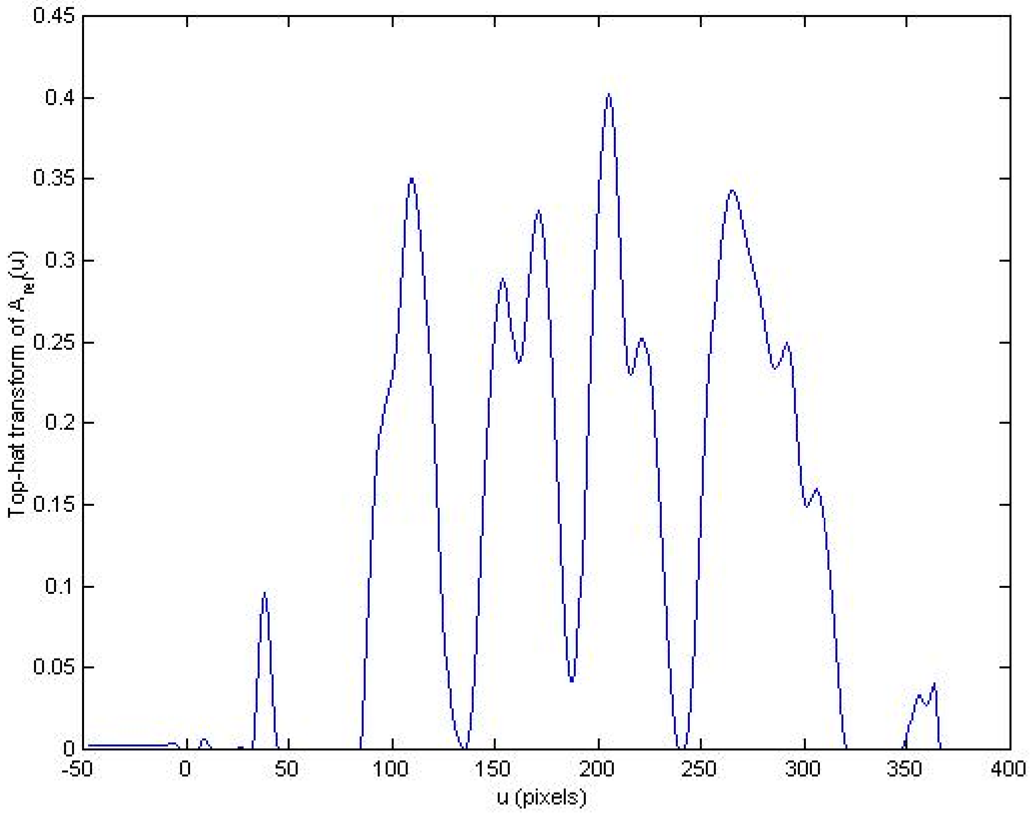


Figure 55. Top-hat transform A_{top} of the $A_{rel}(u)$ signal of Figure 54(b).

Now that we have exposed the periodic structure of the signal, we calculate the autocovariance signal to estimate the distance between repetitions of the template. Recall that the autocovariance sequence is the autocorrelation sequence of the signal after removing the mean of the signal. Before doing so, however, we first threshold to zero any residual values that are less than 0.05 (relative to the maximum value of $A_{rel}(u)$) and shorten the signal by removing the zero values on either end. Figure 56 contains the autocovariance signal for our example.

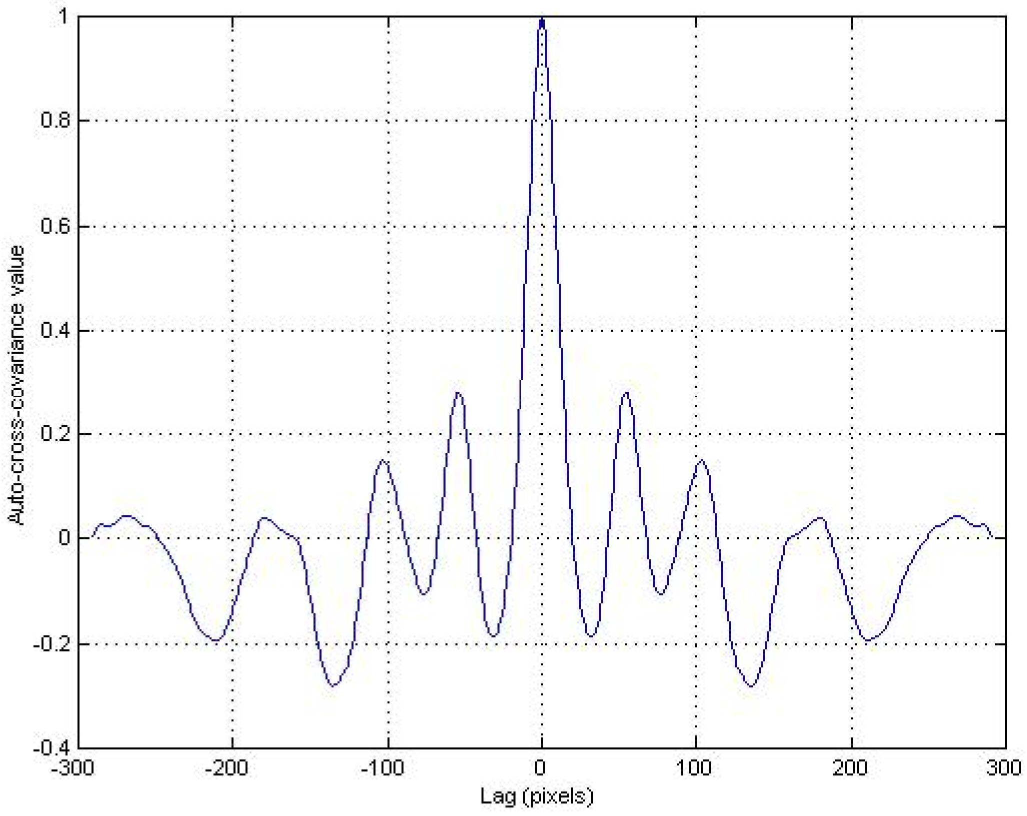


Figure 56. Autocovariance sequence of the signal in Figure 55 after thresholding and shortening it.

Having calculated the autocovariance sequence, we identify the distance between templates by finding the lag τ_1 of the first peak of the sequence away from zero, e.g., 53 for the example of Figure 56. We require the peak to have a minimum value of 0.2 with a lag value of less than $0.4W$. These requirements ensure that the peak is reasonably strong and that there is room for at least two-and-a-half lanes within the width of the image. If these requirements are not met, we assign τ_1 a value of one-third of the $A_{rel}(u)$ sequence length.

We note that some lanes have dual peaks in Figure 55. This is due to the grooves in the pavement left by the vehicles. They cause a greater average frame difference because of the higher contrast. On occasion, these can become so extreme that our algorithm can introduce a false autocovariance peak that can be quite significant. If this occurs, the true peak in the autocovariance is often stronger than usual because the extra valleys in the signal make each period more distinctive. We also expect the true peak to be greater than the false peak because the pavement groove signatures are not as uniformly periodic as are the lane signatures. Thus, if the maximum autocovariance peak within the first 40 percent of the signal exceeds 0.35, then we use it instead of the first peak.

We then use the lag value to define the width of the kernel for a second, more restrictive top-hat transform A_{top}' of $A_{rel}(u)$ to obtain the peaks. We also calculate the bottom-hat transform A_{bot}' [45], which has large values where the valleys of $A_{rel}(u)$ occur. Thus, even if we used the default autocovariance lag, we can still proceed because $A_{top}' = A_{top}$. These signals are all contained for our example in Figure 57. We locate the peak template by finding the location u_{max} of the maximum value of $A_{rel}(u)$, e.g., $u = 205$ in this case. We then search A_{top}' and A_{bot}' to the left and to the right of u_{max}. We record the smallest distance d_{min} from u_{max} such that $A_{top}' < 0.1$ or $A_{bot}' > 0.2$. If we find an autocovariance peak, then we are assured that there is some strong periodicity present in A_{top}', and we want to use the largest template in case there are some false little peaks. In this case, we set the template width W_{temp} to $2 \cdot \max(d_{min}, \tau_1/2)+1$. If we do not find an autocovariance peak, then the template size depends nearly entirely on the searching algorithm just described. We therefore set the template width W_{temp} to $2 \cdot \min(d_{min}, \text{length}(A_{top}')/3)+1$. Thus, we define the template as the subset of A_{top}' located at $u \in [u_{max} - (W_{temp} - 1)/2, u_{max} + (W_{temp} - 1)/2]$. The A_{top}' template located for our example is highlighted in bold in Figure 57. We note that good peak is found in the autocovariance function, which indicates periodicity in $A_{rel}(u)$. Thus, we use the largest template possible, based on the location of the autocovariance peak, even though the template includes a portion of another peak.

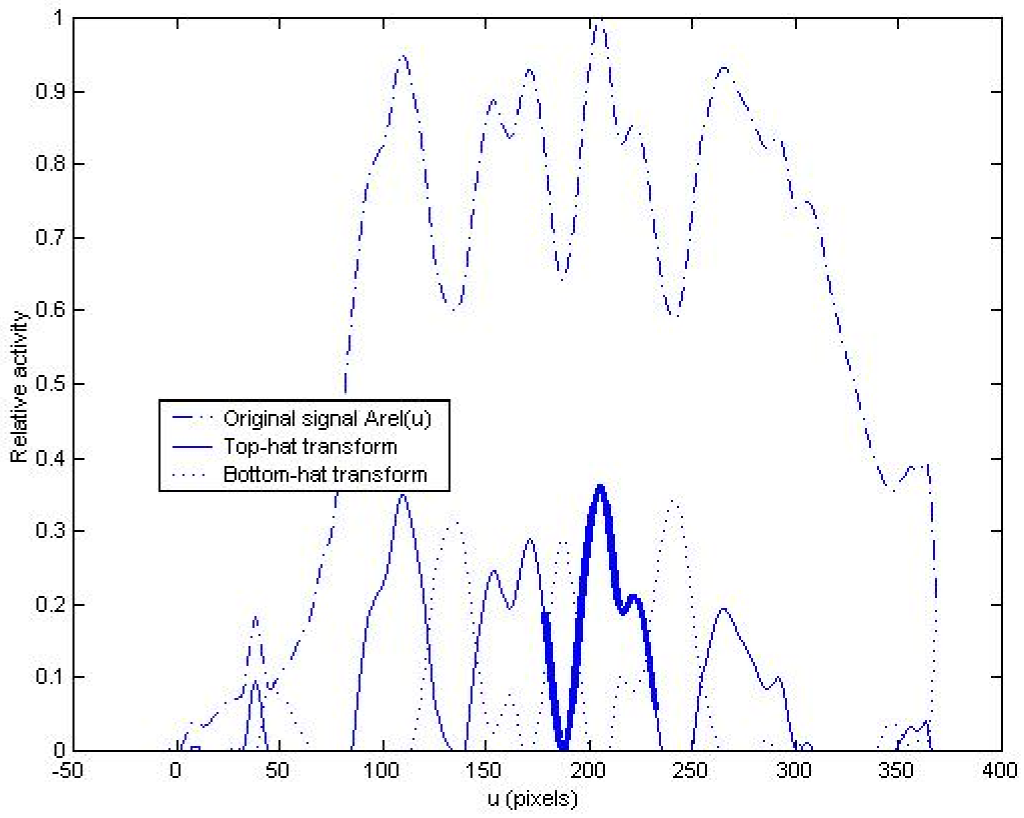


Figure 57. $A_{rel}(u)$ signal and its top-hat A_{top}' and bottom-hat A_{bot}' signals used to extract the peak template in bold.

3.3.2 *Locating activity peaks*

Once the template is identified, we search A_{top}' for other instances. We search by shifting the template T along A_{top}' and calculating the correlation coefficient for the subset A_{top}'' of A_{top}' where the two signals overlap. We define the correlation coefficient ρ_{corr} as

$$\rho_{corr} = \frac{E\left[\left(A_{top}'' - E\left[A_{top}''\right]\right)(T - E[T])\right]}{\sigma_{Atop'}\sigma_T} \tag{65}$$

where $\sigma_{Atop'}$ and σ_T are the standard deviations of the overlapping portions of the signals, and E[·] denotes the expectation operator. This method is a powerful one because it normalizes for the signal amplitude and focuses primarily on detecting similarly shaped signals. However, to avoid responding to noise, we require max(A_{top}'')/max(T) to exceed a threshold T_{min}, which we choose to be 0.2 on the basis of experience, where A_{top}'' is the subsignal of A_{top}' at a given lag value. This value for T_{min} is low enough that we can respond to weak peaks when the template is a strong peak while eliminating the response to much of the signal noise. We also require the relative energy (variance) of the subsignal A_{top}'', i.e., E[(A_{top}'' - E[A_{top}''])^2]/E[(A_{top}'' - E[A_{top}''])^2], to exceed T_{min}^2. Calculating ρ_{corr} as a function of u with these restrictions yields a result similar to that in Figure 58. We see that the correlation coefficient method is selective in that its peaks correspond only to portions of A_{top}' that are similar to the template. The method is also sensitive because it produces a strong response even when the amplitude of A_{top}' is small, e.g., the leftmost peak. The correlation coefficient method is a very satisfying approach to use because its properties are well-founded theoretically in statistics, and choosing a minimum match value is intuitive.

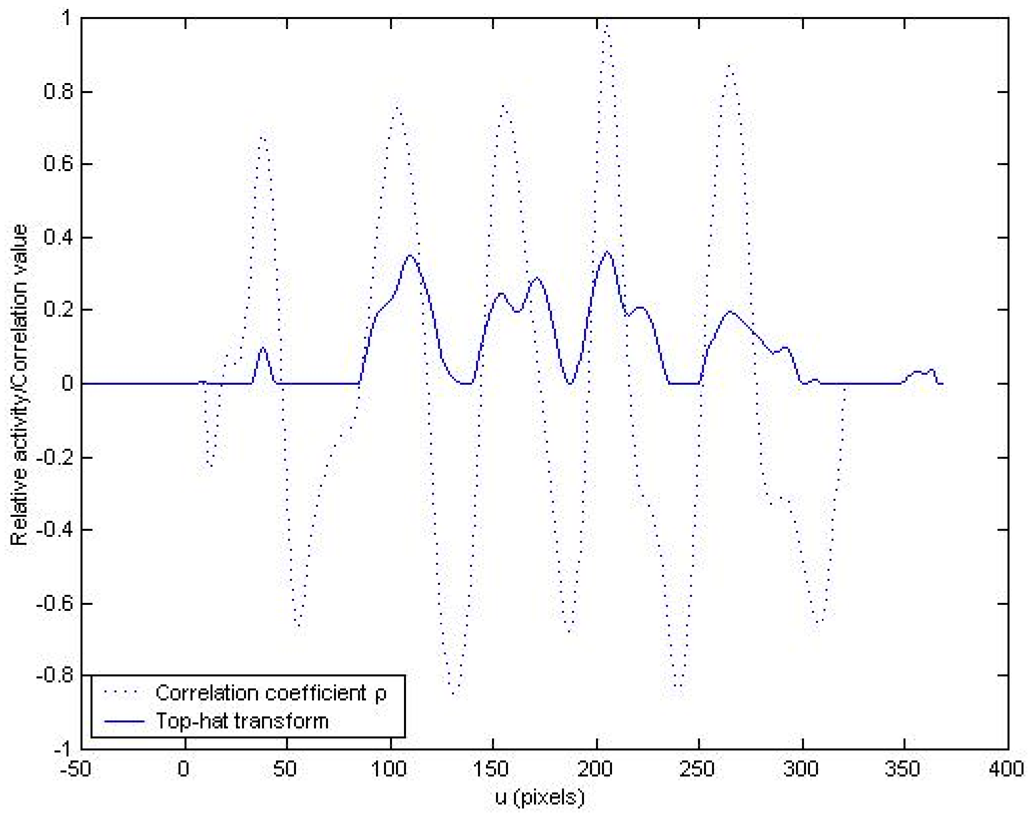


Figure 58. Locating the peaks of A_{top}' from the maxima of the correlation coefficient signal ρ_{corr} (u) found by matching the template with A_{top}'.

3.3.3 Removing bad peaks

As previously mentioned, the selectivity and sensitivity of the template correlation method are useful properties. However, we must occasionally remove false maxima of the correlation coefficient sequence. As a first criterion, we require the maxima to exceed a threshold ρ_{min}. We chose $\rho_{min} = 0.7$, which requires a reasonably strong match, but it can occasionally allow false peaks to pass, e.g., the leftmost peak in Figure 58.

Another observation we can use to eliminate peaks is the uniformity of the peak-to-peak distance. If one peak (e.g., the leftmost one above) is too far from its neighbor, then it should be eliminated. Suppose we label the location of each candidate peak as $u_{peak}(i)$, where $1 \leq i \leq N_{peak}$. We let i_{max} denote the value of i corresponding to the maximum peak in $A_{rel}(u)$. In the example of Figure 58, $u_{peak}(i_{max}) = 205$. Now let $D_{max} = 1.3 \cdot \max(u_{peak}(i_{max}) - u_{peak}(i_{max} -1)$, and $u_{peak}(i_{max}+1) - u_{peak}(i_{max}))$ be the maximum distance allowed between adjacent peaks, i.e., 30 percent more distance than the largest peak-to-peak distance surrounding the maximum of $A_{rel}(u)$. Next, if $u_{peak}(i) - u_{peak}(i-1) < D_{max}$, then we remove all peak indices less than i if $i < i_{max}$, or we remove all peak indices greater than $i - 1$ if $i - 1 > i_{max}$. This strategy enables us to not only eliminate false isolated peaks but to remove whole sections of the freeway that might be traveling in a different direction, e.g., the left set of peaks in Figure 53.

3.3.4 Estimating the lane boundaries

It is straightforward to interpolate the lane boundaries from the activity peaks. However, it is more accurate to locate the valleys in the $A_{rel}(u)$ signal, if possible. To do this, we first select a template from the A_{bot}' signal centered about the maximum of A_{bot}' with width W_{temp} as found above. Next, we apply the same correlation coefficient template matching procedure described above in Section 3.3.2 to estimate the valley locations from the signal A_{bot}'. We use a slightly lower matching threshold of 0.6 because A_{bot}' is less uniform than A_{top}', and we now have the peak locations for A_{top}', which we use to guide our search as follows. We keep only valleys that are found in the range $u \in [u_{peak}(1), u_{peak}(N_{peak})]$, where N_{peak} is the number of peaks previously found. We also retain only the largest peak of A_{bot}' if multiple A_{bot}' peaks lie between peaks of A_{top}'. We interpolate the valley locations using the A_{top}' peaks for any valleys that are missing. Finally, we interpolate the rightmost and leftmost lane boundaries using the two rightmost and leftmost peaks of A_{top}', respectively. The result is the lane boundary estimates $u_{bound}(i)$, $i \in \{1, 2, \ldots, N_{peak} + 1\}$. These are illustrated in figures 59 and 60 for the example scene.

One should note that we have just described how to estimate the location of the boundaries that separate the different regions of activity in the activity map. In a sense, the outside lane boundary lines can provide a rough estimate of b_1 and b_2, the intersection of the road boundaries with the u-axis. Figure 60 shows that the estimated lane boundaries happen to be reasonably close to the true values of b_1 and b_2 in this particular case. In general, however, these boundary locations are different than the location of the lane markers on the road because the vehicles have a nonzero height. For example, note that the bus in Figure 60 will contribute more pixels of activity to the rightmost lane than to its own, illustrating the rightward bias present in this particular image. As we will show later, this bias makes the locations unusable for camera calibration, but the vanishing point estimate remains unaffected by the finite vehicle height.

Now that we have estimates of the road boundaries, it is convenient to record the left and right boundaries, u_{left} and u_{right}, on the bottom row of the image outside of which we do not expect to find any features for that section of the road. Later on, these boundaries will serve to constrain our search for the precise locations of the boundary lines painted on the road. If the initial estimates of the left and right road boundary positions are $u_{bound}(1)$ and $u_{bound}(N_{peak}+1)$, respectively, then we assign u_{left} and u_{right} with a full or half-lane margin of safety as follows, depending on the vanishing point coordinate u_0:

$$\begin{aligned} u_{left} &= \begin{cases} u_{bound}(1) - \left(u_{bound}(2) - u_{bound}(1)\right), & u0 < 0 \\ u_{bound}(1) - \left(u_{bound}(2) - u_{bound}(1)\right)/2, & u_0 > 0 \end{cases} \\ u_{right} &= \begin{cases} u_{bound}\left(N_{peak}+1\right) + \left(u_{bound}\left(N_{peak}+1\right) - u_{bound}\left(N_{peak}\right)\right)/2, & u0 < 0 \\ u_{bound}\left(N_{peak}+1\right) + \left(u_{bound}\left(N_{peak}+1\right) - u_{bound}\left(N_{peak}\right)\right), & u_0 > 0 \end{cases} \end{aligned} \tag{66}$$

In other words, if $u_0 < 0$, then we expect the activity map to be biased to the right, and therefore we need to compensate u_1 more than $u_{bound}(N_{peak}+1)$, with the opposite logic when $u_0 > 0$. Estimating u_{left} and u_{right} can noticeably reduce the search space when locating the road boundary lines while increasing the accuracy of the estimates by eliminating distracting image features outside the active portion of the road.

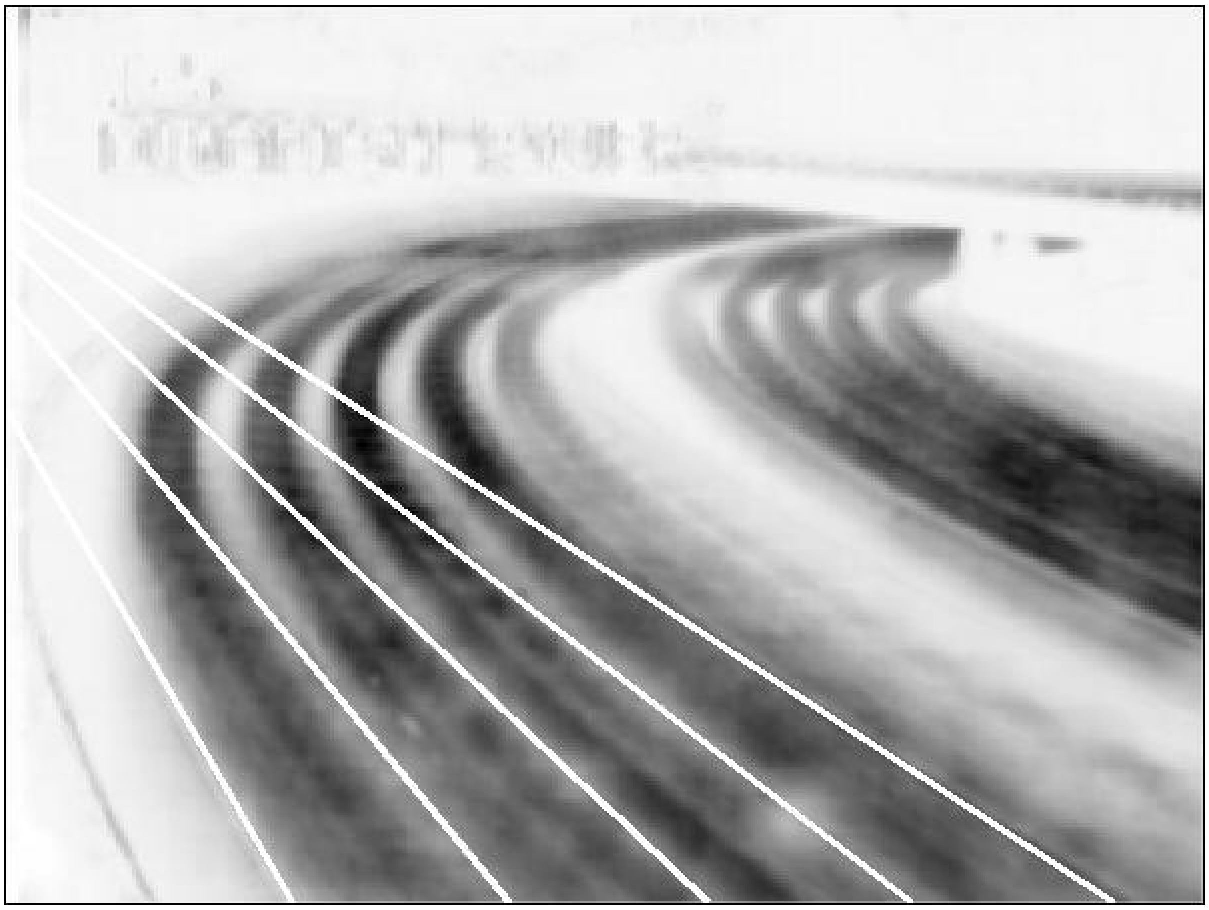


Figure 59. Lane boundaries found with the proposed algorithm superimposed on the activity map.

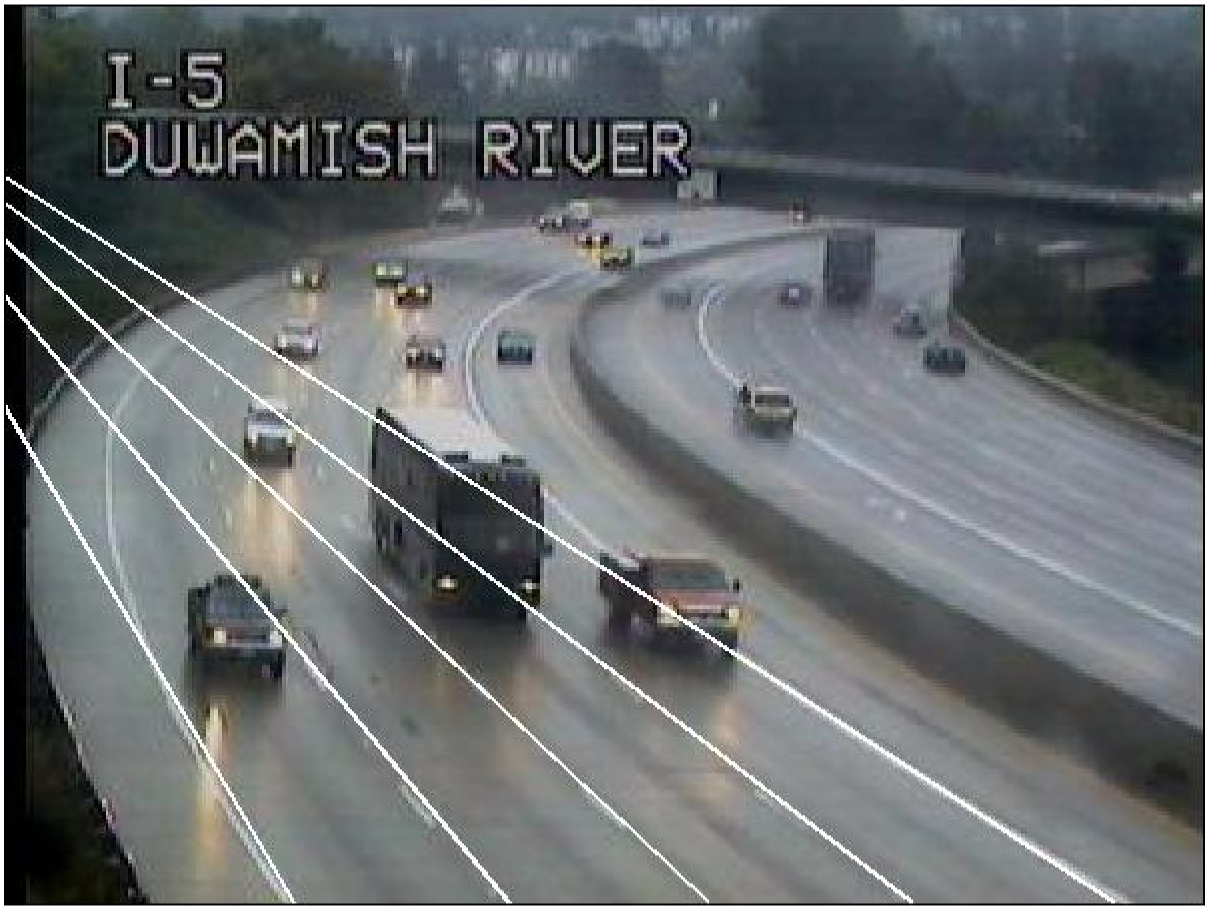


Figure 60. Lane boundaries found with the proposed algorithm superimposed on an image of the scene.

3.3.5 Estimating the lane masks

We use the procedure described in Sections 3.3.1-3.3.4 to estimate the boundaries of the activity along the bottom row of the image. The lane masks are generated simply by forming a triangle by connecting the points ($u_{bound}(i)$,-H/2), (u_0,v_0), and ($u_{bound}(i + 1)$,-H/2) and filling in the pixels within the triangle.

We note that because the algorithm selects the highest peak in $A_{rel}(u)$, it automatically picks the set of traffic lanes with the most activity. On the other hand, it is certainly possible that lanes with traffic in opposite directions could lie so close together that the algorithm would consider them to be part of the same set of lanes. However, our tracking results will easily allow us to discriminate the different traffic directions as long as we properly extract the lane mask. We also note that, as we shall see later on, the object tracker does not require highly accurate lane masks, so we need not perform an error analysis.

3.3.6 Performance and limitations

Our algorithm for lane segmentation was designed in anticipation of the worst possible conditions. We have tested it on over 20 sequences and have found it to perform well even in dark or rainy conditions, as long as no raindrops are on the camera lens. It also works fine at night as long as traffic is reasonably dense and there are lampposts to help illuminate the scene. It is certainly not a limiting factor in the system; the camera calibration procedures own this distinction. In fact, if we simply require the average speed for the road, then even one lane of data can potentially provide a reasonable estimate of the average speed. However, if we desire individual estimates for each lane of traffic, then we must note that the carpool lane may not be detected by our algorithm because of the low level of traffic activity.

The one limitation of the algorithm is that it performs best when the traffic flows toward or away from the camera. As the pan angle θ increases beyond about 20° while ϕ remains less than about 10°, the algorithm performs increasingly worse until it no longer can distinguish the lanes. This is due to the severe overlap of the vehicles into the next lane because of their nonzero height. These problems are mitigated when the camera is zoomed in (f greater than usual) or if ϕ increases beyond 15°.

3.4 Detecting bad activity maps

As seen previously, the activity map is essential for estimating the vanishing point and determining the location of the lane boundaries. A faulty activity map invalidates the entire process of Figure 45. Thus, it is critical to determine the validity of the activity map before accepting the image processing results obtained through the algorithm of Figure 45. Many variables affect the position and quality of the scene. Obstacles such as road signs or overpasses can obstruct the camera's view of the vehicles. Raindrops on the camera lens noticeably affect the image quality and therefore the ability of our algorithms to extract image features and calibrate the camera. Shadows from nearby trees often produce image artifacts (i.e., false edges) that may affect the vehicle tracking results. Finally, although the road is usually approximately straight in the bottom one-third of the image, occasionally it can deviate from this assumption. This can be particularly troublesome for our algorithm that estimates the vanishing point from the activity map when the lanes are very narrow and the vanishing point is high above the image. We give examples of some of these conditions in Figure 61. Any of these conditions affect the activity map and can therefore bias the estimate of the vanishing point.

Based on Figure 45, we again note that much of the system depends on results from the activity map and, therefore, it behooves us to prematurely cease to process the image sequence if we detect untoward conditions within the activity map. We now present an algorithm that can detect each of these conditions if they are present in the activity map. As a side benefit, it also alerts the user if the vanishing point is inconsistent with the trends in the activity map.

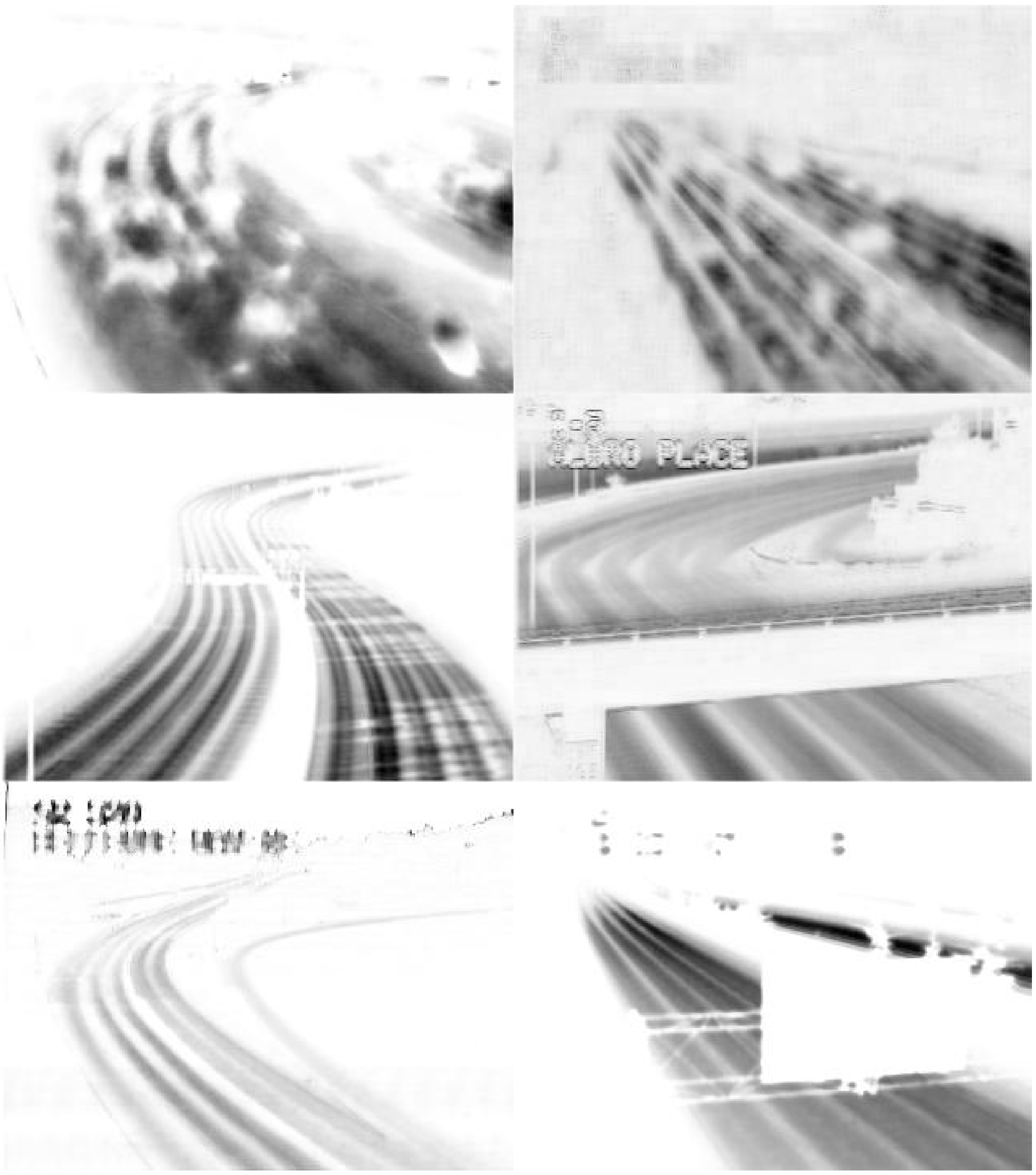


Figure 61. Examples of bad activity maps. In order left to right, top to bottom:
a) Raindrops on camera lens. b) Raindrops on camera lens. c) Shadows on the road. d) Overpass blocking the road. e) High vanishing point and narrow lanes with curved road. f) Road sign blocking the road.

The basic principle in detecting poor activity maps is to find irregularities in the activity along the longitudinal direction of the road. Each of the scenes in Figure 61 possesses this property. As described in Section 3.3, we linearly sample the activity map along lines emanating from the vanishing point to generate the signal $A_{rel}(u)$. During this sampling process, we also take order statistics when we estimate the mean activity. Specifically, we sort the data samples along each line and record the fifth-lowest value and the maximum. We do not record the lowest value because there may be artifacts along the bottom of the image. Next, we set portions of the signal outside the detected road boundaries to zero. We normalize all the signals by the maximum of A_{rel}. Figure 62 contains the relevant portions of the A_{max}, A_{rel}, and A_{min} signals for the rainy example of Figure 61(b). In this particular case, we want to design an algorithm to detect the raindrop evidenced by the low values of A_{min} on the interval $u \in [250,275]$.

In general, we expect the valleys of A_{rel} to contain little activity, which means the difference between A_{max} and A_{min} will be small. As a result, we eliminate these regions as unreliable for determining where activity map anomalies, e.g., raindrops, are present. We morphologically close [45] A_{rel} and identify where closing(A_{rel}) = A_{rel}. These regions are highlighted in bold in the example of Figure 62. The kernel size is chosen as half of the smallest peak-to-peak distance found by estimating the lane boundaries, i.e., 16 pixels in the current example. Within the regions found, we search for samples where $A_{max}(u) > 0.5 \cdot \max(A_{max}(u))$ and $A_{min}(u) < 0.25 \cdot \max(A_{max}(u))$. For the example in Figure 62, the anomalies are clearly evident by observing where A_{min} dips below its threshold and then observing that the left-hand portions that satisfy this criterion have a very low value of A_{max} that disqualifies them from being labeled as an anomaly. The thresholds were chosen on the basis of our experience with more than 20 activity maps, of which half contained anomalies we wanted to detect.

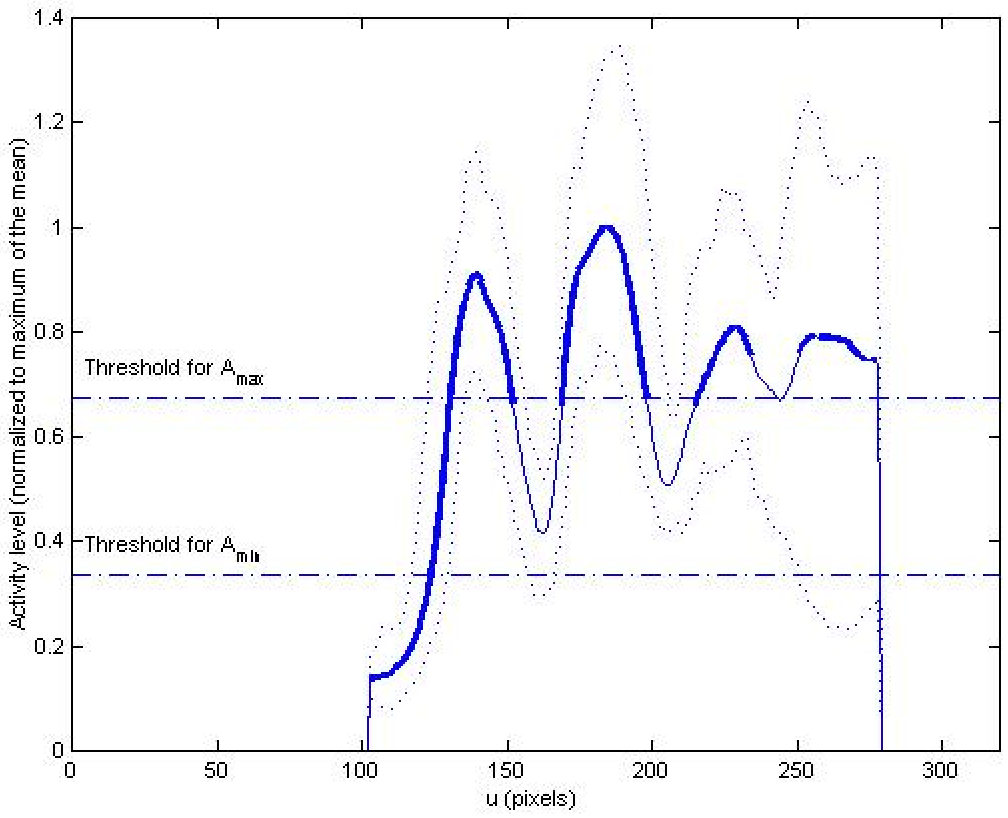


Figure 62. Line sampling maximum, mean, and minimum for the activity map in Figure 61(b) and the thresholds used to identify the anomalies. The bold portion indicates the valid regions for detection.

3.5 The top-hat image

The top-hat image is one of the fundamental feature images found by the framework of Figure 45. It is used to highlight the road boundaries and lane markers in the image sequence. As shown in Figure 45, we use this feature image to obtain the quantities b_1, b_2, and τ_L, each of which is essential for one or more of the three camera calibration methods presented in Chapter 2. For example, Figure 63(a) contains a 640 X 480 image of a night-time scene that has been log-transformed so it can be visualized. One can see some of the road lines and faint indications of lane markers, but only with much effort. The two-dimensional morphological top-hat [45] extracts any portions of the image that are peaked relative to their surroundings. For example, note that in Figure 55, the peak at u = 35 has a much greater magnitude in the A_{top} signal relative to the other peaks than it did in the A_{rel}

signal. The two-dimensional top-hat operator has an analogous effect on images, as seen in Figure 63(b), where the road lines and lane markers are now quite visible. In our experience, kernel sizes of 5 and 3 seem to work well for image sizes of 640 X 480 and 320 X 240, respectively. The kernel size determines the maximum size of image features that are detected by the top-hat transform; road lines that are wider than this may not appear in the output image.

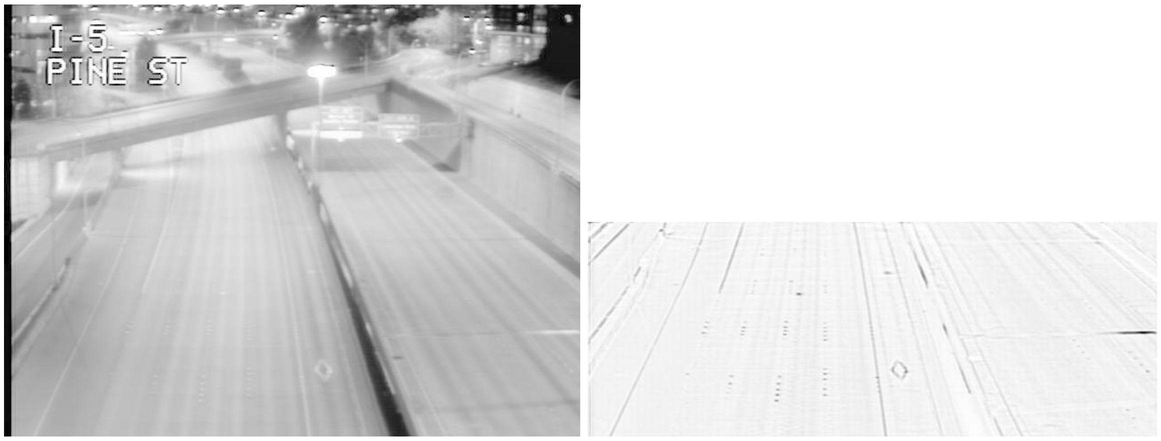


Figure 63. a) Log-stretched image of a night-time scene. b) The top-hat image of the original scene (linear scaling).

At least two approaches exist for estimating the top-hat image, I_{top}, of a scene. 1) Estimate the background image I_{bg} = average(I_i) from all images I_i in the sequence and then let I_{top} = tophat(I_{bg}) = tophat(average(I_i)). 2) Compute the top-hat image for all images in the sequence and then compute the average top-hat image, i.e., I_{top} = average(tophat(I_i)). The system is not a linear one, so the approaches are not equivalent, i.e., the average(·) and tophat(·) operators do not commute. It turns out that for large sample sizes, it is best to use the power of averaging in the second method to reduce the noise in the top-hat operator. Thus, for each of the 1,000 RGB input images (about 8.5 minutes at the anticipated 2 frames/second rate for 640 X 480 images), we do the following. 1) Calculate the intensity by averaging the color components for each pixel (I = R + G + B). 2) Perform 3 X 3 median filtering over the entire image to remove JPEG artifacts and remove noise. 3) Apply the top-hat operator to obtain the i^{th} top-hat image estimate $I_{top,i}$. Lastly, estimate I_{top} by averaging the set of top-hat images, yielding a result like Figure 63(b).

Note that the single estimate of the top-hat image in Figure 63(b) from the background image is reasonably good even in a night-time scene illuminated by roadside lamps. Thus, we conclude that our algorithm should perform fine in the light conditions at early morning and dusk, even if no roadside lamps are present.

3.6 Refining the estimates of u_0, v_0, b_1, and b_2

As we noted in our sensitivity analysis, accurate estimates for u_0, v_0, b_1, and b_2 are essential for each of the calibration methods. Continuing with our exposition of Figure 45, we now show how to refine the rough estimates of u_0, v_0, b_1, and b_2 obtained from the activity map using the methods of sections 3.2 and 3.3.4. Given the low quality of the image in Figure 54(a), it is surprising that we achieve the accuracy evident in Figure 60. However, once the top-hat image I_{top} is available, we can expect even more accurate estimates of these four parameters using the road boundaries themselves. As seen in Figure 64, the road lane markers and boundaries stand out very clearly in the top-hat image for this scene, despite the rainy conditions.

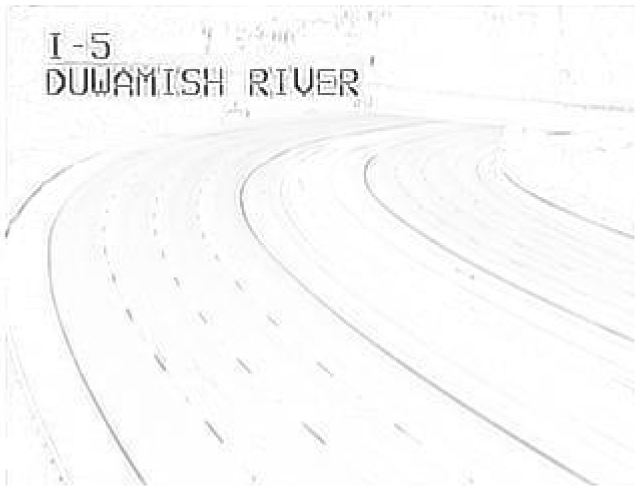


Figure 64. Top-hat image for the scene of Figure 60.

We begin refining our estimates by locating the intersections of the road boundary lines with the bottom of the image. To locate the lines, we average the line samples of I_{top} emanating from the vanishing point in the same way used for sampling the activity map in Section 3.3. We use the bottom one-fourth of the image because it is least affected by road curvature. This results in a signal I_{spike}' containing multiple spikes at the intersections of the road lines with the bottom of the image, e.g., Figure 65. We smooth I_{spike}' with a three-point boxcar filter to obtain I_{spike}. Next, we use the kernel [-1 1] to determine the sign of the difference between the samples of I_{spike}. The values of u where this signal changes sign are recorded, along with the corresponding values of I_{spike}. The boxcar filter helps to reduce the noise when we extract the spike locations.

Some care must be taken for cases in which v_0 is large, i.e., $v_0 > H$. In this case, the road lines may exit the side of the image rather than the bottom, or they may enter the side of the image instead of from the top. Thus, we set wider limits on u_{int} in the line sampling so that the intersection of the line with the edges of the image occurs three-fourths of the way down the image (or more). Furthermore, we process the entire image because the road boundaries may be primarily present in either the upper or lower half of the image.

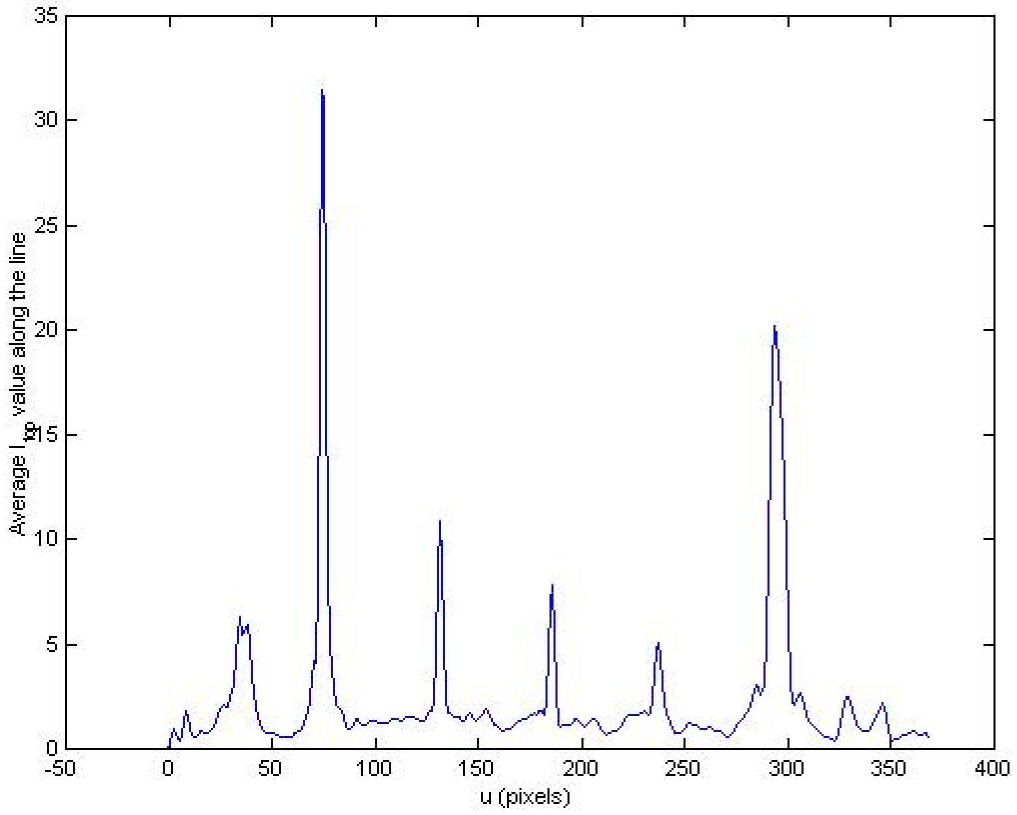


Figure 65. Average value of the line samples emanating from the vanishing point in the top-hat image (after smoothing with a three-point boxcar filter).

After the candidate locations have been found, we wish to determine which of the spikes correspond to the left and right boundaries of the road. We assume that the white road boundaries will have the strongest I_{spike} response, so we assign the spike locations $u_{spike,left}$

and $u_{spike,right}$ by finding the maximum spike within the ranges $u \in [u_{left},(u_{left}+u_{right})/2]$ and $u \in [(u_{left}+u_{right})/2,u_{right}]$, respectively, where u_{left} and u_{right} are as shown in Section 3.3.4. We can then obtain initial estimates of b_1 using the following formula:

$$b_1 = u_0 + \frac{v_0\left(u_{spike,left} - u_0\right)}{v_0 - \left(-H/2\right)} \tag{67}$$

with a similar result for b_2 and $u_{spike,right}$.

Once we have initial estimates for u_0, v_0, b_1, and b_2, we use the BFGS quasi-Newton method [46][47][48][49] to determine the four-parameter vector that maximizes the mean value of the samples along the two road lines. In our objective function, we use the four parameters to obtain samples along each line within the lower one-fourth of the image. We obtain the samples using bilinear interpolation of I_{top}. We use a stopping criterion tolerance of 0.0001 on the objective function and 0.01 on the parameter values so that we guarantee a minimum precision of 0.01 in the parameter values.

We begin by processing the bottom fourth of the image to handle curved roads. However, we also wish to fully utilize the additional information farther up in the image if the road is straight. Thus, we employ a tiered approach as follows. In the first stage, we obtain the optimal parameters using only the bottom one-fourth of the image. If our algorithm determines that it has successfully located both lines in this portion of the image, the process is repeated for the bottom one-third of the image, using the results of the previous stage to initialize the next optimization. We expect to obtain a more accurate result by measuring longer lines. After completing the search, the same criteria for success are applied, and if the search is again successful the process is repeated a final time for the bottom one-half of the image.

For the case in which $v_0 > H$, we do not use the tiered approach. Instead, we only search the bottom half of the image for two reasons. 1) It contains the lines that are the most straight, and including the upper half may cause the success criteria below to fail because of road curvature. 2) We allow the lines to exit and enter the sides of the image, and therefore, the bottom fourth or third of the image may provide an inadequate amount of information to locate the lines.

The criteria for a successful line search stage are as follows. First, we define the bottom half of the I_{top} image as I_{top2}. We normalize I_{top2} by its maximum and perform histogram equalization with 100 levels of precision. Next, we binarize I_{top2} using the 90th percentile as a threshold. Finally, we sample this binary image along the road boundary lines found from the previous stage using bilinear interpolation to obtain signals B_{L1} and B_{L2}. We require 50 percent of the samples on both lines to be within the quarter-image window for the first stage, and 60 percent of the samples to be within the image window for the second and third stages. If these criteria are met, then we compute the mean values of B_{L1} and B_{L2}, each of which may be somewhat shorter than the height of the image window. If both of the mean values are 0.85 or greater, then we consider the search to be successful. In other words, we require about 85 percent of the line pixels in the image window to have a value of at least the 90th percentile for I_{top2}.

Figure 66 contains the results for the example scene. There the changes in the vanishing point estimate are noticeable in the upper portion of the image, though the two lines are nearly indistinguishable in the lower portion. The vanishing point moved from (-218.2,122.6) to (-209.9,100.1). This large change is atypical for most scenes, particularly those where u_0 is smaller in magnitude and where the road is not curved.

3.6.1 *Error estimates for* u_0, v_0, b_1, *and* b_2

Because of the single-shot character of the parameter estimation, it is difficult to get a grasp of the errors in u_0, v_0, b_1, and b_2. We offer the following general approach: vary the individual parameters until the mean value along the line deviates by more than 10 percent from its original value. This then becomes our 95 percent confidence interval for that particular parameter, since we have the goal of a relative standard deviation of 0.1 for camera calibration Method 2. We search on either side of the nominal parameter value using the Newton method search algorithms in [50][51].

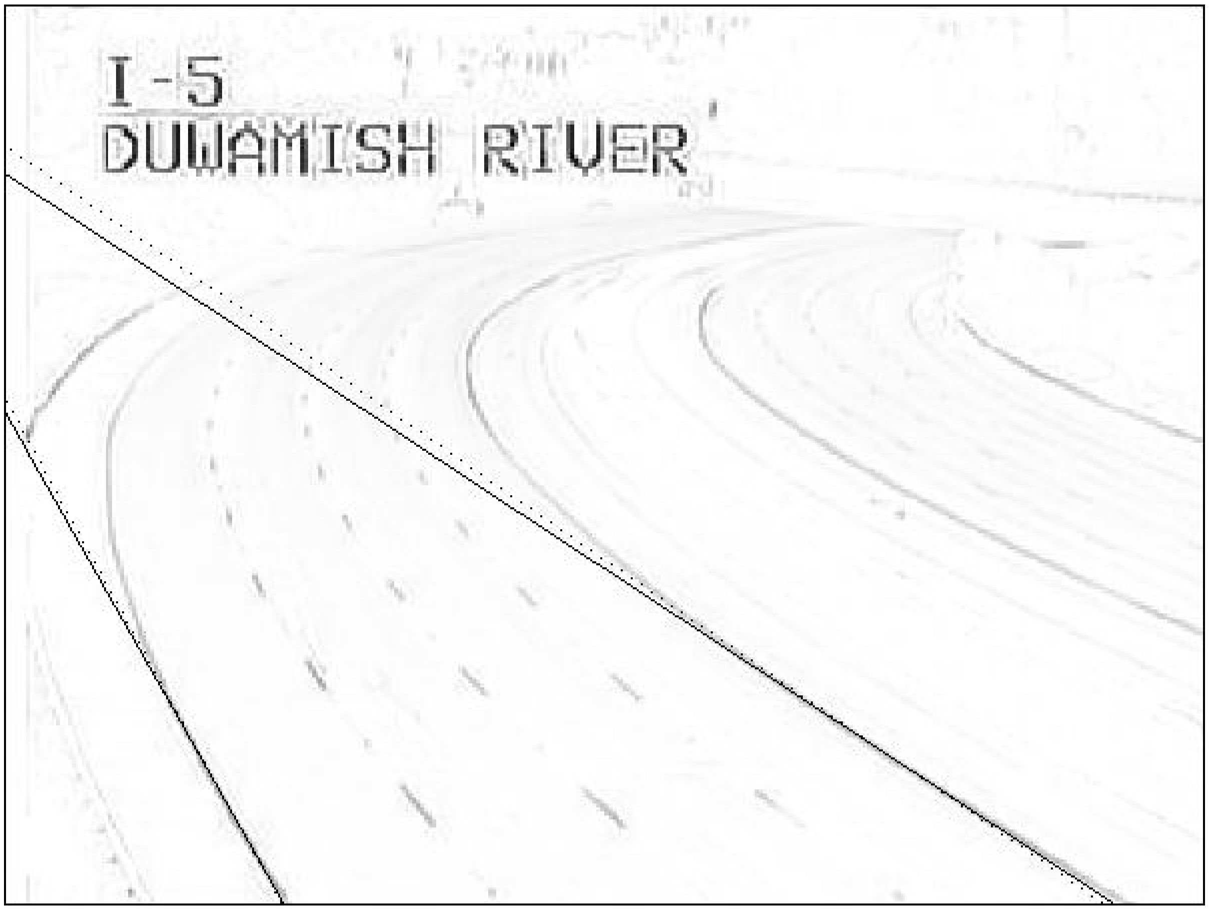


Figure 66. Top-hat image with initial estimate (dotted line) and final estimate (solid) of the road lines used to estimate u_0, v_0, b_1, and b_2 in the example scene.

3.7 Estimating the lane marker interval

Referring to Figure 45, now that we have estimates of u_0, v_0, the lane boundaries, and the top-hat feature image, we can search for the lane markers and estimate the tip-to-tip distance between consecutive lane markers. As discussed in Chapter 2, we encode this distance from the image as the parameter τ_L, which is critical to the success of Method 3.

To this point, when sampling along the lines emanating from the vanishing point, we have only used the statistical properties of the samples, e.g., the mean or maximum.

In many cases, the lane markers may not be very visible. We must look closely at the top-hat image of the night-time scene of Figure 63 in order to see the turtles (little bumps placed by the DOT) that are visible in the lower half of this 640 X 480 image, e.g., Figure 67. Sampling along one of the lines emanating from the vanishing point reveals a strong pattern in the turtles, as evidenced in Figure 68. We define the signal $T_1(u,v)$ along any line as a function of the *u*-coordinate of the intercept of the line with the bottom of the image and the *v*-coordinate of the pixel samples within the image.

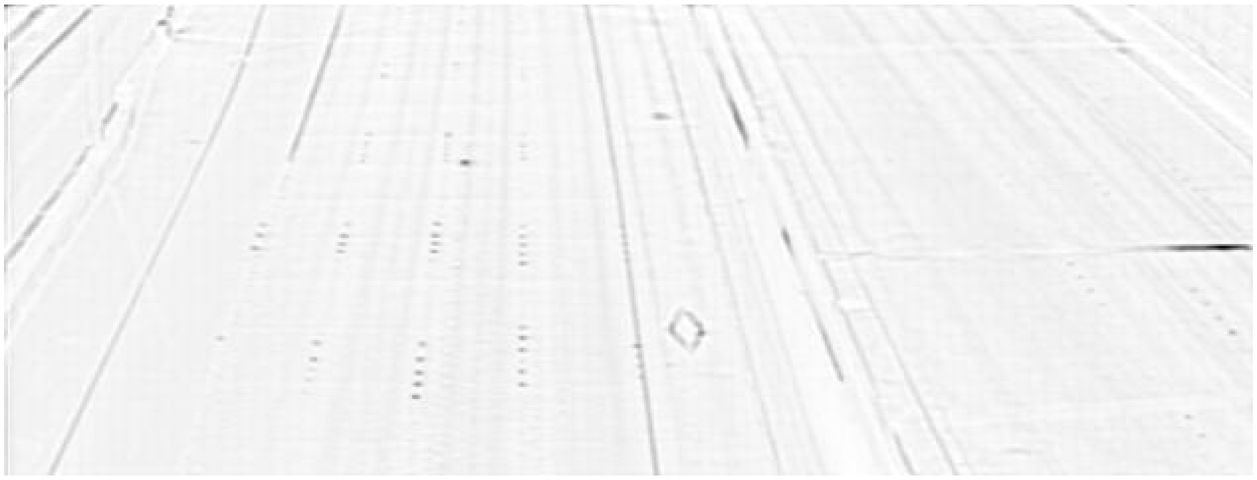


Figure 67. Bottom half of a top-hat image of a dark scene containing turtles as lane markers.

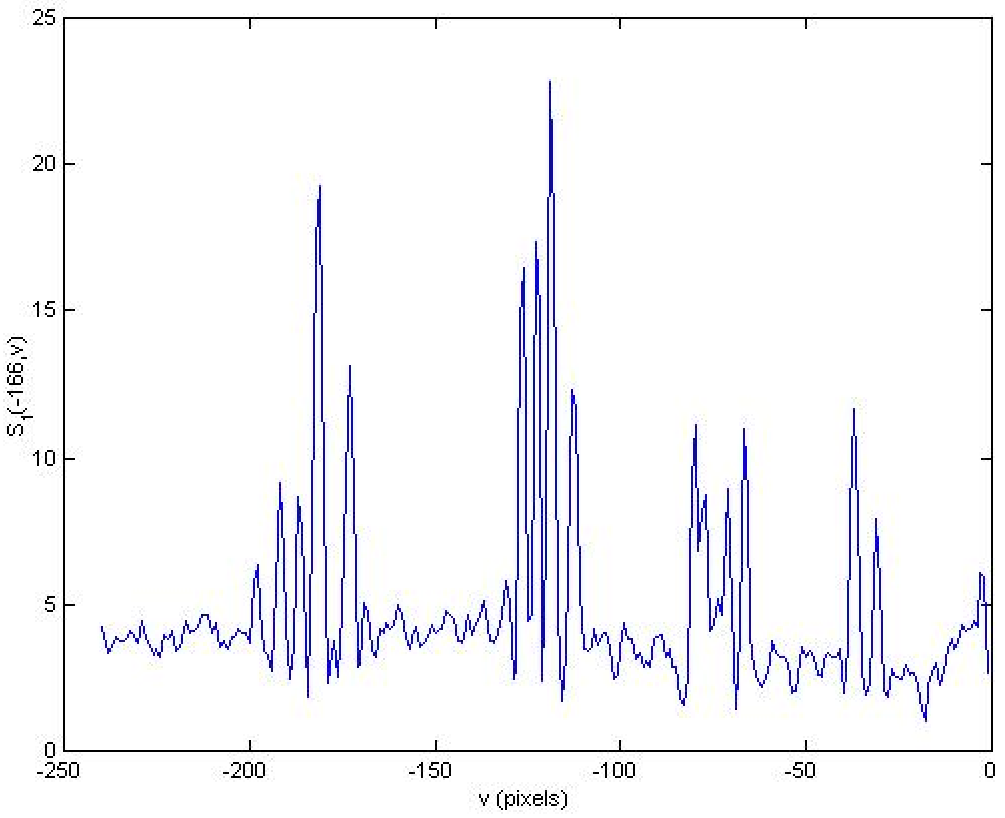


Figure 68. Values of I_{top} sampled along the lane markers second from the left in Figure 67 using image-centered coordinates.

3.7.1 *Linear resampling of the* $T_1(u,v)$ *signal*

Though the lane marker pattern is quite clear in Figure 68, it is also evident that the distance between markers is nonlinear, increasing as *v* increases. We now recall our simplest position formula that relates the vertical image coordinate, *v*, to the position *Y’* along the road.

$$Y' = S'\left(\frac{v}{v_0 - v}\right) \tag{68}$$

Using Equation (43) with *S’* selected arbitrarily, we can use our theoretical model of Chapter 2 and our knowledge of v_0 to remove the nonlinearity in the *v*-scaling of the line sample, e.g., Figure 69.

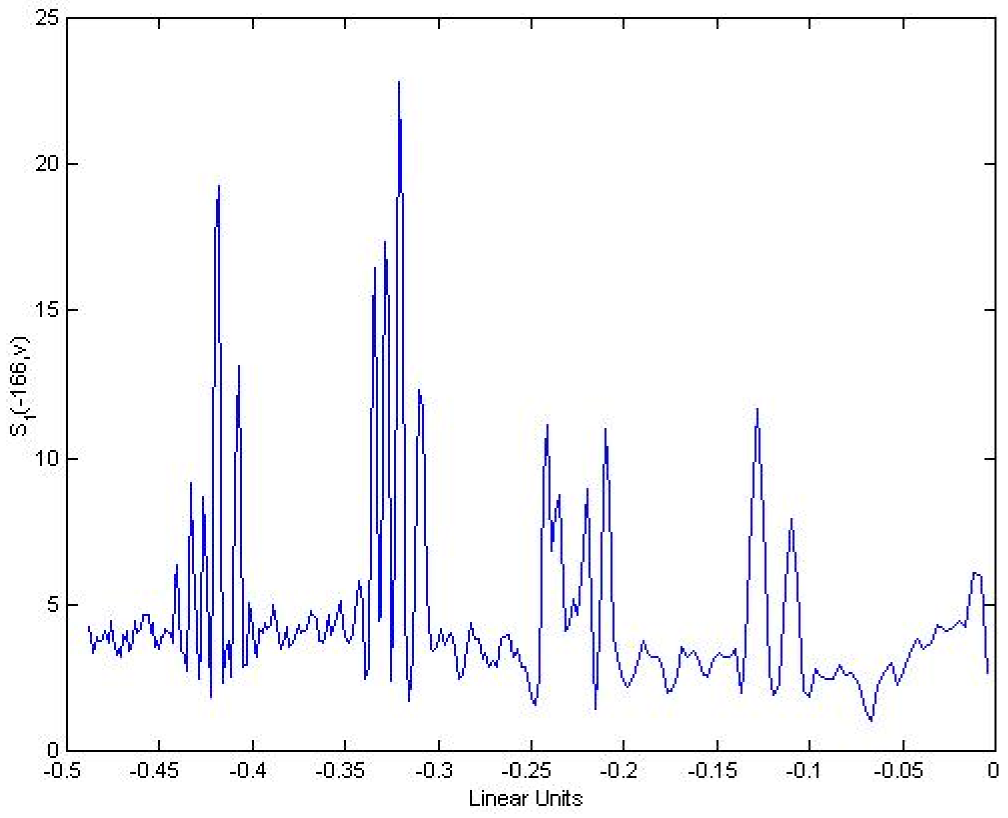


Figure 69. The signal of Figure 68 with the nonlinear scaling removed (*S'* = 1).

In essence, we have simply changed the linear sampling in *v* into nonlinear sampling in the new domain *Y'*, where we expect to perform much of our signal processing. In order to process the signal in the *Y'* domain, we must resample the signal T_1 to obtain a signal T_2 that has linear spacing in the *Y'* domain. We use linear interpolation with an upsampling factor that depends on the value of v_0. Upsampling is necessary in order to preserve the signal detail when $(v_0 - v)$ is large in magnitude, i.e., at the bottom of the image. Thus, the smallest increment of *Y'* is

$$\begin{aligned}\delta Y' &= S'\left(\frac{-H/2+1}{v_0-(-H/2+1)}-\frac{-H/2}{v_0-(-H/2)}\right)\\ &= S'\left(\frac{v_0}{(v_0-(-H/2+1))(v_0-(-H/2))}\right)\end{aligned} \tag{69}$$

If v_{top} is the uppermost v-coordinate that we will sample in the image, then we will need at least

$$N_{samp,\min} = \frac{1}{\delta Y'}\left(S'\left(\frac{v_{top}}{v_0 - v_{top}} - \frac{(-H/2)}{v_0 - (-H/2)} \right)\right) \tag{70}$$

data points in T_2 to avoid losing any details when v is near $-H/2$. We recommend a value of two to three times this theoretical minimum to enable us to measure the peak-to-peak distances with greater precision.

3.7.2 Estimating the distance between lane markers

Having described how to generate T_2, we now describe how to estimate the distance between lane markers, assuming T_2 contains lane markers. Using one of the most standard signal processing techniques, we generate the autocovariance sequence $R(\tau)$ for T_2 and measure the location of the largest peak away from $\tau = 0$ that we find. Figure 70 contains $R(\tau)$ for the example we have been using. This particular signal has very obvious and sharp peaks at $\tau = 0$, $\tau = 284$, and $\tau = 571$. Observing that $2 \cdot 284 = 568$, we see that there is a great deal of periodic structure to this particular instance of T_2. This value represents 0.0988 linear units in Figure 69.

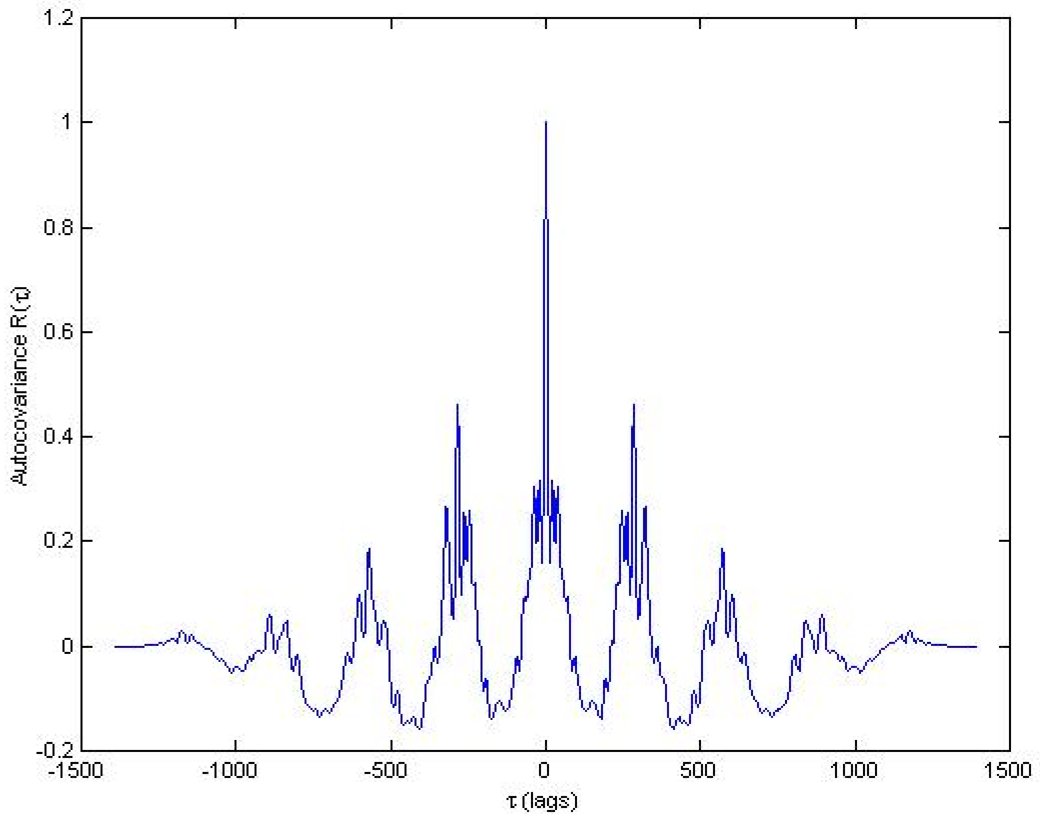


Figure 70. Autocovariance sequence for the signal of Figure 69.

3.7.3 Relating the autocovariance technique and the camera calibration

The connection between this approach for measuring the distance between lane markers and the third camera calibration method is quite startling. We recall from Chapter 2 the equation for the scale factor *S'* in Equation (43)

$$S' \equiv \frac{h}{f}\csc(\phi)\frac{\sec(\theta)}{\sin(\phi)}v_0 = \frac{Sv_0\sec(\theta)}{\sin(\phi)} = L\frac{(v_0-v_a)(v_0-v_b)}{v_0(v_b-v_a)} = \frac{L}{\tau_L} \tag{71}$$

where we had defined

$$\tau_L \equiv \frac{v_b}{v_0-v_b}-\frac{v_a}{v_0-v_a} \tag{72}$$

We can therefore see that the autocovariance procedure is directly measuring the quantity τ_L to within a scale factor (dependent on the upsampling factor above) without identifying the quantities v_a and v_b. Furthermore, if we measure the linear distance τ_{12} between two vertical image levels v_1 and v_2 with the same upsampling factor, then our estimate of the physical distance along the road is simply

$$\Delta Y' = L\frac{\tau_{12}}{\tau_L} \tag{73}$$

where L is the known distance between lane markers. We see that the upsampling factors of τ_{12} and τ_L cancel, and the result is scaled to a physical distance by L. To convert τ_{lag} measured in the autocovariance function to the pixel ratio τ_L, we simply divide τ_{lag} by the upsampling factor (and by S' if we assumed $S' \neq 1$ in the interpolation process).

3.7.4 *Increasing the robustness of the method*

Even though the image in Figure 67 was estimated from a night-time scene, it is quite clean and rather idealized because street lamps are present. If we wish to use this approach as the primary technique for calibrating the camera, then we must make each step robust in our algorithm above.

3.7.4.1 Theoretical model

In the discussion above, we proved that our method has merit using a pragmatic approach and forming a connection with our camera calibration methodology. We now offer some theory to bolster our broad claims for the autocovariance method in Section 3.7.2. In the Northwest region of the United States [42][53], the lane markers are specified to be 10 feet long with 30 feet of blank pavement in between, for a head-to-head spacing of 40 feet. Although two styles of markers exist, i.e., turtles and painted lines, we will develop our theory for the case of painted lines for the sake of simplicity. We can model this situation as a cyclical rectangular-wave signal along the axis of the road with a value of 1 for a fraction of the period (i.e., the duty cycle) and a value of 0 for the rest of the cycle. For the example above, the duty cycle is 12/40 = 0.3, i.e., we expect the signal to have a relative value of 1 for 30 percent of the cycle and a value of 0 for 70 percent of the cycle.

We now develop an intuitive sense of how the autocovariance function, $C_{xx}(\tau)$, will behave for a finite multi-cycle square wave. Figure 71(a) contains an example of a finite square wave containing many cycles. We use an amplitude 0.5, 100-sample, 50 percent duty pcycle square wave for simplicity and ease of presentation. As shown in Figure 71(b), the autocovariance function of any finite square wave has a triangular window superimposed upon it because of the finite length of the correlation. Within the triangular window, the autocovariance function is composed of many identical saw-tooth functions with the same period as the square wave (e.g., 100 samples), as shown in the close-up view of Figure 71(c). We can also see in Figure 71(c) that when the square-wave function (nearly) completely overlaps itself, the response is a nonattenuated triangle wave. For lags of even multiples of 50, the square-wave is completely correlated with itself, and at lags of odd multiples of 50, it is completely anticorrelated with itself.

In the actual road scenes, we typically expect two or three cycles of lane markers, with six to eight at the most. Thus, the effect of the triangular window will be quite strong, as shown in the autocovariance function for a three-cycle square wave in Figure 71(d). The shape of each peak within the autocovariance function will also be somewhat distorted when the duty cycle is not 50 percent. The effect of a smaller (or larger) duty cycle is to contract the triangular portion of the saw-tooth wave to the duty cycle fraction, as shown in Figure 71(e) for a 10 percent duty cycle wave with a period of 100 samples. However, we note that a duty cycle other than 50 percent should not affect our ability to extract the peaks in $C_{xx}(\tau)$, even if they are compressed relative to their original width.

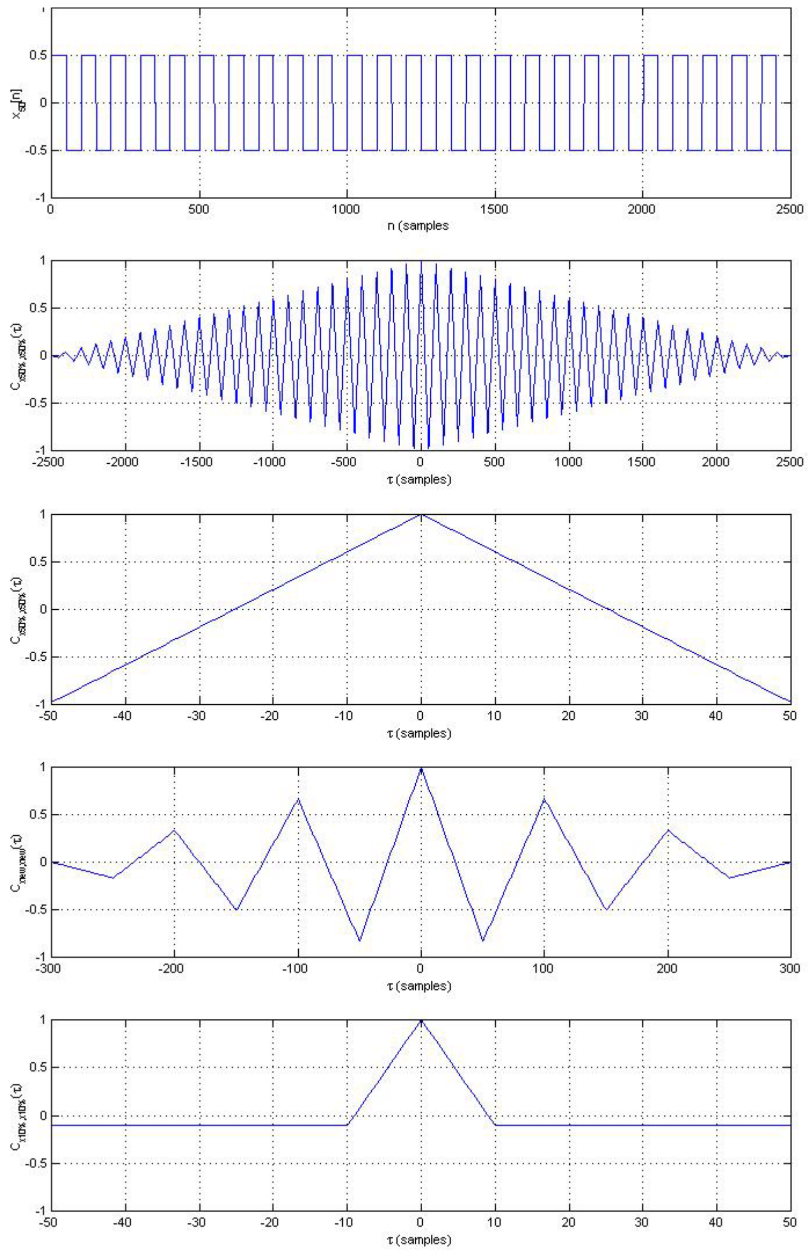


Figure 71. Behavior of the autocovariance function for a square wave. a) Finite square wave with 50 percent duty cycle and period of 100 samples. b) Autocovariance function for the square wave. c) Close-up of the autocovariance function showing the nearly ideal response for a single cycle. d) Autocovariance function for a three-cycle 50 percent duty cycle square wave with a period of 100 sample. e) Close-up showing the nearly ideal, single-cycle response for a 10 percent duty cycle square wave.

3.7.4.2 Image processing for more robustness

As we indicated earlier, it is easier to locate the pattern of the painted stripes than the turtles in the images. The signal obtained from the example of Figure 67 was a nearly ideal case. In reality, we can expect some of the turtles to be wholly or partially missing, which will affect the accuracy of the autocovariance method. Irregular turtle spacing can also cause errors, particularly because the turtles occupy so little space. We also note that even when using images of 640 X 480, only a small number of the vector samples from Figure 67 will contain a signal appropriate for estimating the lane marker spacing. To emphasize these features as much as possible, we normalize I_{top} by its maximum value and applying a gamma correction of 0.2. This boosts the middle and lower values. Mathematically, $I_{top,new} = (I_{top}/\max(I_{top}))^{0.2}$.

To address the errors caused by the turtles and amplify the number of useable line vectors, we preprocess the I_{top} image with mathematical morphology, and again apply similar techniques after the nonlinear scaling is removed before calculating $C_{xx}(\tau)$. We begin by normalizing I_{top} by its maximum values and applying a gamma correction of 0.2 to boost the middle and lower values. Mathematically, $I_{top,new} = (I_{top}/\max(I_{top}))^{0.2}$.

After the preprocessing, we obtain the image $T_1(u,v)$ by sampling I_{top} along the lines emanating from the vanishing point. As described above, we then resample each column of $T_1(u,v)$ to obtain the image $T_2(u,r)$, where even sampling of r produces linear spacing of T_2 in the Y' domain. This yields an image similar to the one contained in Figure 72(a) for the example of Figure 67. We use bilinear interpolation to ensure a reasonably accurate transformation. We generally sample the bottom one-third of the image to strike a balance between obtaining multiple stripes and maintaining their clarity. The horizontal sampling boundaries are determined either by b_1 and b_2, if these measurements are available, or else the rough estimates of the road boundary locations obtained by locating the lanes in the activity map.

Next, we apply the dilation operator from mathematical morphology [45], which assigns to a pixel the maximum value in a neighborhood surrounding that pixel. This yields the image $T_{2d}(u,r)$. To emphasize the vertical lane stripes and connect the turtles, we use a fairly tall kernel with some a small width, i.e., five pixels for an image width of 640. The small kernel width is chosen because of the high-intensity noise in the middle of the lanes, which could contaminate the periodicity of along the lane markers. The kernel height is determined by dividing the number of vertical samples in $T_2(u,r)$ by the maximum number of lane markers expected in that portion of the image, yielding the minimum number of samples per lane marker. This value is multiplied by (20 - 12)/40 = 0.2 so that we obtain a 50 percent duty cycle. In the example of Figure 72, we expected a maximum of 10 lane stripes, whereas only about five are present. Note, however, that the dilation operator connected the lane markers and extended them so the blank space and lane marker are approximately balanced in length.

Performing the dilation operation slightly widens the lane stripes, which ultimately provides more estimates of τ_L. It also merges and extends each lane marker so that it more closely matches the theoretical model of Section 3.7.4.1. For example, Figure 73 contains the signal of Figure 68 after applying all the image processing operations. The signal is much more square, and the peaks are all at even levels, making it much more amenable to the autocovariance method.

If the vanishing coordinate $v_0 > H$, then the camera has either a significant value for the down angle ϕ or the focal length f. The user is probably trying to enable the system to measure d. In this case, the image is typically occupied completely by the road, and the road lines often exit or enter the sides of the image. In such cases, because of the close-up view, the lane markers are quite clear in the image. In addition, the average top-hat image typically contains additional noise in the middle of the lanes because the road imperfections are larger. Thus, we obtain the top-hat image from the average background image (see Figure 74(a)) rather than by averaging the individual top-hat images from the sequences. We clean the top-hat image slightly by morphologically opening it with a 3 X 3 “cross” (plus sign +) kernel.

In the case of $v_0 > H$, when processing $T_2(u,r)$, we widen the morphology kernel to 15 pixels for a 640 X 480 image because the lane markers are much brighter than the lane features. This allows us to combat any slight curvature in the road or errors in the estimation of the vanishing point (see the top portion of Figure 74(b)). Since v_0 is large, we expect at most four lane markers, and we increase the vertical kernel size accordingly. Furthermore, we expand our sampling to the entire image rather than the typical bottom one-third limit. This allows us to capture more lane markers, which increases the accuracy of the autocovariance method. In addition, because the line samples may enter or exit the sides of the image, we track the beginning and ending indices of each column in $T_2(u,r)$. This allows us to shorten the sequence and thereby obtain sharper peaks in the autocovariance function.

Tracking these indices also proves useful for normal scenes where the lane markers may enter or exit the image from the side when the pan angle is large.

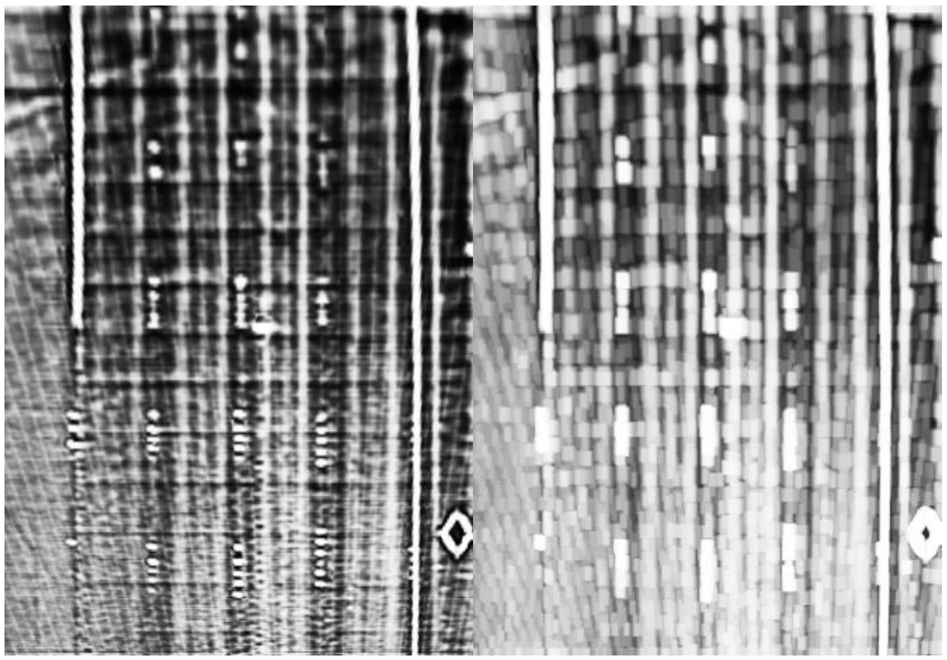


Figure 72. a) Relevant portion of the top-hat image of Figure 67 after log-stretching, histogram equalization, and linear resampling. b) Result of (a) after morphological dilation.

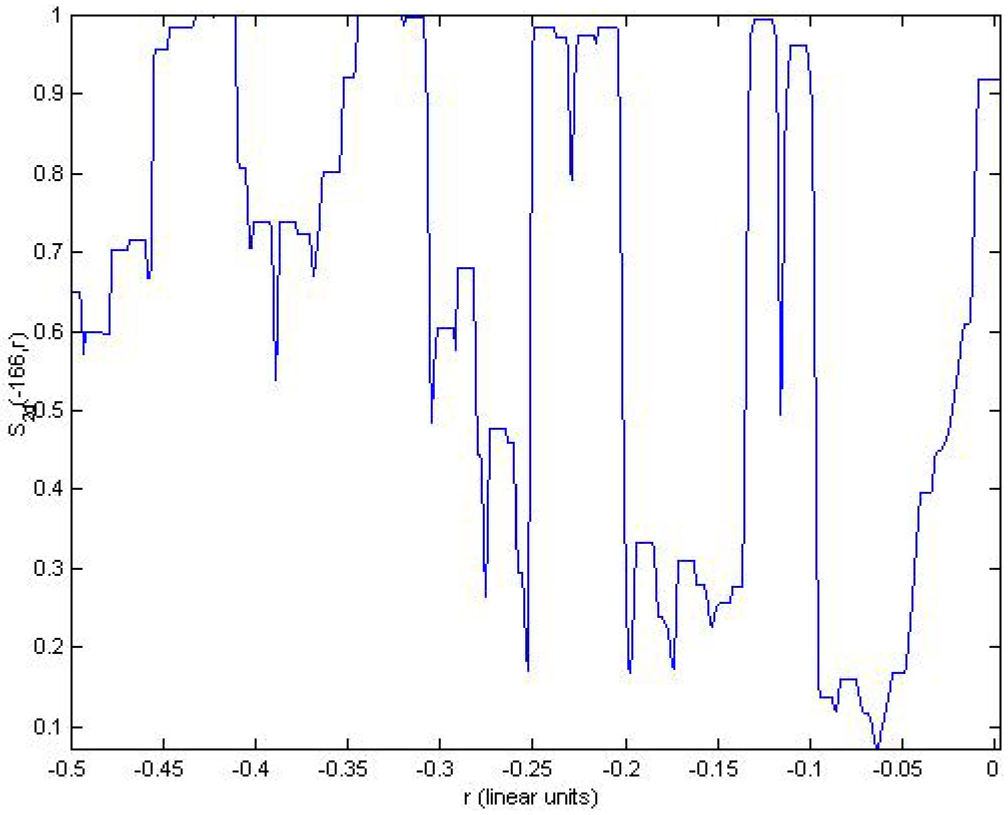


Figure 73. Line sample signal from Figure 68 after applying the image processing operations.

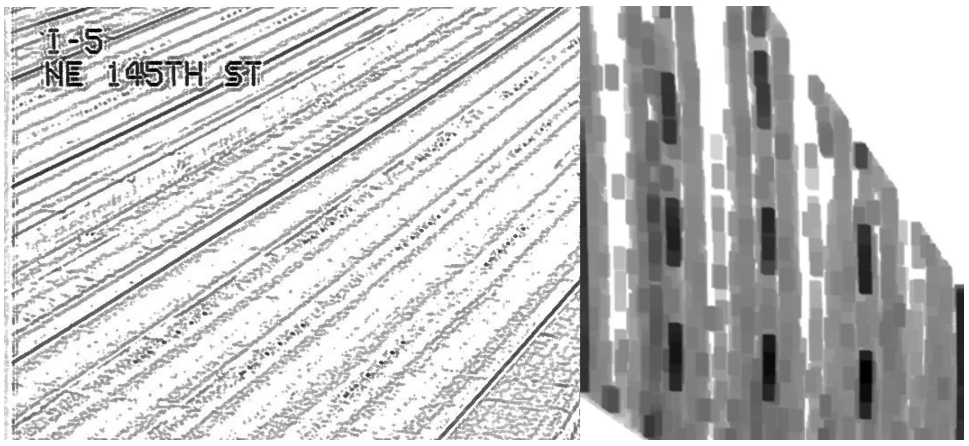


Figure 74. a) Typical top-hat image obtained from a background image where v_0 is large. b) $T_2(u,r)$ sampled from the whole image (excepting the text label) after applying the dilation kernel.

3.7.4.3 Robust estimation of τ_L via the autocovariance function

Although many scenes produce signals and images with the level of quality of the previous example, the quality can be degraded greatly. For example, Figure 75 contains a version of $T_{2d}(u,r)$ taken from a rainy scene where the road is curved. Even when using the front one-third of the image, the features are difficult to perceive. We must design our estimation procedure with these scenes in mind for two reasons: 1) we want to estimate τ_L in even the most difficult cases, and 2) we want to develop criteria to know when our algorithm cannot work.

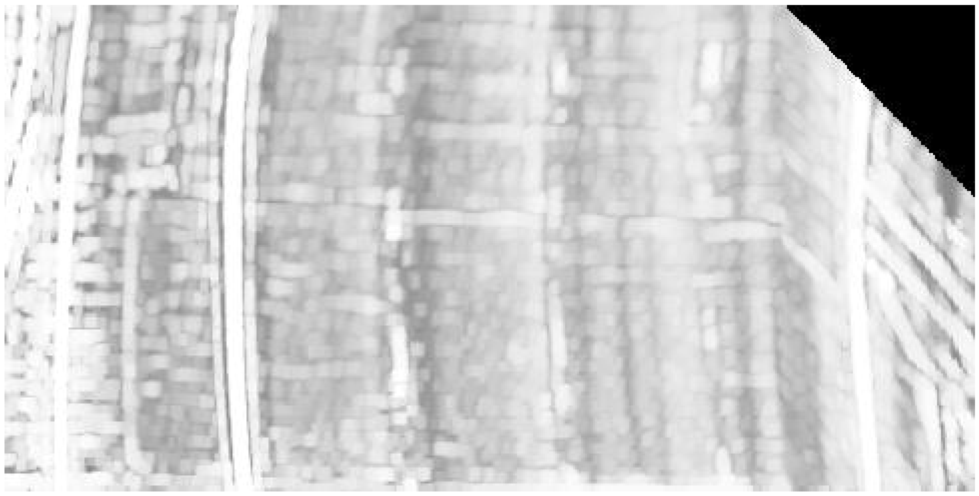
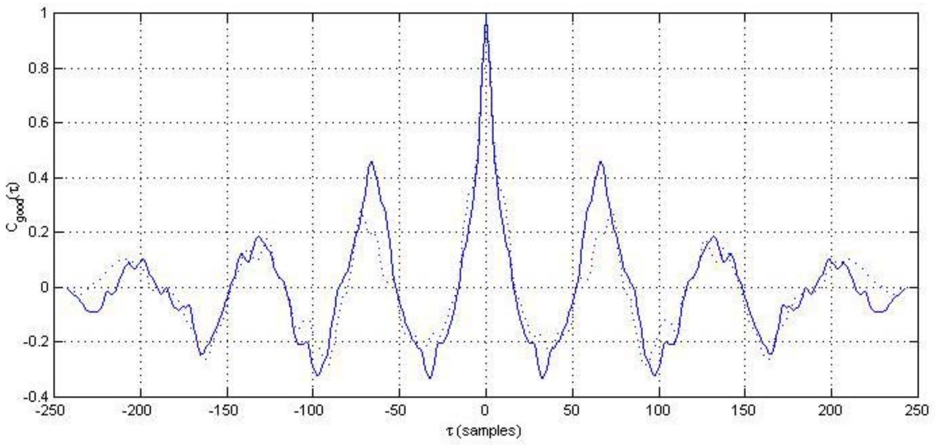


Figure 75. Example where the lane markers are difficult to measure in the $T_{2d}(u,r)$ image.

The problem of estimating the interval τ_L between lane markers can be divided into two stages: 1) identifying the horizontal positions, $u_{lane}(i)$, where the lane marker signal is present, and 2) estimating the lane marker interval from the autocovariance functions of $T_{2d}(u_{mark}(i),r)$. The difficulty of the first stage can be appreciated in the examples of Figure 76, which contains examples of autocovariance functions that we wish to accept for the second stage (Figure 76 (a)), and those we should reject (Figure 76 (b)). Whereas the $T_{2d}(u,r)$ image here is 485 pixels wide, less than 15 of its columns contain useful data from which we can estimate τ_L.



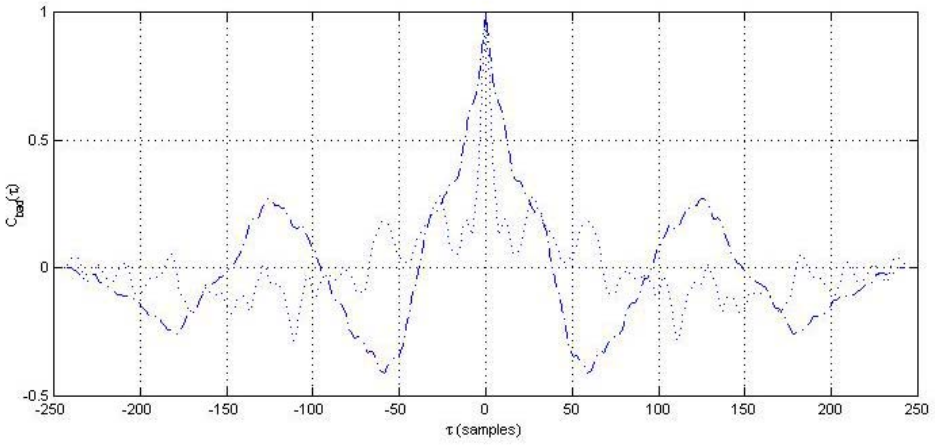


Figure 76. a) Examples of good autocovariance functions for columns in the $T_{2d}(u,r)$ image. b) Examples of bad autocovariance functions for columns in the $T_{2d}(u,r)$ image.

If we are to identify all the horizontal positions $u_{lane}(i)$ of the lane markers, we cannot avoid calculating the autocovariance function $C_{ss}(u,\tau)$ for every value of u. Once this is complete, however, we can filter the ensemble of autocovariance functions to determine the values of $u_{lane}(i)$. We can observe several things about the samples of $C_{ss}(u,\tau)$ in Figure 76. 1) Bad sequences may have a few broad or many narrow peaks, none of which are correct. 2) Bad sequences may have tiny bumps near the main lobe that could falsely be considered a peak. 3) The good sequences possess a character similar to that of Figure 71(d) and they are fairly similar within the $T_{2d}(u,r)$ image. 4) Because of the normalization, bad sequences could randomly have strong autocovariance peaks off the main lobe even if the signal is noisy or its amplitude is small.

On the basis of our analysis, we offer several criteria to reduce the full ensemble of autocovariance functions E_1 to a subset E_2. 1) The location of the maximum peak off the main lobe must be within 40 percent of the height of the $T_{2d}(u,r)$ image. 2) The maximum peak height, C_{max}, must exceed a threshold of $T_{Cmin} = 0.35$. 3) The difference in $C_{ss}(u,\tau)$ between the lowest trough C_{min} and C_{max} must exceed $0.2 \cdot C_{max}$. 4) The correlation coefficient ρ_{corr} (see Eq. (65)) between the candidate $C_{ss}(u,\tau)$ and a synthesized autocovariance function must exceed 0.75. 5) To achieve a value of 1 at the peak of the autocovariance, we normalize by the variance of the signal, i.e., its energy. The signal will range between -1/2 and 1/2. We require the signal to have an energy value of at least 0.03 for normal scenes and 0.06 when $v_0 > H$.

We now explain the criteria above as follows. Requiring τ_L to be less than 40 percent of the height of the $T_{2d}(u,r)$ image ensures that we have at least two-and-a-half lane marker periods with which to estimate τ_L. If $v_0 > H$, we raise the threshold to 50 percent since we are utilizing the entire image, and often there are only two lane markers in the image. Our experience suggest that a minimum value of $T_{Cmin} = 0.35$ works well. Because of the data windowing effect illustrated in Section 3.7.4.1, when τ_L reaches the 40 percent point, the theoretical maximum of $C_{ss}(u,\tau)$ is 0.6, but the value of $C_{ss}(u,\tau)$ is appreciably lower than this in practice. Candidates of $C_{ss}(u,\tau)$ located at or near the lane marker position can sometimes have a threshold in the 0.25 or 0.3 range, but false peaks reach this value much more frequently than when the value is higher, such as $T_{Cmin} = 0.35$. The choice of the C_{max}-C_{min} difference is also based on experience. The relative value of 0.2 is not very discriminating, but it occasionally reduces the number of errors. As for the fifth criterion, the energy values seem to work well in practice for a variety of scenes. The higher threshold when $v_0 > H$ enables us to eliminate periodic signals of low amplitude whose periodicity is due to extraneous road features that were enhanced by the morphology.

Of all the criteria, the fourth one is the most grounded in theoretical considerations. The correlation coefficient ρ_{corr} is an excellent way of characterizing how well the shapes of two signals match. We synthesize the autocovariance function for comparison with $C_{ss}(u,\tau)$ by generating a square-wave with a 40 percent duty cycle with the period given by the estimate for τ_L provided by the location of the first peak in $C_{ss}(u,\tau)$. We choose a 40 percent duty cycle because it lies in the middle of the duty cycle range we expect in practice. A particularly nice feature of this approach is that it implicitly includes the locations of the second, third, and subsequent peaks in the calculation of ρ_{corr}. Another advantage of this approach is that the tapering of the autocovariance function due to the data window automatically weights the main lobe and first peak more heavily than the peaks farther from the main lobe. Choosing a correlation threshold of 0.75 allows many autocovariance functions with the same general shape as the synthetically generated function to pass. For example, the two signals in Figure 77 have a correlation coefficient of 0.80.

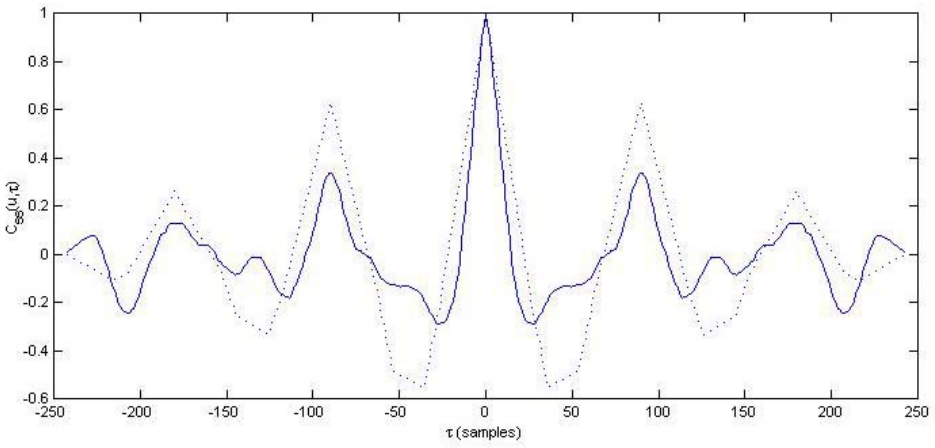


Figure 77. Comparison between a candidate autocovariance function and a synthetically generated one, where ρ_{corr} = 0.80. The candidate passes criterion 4 but fails criterion 2.

On occasion, the image contains periodic features that occur at multiple periods. For example, the lane markers on the right in Figure 78 are surrounded by horizontal lines that are strong enough to produce a false peak in the autocovariance function. Figure 79(a) contains the average autocovariance function from such a case where lags of 73 and 198 are competing; we originally measure 9 and 13 values of each, respectively. Thus, we check each member of E_2 as follows. We use the average of ensemble E_2 as a template and calculate the correlation coefficient ρ_{corr} between it and the members of E_2. We create a subensemble E_3 composed of members of E_2, where $\rho_{corr} > 0.75$. In the case below, all nine bad members of E_2 are eliminated and all 13 good members are included in E_3. As seen from Figure 79(b), the average of E_3 matches the synthetically generated autocovariance function extremely well.

Finally, we calculate the average of E_3 and check it against criteria 1, 3, and 4 from above. Checking the average autocovariance function against a synthetically generated version reassures us that our estimate of τ_{ab} was obtained from a signal whose shape approximately matches the theoretical ideal of Section 3.7.4.1. If the average autocovariance function passes these checks, then we collect samples of the lags of the largest peaks off the main lobe for each of the members of E_3. We estimate τ_{ab} using the average of the data. If the average autocovariance function fails any of the criteria, our algorithm alerts the user that it is unable to calibrate the camera.

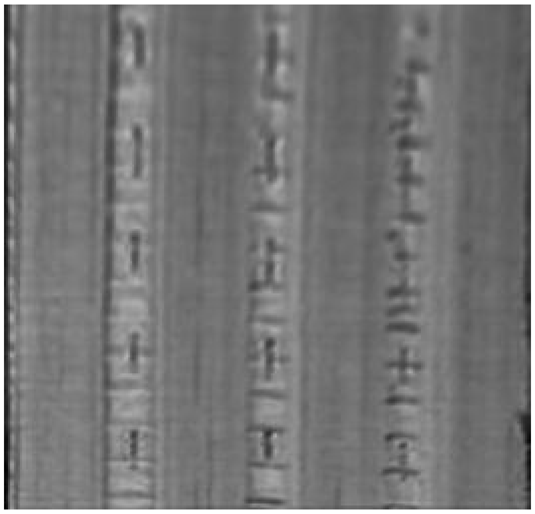


Figure 78. Example of $T_2(u,r)$ containing features with multiple periods.

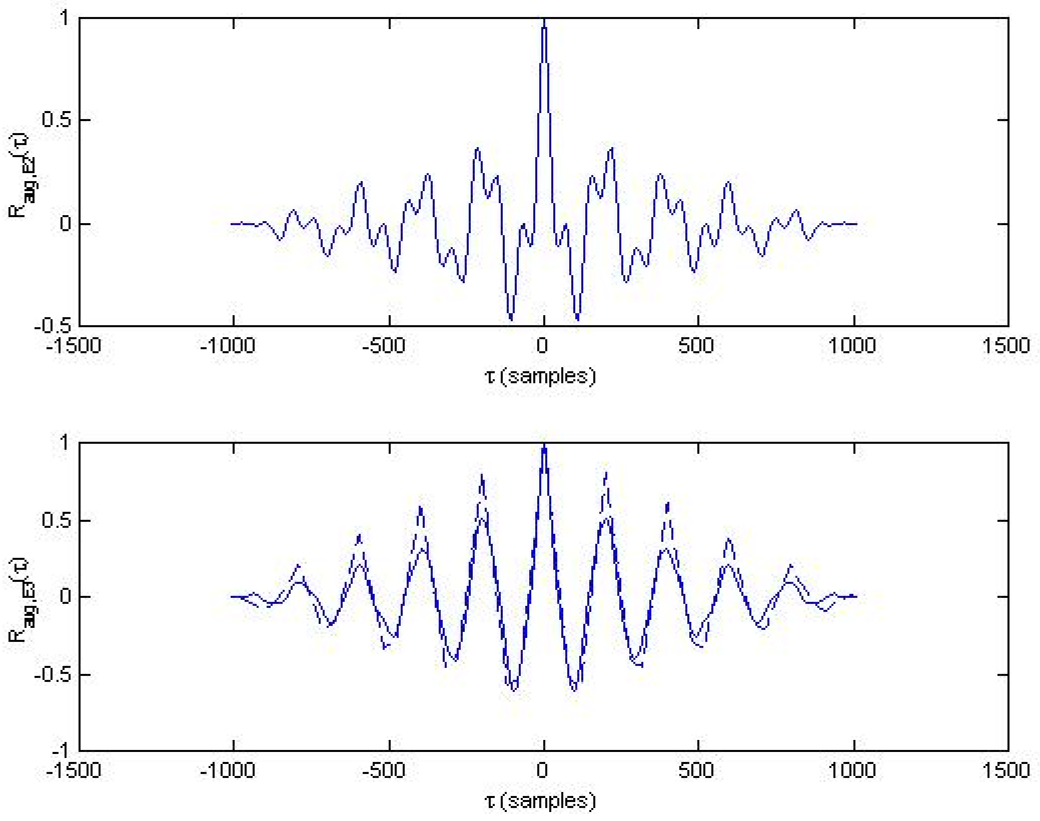


Figure 79. a) Average autocovariance function for E_2 when the image contains multiple periods. b) Average autocovariance function for E_3, i.e., using members of E_2 that are sufficiently similar to the ensemble average of E_2 in (a). A synthetically generated autocovariance function is shown using dashed lines.

3.7.4.4 Confidence interval estimation for τ_L

Estimating the confidence interval for τ_L is nontrivial because there are two sources of variation: image measurement errors and the variability of the physical lane marker intervals. Furthermore, we typically collect a sparse data set. We combat this problem by bootstrap-sampling the lag data collected to generate many samples of the mean of the data. The law of large numbers ensures that the samples of the mean are normally distributed. We generate 10^4 samples of the mean and use $\pm1.96s$ about the mean of the lag data as our confidence interval, where s is the sample standard deviation for the samples of the mean.

3.7.5 Monte-Carlo analysis

Our image and signal processing algorithms above seem to perform in a reasonably robust fashion for various light and road conditions. However, we must also objectively characterize how they behave when the lane markers are physically not ideal. The diagram in Figure 6 defines the total interval between lane markers as $L_{tot} = L_{1,mark} + L_{2,mark}$, where $L_{1,mark}$ is the length of the blank space and $L_{2,mark}$ is the length of the marker itself. Typically L_{tot} is 40 feet and $L_{2,mark}$ is 10-12 feet. We define the duty cycle D as $L_{2,mark} / L_{tot}$. However, because we morphologically dilate the lane marker image above, the apparent duty cycle is closer to 35-40 percent as opposed to its physical value of 25-30 percent.

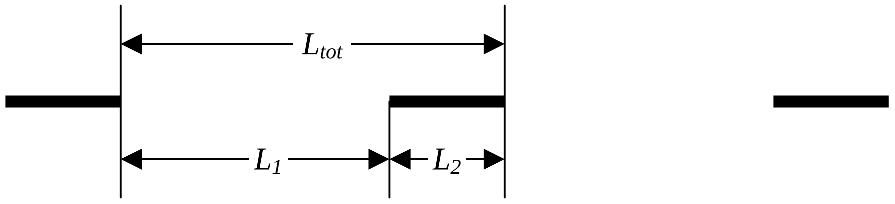


Figure 80. Periodic lane markers with distance definitions.

We developed a Monte-Carlo simulation to examine the performance of the combination of morphological preprocessing and peak extraction from the autocovariance function. We used a signal that was $N_{length} = 800$ samples long, which is typical for our data. We fixed the number of pulses, N_{pulse}, for the duration of the simulation. For each run of the simulation, we selected random values for the duty cycle D of each square-wave signal, such that $D \sim U(0.25,0.5)$. For each lane marker interval, we selected random values for L_{tot} and L_2, defining their standard deviations relative to the mean values of L_{tot} and L_2. The specific distributions in (μ,σ) format are $L_{tot} \sim N(N_{length}/N_{pulse}, 0.1 \cdot N_{length}/N_{pulse})$, and $L_{2,mark} \sim N(D \cdot N_{length}/N_{pulse}, 0.04 \cdot D \cdot N_{length}/N_{pulse})$. In particular, the choice for $\sigma_{Ltot} = 0.10 \cdot N_{length}/N_{pulse}$ was motivated by collecting physical data from a real traffic scene, as discussed later in Chapter 4. Figure 81 contains an example of a sample signal undergoing distortion by gaussian noise followed by noise suppression with the dilation operator. Comparing Figure 81(c) to an actual signal in Figure 73 shows that our simulation gives a reasonable reproduction of realistic conditions.

To make our simulation even more realistic, we include in our model the possibility that a lane marker is only half-present, i.e., $L_2 = 0.5 \cdot D \cdot N_{length}/N_{pulse}$. For any given lane interval, we allow a 1/3 probability that it is half-missing. Our experience suggests that this probability is likely somewhat high because departments of transportation typically perform road striping maintenance annually [52].

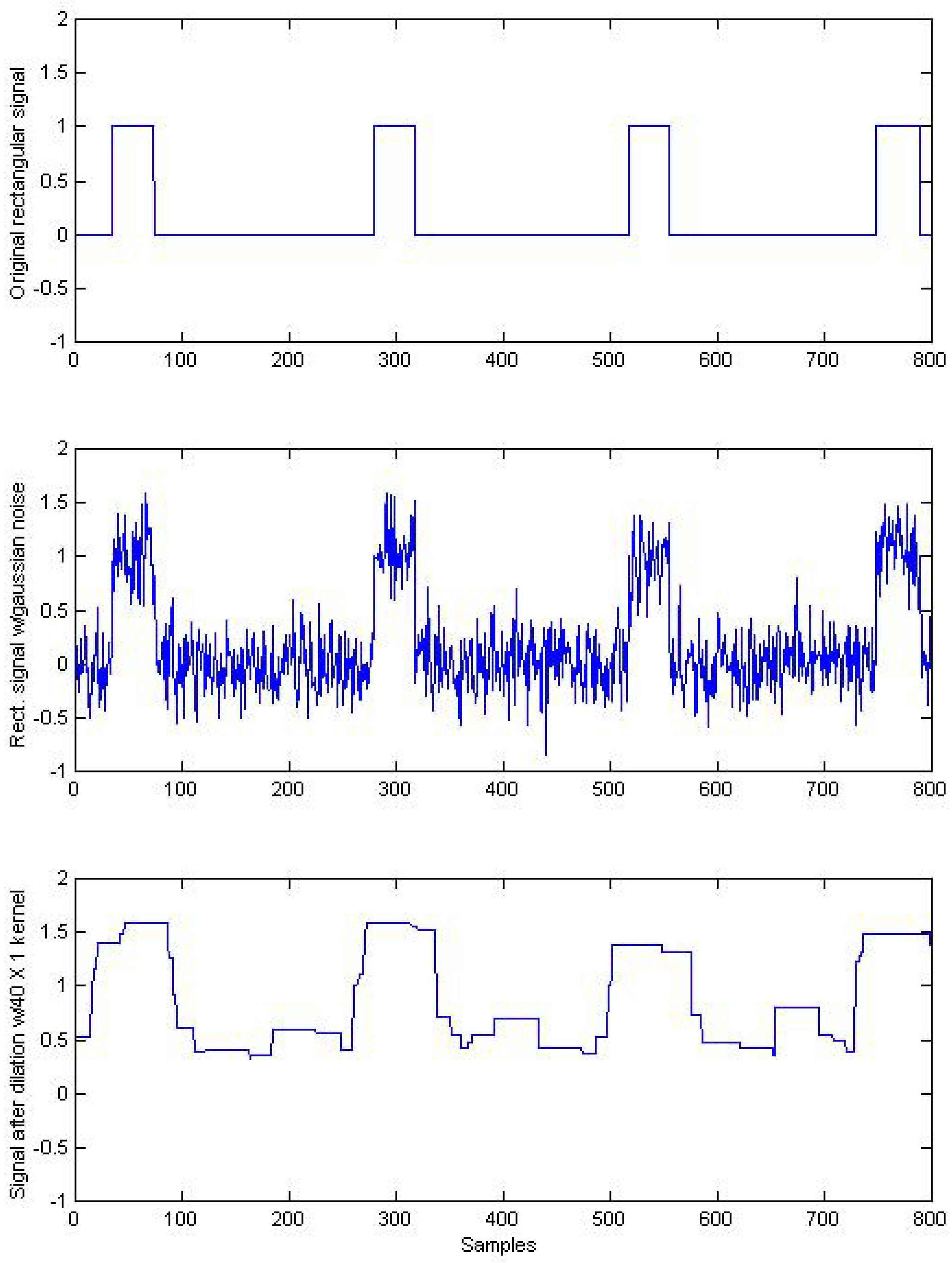


Figure 81. a) Original rectangular signal. b) Signal with additive gaussian noise σ = 0.25. c) Signal after morphological dilation with 40-sample kernel.

We have already shown that the autocovariance function is generally a reasonable way to estimate the lane marker interval. However, we wish to know the variability in that estimate as a function of the number of estimates we make from the lane stripe image. Furthermore, when we only obtain a few estimates, the lack of data can sometimes completely sabotage our estimation process. Thus, we would also like to determine the likelihood of obtaining a valid measurement as a function of the number of measurements we perform. By “measurement,” we mean the processing of a column in the lane stripe image that we know physically ought to contain lane stripes.

Figure 82(a) shows that the relative variability of our estimate of the lane marker interval decreases with the number of measurements we are able to make on the lane stripe image. The decrease is due to the increased sample size, as well as the increased probability of getting multiple valid measurements, i.e., those without partially missing lane markers. It is also interesting to note that the relative standard deviation of the estimate (0.05) for a single estimate is half of the input relative standard deviation (0.10). This is because the filtering criteria of Section 3.7.4.3 reject many of the outliers. For example, if one of the two lane markers is only half-present, then the autocovariance will probably be too low, and that estimate will be rejected. We also note that the autocovariance method induces an averaging effect that reduces the estimate variance.

Figure 82(b) shows that if we can obtain ten or more measurements, we can almost guarantee that at least one of them will give us a reasonable estimate of the lane marker interval, i.e., a valid autocovariance function. For this number of measurements, we can also expect a reasonably low relative standard deviation, as shown in Figure 82(a). This result provides additional support for our image processing strategy above to use horizontal dilation on the lane stripes to magnify the effect of each set of lane markers to obtain more measurements.

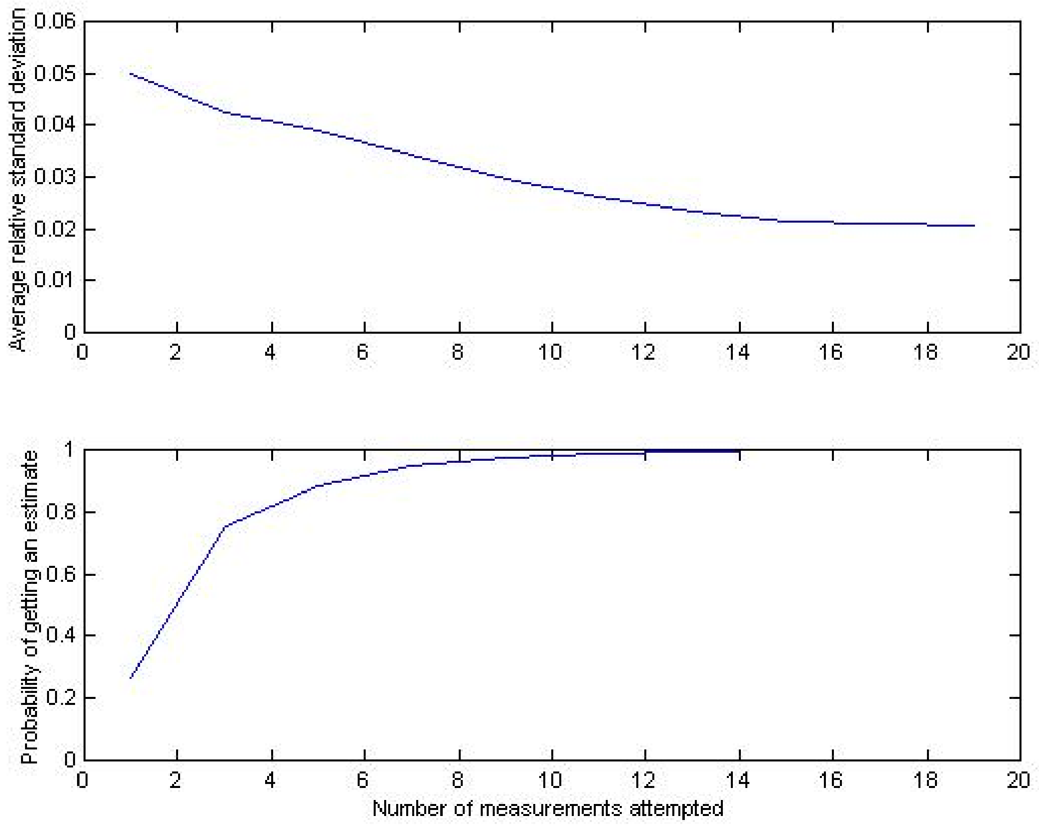


Figure 82. Trends for a Monte-Carlo simulation for the measurement of the interval involving two lane stripes. a) The trend in relative variability of the lane marker interval estimate with the number of measurements attempted. b) The trend in the probability of a valid measurement with the number of measurements attempted.

We also offer Figure 83 in comparison for the case of four markers visible in the lane stripe image. The result of Figure 83(a) is essentially the same as that of the single-interval case of Figure 82(a), apart from the larger initial value for a single measurement. Because there are three intervals instead of one, we note from Figure 83(b) that we are much more likely to obtain a good estimate of the cycle length than when we measure only a single interval, as in Figure 82(b).

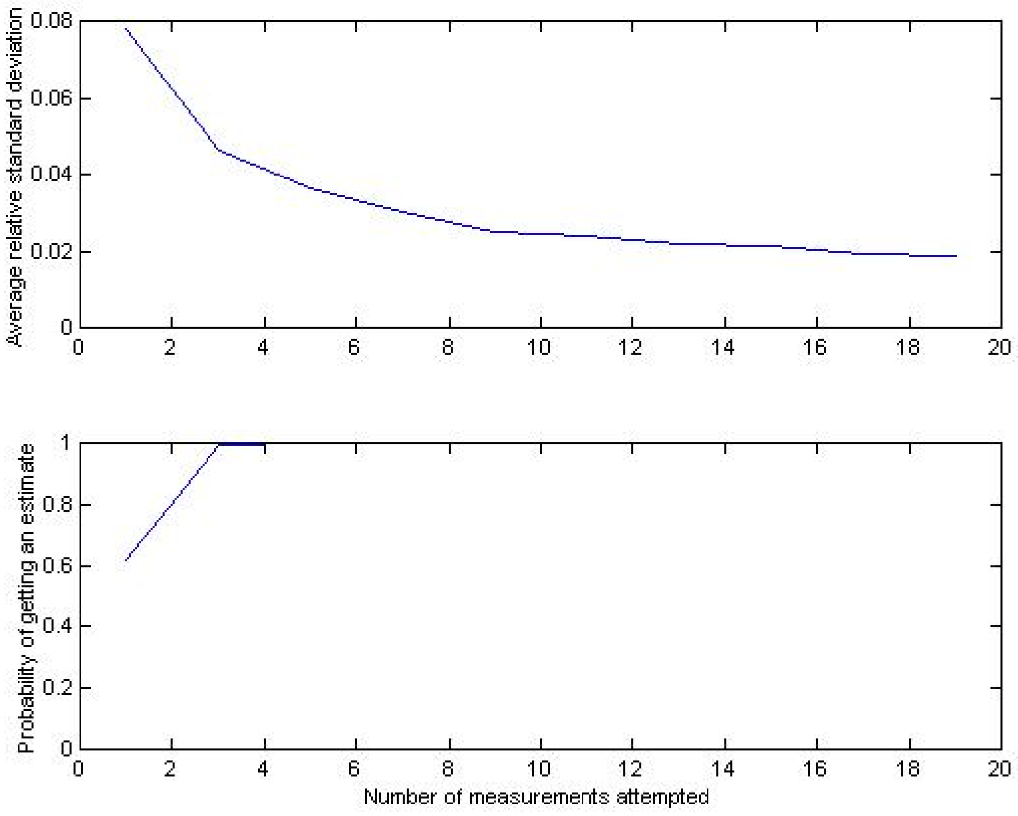


Figure 83. Trends for a Monte-Carlo simulation for the measurement of the interval involving four lane stripes. a) The trend in relative variability of the lane marker interval estimate with the number of measurements attempted. b) The trend in the probability of a valid measurement with the number of measurements attempted.

In our data sets, we typically achieve at least ten separate measurements. Thus, we can expect reasonably low variability in the output of our algorithm, even with a large relative standard deviation for *L* of about 0.1. It is also encouraging to note that if we have multiple intervals, the threat of a half-missing lane marker is not that great and, on average, should not sabotage our measurements.

3.8 Estimating vehicle lines

The last parameter from Chapter 2 for which we need an estimate is u_1, the u-coordinate of the vanishing point for the lines perpendicular to the road. This parameter is essential for calibrating the camera using Method 1. However, there are no guarantees that any lines perpendicular to the road exist on the roadway or the surrounding scene. Because the vehicles themselves contain (small) line segments that are perpendicular to the road, we can calculate u_1 by estimating the parameters of these line segments. This procedure is complicated by the discrete nature of the image digitization. Even for reasonably large camera angles, there is not much slope to the vehicle lines at the 320 X 240 image resolution. Fortunately, because our system uses an NTSC camera, we have higher-resolution, 640 X 480 interlaced images available. However, the vehicles often travel a significant distance between successive scans of the interlaced image frame. Thus, we deinterlace the image by doubling the two sets of scan lines, creating two 640 X 480 noninterlaced images captured 1/60 second apart. To ensure the best estimates of u_1 (and d), we require the vanishing coordinate v_0 to be large, i.e., $v_0 > H$.

As shown in Figure 45, we simply need an appropriate image sequence, an estimate of the vanishing point coordinate v_0, and a set of lane masks in order to estimate u_1. Our general strategy is to identify the edge pixels of the vehicles with orientations within an expected range. After thresholding to create a binary image, we remove any pixels outside of the lane masks. We thin the mask using morphology and perform a connected components analysis to remove small islands. Next, we use the Hough transform [44] to find any lines that exist in the image. Finally, we calculate the intersection of each line with the vanishing line $v = v_0$ and estimate u_1.

3.8.1 General image processing

We follow the procedure below to identify vehicle edge pixels that might belong to a line that is perpendicular to the road. 1) Calculate the image intensity by averaging the RGB color components. 2) Deinterlace the image and create two images by duplicating the odd rows in the even rows of the image, and vice versa. 3) Estimate the horizontal and vertical image gradients, I_x and I_y, of the current frame using a derivative-of-gaussian kernel

$$K_1 = -(2\pi)^{-1/2}\sigma^{-3} n e^{-\frac{n^2}{2s^2}} \qquad (74)$$

where $n \in \{-6, -5, \dots, 5, 6\}$ and $\sigma = 2$. The "x" subscript denotes a 1 X 13 convolution kernel that detects vertical edges, with the converse (transpose) for "y." n and σ may be scaled linearly according to the current image dimensions (640 X 480). 4) Estimate the angle of the line direction for each pixel by $\gamma = \tan^{-1}(I_y/I_x)$. 5) Calculate the gradient magnitude image $\|\nabla I\| = (I_x^2 + I_y^2)^{-1/2}$. 6) Create a binary image in which pixels are 1 if $\|\nabla I\| > 0.50$ and $0 < \gamma < 40°$, and 0 otherwise for $u_1 > 0$. The inequality becomes $0 > \gamma > -40°$ if $u_1 < 0$.
7) Threshold $\|\nabla I\|$ and set to 0 any pixels in the binary image where $\|\nabla I\| < 0.085 \cdot \max(I_i)$; the threshold is chosen as a fraction of the maximum image intensity. An ideal step edge has a maximum filter response of approximately 17 percent of the step magnitude. Thus, we detect step edge magnitudes that are at least half of the maximum image intensity.
8) Remove any pixels outside the logical addition of the lane masks. We generally use only the bottom half of the image, since the lines have greater slope in this region. Figure 84(a) contains a typical mask found by applying these eight steps.

Filtering the edges on the basis of their orientation does a pretty good job of eliminating false edges. However, thick lines (see Figure 84(a)) add unnecessary noise to the Hough transform because a wider range of angles fit within a thicker block. Morphological skeletonization is a perfect solution to this problem because it reduces a region to its minimum number of connected pixels at the center of the region [45]. Figure 84(b) illustrates the effects of morphological skeletonization on a binary image generated by the process above. Next, we perform a connected components analysis on the skeleton image to remove any features that are less than 20 pixels long.

Figure 84. a) Binary image B generated by thresholding the gradient magnitude of a traffic image. b) Morphological skeleton of B. c) Binary image B after skeletonization and connected components analysis.

3.8.2 Estimating the line parameters

After we finalize the binary image, we calculate the Hough transform [44] of the binary image using one of the standard line parameterizations

$$\rho = u \cos(\Phi) + v \sin(\Phi) \tag{75}$$

where ρ is the perpendicular distance from the origin to the line, and Φ is the angle between the line and the u-axis. We use non-image-centered coordinates because it makes the parameterization of ρ more convenient. Since we operate on the lower half of the image, we expect the extreme ranges to be $\rho \in (3H/8, (W^2 + H^2)^{1/2})$. Since we require the vehicles to be passing through the bottom of the image, we require $\Phi \in (60,110)$ degrees when $u_0 < 0$. The range of Φ past 90° is to enable us to analyze the statistical properties of Φ later on. We calculate the extrema of ρ that we will encounter by calculating its maximum and minimum at the four corners of the image window. To use the Hough transform we quantize ρ into 1-pixel increments, and quantize alpha into 0.15-degrees increments for 640 X 480 images. We select (ρ,Φ) bins from the Hough transform containing at least 20 pixels and record the (ρ,Φ) pair. Using a finer ρ quantization noticeably reduces the number of pixels in each bin, reducing the number of lines found. Because of the fine quantization of Φ, each line is usually measured as multiple (ρ,Φ) pairs distributed about a nominal angle Φ. Because each nominal angle is slightly different, we are essentially convolving a bunch of distributions for Φ. If this hypothesis were completely true, this convolution should eventually lead us to an asymptotically normal distribution for Φ because of the large number of samples that we collect. Figure 85 contains a distribution of 101,721 samples of Φ that appears to be normally distributed. However, as shown from the curve fit of the normal distribution specified by the estimates of the mean and standard deviation of the data, the underlying distribution is probably not normal.

The algorithm above assumes that that $u_0 < 0$. We know that $u_0 > 0$ implies that $u_1 < 0$, which changes the ranges we expect of ρ and Φ in the Hough transform. However, if we flip the image and let $u' = W - u$, then we can proceed with the analysis above without any further modifications.

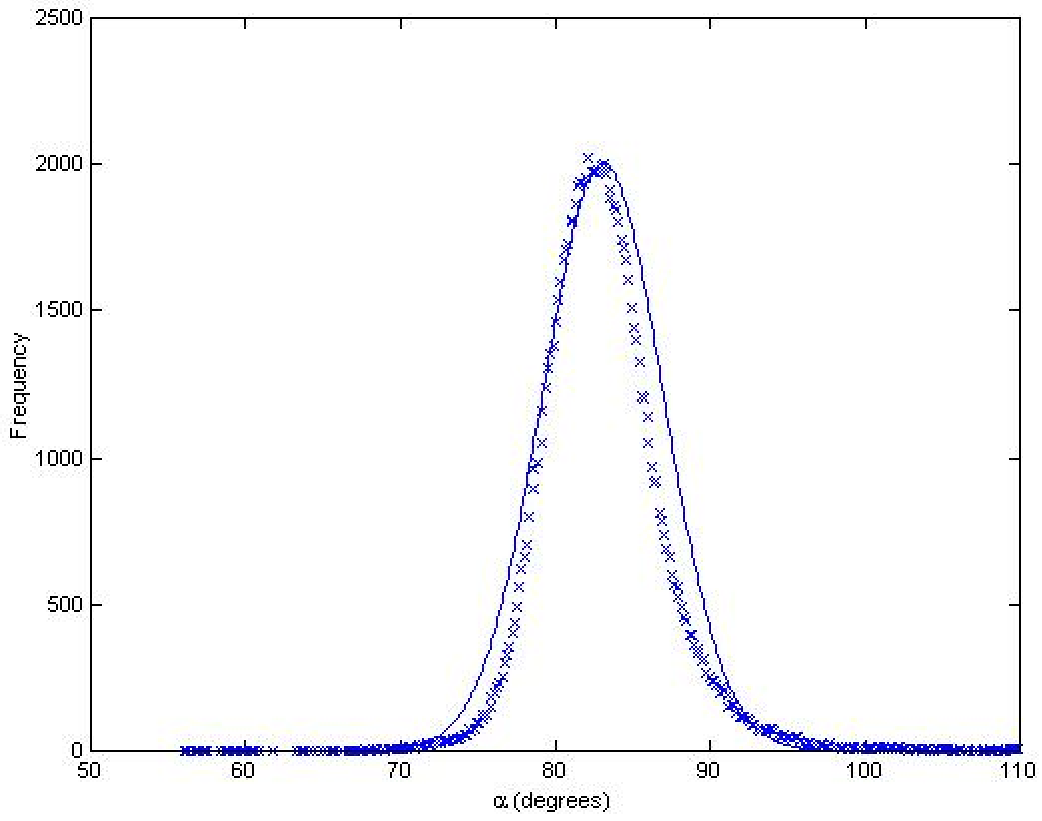


Figure 85. Empirical (x) and theoretical gaussian (-) distributions of the line angle Φ for a scene when the perpendicular lines to the road have a noticeable tilt in the image. The theoretical distribution is based on estimates of the mean and variance from the data.

3.8.3 *Estimating* u_1 *from the data*

There are two main problems to be solved in trying to estimate u_1 from the line parameter data. First, we must determine whether the data will allow us to estimate u_1, or the line slope is too small. As we argued earlier in Chapter 2, we cannot make claims to measure line slopes less than a certain level. We choose a minimum slope of 1/15, which is consistent with our choices for a minimum line length of 20 pixels. Thus, if our estimate of the mean value for Φ exceeds $90° - \tan^{-1}(1/15) = 86.19° \equiv \Phi_{max}$, then we conclude that the quantization errors in our estimates will prevent us from reliably estimating u_1. For example, Figure 86 contains the distribution of Φ for a scene where the line slopes are very slight. The estimate of the mean for this distribution is 88.9°, so we conclude that u_1 cannot be estimated from the data of the scene.

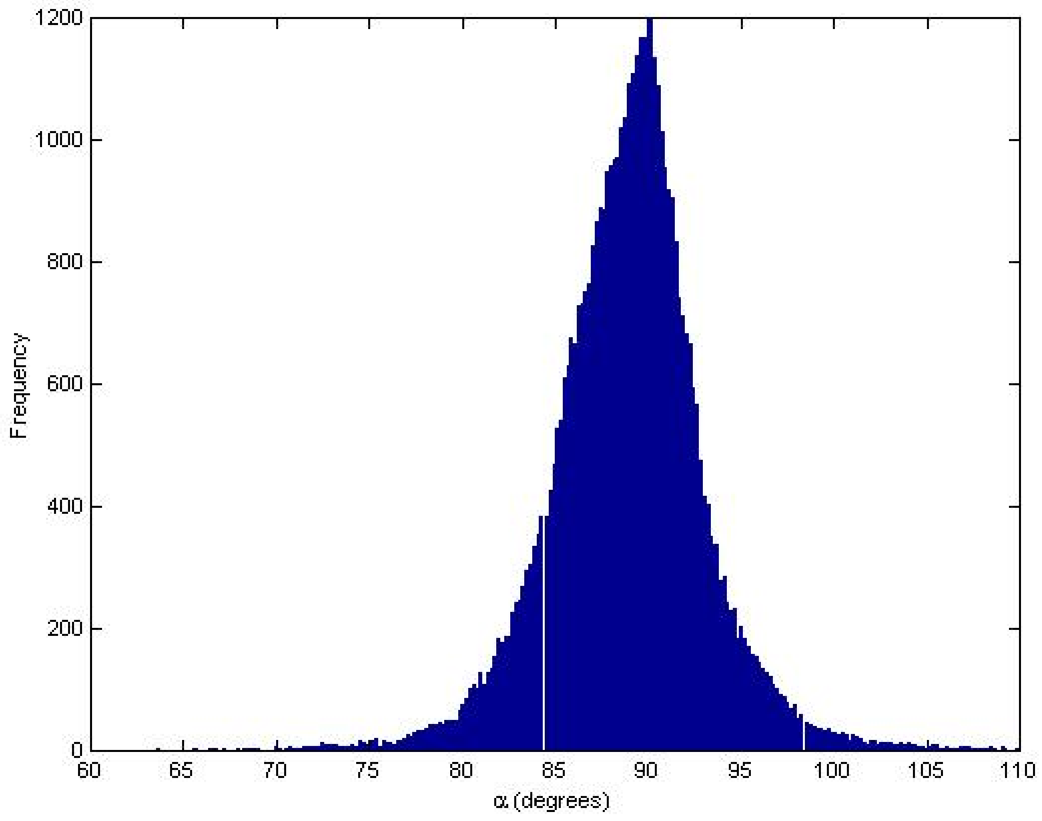


Figure 86. Empirical distribution of the line angle Φ for a scene where the tilt of the lines perpendicular to the road is very small.

The second problem to solve is estimating u_1 itself, given the distributions of the line parameters Φ and ρ. We recall that the v-coordinate of the second vanishing point is v_0 under the assumptions of our camera model from Chapter 2. Solving for $u = u_1$ in Equation (75) yields

$$u_1 = \begin{cases} -\left(\dfrac{\rho - (H/2 - v_0)\sin(\alpha)}{\cos(\alpha)} - W/2\right), & u_0 > 0 \\ \dfrac{\rho - (H/2 - v_0)\sin(\alpha)}{\cos(\alpha)} - W/2, & u_0 < 0 \end{cases} \tag{76}$$

where we have included correction terms for the fact that u_0, u_1, and v_0 assume image-centered coordinates, and the measurements of Φ and ρ do not. We can now see why u_1 cannot be estimated for small slopes, i.e., cos(Φ) → 0 as Φ → 90°, and the uniformly quantized values of Φ run out of resolution beyond Φ_{max}. We can also see that because of the transcendental transformations of Φ, we should not expect the distribution of u_1 to be symmetrical. For example, the distribution in Figure 87 has a peak around 4200, whereas the true value is more than 5000; the average and median values are 5822 and 5018, respectively.

As shown in the example of Figure 87, the distribution of u_1 is quite skewed. It is not obvious how we should estimate u_1, since Equation (76) is nonlinear. However, the median and mean are reasonable estimators of u_1 that are worth trying. Thus, we performed a Monte-Carlo simulation with known values of u_0, v_0, and u_1 and generated random lines by connecting the point (u_1,v_0) to points in the measurement window. After calculating Φ and ρ for the line, we added random noise according to $N(0,3.7°)$ to each value of Φ. The resulting distribution of Φ had approximately the same shape and standard deviation as the distribution in Figure 85. Next, we calculated u_1 according to Equation (75) and examined the efficacy of the average and median values as estimators of the known value of u_1. We found that as long as samples are ordered properly across the branch cut of Φ = 90°, the median is an unbiased estimator of u_1, whereas the mean is not. One way to handle the branch cut is to add a very large value with the same sign as u_1 to those estimates of u_1 with the same sign as u_0.

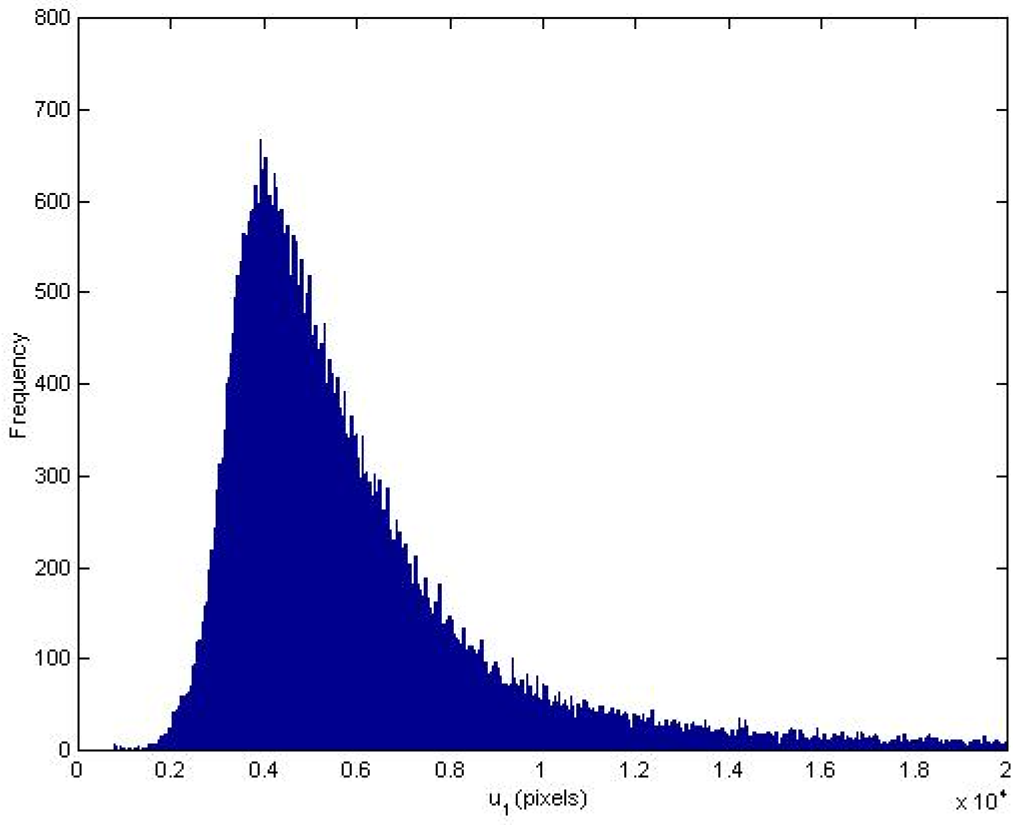


Figure 87. Empirical distribution of u_1 for the data in Figure 85. This heavy-tailed distribution continues up to 10^5.

3.8.4 *Estimating the error in* u_1

The u_1 estimation procedure has two interesting properties. For one thing, the distribution of u_1 is inherently skewed, and therefore, it is nontrivial to estimate u_1 and even more difficult to estimate its variance. On the other hand, the estimation procedure produced a lot of data. We use this fact to our advantage by bootstrapping the samples on hand to estimate the variance of u_1. Specifically, we sample the data we collected for u_1. If N_{all} is the total data size, we obtain $\eta \cdot N_{all}$ samples, where η is a fraction between 0 and 1. We choose $\eta = 0.2$, both for practical considerations of computational cost and to avoid repeating samples as much as possible because we sample with replacement. After obtaining the samples, we estimate u_1 by finding the median of the samples and record it as one estimate of u_1. On the basis of the law of large numbers, we expect this distribution of the estimates of u_1 obtained through bootstrapping to converge to the normal distribution. After performing this procedure a sufficient number of times, we then use the sample variance $s_{u1,\eta}{}^2$ as an estimate of the population variance $\sigma_{u1,\eta}{}^2$ for the case of $\eta \cdot N_{all}$ samples. To estimate the population variance σ_{u1} for the u_1 estimate obtained using N_{all} samples, we simply multiply by η:

$$\sigma_{u1}{}^2 = \eta \sigma_{u1,\eta}{}^2 \tag{77}$$

For example, the median of the distribution in Figure 87 is 5603 pixels. When we perform 5000 runs with $\eta = 0.1$, we obtain $s_{u1,\eta}{}^2 = 890.1$. Therefore, our estimate of $\sigma_{u1,\eta}{}^2 = 0.1(890.1) = 89.00$. The same experiment with $\eta = 0.2$ yields $s_{u1,\eta}{}^2 = 452.0$ and the estimate $\sigma_{u1,\eta}{}^2 = 90.41$. The consistency in these two estimates of $\sigma_{u1,\eta}{}^2$ greatly increases our confidence in Equation (77). The variance is actually quite small because of the large sample size ($N_{all} = 101721$).

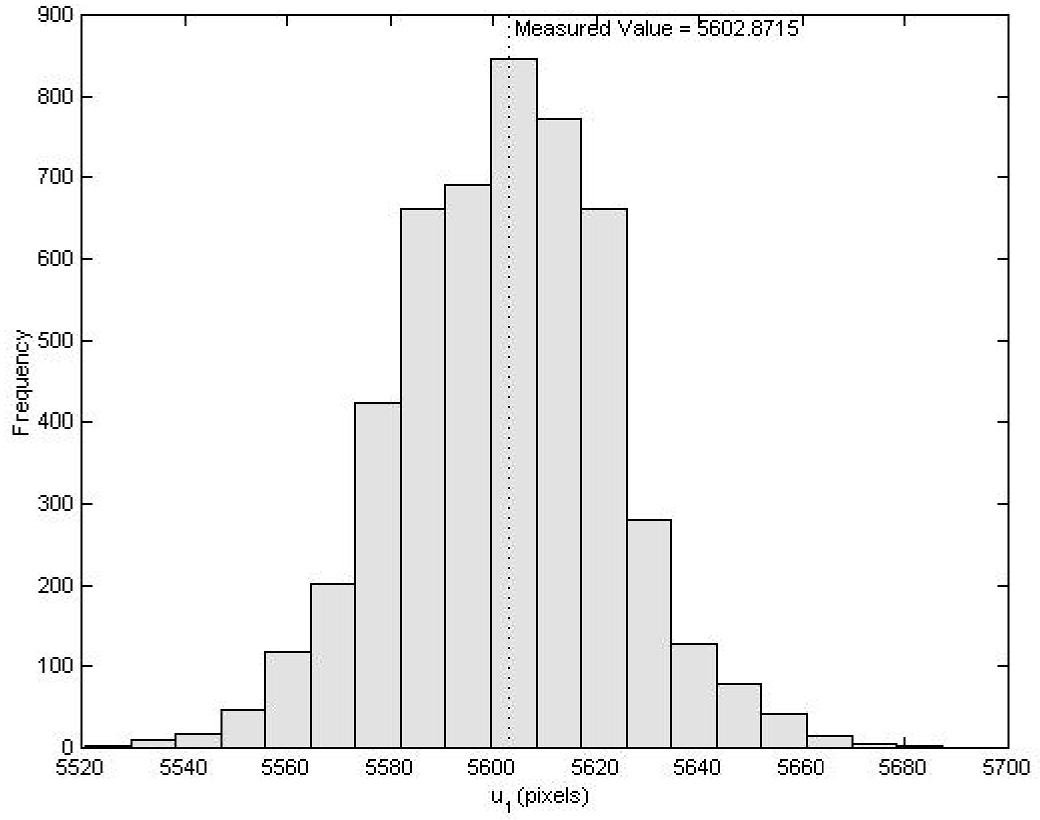


Figure 88. Distribution of the median values of u_1 found by bootstrapping the data displayed in Figure 87.

3.9 Summary

Chapter 2 provides a fundamental model for the camera and road scene and outlines several ways of calibrating the camera by performing measurements in the digital images. In this chapter, we describe algorithms for automatically extracting these quantities (i.e., u_0, v_0, u_1, b_1, b_2, and τ_L) in a reliable fashion. All of the algorithms depend on the two fundamental feature images—the average top-hat image and activity map—that we generate from 1,000 images sampled from the scene. The activity map enables us to estimate the approximate vanishing point from which we can further analyze the activity map and extract the lane boundaries of the road. The lane boundaries are critical to the success of our vehicle tracker, which assumes that the vehicles are constrained to move down the road in single fashion. We also describe a raindrop and obstacle detection algorithm to disqualify certain scenes from being processed since the camera would not be able to be calibrated. Once the lane boundaries and rough vanishing point are estimated, we show how to precisely locate the road boundaries and vanishing point using the top-hat image. This, in turn, constrains our search process while estimating the interval between lane markers with the autocovariance signal processing technique. Finally, our knowledge of the activity region of the vehicles enables us to develop a method for estimating the vanishing point for the lines perpendicular to the road using the vehicles themselves. We believe that our algorithms should perform reliably under any daytime or dusk conditions in the context of a system using our rain detector where the pan angle $|\theta| < 20°$.

4 APPLICATION OF IMAGE PROCESSING TECHNIQUES TO CAMERA CALIBRATION

4.1 Introduction

The image processing techniques of Chapter 3 enable us to perform the measurements required to calibrate the camera using the various methods described in Chapter 2. Given the two options for calibrating the camera on a day-to-day basis from Chapter 2, we must test two major types of scenes to determine the performance of each approach. In the first type of scene, the camera has a large down angle ϕ, giving it a large value of v_0. Such scenes are appropriate for estimating the distance, *d*, between the camera and the road with either Method 1 or Method 3. This estimate enables us to use Method 2 on future scenes from that camera.

In the second type of scene, the camera is usually zoomed out all the way with a slight to moderate down angle so it can view the whole road, usually including the horizon. This is the typical operating position for the camera on a day-to-day basis. Road curvature is much more evident in this type of scene than in the first type of scene. Because of road curvature and perspective effects, only the bottom portion of the image possesses useful calibration information. Camera calibration methods 2 and 3 are potentially applicable to this type of scene. As explained in Chapter 2, Method 1 typically won't work because the lines perpendicular to the road appear nearly horizontal in the image, and thus their slope cannot be reliably estimated.

Within this second major type of scene, there are two subtypes. In one subtype, the camera is located on an overpass (e.g., Scene 3 of Figure 8(c)) where the perpendicular distance, *d*, between the road boundary and the camera is nearly zero or negative. Operators typically orient the camera with a small pan angle θ for such a camera geometry. In the second subtype, the camera is located off to the side of the road (e.g., Scene 1 of Figure 8(a)) where *d* is greater than 10 feet. Operators typically orient a camera in this position with a reasonably large pan angle θ to capture the full road.

In order to characterize the system performance, we must compare it to a reference. We begin by describing how we obtain reference data for the camera parameters before we discuss the calibration results for our algorithms.

4.2 Estimating the reference parameters

Although we have many operator-controlled cameras from which we can get vehicle image sequences, none of them are calibrated. Thus, we must use our knowledge of the scene, together with the equations for Method 3 from our theoretical model of Chapter 2, to estimate the true camera parameters. We note that if the road slope and tilt are zero, the equations for all three calibration methods are fully compatible. We chose to use the equations for Method 3 because known distances were the easiest features to accurately measure in the images.

By using the same assumptions as those used in the model of Chapter 2, we are essentially comparing the ability of the computer to measure image features with a human observer who identifies points in the image. We expect that both the computer and the human observer will make errors. We lump our imperfect knowledge of the scene into the measurement errors of the human observer. At the end, we expect to obtain 95 percent confidence intervals for the three camera parameters f, ϕ, and θ, the camera position d, and the overall scale factor S', which is the greatest determinant of the accuracy of distance measurements. We also estimate u_1, the u-coordinate for the vanishing point of the lines perpendicular to the road, in order to compare it to the results of the image processing.

We calculate estimates for f, ϕ, θ, d, S', and u_1 by estimating the intermediates b_1, b_2, u_0, and v_0. These intermediates are estimated on the basis of points clicked by the human observer. We requested the human observer to perform 25 iterations of the following procedure: click twice on the left-hand road boundary, twice on the right-hand road boundary, and sequentially on road features that are separated by a known distance. This yields N_{dist} pairs of points, each with 25 observations, and 50 observations along each of the road boundary lines. Figure 89 contains a top-hat image and points clicked by the user. According to the Manual on Uniform Traffic Control Devices [42] and WSDOT engineering staff [53], the Northwest region uses 10-foot-long lane markers on all freeways,m with a blank interval of 28-30 feet, for a total required end to end distance of 40 feet. This known distance corresponds to L in the equations that follow. Of course, individual instances of this distance will contain errors because of imperfections in painting the road or placing the turtles.

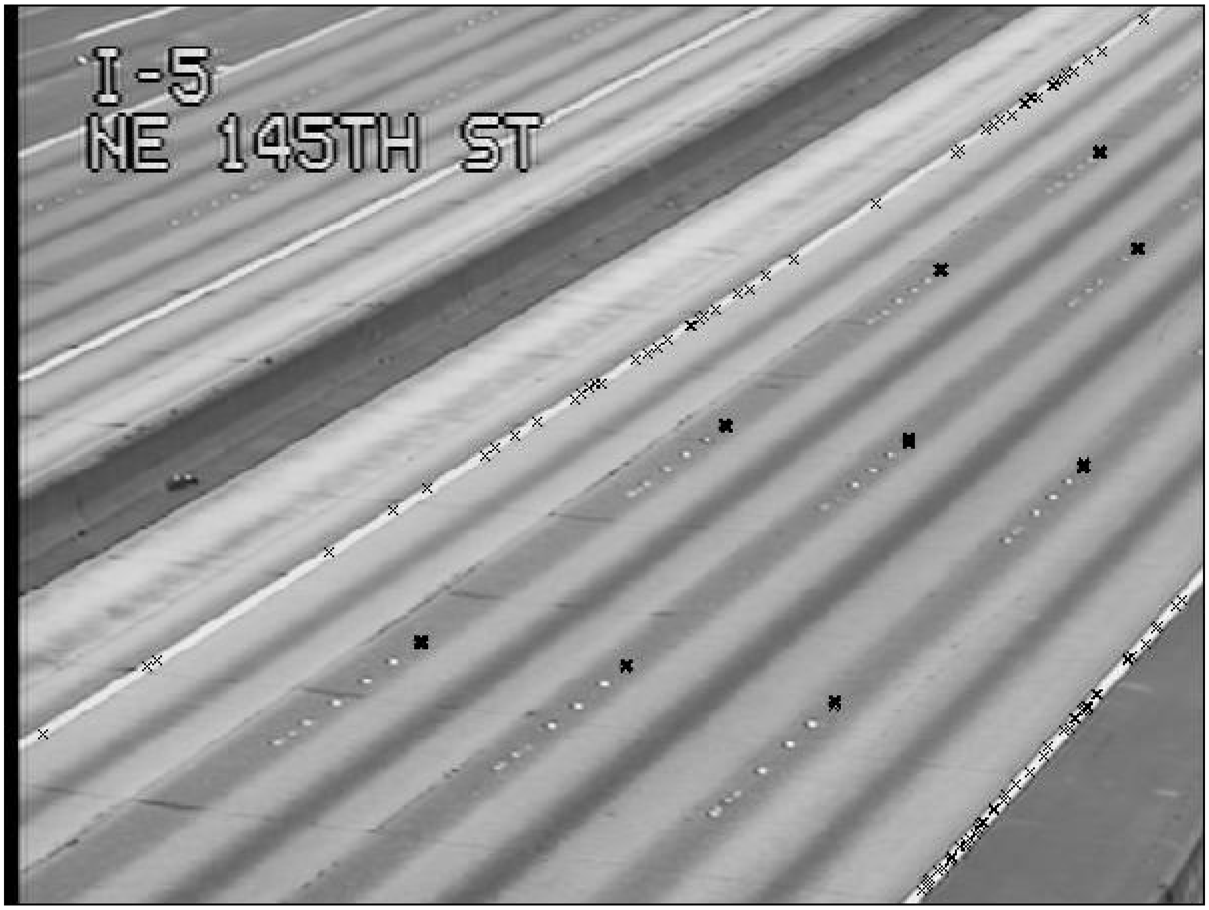


Figure 89. Sample image used in hand-calibration of the camera with superimposed points clicked by a human observer.

4.2.1 Reference parameter estimation using the theoretical model

Our strategy for hand-calibrating the camera for a given scene is to use the equations of Method 3 from Chapter 2, which we selectively repeat here without derivation. The following equation defines the useful quantity τ_L, which is a function of the vertical positions v_a and v_b of the two points defining the known 3-D distance L in the image. The calculation for τ_L uses the vanishing point coordinate v_0 to remove the perspective distortion and obtain a quantity that is proportional to the linear distance along the road. τ_L is extremely important because, together with v_0, it encodes all of the calibration information that can be gathered from the image. We will find that it is constant regardless of position in the image, and it is directly proportional to the physical distance L.

$$\tau_L \equiv \left(\frac{v_0 (v_b - v_a)}{(v_0 - v_b)(v_0 - v_a)} \right) \tag{78}$$

The next equation relates the known distance *L* to other quantities we can measure (the *u*-axis intercepts b_1 and b_2 of the road lines in the image, τ_L, u_0, and v_0), the unknown focal length *f*, and the road width *w* (assumed to be known).

$$L^2 = \tau_L{}^2 \left(\frac{w}{b_2 - b_1} \right)^2 \frac{\left(f^2 + v_0{}^2 + u_0{}^2 \right)^2}{f^2 + v_0{}^2} \tag{79}$$

In Chapter 2, we derived formulae for u_0, v_0, b_1, and b_2 in terms of the camera parameters and scene/camera geometry. Thus, we deduce that the ratio L/τ_L depends entirely on the camera parameters and scene/camera geometry and is therefore constant for any given situation. This result will prove very useful later on.

To solve for *f*, we first define

$$\alpha_0 \equiv \left(\frac{L(b_2 - b_1)}{w \tau_L} \right)^2 \tag{80}$$

where we assume knowledge of *w* and *L* and we estimate b_1, b_2, and τ from the mouse-clicks of the user upon the image. Following the derivation of Chapter 2, we define

$$\begin{aligned} \alpha_1 &\equiv 2\left(u_0{}^2 + v_0{}^2\right) - \alpha_0 \\ \alpha_2 &\equiv \left(u_0{}^2 + v_0{}^2\right)^2 - \alpha_0 v_0{}^2 \end{aligned} \tag{81}$$

Finally, we calculate *f*:

$$f = \sqrt{-\frac{\alpha_1}{2} + \sqrt{\frac{\alpha_1{}^2}{4} - \alpha_2}} \tag{82}$$

The remaining parameters θ, ϕ, *d*, and *S'* can now be found as described in Chapter 2 using their theoretical relationship to *f*.

We note that because these equations are the very ones used by Method 3, we must take care to acknowledge the weaknesses in this calibration method. Even if Method 3 perfectly mimics the hand-calibration results, we cannot guarantee that our algorithm found the true camera parameters unless our assumptions about *w*, *L*, and our two-angle camera model are completely valid. At the very least, however, given the analysis of road slope and tilt in Chapter 2, we can assert that our scale factor *S'* for road distance measurements is essentially unbiased by nonzero road slope or tilt. The truth about the bias of other output parameters is less certain.

4.2.2 *Estimating* b_1, b_2, *and* τ_L *from the human observer's mouse-clicks*

We estimate b_1 and b_2 by performing a linear regression of the points along the left and right road boundaries defined as lines L_1 and L_2 in Chapter 2. Their equations within the image are $u = m_1v + b_1$ and $u = m_2v + b_2$, respectively. From the estimates of m_1, b_1, m_2, and b_2 we can estimate u_0 and v_0, the coordinates of the vanishing point as previously derived in Chapter 2:

$$v_0 = \frac{b_2 - b_1}{m_1 - m_2}$$
$$u_0 = \frac{b_2 m_1 - b_1 m_2}{m_1 - m_2} \tag{83}$$

Given an estimate of v_0, we may then estimate the average value of τ_L from Equation (78) using all the pairs of points collected from the human observer that we assume are separated by a uniform distance L in 3-D.

4.2.3 *Estimating the confidence intervals in the output parameters*

The errors in the output parameters have two sources: errors in the human observations and errors in the assumed values of w and L. It is important to keep in mind that while these errors give the output parameters a certain distribution, this is different than the distribution of the mean values for the output parameters, which is the range we intend to quantify. Implicit in this statement is our (reasonable) assumption that the mean is an unbiased estimate of the true value.

We provide the following procedure for establishing the 95 percent confidence intervals of the output parameters, using a Monte-Carlo simulation of 10^4 runs to build our confidence interval estimates. To start with, we use the standard errors in m_1, m_2, b_1 and b_2 from the linear regression of the mouse clicks along the road boundary as estimates of σ_{m1}, σ_{m2}, σ_{b1}, and σ_{b2}. These values are the standard deviations for the errors associated with the subscripted parameters, assuming they are approximately normally distributed. Theoretically, the errors in the regression parameters have a *t*-distribution, but since we have 50 data points on each line, our approximation is reasonably good. Next, we generate 10^4

samples for each of the line parameters using a normal distribution with the appropriate standard deviation just found from the standard errors of the linear regression. From these normally distributed samples of m_1, m_2, b_1 and b_2, we obtain 10^4 samples of u_0 and v_0 using Equation (83). These samples of u_0 and v_0 will form the basis for further Monte-Carlo simulations to determine the statistical properties of f and the other output parameters.

When determining the spread of τ_L, we note that it is affected both by errors in human observations and errors in the physical spacing between lane markers (i.e., inconsistent values of L). In other words, each estimate of τ_L from a pair of points observed by a human contains observer error and a bias due to imperfections in placing the lane markers. Using the average value of v_0 to calculate a value of τ_L from each of 150 pairs of points (6 different distances times 25 observations), we obtain a distribution for τ_L such as that contained by Figure 90(a) for the example scene in Figure 89. However, we again emphasize that the distribution of τ_L is not the same as the distribution of its mean value. Thus, we calculate the mean value of τ_L for the 150 pairs of points using each of the 10^4 samples of v_0 generated above. This yields a much tighter distribution for the mean of the τ_L data, as shown in Figure 90(b).

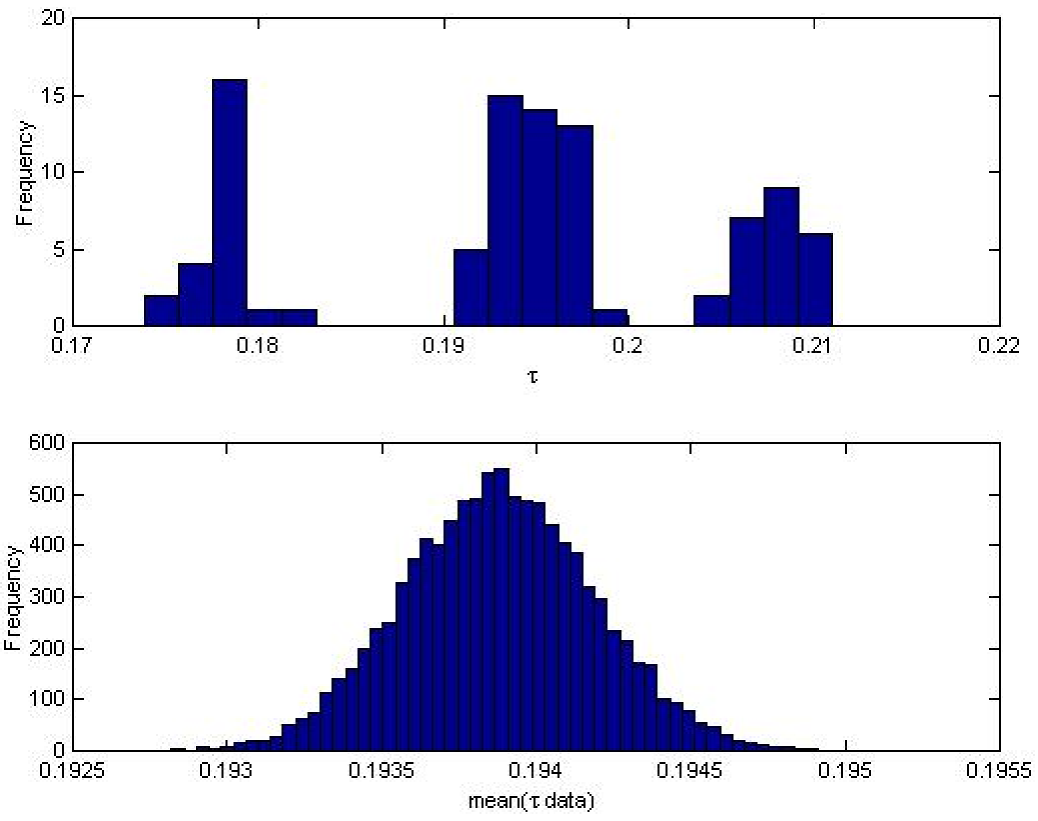


Figure 90. a) Histogram of the values of τ_L calculated from 150 human observations and an estimate of v_0 for the example scene in Figure 89. b) Distribution of τ_L found by estimating its mean value using the distribution of v_0 for the example scene in Figure 89.

The variability in the outputs of Method 3 is primarily due to three inputs, which contribute equally. It is critical to obtain an accurate estimate of τ_L from the hand-calibrated data because it is one of these inputs to which Method 3 is most sensitive.

Given the 10^4 samples of b_1, b_2, τ_L, w, u_0, and v_0, we may then calculate f, θ, ϕ, d, S', and u_1 using the equations previously derived. By sorting the calculated values, we can determine the 2.5 percentile and 97.5 percentile points, yielding the 95 percent confidence interval for each parameter. We used a normal distribution for w and L such that each has a ±1-foot 95 percent confidence interval. Since our best estimates of the input parameters each has an approximately normal distribution, the distributions of the output parameters are also approximately normal, as shown in the results of figures 91 and 92 or the example from Figure 89.

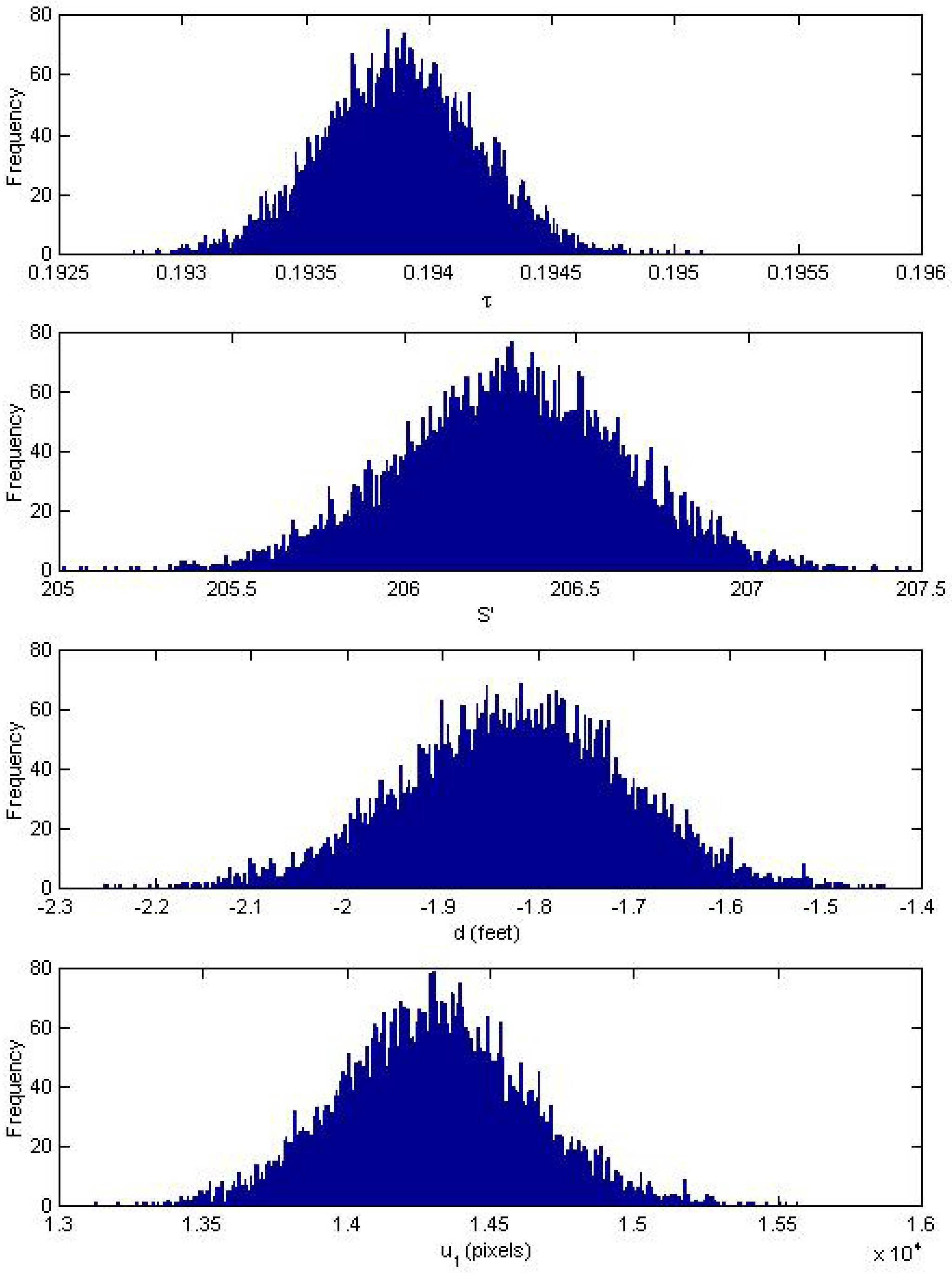


Figure 91. Sample histograms of τ_L, S', d, and u_1 from 10,000 runs of a Monte-Carlo simulation used to generate confidence intervals for the example of Figure 89.

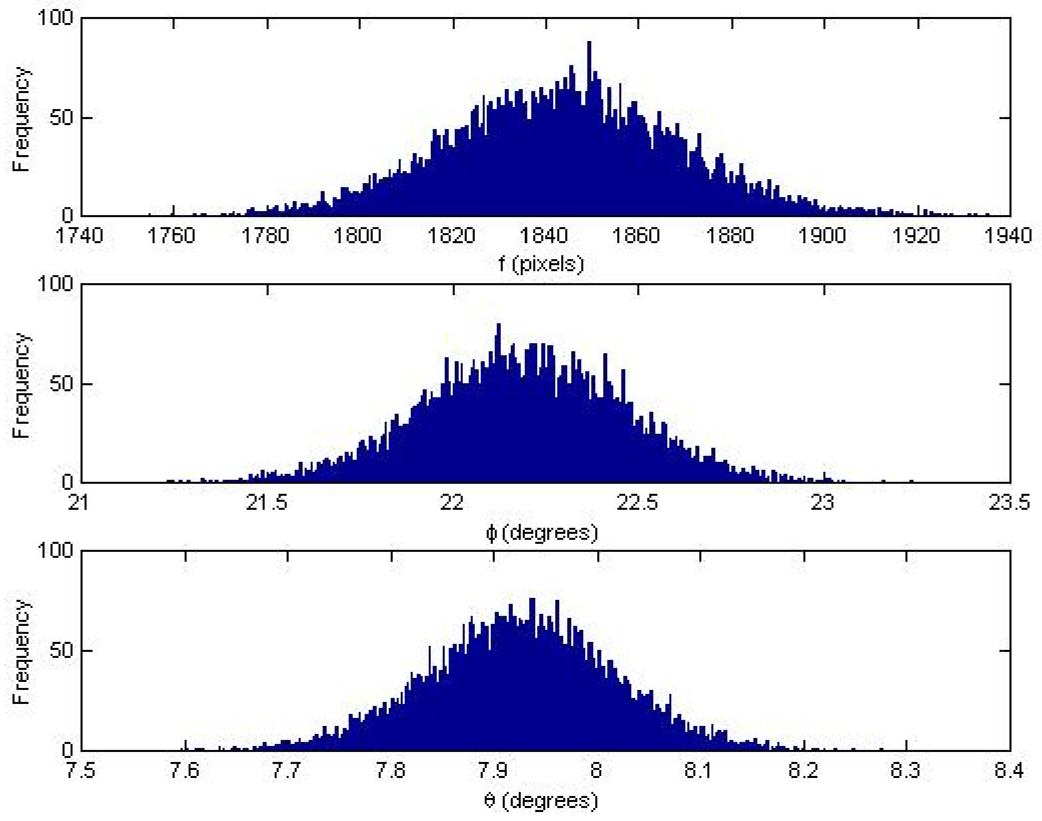


Figure 92. Sample histograms of f, θ, ϕ from 10,000 runs of a Monte-Carlo simulation used to generate confidence intervals for the example of Figure 89.

On the basis of conversations with WSDOT [54], we know that a machine originally paints the lane stripes and marks where to place the RPMs to mark the lanes. These machines have an accuracy of better than 1 percent. However, the markers must eventually be repainted or replaced. To estimate the parameters of the actual distribution of L, we measured the individual distances from tip-to-tip of 101 lane marker intervals using a tape measure on a section of the express lanes. The nominal interval specified in the road design is 40 feet [42]. However, as shown in the histogram for the data in Figure 93, the data have a fairly wide, centroidal distribution, with a standard deviation of 3.5 feet. The mean is 40.5 feet, with a 95 percent confidence interval of $39.9 \leq \mu \leq 41.2$ feet. Thus, the data are unbiased with a bigger spread than we might prefer. However, all three people who helped gather the data agreed that the the section of freeway from which the data were obtained was noticeably worse than other sections of the express lanes and the regular traffic lanes. This is probably due to poor maintenance and to the fact that cars travel bidirectionally and therefore wear off the markers on both ends. Thus, we offer these data as a worse-case scenario rather than as typical data for Seattle freeways.

Averaging the distances from multiple lane intervals can help to noticeably reduce the variability in the estimate of L, i.e., on the order of $\sigma/N^{1/2}$ as expected from the law of large numbers. In this way, the average of four measurements should yield a 95 percent confidence interval that remains within 10 percent of the mean, even for this noisy set of data.

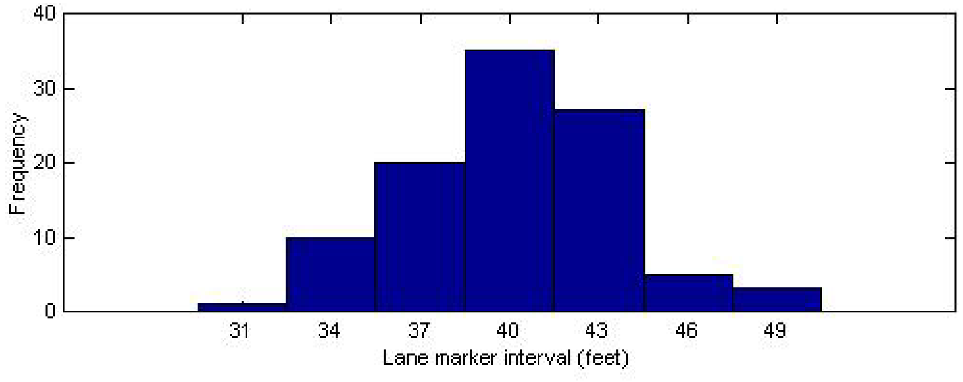


Figure 93. Histograms of the lane marker intervals measured using a distance wheel.

4.3 Camera calibration results

4.3.1 Overview of sample scenes

We captured four image sequences, two from one camera and two from another camera. For each camera, one sequence was from a close-up view and the other was from a normal far-field perspective. Sample images contained in Figure 94 illustrate several interesting and challenging properties of the scenes. Note that even with the extreme down angle in Figure 94(a,c), the slope of the vehicle lines perpendicular to the road is not very noticeable. However, the large value of v_0 is quite noticeable and distinguishes this type of scene from those in Figure 94(b,d). Also note the road imperfections and extraneous, distracting features that are present in the close-up scenes that are not visible in the far-field perspective. The mix of sunshine and shadows in the scene of Figure 94(d) will also present some unique challenges to our algorithms because of the additional dynamic and static strong edges. In addition, because the road is curved, the value of d measured in Figure

94(c) may not necessarily apply to every camera viewpoint, since the apparent road direction differs as the road bends; calibration Method 2 may fail in this case.

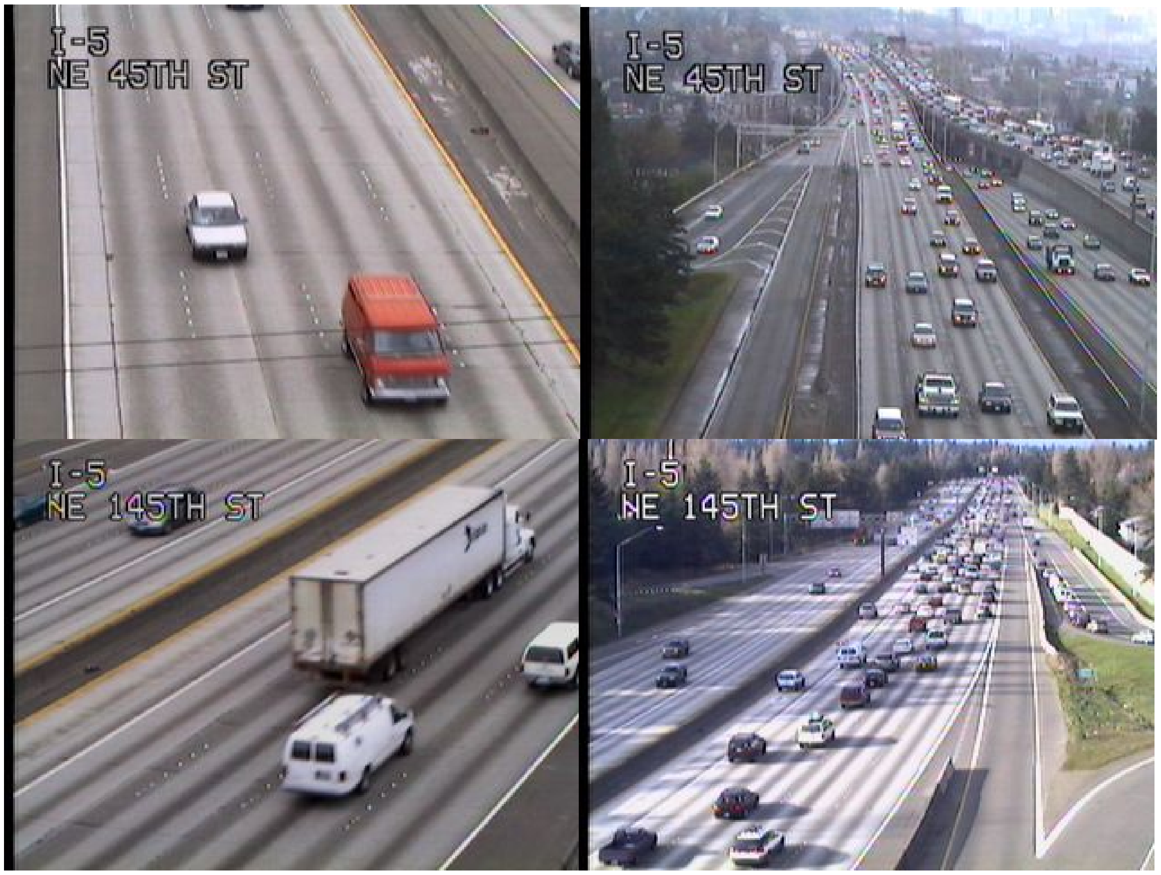


Figure 94. Sample images from the image sequences for which we calibrated the camera. From left-to-right, top-to-bottom: a) Close-up view of NE 45th St. b) Far-field view of NE 45th St. c) Close-up view of NE 145th St. d) Far-field view of NE 145th St.

4.3.2 *Image processing results*

We applied the image processing techniques of Chapter 3 to each of the sequences of Figure 94. The algorithms successfully found a rough estimate of the vanishing point and lane boundaries. They were also successful in refining the vanishing point estimate and were able to locate the road boundary lines. Finally, in each sequence, the algorithms were able to estimate the lane marker interval. These results are presented visually on the average top-hat feature images in the following four figures. The exception is the background image used in Figure 97 because the top-hat image was too distracting.

The lane boundaries are quite accurate in Figure 96 and Figure 98, where it matters for tracking vehicles. The activity maps for the scenes of figures 95 and 97 are quite spread out, and the pan angle is fairly large, so the lane segmentation is noticeably distorted.

The road boundaries are reasonably strong in all the images, though the right and left boundaries of Figure 94(b) and Figurer 94(d) present some challenges, respectively, because they are farther from the image center than the other boundary. However, because the lane location worked reasonably well, our algorithm is also able to locate the road boundaries very accurately.

We note from the horizontal lines in Figure 95 that our algorithm properly matched the left set of lane markers, but the other two clearly do not match. This is an excellent example of where the separation between lane markers is not exactly uniform. Another deviation is evident in Figure 97 in the upper marker of the middle set.

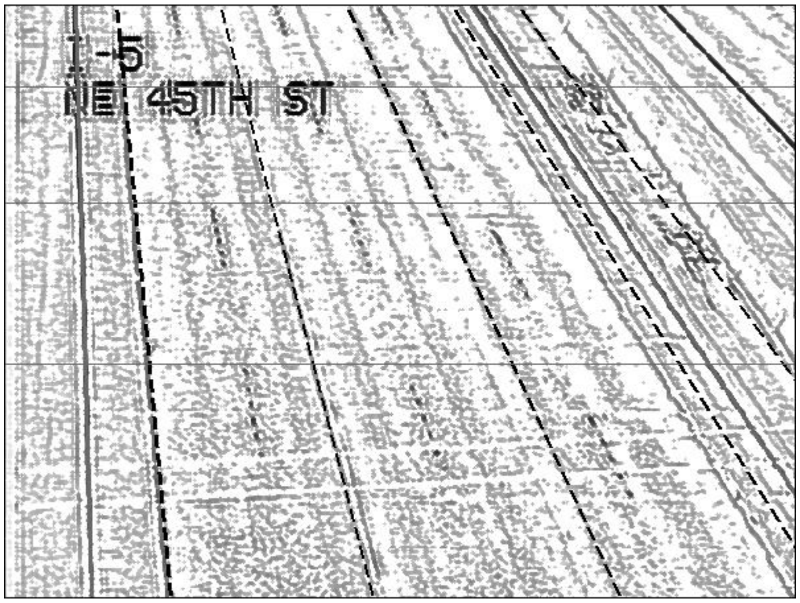


Figure 95. Average top-hat image and features for the sequences of Figure 94(a). The road boundaries are marked with solid lines, and the lane boundaries are marked with dashed lines. The 40-foot intervals are marked with horizontal lines.

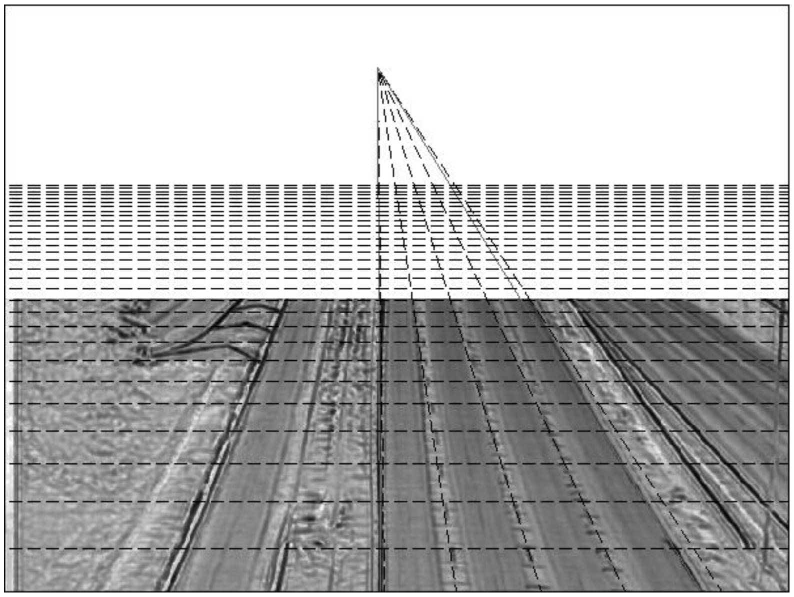


Figure 96. Average top-hat image and features for the sequence of Figure 94(b). The road boundaries are marked with solid lines, and the lane boundaries are marked with dashed lines. The 40-foot intervals are marked with horizontal lines.

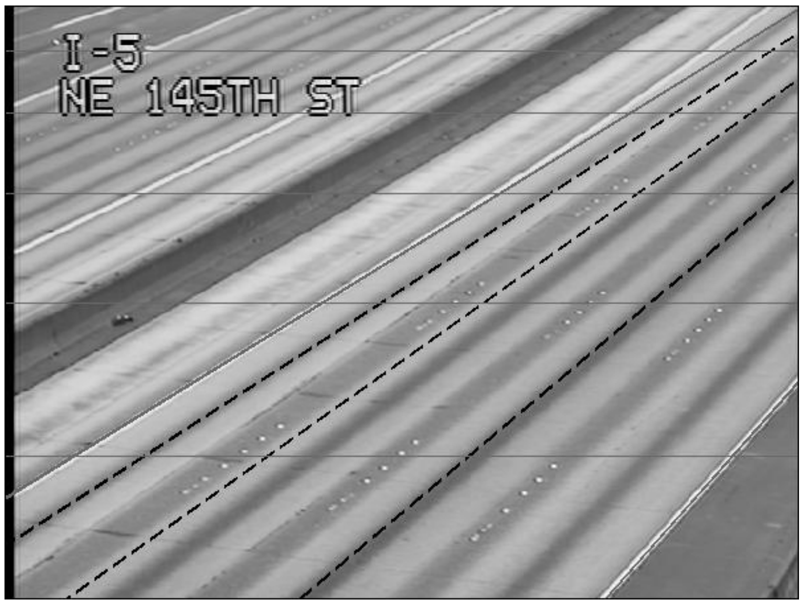


Figure 97. Background image for the sequence of Figure 94(c). The road boundaries are marked with solid lines, and the lane boundaries are marked with dashed lines. The 40-foot intervals are marked with horizontal lines.

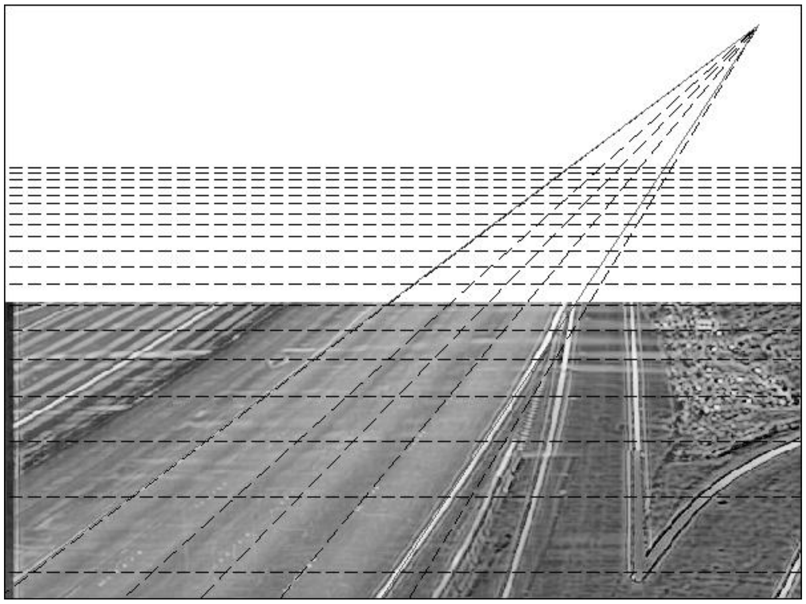


Figure 98. Average top-hat image and features for the sequence of Figure 94(d). The road boundaries are marked with solid lines, and the lane boundaries are marked with dashed lines. The 40-foot intervals are marked with horizontal lines.

In addition to estimating b_1, b_2, u_0, v_0, *and* τ_L, we also applied the techniques of Chapter 3 to estimate u_1, the horizontal coordinate for the vanishing point for the lines perpendicular to the road. Because the scenes in Figure 94(a) and Figure 94(c) were the ones with a vanishing point coordinate $v_0 > H$, our algorithm estimated u_1 only for these scenes and ignored the other two. The histograms for the estimates of u_1 for the two valid scenes are contained in figures 99 and 100, respectively. Given the centroidal tendency of both histograms with a median value of Φ that is less than 87°, we anticipate that our estimates of u_1 will be reasonably accurate for both scenes and, therefore, provide decent estimates of f, ϕ, θ, S', and especially d.

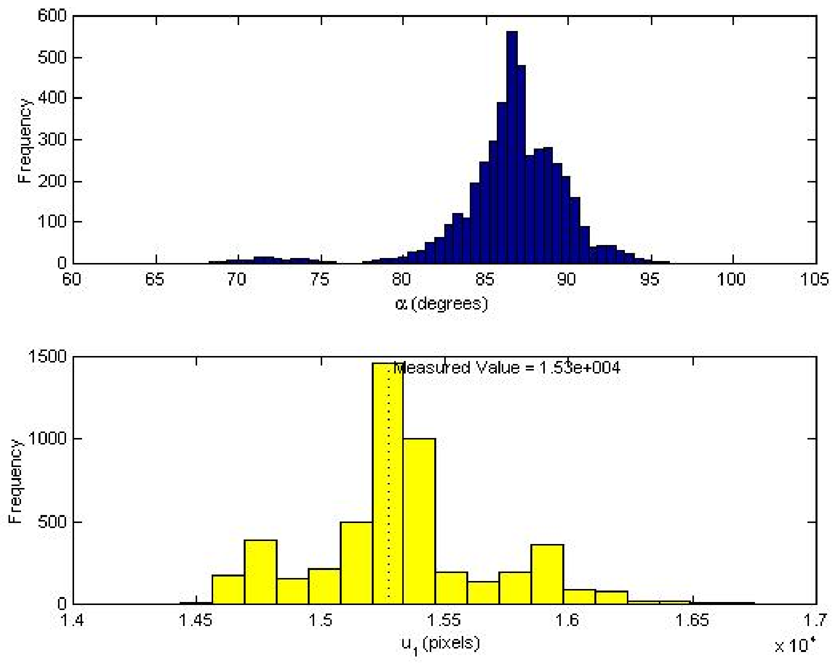


Figure 99. Histograms from estimating the vanishing point (u_1,v_0) for the scene in Figure 94(a). a) Distribution of line angles. b) Distribution of u_1 estimates.

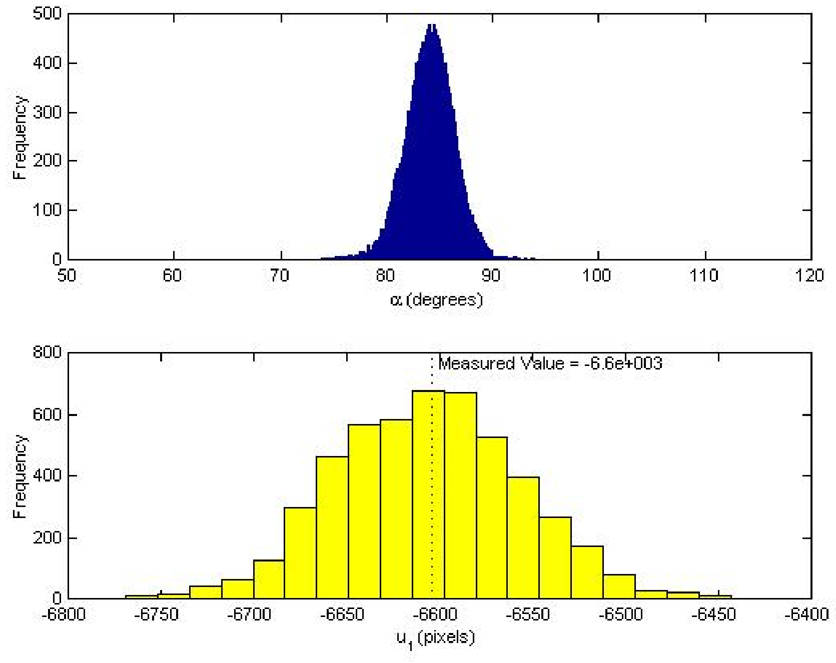


Figure 100. Histograms from estimating the vanishing point (u_1,v_0) for the scene in Figure 94(c). a) Distribution of line angles. b) Distribution of u_1 estimates.

4.3.3 Results for close-up scenes

As presented in Chapter 2, camera calibration Method 1 assumes that we can estimate u_1, the *u*-coordinate of the vanishing point for the lines perpendicular to the road. This information, together with u_0, v_0, b_1, and b_2, enables us to estimate the camera parameters f, ϕ, and θ, and the road distance scale factor S'. More importantly, we can also estimate d, the distance of the camera to the road, which should be fixed for any given camera geometry (neglecting road curvature). As pointed out earlier, Method 1 is only appropriate when the camera has been specially positioned by the operator with large values of ϕ, θ, and v_0, as in the scenes of Figure 94(a) and Figure 94(c).

In Chapter 2, we also derived Method 3, which uses estimates of the vanishing point (u_0,v_0) and the interval τ_L between tips of the lane markers to calibrate the distance along the road. By measuring b_1 and b_2 and assuming knowledge about the width of the road, we can also estimate the parameters of the camera and its position relative to the road.

We applied methods 1 and 3 to the scenes in Figure 94(a) and obtained estimates of all the camera parameters in Table 7. Before we compare the results to the hand-calibrated estimates, we note that the automated results used exactly the same image processing techniques to estimate u_0, v_0, b_1, and b_2. However, Method 1 used an image processing technique to estimate u_1 and derived an estimate of τ_L from u_1. On the other hand, Method 3 estimated τ_L first and then used this value to estimate u_1.

Table 7. Calibration results for the scene in Figure 94(a).

	Hand-calibrated 95% confidence intervals (estimated value)			Method 1 95% confidence intervals (estimated value)			Method 3 95% confidence intervals (estimated value)		
u_0 (pixels)	-278.10	(-277.17)	-276.25	-270.25	(-274.70)	-280.34	-270.25	(-274.70)	-280.34
v_0 (pixels)	749.69	(751.81)	753.89	725.66	(740.40)	752.84	725.66	(740.40)	752.84
u_1 (pixels)	$1.37\text{x}10^4$	($1.44\text{x}10^4$)	$1.49\text{x}10^4$	$1.46\text{x}10^4$	($1.53\text{x}10^4$)	$1.60\text{x}10^4$	$1.29\text{x}10^4$	($1.37\text{x}10^4$)	$1.45\text{x}10^4$
b_1 (pixels)	-255.74	(-255.57)	-255.41	-255.63	(-255.13)	-254.63	-255.63	(-255.13)	-254.63
b_2 (pixels)	216.37	(216.56)	216.76	215.72	(216.22)	216.72	215.72	(216.22)	216.72
τ_L	0.1933	(0.1939)	0.1945	0.1764	(0.1884)	0.2006	0.1960	(0.1983)	0.2006
f (pixels)	1797	(1843)	1891	1855	(1910)	1969	1740	(1795)	1851
ϕ (deg.)	21.67	(22.19)	22.71	20.39	(21.19)	21.91	21.58	(22.41)	23.20
θ (deg.)	7.753	(7.927)	8.103	7.457	(7.639)	7.847	7.797	(8.052)	8.346
S'	205.7	(206.3)	206.9	205.0	(212.3)	220.1	199.5	(201.7)	203.9
$\Delta Y'$ (feet)	39.89	40.0	40.12	39.75	41.17	42.68	38.68	39.11	39.54
d (feet)	-2.06	(-1.82)	-1.60	-2.17	(-1.66)	-1.01	-2.59	(-2.07)	-1.46

Table 7 contains the calibration results for the scene of Figure 94(a). We note that our algorithm estimated b_1 and b_2 to better than ½-pixel accuracy of the hand-calibrated result, and indeed our 95 percent confidence intervals include the estimate provided from the human observer. Similarly, with u_0 and v_0, the 95 percent confidence interval for the results of the algorithm contains the hand-calibrated estimate, with relative errors of only 0.9 percent and 1.5 percent for u_0 and v_0, respectively.

We now compare the confidence intervals for u_1 and τ_L to the hand-calibrated estimates. Method 1 estimates a confidence interval for u_1 that is too tight and does not contain the hand-calibrated estimate. Method 3 produces a similarly flawed result for τ_L. On the other hand, the derived intervals for τ_L and u_1 from methods 1 and 3, respectively, do contain the hand-calibrated estimate. Thus, we infer that our confidence intervals for the measured parameters u_1 and τ_L are overly narrow for methods 1 and 3, respectively.

Method 1 provides a fairly accurate result relative to the hand-calibrated reference, and Table 7 shows that the hand-estimates of *S'* and *d* are within the computer-estimated confidence interval. However, the confidence intervals from the algorithm for *f*, ϕ, and θ do not contain the hand-calibrated estimates for these parameters. On the other hand, the computer-generated confidence intervals for *f*, ϕ, θ, and *d* from Method 3 contain the hand-calibrated estimates, though the confidence interval for *S'* does not. However, we note that our confidence intervals for *w* and *L* were chosen somewhat arbitrarily, and expanding them would eventually lead to the inclusion of the hand-calibrated estimate.

We also applied methods 1 and 3 to the scene in Figure 94 (c) and obtained estimates of all the camera parameters, which are contained in Table 8. Comparing the results of Method 3 to the hand-calibrated results shows that the Method 3 intervals for u_0, v_0, b_1, b_2, and τ_L do not contain the hand-calibrated estimate. This indicates that our confidence interval calculations may yield intervals that are too restrictive or somehow do not account for some of the variability that is present. Nonetheless, the confidence intervals of Method 3 for *f*, ϕ, and θ do contain the hand-calibrated estimate, although the intervals for the (more important) parameters *S'* and *d* do not.

In contrast to the results of Method 3, Method 1 yields values that are not even close to the hand-calibrated estimate. This is due to its estimate of u_1, which was obtained directly from the set of images by measuring the angles of vehicle lines that were judged to be perpendicular to the road. Fortunately, because of the strong down-angle, the background image contains linear features that we believe are perpendicular to the road in 3-D. These features can provide independent verification of the vanishing point for the lines perpendicular to the road. We obtained human observations of three of these lines and found their intersection at the vanishing point (u_1,v_1) for the lines perpendicular to the road; these data are contained as a special entry in Table 8. We recall that in our original two-angle camera model, $v_0 = v_1$, i.e., the vertical image coordinate for all sets of parallel lines on the same 3-D plane was the same. However, we hypothesize that the road slope is sufficiently large that this model is no longer correct. The data of Table 8 strongly support this hypothesis, as we note that the 95 percent confidence interval for the hand-estimate of u_1 contains the u_1 estimate from Method 1 (when we substitute the hand-estimate of v_1 for v_0 in our calculations). These results are consistent with our analysis in Chapter 2, which showed that Method 1 is susceptible to nonzero road slope and tilt, whereas Method 3 is relatively immune.

Table 8. Calibration results for the scene in Figure 94(c). * indicates the calculation of u_1 using the hand-calibrated value of v_1. ** denotes alternate calibration procedure using lines perpendicular to the road.

	Hand-calibrated 95% confidence intervals (nominal value)			Method 1 95% confidence intervals (nominal value)			Method 3 95% confidence intervals (nominal value)		
u_0 (pixels)	934.77	(942.96)	**951.15**	906.34	(909.58)	919.80	906.34	(909.58)	919.80
v_0 (pixels)	625.60	(632.35)	639.10	595.10	(600.49)	604.07	595.10	(600.49)	604.07
u_1 (pixels)	-3114	(-2816)	-2538	-6701	(-6604)	-6507	-3093	(-2861)	-2572
u_1** (pixels)	-6233	-5586	-5044	-5904*	-5824*	-5740*			
v_1** (pixels)	452.8	520.3	602.0						
b_1 (pixels)	-63.25	(-62.84)	-62.43	-61.42	(-60.92)	-60.42	-61.42	(-60.92)	-60.42
b_2 (pixels)	377.36	(378.25)	379.15	377.21	(377.71)	378.21	377.21	(377.71)	378.21
τ_L	0.1671	(0.1690)	0.1709	0.1233	(0.1311)	0.1383	0.1708	(0.1719)	0.1730
f (pixels)	1413	(1501)	1590	2359	(2376)	**2404**	1413	(1497)	1568
ϕ (deg.)	21.6	(22.9)	24.2	14.0	(14.2)	14.3	20.9	(21.9)	23.0
θ (deg.)	-31.4	(-30.1)	-28.8	-20.6	(-20.4)	-20.2	-30.8	(-29.4)	-28.5
S'	234.0	(236.6)	239.4	299.8	(305.2)	312.0	231.2	(232.7)	234.1
$\Delta Y'$ (feet)	39.55	40.00	40.46	50.66	51.58	52.73	39.07	39.32	39.56
d (feet)	-94.9	(-93.9)	-92.9	-102.9	(-100.2)	-98.4	-93.7	(-92.4)	-91.5

4.3.4 Results for far-field scenes

As presented in Chapter 2, camera calibration Method 2 assumes that we can have an estimate of d, the perpendicular distance between the camera and the road. This information, together with u_0, v_0, b_1, and b_2, enables us to estimate the camera parameters f, ϕ, and θ, and the road distance scale factor S'. We anticipate from our sensitivity analysis of Chapter 2 that Method 2 is more likely to fail on a straight-on scene (i.e., Figure 94(b)) than on a scene where the camera is located farther from the road (i.e., Figure 94(d)). We may use Method 3 any time we are able to estimate the lane marker interval; our algorithm successfully estimated τ_L for both far-field scenes, as shown in Section 4.3.2.

We applied methods 2 and 3 to the far-field scene in Figure 94(b). However, Method 2 gave completely wrong results, even when the hand-estimated interval for *d* was substituted. For example, given our knowledge of the focal length range of the camera, it is obvious that a confidence interval of $226 < f < 371$ pixels is physically impossible. Thus, Table 9 contains a comparison of just the hand-calibrated estimates with the estimates from Method 3.

Examining the fundamental measurement quantities u_0, v_0, b_1, b_2, and τ_L, we note that the nominal values seem fairly close in absolute terms but are somewhat distant in relative terms. We measure the relative deviation error E_{rel} as a fraction normalized to as follows:

$$E_{rel} = 1 - \frac{computer\ estimate}{hand\ estimate} \tag{84}$$

This creates a situation in which, of the derived parameters, the confidence intervals for ϕ, θ, and *d* overlap, but those of *f* and *S'* do not. However, we note that the estimated values for the fundamental parameters that affect speed estimation, i.e., v_0 and *S'*, differ by only 2.8 percent and 5.5 percent from the hand-estimates, respectively, so we expect a similarly small relative bias in the speed estimates.

Table 9. Calibration results for the scene in Figure 94(b).

	Hand-calibrated 95% confidence intervals (nominal value)			Method 3 95% confidence intervals (nominal value)		
u_0 (pixels)	-15.66	(-13.68)	-11.70	-16.05	(-15.44)	-14.60
v_0 (pixels)	178.51	(182.74)	186.98	190.46	(187.86)	185.84
u_1 (pixels)	2.13×10^5	(2.48×10^5)	2.92×10^5	2.33×10^5	(2.49×10^5)	2.69×10^5
b_1 (pixels)	-15.08	(-13.50)	-11.93	-14.71	(-14.35)	-14.07
b_2 (pixels)	98.37	(100.31)	102.24	100.31	(100.71)	101.06
τ_L	0.0514	(0.0516)	0.0518	0.0486	(0.0489)	0.0493
f (pixels)	1779	(1828)	1878	1907	(1951)	1997
ϕ (deg.)	5.51	(5.71)	5.91	5.36	(5.50)	5.66
θ (deg.)	0.36	(0.43)	0.49	0.42	(0.45)	0.47
S'	772.2	(774.9)	777.6	812.4	(817.8)	823.2
$\Delta Y'$ (feet)	39.9	40.0	40.1	41.9	42.2	42.5
d (feet)	-1.03	(0.02)	1.07	-0.01	(0.39)	0.68

We also applied methods 2 and 3 to the far-field scene in Figure 94(d), yielding the measurements and calibration results of Table 10. When applying Method 2, we must use an estimate of *d* obtained by either Method 1 or Method 3. However, referring to the last row of Table 8, we note that methods 1 and 3 gave very different results. When using the confidence intervals for *d* from Method 1 for the scene in Figure 94(d), we obtained imaginary numbers that are clearly wrong. Thus, even though the estimate from Method 1 was correct in terms of its image processing, the reduced-order camera model gave an incorrect result because of the nonzero slope and tilt of the road. Applying the confidence interval for *d* from the Method 3 result gave very wide confidence intervals, as shown in the Method 2 results of Table 10. This is consistent with our sensitivity analysis of Chapter 2.

Nevertheless, the estimate of *S'* (which is critical for speed estimation) was off by only about 8 percent, which is surprisingly low given the immense 95 percent confidence interval. The results for Method 3 are also contained in Table 10, which shows Method 3 to be vastly preferable to Method 2 in terms of its precision, though its accuracy relative to the hand-calibration result is roughly comparable to that of Method 2.

Table 10. Calibration results for the scene in Figure 94(d).

	Hand-calibrated 95% confidence intervals (nominal value)			Method 2 95% confidence intervals (nominal value)			Method 3 95% confidence intervals (nominal value)		
u_0 (pixels)	289.43	(293.99)	**298.55**	284.07	(285.87)	287.27	284.07	(285.87)	287.27
v_0 (pixels)	217.73	(222.02)	226.32	225.46	(223.17)	221.36	225.46	(223.17)	221.36
u_1 (pixels)	-9826	(-9271)	-8750	-7.911×10^4	(-6633)	-3474	-9577	(-9106)	-8671
b_1 (pixels)	-6.83	(-5.59)	-4.33	-9.90	(-9.40)	-8.90	-9.90	(-9.40)	-8.90
b_2 (pixels)	143.13	(144.16)	145.18	139.59	(140.09)	140.59	139.59	(140.09)	140.59
τ_L	0.0795	(0.0800)	0.0804	0.0262	**(0.0867)**	0.1142	0.0743	(0.0749)	0.0754
f (pixels)	1590	(1636)	1682	972	(1359)	4729	1559	(1598)	1638
ϕ (deg.)	7.46	(7.73)	8.00	2.69	(9.33)	12.96	7.74	(7.95)	8.18
θ (deg.)	-10.42	(-10.10)	-9.78	-16.02	(-11.73)	-3.44	**-10.30**	(-10.05)	-9.79
S'	497.6	(500.3)	503.0	352.6	(461.2)	1506	530.8	(534.3)	538.0
$\Delta Y'$ (feet)	39.8	40.0	40.2	28.2	36.9	120.	42.5	42.7	43.0
d (feet)	-88.9	(-86.5)	-84.1				-95.0	(-93.0)	-91.0

4.4 Summary

We analyzed four scenes described in detail in Section 4.3: one close-up and one far-field view for each of two cameras. We generated a reference for the output parameters of the system on the basis of our reduced-order camera and scene model of Chapter 2 by using the observations of a person that clicked on multiple features in the image. Our image processing algorithms successfully estimated the input parameters b_1, b_2, u_0, and v_0, which are critical for the success of each calibration method.

We obtained mixed results when applying Method 1 to the close-up scenes, i.e., where v_0 exceeded H. Our image processing algorithm accurately estimated the slopes of the perpendicular lines in both scenes. However, the road slope and tilt were negligible for one of the scenes but not for the other one. Consequently, the camera calibration was fairly accurate for the former scene but completely wrong for the latter. These experiments support our recommendations in Chapter 2 that Method 1 be applied only when the road is known to be essentially flat.

The results for Method 2 strongly support the sensitivity analysis of Chapter 2. Just like the computer simulation of the similar scene in Figure 8(c), Method 2 completely failed for the scene where the camera had a straight-on view. In the scene with a moderate pan angle, Method 2 gave an estimate of *S'* that was within 8 percent of the hand-estimate, albeit with an extraordinarily wide confidence interval. Such a wide confidence interval strongly supports our conclusions about the sensitivity of this method that were predicted in Chapter 2. Nevertheless, the latter scene (i.e., Figure 94(d)) approximately conforms to our recommendations at the end of Chapter 2. Specifically, the camera is located on an overpass, so it is about 70 feet above the road, and if we subtract the road width of 48 feet, then the camera is located only about 45 feet from the nearest road boundary marker.

Of the three methods, Method 3 was the only one we successfully applied to all four scenes. Even though the hand-calibrated and Method 3 confidence intervals did not always overlap, the output parameters for each scene were consistently within 7.5 percent relative to the hand-based estimates. Although this is not an exhaustive study, the Method 3 approach successfully surmounted difficulties (e.g., extraneous road features, shadows, closely spaced lane markers) that typically occur in freeway scenes. Thus, we can conclude that at the very least, this algorithm holds a great deal of promise for calibrating the roadside cameras for the wide variety of orientations we expect them to encounter

5 ESTIMATING THE AVERAGE VEHICLE SPEED

5.1 Introduction

In Chapter 1, the original problem statement outlined our goal of estimating the average vehicle speed for each of the lanes for a section of freeway. We also discussed the previous work in tracking vehicles and other objects in image sequences. Traditional methods that track objects individually are inadequate for the present task because the vehicles often occlude one another as a result the camera perspective, especially during high traffic volumes. Instead of estimating the image position of individual vehicles, we now present our algorithm, which treats the vehicles like a particle flow problem and estimates the average flow speed along each of the lanes of traffic. This approach estimates the average speed in a natural way without identifying the positions of individual vehicles.

Our algorithm assumes that vehicle lane masks have been extracted from the image sequence and that the vanishing point coordinate v_0 has been estimated. The algorithm can be used with a wide range of image sizes, though the spatial resolution is lower at lower image resolutions. The vehicles typically move rapidly through the useful range of the image, so we require a known sampling frequency of at least 3 frames/second to properly track the vehicles.

We now present the tracking algorithm, its connection to the computer vision equations, and the results for several image sequences.

5.2 Tracking algorithm

5.2.1 Theoretical model

As described in Chapter 2, when the road can be modeled as a flat plane and we reduce the camera model to only two camera angles, the position along the road has a very simple relationship to the vertical position in the image:

$$Y' = \frac{h}{f}\csc^2(\phi)\sec(\theta)v_0\left(\frac{v}{v_0 - v}\right) = S'\left(\frac{v}{v_0 - v}\right) \qquad (85)$$

We found it convenient to define the difference between two vertical positions *v₁* and *v₂* as follows:

$$\tau_{12} \equiv \frac{v_2}{v_0 - v_2} - \frac{v_1}{v_0 - v_1} = \left(\frac{v_0 (v_2 - v_1)}{(v_0 - v_2)(v_0 - v_1)} \right) \tag{86}$$

In Chapter 2, we showed that if we know the physical distance $L = \Delta Y'$ between two known image coordinates v_a and v_b, then the scale factor *S'* for the scene can be determined:

$$\begin{aligned} L = \Delta Y' &= S' \left(\frac{v_0 (v_b - v_a)}{(v_0 - v_b)(v_0 - v_a)} \right) \\ S' &= \frac{L}{\tau_L} \end{aligned} \tag{87}$$

Thus, to find the physical distance along the road between any two image coordinates v_1 and v_2, we simply calculate $\Delta Y' = L\tau_{12}/\tau_L$, where τ_{12} is defined above. Using the same line of thinking as that used in Chapter 3 for estimating τ_L, we now describe the image processing steps necessary to estimate the shift τ_{12} for a lane of traffic directly from the images without specifically identifying v_1 or v_2. This will enable us to calculate the average distance traveled by the vehicles between the known frame interval and thereby calculate the average speed for the lane of traffic.

5.2.2 Image processing

Before we can estimate τ_{12}, we must obtain the features from the moving vehicles. In order to avoid including background features (e.g., shadows) that would bias the estimate of τ_{12} towards a zero shift, we subtract the background from the current image as part of our process of extracting vehicle features.

5.2.2.1 The background image

Not only is the background image a useful presentation tool, but it enables us to perform background subtraction to isolate the vehicles. The background image is simply the scene without any vehicles present. For large sample sizes (500 or more frames), we may estimate the background image as the expected value of the set of frames. Since our system continuously receives data, we need not worry about the possibility of a sparse data set. On occasion, we must manipulate the histogram of the background image before processing it, e.g., the night-time scene in Chapter 4. Having the background image available makes it easier (via image subtraction) to determine whether image features belong to the foreground or background.

5.2.2.2 Vehicle feature extraction

Once we have obtained the background image, the lane masks, and the vanishing point estimate v_0 from our other algorithms, we can appropriately extract vehicle features from each of the images. By properly processing these features in a manner similar to that of Chapter 3, we can estimate τ_{12}, the average linear shift along the road. In Chapter 3, we used the top-hat image as the basis for our features. In the present case, we apply a vertical derivative-of-gaussian convolution kernel to the difference between the background image, I_{bg}, and the i^{th} image frame, I_i, to detect horizontal edges in (I_i - I_{bg}), i.e.,

$$K_2 = -255^{-1}(2\pi)^{-1/2}\sigma^{-3}ne^{-\frac{n^2}{2s^2}} \tag{88}$$

where $n \in \{-3, \ldots , 3\}$ and $\sigma = 1$ for 320 X 240 images. We use a value for σ that slightly smoothes the image yet preserves the details of all the important edges. We normalize the kernel by 255 to uniformly scale the image. Subtracting helps to isolate the features of the moving vehicles. We then take the absolute value of the feature image to equally emphasize both rising and falling edges. Applying the lane mask yields an image similar to that in Figure 101(a). Next, we binarize the image using a threshold of 0.15, which preserves only the moderately strong edges, producing a result similar to Figure 101(b). Binarizing the signal removes a great deal of noise and sharpens the distinctive features of the vehicles later on. Note that we only process the bottom one-third of the image to limit the effects of road curvature and because of greater quantization error higher in the image.

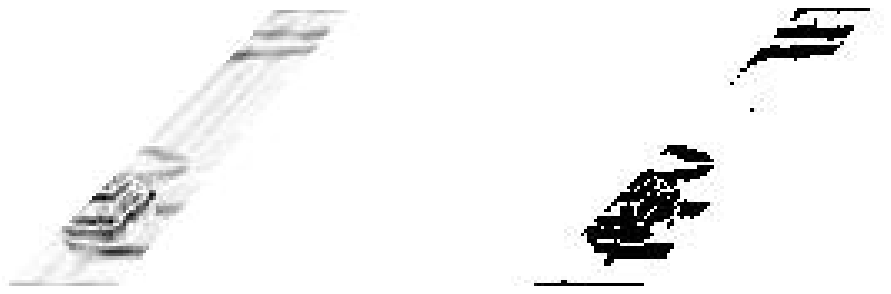


Figure 101. a) Horizontal image gradient features found in a lane of uncongested traffic. b) Binary edge features remaining after thresholding.

Once we have isolated the features for a single lane, we chop off the top and bottom five rows to remove the effects of a finite convolution kernel (note the false edges at the top and bottom of Figure 101(b)). Next, we generate a one-dimensional signal, S_1, by horizontally summing the feature image and normalizing each sample by the number of pixels in each row of the lane mask. This creates a signal ranging between 0 and 1, since the binary image has values of only 0 and 1. Figure 102(a) contains examples of S_1 from the same lane at two adjacent time steps.

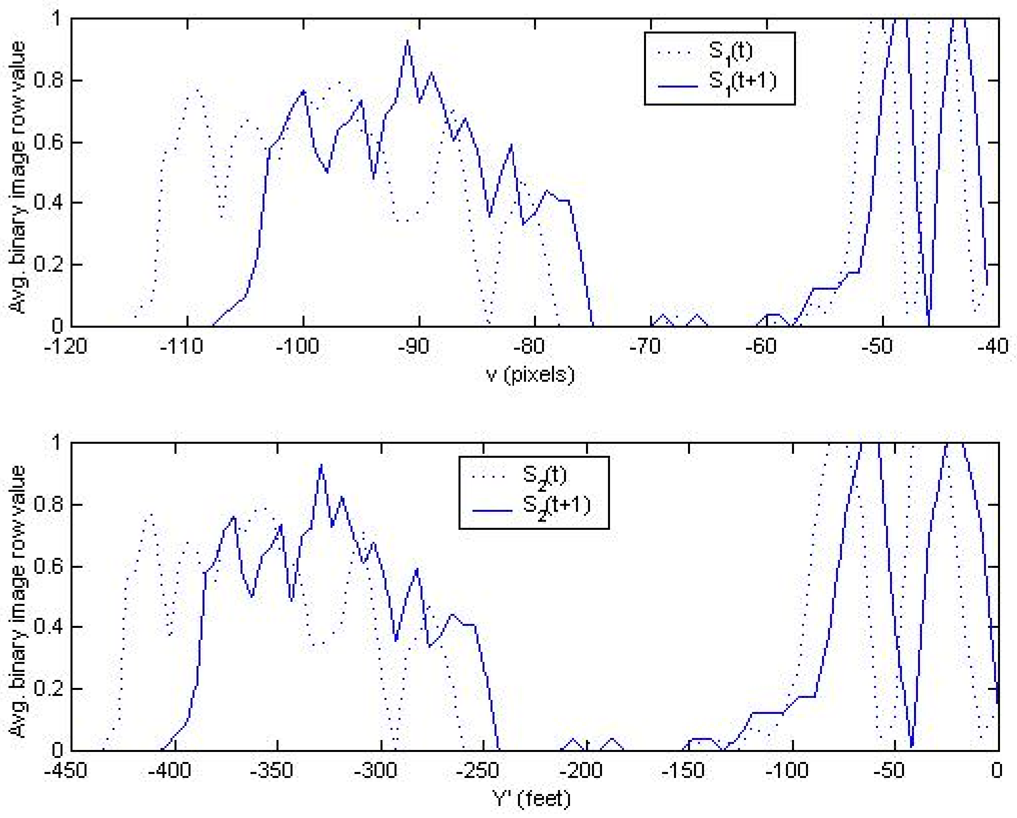


Figure 102. a) Average feature signals $S_1(t)$ and $S_1(t + 1)$ (nonlinear spatial sampling). b) Average feature signals $S_2(t)$ and $S_2(t + 1)$ (linear spatial sampling).

As shown in the two signals of Figure 102(a), there is a reasonably strong correlation between time samples of S_1. However, the figure also illustrates some of the nonlinear spatial distortion for which we must use Equation (43) to compensate. Specifically, the shift of the new signal relative to the older sample is much smaller on the right than on the left of $S_1(t)$, although we acknowledge that the vehicle on the left in $S_1(t+1)$ is noticeably more compressed than in $S_1(t)$. We reassign the spatial sampling points as follows to create a linear scaling for S_1:

$$
\begin{aligned}
&Y' = S' \frac{v}{v - v_0} \\
&r = \frac{3600 \text{ seconds / hour}}{5280 \text{ feet/mile}} (Y' \text{ feet}) \left(\frac{1}{\Delta t \text{ seconds}} \right)
\end{aligned}
\tag{89}
$$

In other words, if we resample S_1 such that *Y'* is sampled linearly, then we obtain a signal that is a function of position along the road, e.g., Figure 102(b). If we perform further scaling as described in Equation (89), where Δt is the time between image frames, then the shift in the signal is an estimate of the vehicle speed in miles/hour. We resample S_1 using linear interpolation to create a new signal, $S_2(r)$, that has a linear resolution in r of 0.5 miles/hour.

5.2.3 Signal processing for speed estimation

As shown in Figure 102(b), the adjacent time samples $S_2(r,t)$ and $S_2(r, t + 1)$ exhibit a high degree of correlation when properly aligned. In fact, with the proper scaling described above, the shift between the two signals is a single estimate of the mean vehicle speed between times t and $t + 1$. Thus, we proceed with calculating the noncircular cross-covariance function $C_{t,t+1}(\tau)$ of the two sequences of length N as follows:

$$
C_{t,t+1}(\tau) = \frac{\sum_{n=-N}^{N} \left(S_2(r+n,t) - E[S_2(r+n,t)]\right)\left(S_2(r+n,t+1) - E[S_2(r+n,t+1)]\right)}{\sqrt{\operatorname{var}(S_2(r,t)) \operatorname{var}(S_2(r,t+1))}} \tag{90}
$$

where var(·) denotes the spatial variance of each signal. We use the noncircular cross-covariance function rather than calculating the circular version with the discrete Fourier transform because there is no guarantee that the vehicles are equally spaced. This also reduces the peak magnitudes that are far from zero because of the natural triangular windowing from the noncircular convolution. Continuing with our example from Figure 102, we obtained a cross-covariance function similar to the signal in Figure 103. The negative offset of the peak indicates that the vehicles are traveling up in the image away from the camera. From Figure 102, we note that the data based on the binary image typically exhibit no noise on the floor of the signal with distinctive triangular and rectangular features that often create a strongly peaked cross-covariance function.

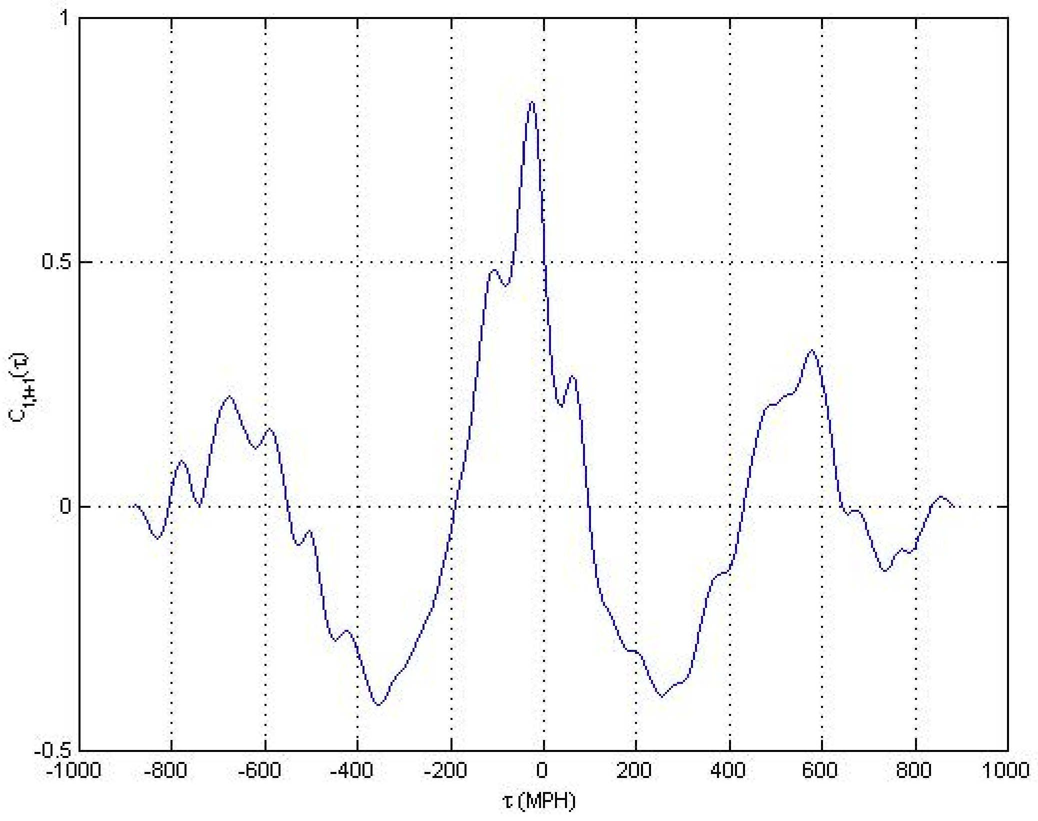


Figure 103. Cross covariance function for the two signals of Figure 102(b). The peak occurs at -26 MPH.

Our approach offers many advantages over the state-of-the-art approach to vehicle tracking [23] for the task of estimating mean vehicle speed. 1) All available data are considered when our algorithm estimates the average speed; the averaging is inherent in the method. All other methods, including that of Beymer et al. [23], depend on locating individual vehicles. 2) Our method is robust even should some vehicle features disappear from frame-to-frame. The work of Beymer et al. [23] will lose the vehicle if its features disappear. 3) The normalization factor of $C_{t,t+1}(\tau)$ allows us to set a matching threshold on the basis of well-understood signal processing techniques below which we may disregard the speed estimate. There are no natural criteria for determining when the approach of Beymer et al. [23] has lost a vehicle. 4) The method can handle very large displacements between frames, whereas the work of Beymer et al.[23] limits the search to a neighborhood around the previous vehicle.

We use the location of the maximum peak of the cross-covariance function as an estimate of the average speed. Given our experience with several sequences under various conditions, we require the maximum peak to exceed 0.5 in order for it to be considered a valid estimate.

5.2.4 Temporal tracking

The mean speed of each lane constantly changes over time, by small amounts in normal conditions and by large amounts when a traffic event occurs. Given the cross-covariance approach for estimating the mean speed, we can imagine a variety of possibilities for dynamically estimating the mean speed. One option is to model the past *M* cross-covariance functions as an ensemble. We could then extract the peak from an FIR- or IIR-filtered version of the ensemble. This requires a moderate amount of memory and a moderate amount of complexity. We choose instead to treat each maximum peak of the cross-covariance function that exceeds 0.5 as a noisy measurement of the mean speed. We then apply a Kalman filter to predict and smooth the data. This approach enables us to take advantage of our knowledge of the statistics and state model of the system when we encounter very noisy measurements or fail to record a measurement at a given time step. Besides its advantages as a well-tested and rigorous technique, the Kalman filter is computationally simple and has a very low memory overhead.

5.2.4.1 Kalman filter recipe

The classic discrete Kalman filter assumes that we have a state model in which the next state vector $\mathbf{X}(k+1)$ is a linear combination of the previous state estimate $\mathbf{X}^{+}(k)$ plus some zero-mean gaussian noise vector $\mathbf{u}(k)$, where the noise has a covariance matrix $\mathbf{Q}$ (we give matrices and vectors in **bold**).

$$\mathbf{X}(k+1) = \mathbf{A}(k)\mathbf{X}^{+}(k) + \mathbf{u}(k) \tag{91}$$

$\mathbf{A}(k)$ is the state-update matrix that can change with time. The “innovations” or new information from the system are often modeled by the noise term $\mathbf{u}(k)$. In the discussion below, we denote predictions by the superscript - and the same quantities after updates using system measurements with the superscript +.

We assume that we obtain measurements **Y** that are linear combinations of the states in **X** plus some zero-mean gaussian noise **w**(*k*) having a covariance matrix **R**. We define **M** as the measurement matrix.

$$\mathbf{Y}(k) = \mathbf{M}\mathbf{X}^{+}(k) + \mathbf{w}(k) \qquad (92)$$

In the Kalman filter recipe, we calculate $\mathbf{X}^{-}(k+1) = \mathbf{A}(k)\mathbf{X}^{+}(k)$ on the basis of Equation (91). Next, we obtain the covariance matrix $\mathbf{P}^{-}(k+1)$ for $\mathbf{X}^{-}(k+1)$ by sing the previous covariance matrix $\mathbf{P}^{+}(k)$ as follows:

$$\mathbf{P}^{-}(k+1) = \mathbf{A}(k)\mathbf{P}^{+}(k)\mathbf{A}(k)^{T} + \mathbf{Q} \qquad (93)$$

We then calculate the Kalman gain **K**, which tells us how to optimally (when **u**(*k*) and **w**(*k*) have a known gaussian distribution) incorporate differences between the state prediction $\mathbf{X}^{-}(k+1)$ and any states **Y**(*k*+1) that we were able to measure.

$$\mathbf{K} = \mathbf{P}^{-}(k+1)\mathbf{M}^{T}\left(\mathbf{M}\mathbf{P}^{-}(k+1)\mathbf{M}^{T} + \mathbf{R}\right)^{-1} \qquad (94)$$

$$\mathbf{X}^{+}(k+1) = \mathbf{X}^{-}(k+1) + \mathbf{K}\left(\mathbf{Y}(k+1) - \mathbf{M}\mathbf{X}^{-}(k+1)\right) \qquad (95)$$

$\mathbf{X}^{+}(k+1)$ is the Kalman filter estimate of **X**(*k*+1) given the state model, noise model, and measurements of the system. Finally, we obtain our new estimate $\mathbf{P}^{+}(k+1)$, the covariance for $\mathbf{X}^{+}(k+1)$

$$\mathbf{P}^{+}(k+1) = (\mathbf{I} - \mathbf{K}\mathbf{M})\mathbf{P}^{-}(k+1)(\mathbf{I} - \mathbf{K}\mathbf{M})^{T} + \mathbf{K}\mathbf{R}\mathbf{K}^{T} \qquad (96)$$

5.2.4.2 State model

We use a first-order predictive model with a state vector composed of the speed, *s*, and acceleration, *a*, of the vehicles. We assume we are only able to measure *s*. Thus, our state model is

$$\mathbf{X} = \begin{bmatrix} s \\ a \end{bmatrix}$$
$$\mathbf{Y} = \mathbf{M}\mathbf{X} = \begin{bmatrix} 1 & 0 \end{bmatrix}\begin{bmatrix} s \\ a \end{bmatrix} = s \qquad (97)$$
$$\mathbf{A}(k) = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$$

where Δt is the time step between our measurements.

5.2.4.3 Advancing the filter

It is important to note that unless we make a measurement **Y** that we accept, we allow Δt to increase without advancing the filter. Our acceptance criteria are outlined below. This filtering strategy enables us to increase the noise variance to respond rapidly when we have not made a measurement for a long time. For example, on occasion the frame-grabber may drop frames or an Internet outage may delay or drop the frame transmission. In other cases, we may have rejected measurements for a while before finding one that we accept. We desire the filter to respond quickly to this new information.

5.2.4.4 Kalman filter noise parameters

The choices for the state-update covariance, **Q**, and measurement covariance matrix, **R**, have always been something of an art form. Recently, researchers [59] developed a theory and procedure for determining optimal values for these parameters based on typical measurements **Y** made by the system. We applied this approach to speed data measured by the process above in two cases: free-flowing traffic (60 MPH) and slow to stop-and-go conditions (0-30 MPH). In both cases, the component of **R** representing the measurement variance for s was approximately the square of 10 percent of the nominal speed. Thus, we choose to let $\mathbf{R}(k+1) = (0.1 \cdot s^+(k))^2$, where $s^+(k)$ is the Kalman-updated estimate of the speed at the previous time step.

The choice for **Q** is based on our choice for the variance σ_a^2 of the acceleration, into which we lump all of the innovations of the system (i.e., we assume $\sigma_s^2 = 0$). The details can be found in [1]:

$$\begin{aligned}
&\mathbf{Q} = \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix} \text{ where} \\
&q_{11} = \sigma_a{}^2 (\Delta t)^3 / 3 \\
&q_{12} = q_{21} = \sigma_a{}^2 (\Delta t)^2 / 2 \\
&q_{22} = \sigma_a{}^2 \Delta t
\end{aligned} \tag{98}$$

The choice of σ_a determines the responsiveness of the system to innovations, i.e., measurements that are quite different than the current state. We found that a value of σ_a = 1.0 MPH/s allowed the system to track the acceleration and decelerations of the vehicles while smoothing the overall trajectory. We also note that as Δt increases, the components of **Q** also increase, resulting in a greater Kalman gain **K**, and a faster response to innovations on the state prediction $\mathbf{X}^{-}(k+1)$.

5.2.4.5 Performing measurements

Our system must work under a wide variety of traffic conditions. At certain times of the day, traffic may be sporadic, and there may be no measurement for the system to make. In this case, the cross-covariance function will indicate a speed of 0, especially if there are extraneous horizontal lines on the road, e.g., shadows. If this happens, then the system will be oscillating between the nominal speed S_{avg} and 0 as it attempts to respond to the input measurements. Another problem we can expect is that the cross-covariance function maximum peak may be too low, and therefore, the measurement will be invalid. We wish to handle these situations in an intelligent and robust fashion.

There are three types of measurements that we will specifically ignore. As indicated previously, we ignore any peak maxima that are less than 0.5. Furthermore, in keeping with our choice for **R**, we ignore any speed estimate that deviates more than 30 percent from the most recent Kalman-filtered speed estimate $s^{+}(k)$. The single exception is the regime in which the speed drops below 15 MPH, when we accept any speed measurements that deviate from $s^{+}(k)$ by 5 MPH or less. We also calculate the energy (i.e., variance) $EE_2(t+1)$ of $S_2(r,t+1)$ and reject the measurement if $EE_2(t+1) < 2.5 \times 10^{-4}$, recalling that $-1 < S_2(r,t+1) < 1$ because we normalized by the width of the lane mask. This prevents the filter from responding to data from weak edges in the current image.

5.2.4.6 Starting the Kalman filter

We selected a value for σ_a that smoothes the output but does not respond very rapidly to large changes. This selection, combined with the criteria of Section 5.2.4.5 for ignoring certain measurements, can cause the Kalman filter to never get properly started and ignore all measurements. To avoid this situation, we analyze 20 seconds' worth of data, collecting all of the cross-covariance lags that have a peak of 0.7 or greater with speed estimates ranging from 5-90 MPH. Within this time interval we expect to obtain a reasonable sampling of vehicle speeds under even the most sparse conditions. Using a threshold of 0.7 that is greater than the typical value of 0.5 adds to our confidence in the data that are collected. After collecting the data for all of the lanes, we classify the speed measurements for each lane by using the K-means algorithm with a setting of two clusters. We take the cluster mean with the most votes as our estimate of the speed with which to initialize the Kalman filter; a tie-breaker goes to the greater speed estimate. If we obtain no measurements, we continue to analyze new data in blocks of 20 seconds until our criteria are met by at least three samples.

5.2.4.7 Restarting the Kalman filter

On occasion, particularly in rapidly changing traffic conditions, the Kalman filter may continuously ignore speed measurements because they are out of range of the last value $s^{+}(k)$. If the time Δt since the last accepted measurement exceeds 20 seconds, we examine all of the measurement data since the last accepted measurement (which may exceed 20 seconds). We apply the K-means algorithm to any speed estimates with a cross-covariance peak of greater than 0.7 and signal energy of greater than 2.5 x 10^{-4}. We take the cluster mean with the most votes as our estimate of the speed with which to restart the Kalman filter; a tie-breaker goes to the greater speed estimate. If there has been only one or fewer such estimates, i.e., the K-means makes little sense, then we use the peak of an IIR-filtered cross-covariance function $C_{total}(\tau)$, which updates after processing every image with the following equation:

$$C_{total}(\tau) = 0.95C_{total}(\tau) + 0.05C_k(\tau) \qquad (99)$$

This enables us to track new developments in the mean speed while retaining a memory of all past data. This approach also naturally emphasizes the estimates with the highest peaks, which we can assume are the most reliable.

5.3 Tracking results

To apply our mean vehicle speed estimation algorithm in day-to-day conditions, we must have a feel for its performance under a wide variety of conditions. Typical challenges it will encounter include bright sunlight with shadows, road glare, light conditions at dusk, night-time light conditions, raindrops on the camera lens, sparse traffic, dense traffic, stop-and-go conditions, curved roads, and both large and small pan angles. We applied the mean vehicle speed estimation algorithm to four sequences that, taken together, present most of the challenges our algorithm will face. Figure 94 shows one or more key frames from each of the sequences. Our objective in presenting these results is not to offer an exhaustive study of the precise behavior of our algorithm. Instead, we hope that by illustrating its robustness under a variety of conditions, we can provide a measure of confidence that the algorithm can do well in new situations.

The camera was calibrated using Method 3 for all the sequences, apart from the dark scene and the scene with raindrops on the camera lens. The calibration results for the scenes of Figure 94 (c-e) are contained in Chapter 4 in more detail. As for the first two scenes, our algorithm was unable to find the vanishing point or the vehicle lanes because of the raindrops and darkness that obscured the features in the image. In these two cases, we calibrated the scenes by hand using the method outlined in Chapter 4. However, we believe that our image processing techniques in Chapter 3 are strong enough to locate any features that a human observer can identify if we take the time to fully develop an expert system to handle the specific cases of raindrops and darkness. We leave this for future research.

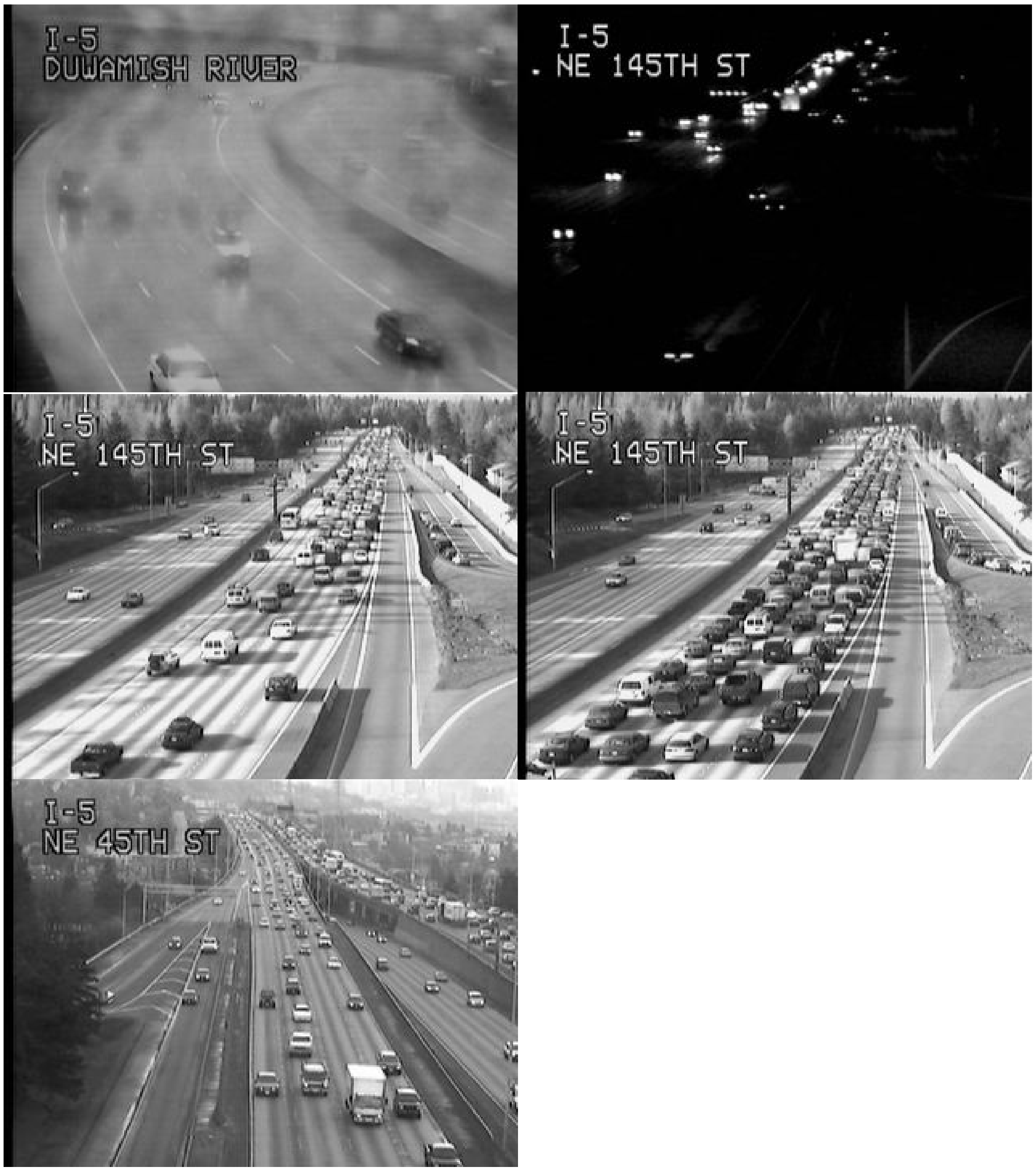


Figure 104. Sample images from the four scenes to which we applied our tracking algorithm (from left-to-right, top-to-bottom). a) Free-flowing daytime traffic scene with raindrops on the camera lens and road glare. b) Dark, sparse traffic scene. c,d) Sunny, slightly curved scene with shadows and congestion that becomes stop-and-go. e) Afternoon commuting scene.

5.3.1 Free-flowing scene with raindrops

The scene contained in Figure 94(a) is very distorted by the multiple raindrops on the camera lens. The images are also challenging for the computer to process because of the strongly curved road and the glare due to the rain. In fact, our standard approach outlined in Chapter 3 was unable to successfully analyze the scene and calibrate the camera, so we calibrated it by hand using the methods described in Chapter 4. The scene contains free-flowing traffic, so we expect data in the range of the speed limit, which is 60 MPH.

Figure 105 contains the raw speed data and the Kalman-filtered mean vehicle speed estimates for the four lanes of traffic. The Kalman filter was started successfully, and the remainders of the sequences are relatively unremarkable, apart from the sparsity of good data and the overall noisiness of the data. It was particularly difficult for our algorithm to get reasonable measurements on lane 3 because it was extremely obscured by the raindrops, and this is evident in the extra noise of lane 3 relative to the other lanes. Several restarts of lane 1 are evidenced by the long straight lines of the Kalman filter signal, i.e., t = 60 and t = 100. This occurred after 20 seconds of no valid measurements, but the restarts seemed adequate to get the system back on track.

We obtained inductance loop data containing 20-second average speed estimates from the same time interval. We also calculated 20-second averages from the output of the Kalman filter. Figure 106 compares the two distributions from the inductance loop sensor and a comparable computer-vision based sensor for lane 1 (slow) and lane 4 (passing). We note that the two distributions have approximately the same mean and rough distribution, indicating a certain equivalence between the sensors. This experiment also demonstrates the robustness of our method of mean vehicle speed estimation even under extremely adverse conditions, i.e., raindrops on the camera lens, road curvature, and road glare.

We present this scene before any of the others for an important reason. Because our third calibration method—the most functional one—is largely mimicking the process we perform by hand, it is reasonable to question the legitimacy of our camera calibration results, particularly given the moderate variability in the lane marker interval. However, this experiment demonstrates that the lane markers contain enough calibration information to place the computer vision results essentially at the center of the distribution of the inductance loop data. This adds a great deal of credibility to the results we present later in which inductance loop data were not necessarily available, but we can see that our algorithm properly identified the lane marker intervals. We also note that these results are also consistent with our general experience that our vehicle speed estimation algorithm is less sensitive to image and signal processing errors than are our algorithms for analyzing the scene and calibrating the camera.

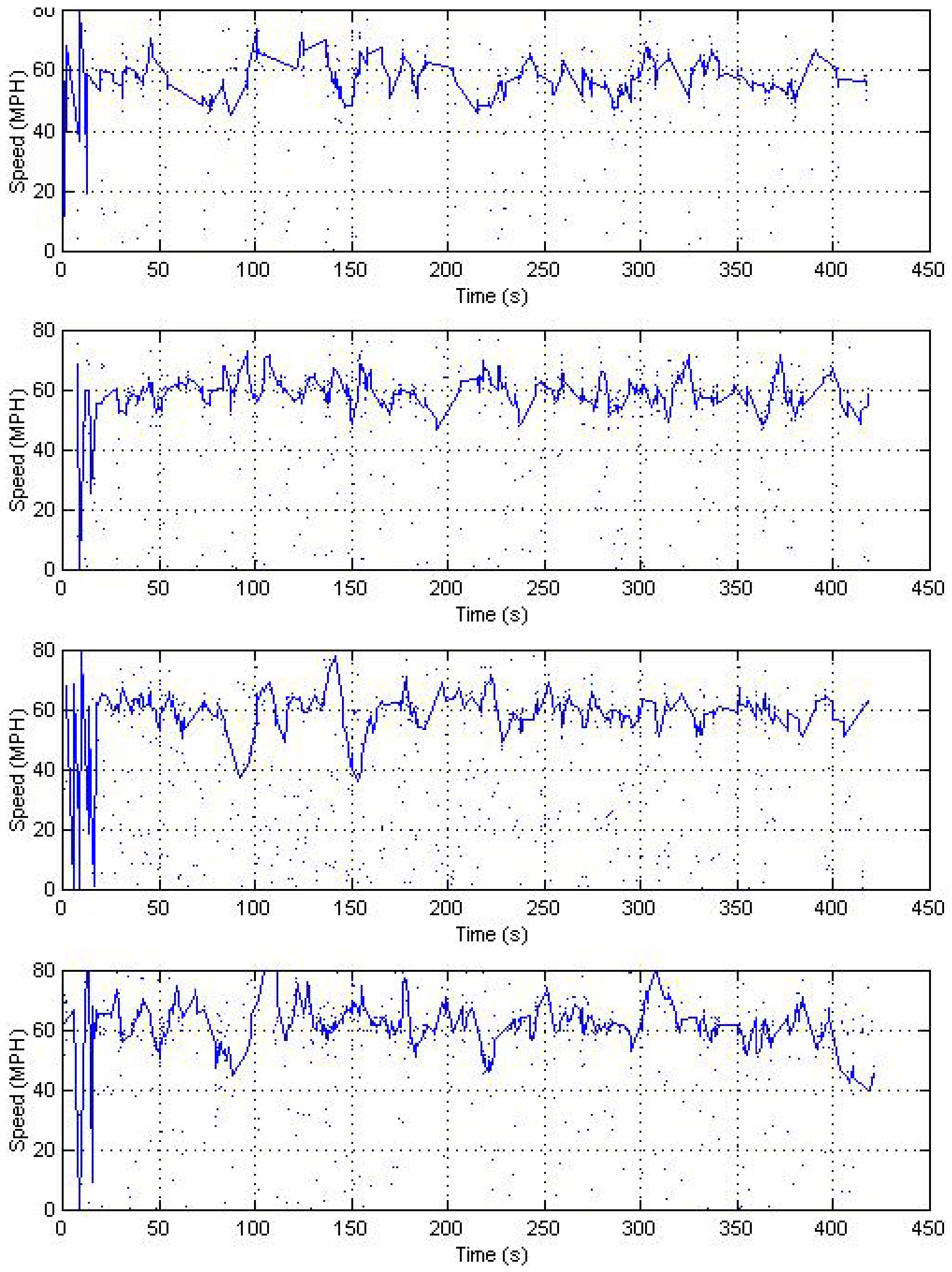


Figure 105. Fine-resolution tracking results for lanes 1-4 (top to bottom) of the scene of Figure 94(c). Lane 1 is on the left and lane 4 is on the right in the scene. The Kalman filter output is solid, and the dots are the speed estimates satisfying the energy and cross-covariance thresholds.

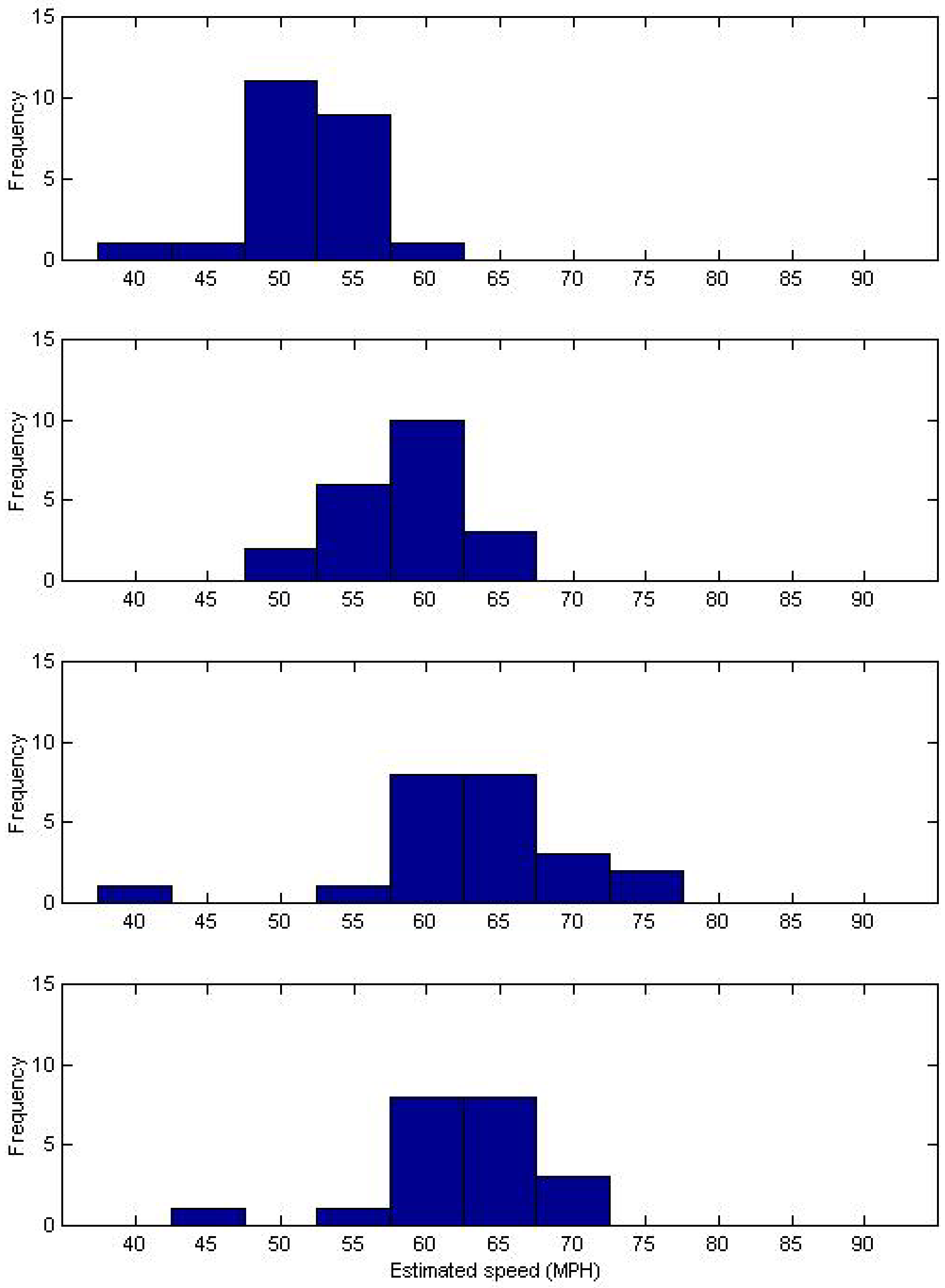


Figure 106. Histograms of speed estimates. a) Inductance loop sensor, lane 1. b) Computer vision sensor, lane 1. c) Inductance loop sensor, lane 4. d) Computer vision sensor, lane 4.

5.3.2 Dark scene with very sparse traffic

If we want to apply our speed estimation approach in a universal fashion, we will also want to know how it performs in dark conditions. In some cases, lamps are present to illuminate the freeway, but we chose a sequence where they were absent. The only illumination was from the taillights of the vehicles traveling away from the camera; these are not nearly as bright as headlights. The area of interest comprises the lanes on the right-hand side of the road that exit the left half of the bottom of the image. We also want to see how the algorithm and Kalman filter behave under sparse traffic conditions—the scene in Figure 94 (b) meets both of these criteria. However, the scene is so dark that our scene analysis and camera calibration algorithms from Chapter 3 cannot calibrate the camera or identify the vehicle lanes. Thus, like the previous sequence, we calibrated the scene and identified the lane masks by hand.

Figure 107 contains the tracking results for the scene in Figure 94(b). Each of the Kalman filters started successfully, though we had to wait for 40 seconds to start the one for lane 4. Each Kalman filter had to be restarted more than once because of the sparsity of the data. Figure 108 contains histograms comparing the inductance loops and the 20-second average output from the Kalman filter for lanes 2 and 3. We note that the distributions are quite similar, strengthening our case for using the lane marker intervals as a calibration method.

The ability of our algorithms to estimate the mean speed in a very dark scene is quite encouraging. It is reasonable to assume that our algorithms will perform even better in scenes where street lamps are present. The calibration algorithms presented in Chapter 3 may even be suitable in certain cases when street lamps are present, yielding a sensor capable of robustly processing any weather or lighting conditions.

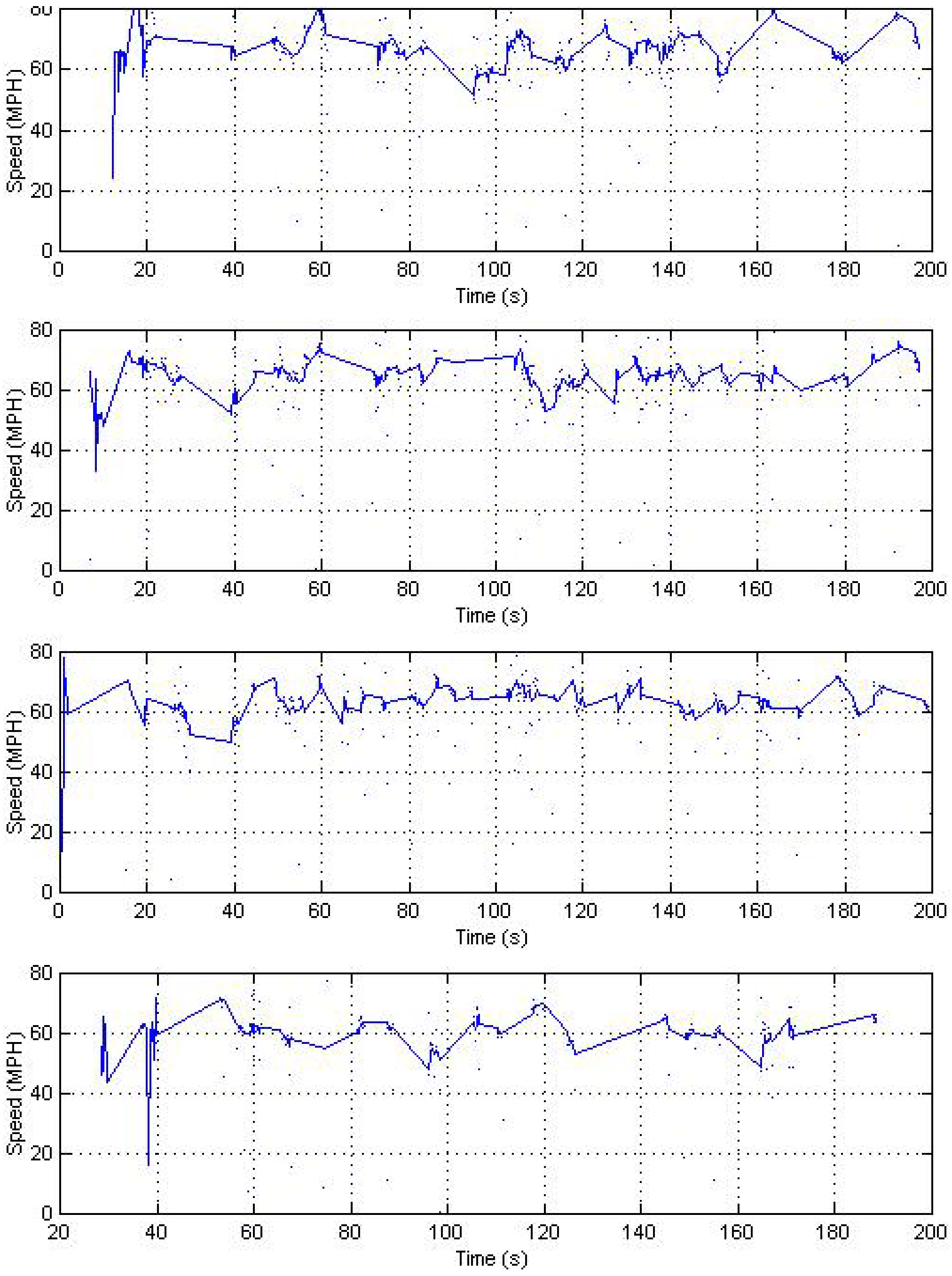


Figure 107. Fine-resolution tracking results for the scene of Figure 94(b). Lane 1 is on the left and lane 4 is on the right. The Kalman filter output is solid, and the dots are the speed estimates satisfying the energy and cross-covariance thresholds.

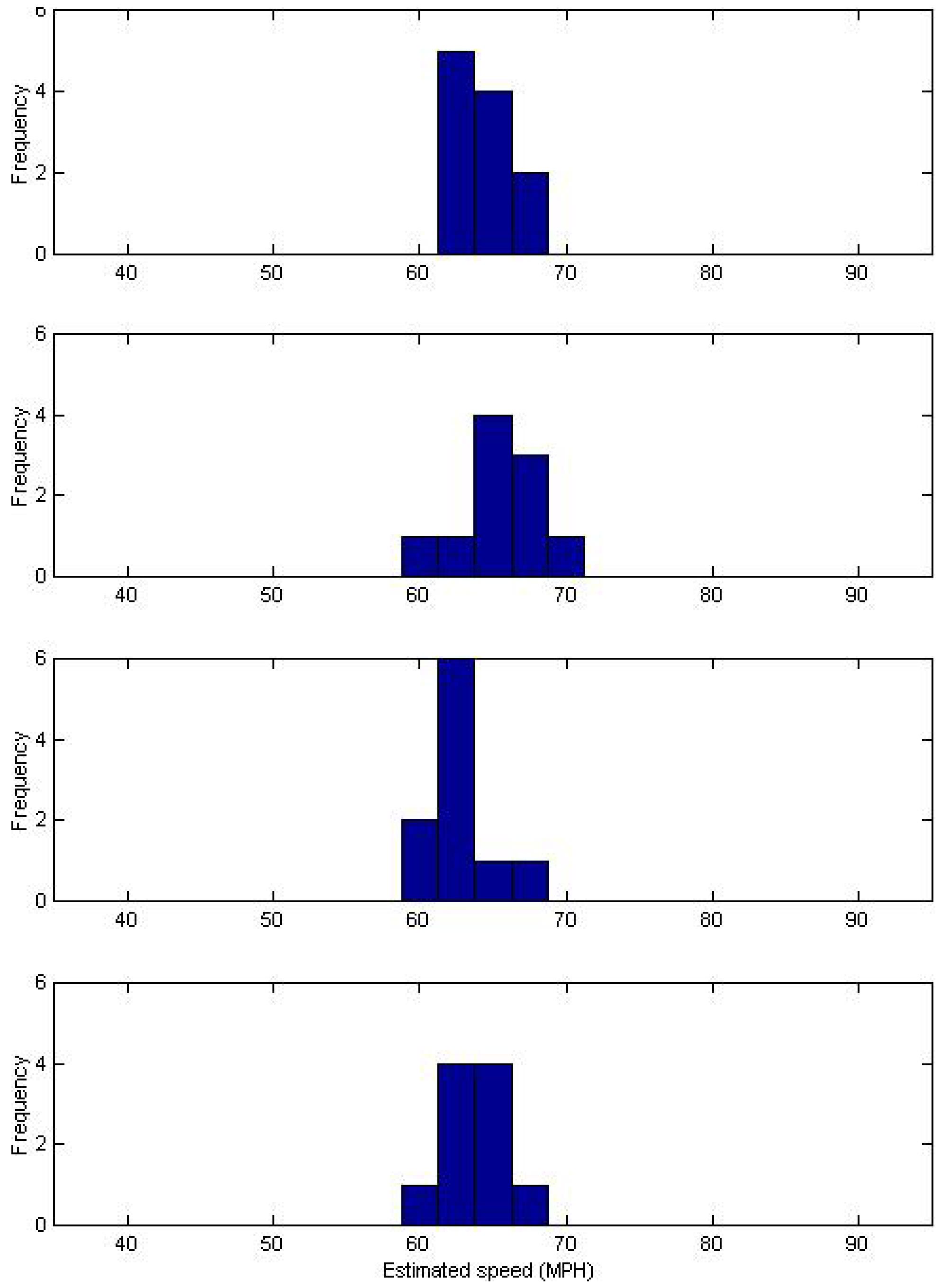


Figure 108. Histograms of speed estimates. a) Inductance loop sensor, lane 1. b) Computer vision sensor, lane 1. c) Inductance loop sensor, lane 4. d) Computer vision sensor, lane 4.

5.3.3 Congested scene with shadows

The scene in Figure 94(c,d) is difficult to analyze because the mix of shadows and sunshine on the road creates many horizontal edges in the image that could confuse the speed estimation algorithm. The images were captured as traffic became heavy in the afternoon commute at about 4:00 PM, and the speed is highly dynamic, creating difficulties for the Kalman filter. High density traffic is evident in the distance, and during the 3-minute segment, the stop-and-go shockwave reaches the viewing area of the camera and the vehicles begin backing up, as shown in Figure 94(d). Figure 109 contains the mean speed estimates from our algorithm and the Kalman-filtered mean vehicle speed estimates for the main four lanes of traffic. The unfiltered start-up noise in the first 20 seconds illustrates the challenges addressed in Section 5.2.4.6. We also note the inability of the Kalman filter to track the rapidly changing data in lane 4 at t = 30, 80, and 120 seconds. This shows that we ought to increase the value of σ_a used to calculate the Kalman gain so the filter is more responsive to the input data. However, it is interesting to note that the data from our mean speed estimator are so clear that the error in the Kalman filter signal is quite visible. This indicates that the mean speed estimator is performing quite well under these conditions.

As far as the traffic patterns are concerned, we note that the rightmost lane experiences the stopped-traffic shockwave first, which reaches the other lanes shortly thereafter in order from lane 4 to lane 1. It is encouraging to note that the left-hand passing lane shows the most free flow of traffic, as we would expect. Visually comparing the speed estimate with the image sequence demonstrates it to be essentially a perfect representation from a subjective perspective.

Although inductance loop data are available, we cannot easily compare these data with our estimates because the 20-second average time resolution from the inductance loop sensors is inappropriate in the dynamic conditions of this scene.

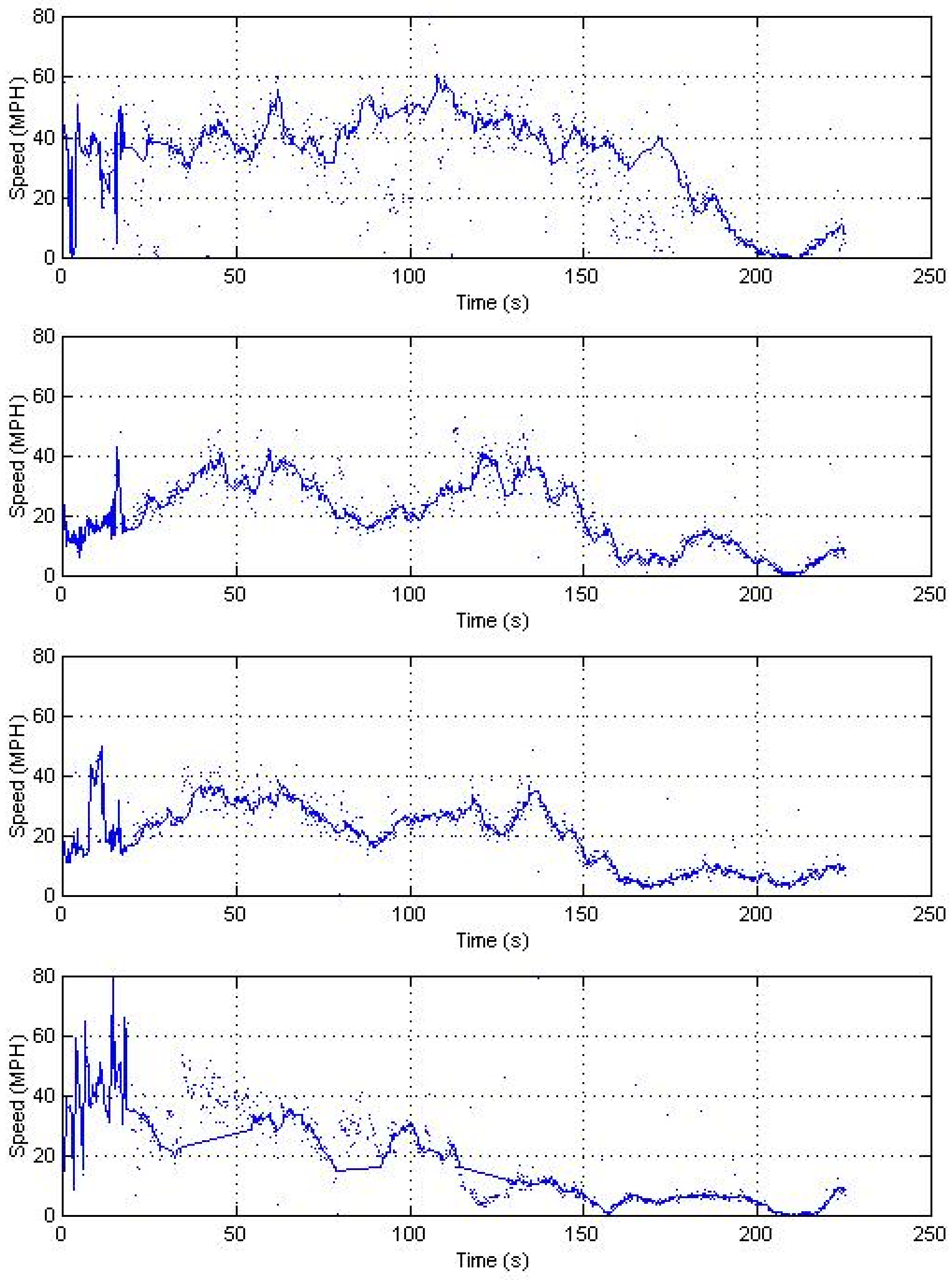


Figure 109. Fine-resolution tracking results for the scene of Figure 94(b,c). Lane 1 is on the left and lane 4 is on the right.

5.3.4 Afternoon commuting scene with medium density

The scene contained in Figure 94(c) is not particularly notable, apart from the low pan angle. Examining the tracking results in Figure 110 shows that it does a reasonable job of tracking the traffic dynamics, especially after the start-up transient. The only spot where the data change too rapidly for the Kalman filter to track is in lane 4 at $t = 80$. The linear portions of lane 1 exist because there are no vehicles present.

We note that most of the data collected from the mean speed estimate algorithm are within range of the Kalman filter output signal, indicating that it is particularly reliable in this scene. The decreased variability of the mean speed estimate when many vehicles are present is an interesting and counter-intuitive property of the mean speed estimation algorithm. However, it is clear that the peak in the cross-covariance function is stronger and more narrow when many features are present in the two signals that are compared. Future work could examine how to take advantage of this property when traffic density is heavy.

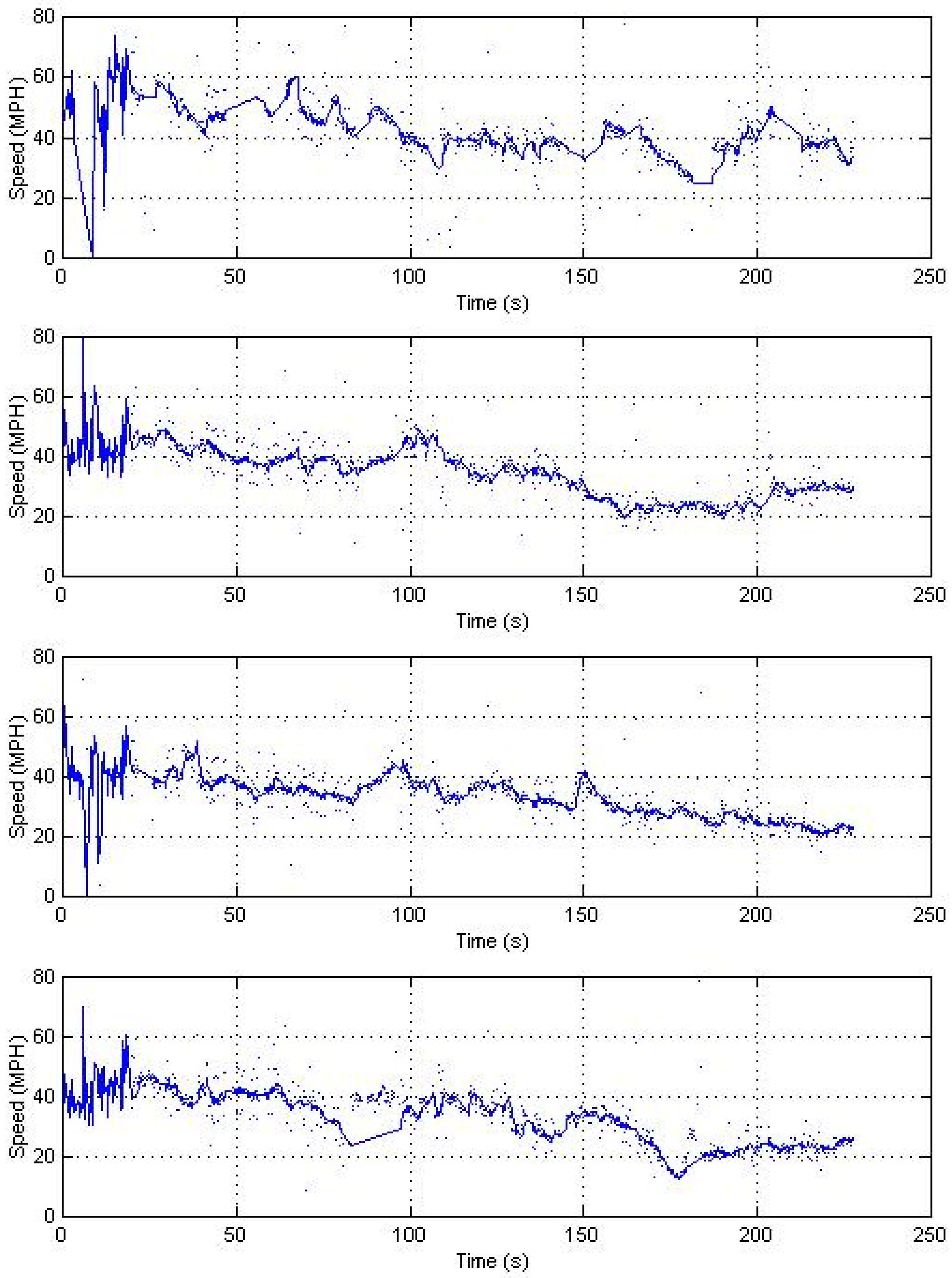


Figure 110. Fine-resolution tracking results for the scene of Figure 94(c). Lane 1 is on the left and lane 4 is on the right.

5.4 *Error analysis*

In order to estimate the vehicle speed, we applied the formula

$$s(k) = \frac{L}{\tau_L}\tau_{12}(k) = S'\tau_{12}(k) \qquad (100)$$

to estimate the mean vehicle speed for a given lane at time *k*. The process described in this chapter assumed it had been given the scale factor *S'*. Multiple sources of error exist that contribute to the output uncertainty of *s*(*k*). Simply starting with the terms of Equation (100), we note that *S'* has a given uncertainty that was estimated in chapters 3 and 4. *S'* itself is composed of *L* and τ_L, both of which have uncertainties associated with them, i.e., the physical lane stripe interval variation and the measurement errors of our image processing.

This chapter devoted itself to estimating $\tau_{12}(k)$, assuming a constant value of *S'* has been provided. Thus, the variability of the data presented in this chapter is due entirely to $\tau_{12}(k)$. Several sources contribute to this variability. First of all, the vehicles themselves have a speed distribution, as shown in many of the figures above. Even if we performed perfect image processing and camera calibration, we would expect temporal variability in the mean vehicle speed. However, our speed estimation process is not perfect, as shown by the many data points that do not necessarily lie near the Kalman-filtered output signal. If we simply took the data as a whole, it would indeed be a fairly noisy process. However, the Kalman filter helps us to select which measurements are appropriate to incorporate into our model. At the end, it also gives us $\mathbf{P}^+(k+1)$, the covariance for $\mathbf{X}^+(k+1)$, which is our mean vehicle speed estimate at *k*+1. In our one-variable case, $\mathbf{P}^+(k+1)$ is simply the variance of our estimate at the current time step. This variance becomes a very convenient way of describing the noise in the estimate $\tau_{12}(k)$, given our state-space model in Equation (97). However, we also acknowledge that its steady-state value is ultimately determined by the input noise model.

As we might expect from intuition, we are able to reduce the variance of our mean speed estimate by averaging over short time periods. From statistics, we know that the variance of the estimate of a normally distributed variable decreases according to $(N_{samp})^{-1/2}$, where N_{samp} is the number of samples used in the average. Thus, we expect that when our process is able to perform many measurements, it can significantly reduce the variance of the mean speed estimate arising from the temporal term $\tau_{12}(k)$. In other words, we can trade time resolution for precision in our mean vehicle speed estimate.

Regardless of how much we reduce the variability due to $\tau_{12}(k)$, we cannot escape the bias introduced by errors in *S'*. However, according to our estimates in Chapter 4, the 95 percent confidence intervals for *S'* for the scenes of Figure 94(c) and Figure 94(e) have a fairly narrow range on the order of 0.6 percent, though they are biased by about 6.8 percent and 5.5 percent, respectively, as compared to the results of our hand calibration.

For the scenes in Figure 94(a), (b), (c-d), and (e), the average standard deviation of the output $\mathbf{X}^+$ were, respectively, 3.5, 3.0, 2.0, and 2.0 MPH over the course of the sequences. Lane 3 in the raindrops scene had an average standard deviation of 3.75 MPH, so we can offer a worst-case 95 percent confidence interval of about ±7.5 MPH for free-flowing traffic and ±4.0 MPH for congested traffic. If we assume a worst-case bias of 7.5 percent, which is very reasonable in our experience, then we can offer 95 percent confidence intervals of ±6.25 MPH at congested levels (30 MPH) or ±12 MPH in free-flowing traffic (60 MPH). These worst-case values amount to about ±20 percent of the nominal value. While these values seem quite large, we note that this interval describes the mean speed estimate at a 0.2-second time resolution. If we perform averaging with 100 samples over 20 seconds similar to the inductance loops, we can reduce the variability in $\tau_{12}(k)$ by a factor of 10, leaving the calibration bias as the major source of error.

5.5 Summary

Rather than tracking vehicles individually, we describe how to track the flow of their features through the image. We showe how the offset of the features between frames has a direct connection to the three-dimensional distance the vehicles travel along the road. By applying appropriate thresholds on the signal energy and minimum cross-covariance, we are able to estimate the mean vehicle speed at a very high time resolution. Finally, we apply a Kalman filter to smooth the output signal.

Our experiments showed that our algorithms perform well under a wide variety of real-world conditions. We showed that in the case of free-flowing traffic, the distribution of the 20-second average speed is quite similar to inductance loop data. When traffic becomes congested, the algorithms perform even better, with lower measurement variance.

Occasionally the traffic will change too rapidly for the Kalman filter to follow. However, we provide reasonable criteria with which to restart the Kalman filter to provide a robust solution. In summary, the experiments support the proposition that our speed estimation algorithm can serve as the basis for a sensor with fine time resolution (i.e., congested conditions) and a sensor with coarse time resolution (e.g., 20-second averages similar to the inductance loops).

6 CONCLUSIONS

6.1 Summary of our work

This document began by reviewing the literature, which contains a great deal of work in the areas of camera calibration and vehicle tracking. However, few of the efforts have been precisely directed at dynamically calibrating cameras that view road scenes. By a similar token, most vehicle trackers focus on individual vehicles for counting and classification rather than tackling the problem of mean speed estimation.

To develop our context, we present analytical models of the camera and the scene that ignores camera roll, road tilt, and road slope. We developed three methods of calibrating the camera in terms of features that are available for measurement in the images, together with information about the scene known *a priori*. We presented results from a Monte-Carlo simulation of errors for each calibration method illustrating the operational limits for each calibration method. We also provide plots illustrating the effects of nonzero road slope and tilt on each calibration method.

We found that in the absence of features separated by a known distance, it is possible to estimate the camera parameters and its position using the vanishing points for lines that are parallel and perpendicular to the road. To use this method, however, the camera must be properly aimed, and the road slope and tilt must be nearly zero. The second calibration method that we proposed assumes that the camera-to-road distance is known. Because this method is quite sensitive to errors in this distance and any slope or tilt to the road, its applicability is limited to a fairly narrow range. However, our third camera calibration method, which utilizes known distances along the road is fairly immune to the slope or tilt of the road. On the basis of our analysis of its sensitivity to errors and range of operation, we recommend this method as a sound approach for calibrating the WSDOT traffic cameras under the wide variety of focal lengths and orientations that they will encounter (assuming that lane markers are visible in the images).

After the development of our fundamental model for the camera and road scene, we describe algorithms for automatically extracting the necessary measurements from digital images to calibrate the camera. We generate the background image, average top-hat image, and activity map as feature images for these algorithms. The activity map enables us to estimate the approximate vanishing point from which we can further analyze the activity map and extract the lane boundaries of the road. We also describe a raindrop and obstacle detection algorithm to disqualify certain scenes from being processed, since the camera cannot be calibrated. This yields a set of algorithms that is robust for any conditions a traffic camera might encounter during the day.

Assuming that the lane boundaries and rough vanishing point have been estimated, we describe how to precisely locate the road boundaries and vanishing point using the average top-hat image. Our knowledge of the activity region of the vehicles also enabled us to develop a method for estimating the vanishing point for the lines perpendicular to the road from the vehicles themselves. Finally, we describe how to use the knowledge of the vanishing point and road lanes to estimate the interval between lane markers by using the autocovariance signal processing technique.

We tested our scene analysis and measurement algorithms on four scenes to determine the performance of our camera calibration methods. We obtained 95 percent confidence intervals for a hand-calibrated reference using many of the equations from one of the camera calibration techniques. Applying our three automatic methods of camera calibration to all four scenes (where appropriate) gave us confidence intervals for the camera parameters f, ϕ, and θ, the camera position d, and the overall scale factor S', which we compared to the hand-calibrated reference. The results supported our hypotheses about the sensitivity and limits of operation that we developed for each method on the basis of our Monte-Carlo simulations of the theoretical models.

Method 1 performed well on one close-up scene but not the other, where the reduced-order camera model could not adequately handle the levels of slope and tilt in the road. The second method also failed on the far-field scene containing the sloped and tilted road. Although it succeeded on the other far-field scene, the resulting confidence intervals were quite large, as anticipated by the simulations in Chapter 2. Of the three methods, Method 3 was the only one we successfully applied to all four scenes. Even though the hand-calibrated and Method 3 confidence intervals did not always overlap, the output parameters for each scene were consistently within 7.5 percent relative to the hand-based estimates. We concluded that because this approach successfully surmounted many difficulties (e.g., extraneous road features, shadows, closely-spaced lane markers) that typically occur in freeway scenes, it has the potential to be able to calibrate the roadside cameras under any reasonable daytime or dusk conditions.

In Chapter 5, we present our algorithm for estimating mean vehicle speed in a given lane of traffic by estimating the flow of edge features between frames with the cross-covariance function. Our method is intimately connected to the camera calibration method based on the lane marker interval. In essence, by removing the nonlinearity from the perspective projection, we only need to take the ratio of the distance traveled along the road to the distance between lane markers and scale it by the known lane marker interval to obtain a speed estimate. We also point out that the method naturally estimates the mean vehicle speed for all vehicles in a lane of traffic without identifying any particular vehicle. We obtained a mean speed estimate signal with high time resolution to which we applied a Kalman filter for smoothing.

Although they remained somewhat noisy, the mean speed estimate signals possessed a very strong correlation with subjective observations of the dynamic traffic patterns in congested and stop-and-go traffic. We found that the mean speed estimate had less variability at lower speeds than in free-flowing conditions. It also performed much better under congestion than in sparse traffic because more image features were available for tracking. If we include the camera calibration bias, we assessed our algorithm with (at a maximum) a 95 percent confidence interval of 20 percent relative error in the mean speed estimate at 0.2-second time resolution. Temporal averaging over 20 seconds greatly reduces this variability, however, leaving the camera calibration bias as the major source of error, for a total maximum error of about 8.5 percent, or 5 MPH at 60 MPH nominal speeds.

In summary, our experiments under many challenging conditions support the hypothesis that our image processing algorithms can generally perform well in real-world conditions. Of course, this assumes that we ensure the scene conditions are appropriate for whichever camera calibration method that we choose to use. We showed that in the case of free-flowing traffic, the distribution of the 20-second average speed is quite similar to inductance loop data. Thus, the experiments support the proposition that our algorithms can serve as the basis for a sensor with fine time resolution (i.e., congested conditions) and a sensor with coarse time resolution (e.g., 20-second averages similar to the inductance loops).

6.2 Contribution of our work to the literature

Having summarized our work, we now highlight the portions that are particularly significant contributions beyond the work of others. For starters, while they are not very complex, the camera and scene model are the first of which we are aware that explicitly model the road boundaries and lane markers for a road scene in terms of computer vision equations. As a direct consequence of our modeling approach, we have contributed three novel and unique calibration approaches based on different assumed quantities and measurements that can be made from the images. We gave this contribution practical import by our sensitivity analysis that described the range of usefulness for each of the calibration methods, including the effects of road tilt and slope. Such an analysis has yet to be performed in the literature for any system, whether it is manually or automatically calibrated.

We have also contributed significantly to the area of road scene analysis. Our activity map composed from the average frame difference extended the binary activity region approaches by Stewart et al. [40] and Tseng et al. [56] and the accumulation of binary change-detection images [57] to new levels of functionality. The activity map enabled us to estimate the vanishing point of the road to a fair degree of accuracy using vehicles alone, without considering cues from the static portion of the scene. It also provided us with the features that we used to extract the lane boundaries for each lane of traffic for reasonable values of the pan angle θ, i.e., $|\theta| < 20°$. To our knowledge, only two other approaches exist for solving this problem [40][41]. One [40] applies morphology to a binary version of the activity map to extract very rough lane boundaries in scenes where the pan angle is nearly zero. This makes it irrelevant to the current problem where the pan angle often exceeds 5 degrees. Another detractor is the method's inability to provide a reasonable estimate of the vanishing point. The other [41] relies on static line features on the road, which may vary considerably from road to road. Such an edge-based approach is likely to fail under real-world lighting and weather conditions. In distinction, the ingenuity of our development of the activity map is that it offers the possibility of feature extraction, vanishing point estimation, and lane boundary extraction even in the face of adverse weather and lighting conditions, as long as the vehicles are somewhat visible.

The activity map also formed the basis for our algorithm to detect raindrops or other obstacles such as highway overpasses that obscure the road from the camera. This is another important contribution that is critical for a system to behave robustly in real-world applications. To our knowledge, no other research has bothered to detect such untoward conditions, probably because nearly all approaches assume a static camera without rainy conditions.

In addition to the activity map, we introduced the morphological top-hat image as an important contributor to the set of features with which we can analyze the traffic scene. This morphological operator is well known but has not been used to process traffic images, to our knowledge. Only the average top-hat image for a video sequence lane marker features that were strong enough to allow us to estimate the lane marker interval. Although the autocovariance function is a classic method of estimating the periodicity of a signal, it has certainly never been tried in the traffic monitoring literature for measuring vehicle displacements. The autocovariance approach holds other possibilities for traffic scene analysis, depending on the camera position and orientation, e.g., applying the method in the orthogonal direction to estimate the width of the road. Thus, our successful attempt at extracting the lane marker intervals from the images and using them to calibrate the camera is yet another novel and fruitful contribution of this work.

When determining features with which to perform camera calibration, nearly all of the approaches in the literature rely exclusively on static features from the scene. The lone exception is the work by Zhu et al. [19], in which the authors calibrated the system by sending a vehicle of known dimensions through the scene. Of course, this is impractical for a large-scale solution, not to mention the need for recalibration each time the operator moves the camera. However, we showed that with a properly oriented camera and road without any slope or tilt, our algorithm could extract important information from the vehicles themselves regarding the vanishing point for the set of lines that are perpendicular to the road in 3-D. This could eliminate the need for a hand-calibration tool for the detection and classification work done by Gupte et al. [22].

Vehicle tracking has long been the focus of a great deal of research, as shown by the plethora of references in our literature review of Chapter 1. A primary reason for this fact is that perfect vehicle tracking and extraction would immediately provide perfect vehicle speed estimates and vehicle counts. It would also give vehicle classification algorithms the best chance of success. However, our speed estimation algorithm focuses solely on estimating the mean vehicle speed without necessitating the extraction of individual vehicles. By contributing the cross-covariance method of speed estimation, we have solved several subproblems. First, we have obviated the problem of occlusion; the algorithm works even better when the image contains more vehicle features. Second, we have shown how to naturally incorporate the imaging information from all vehicles into the speed estimate. Third, the cross-covariance method carries with it all the weight of signal processing rigor, including an indication of our confidence in the speed estimate, i.e., the cross-covariance peak height that ranges from zero to one. Fourth, by averaging the features across the lane masks, we greatly simplify the tracking challenge to a one-dimensional problem while adding robustness in the averaging process. This means that the tracking portion of the algorithm is computationally very inexpensive, and a real-time implementation is well within reach. Fifth, unlike inductance loops, which measure time mean speed, our algorithm inherently measures space mean speed, which has desirable theoretical properties [58]. The differences between the two types of mean speed are accentuated at lower speeds, exactly where our algorithms perform best. Finally, our fresh approach to speed estimation also has implications for the estimation of other fundamental traffic parameters that will be discussed below.

6.3 Computational complexity and implementation

Because we developed these algorithms exclusively in the Matlab environment, it is difficult to assess their computational requirements with complete accuracy. The following represents our best attempt to indicate the algorithms from which we should expect real-time performance. We expect the feature images for the scene analysis and camera calibration to operate at the 640 X 480 pixel resolution at 1-2 Hz, and the images for speed estimation to operate at 320 X 240 pixel resolution at 5 Hz. We can reasonably assume that we can request gray-scale, JPEG-compressed images rather than color ones; this will somewhat shorten the processing time.

6.3.1 Feature extraction

Generating the primary feature images should take little overhead, and this could even operate as a background process. Specifically, we must update accumulators for the average image, the average top-hat image, and the average frame-difference. Besides a lot of amount of memory, this involves three image additions, two image subtractions, one morphological dilation, and one morphological erosion. A computer can perform each of these operations in-place. We can either apply an IIR filter with an update factor of about 0.005 or 0.001, or else perform conventional averaging. The latter will require many fewer multiplication operations, whether they are fixed or floating-point.

We note that because we calculate the frame difference in order to estimate the level of image activity, we do not require a specific sampling rate. In fact, 1 Hz or 0.5 Hz would be perfectly satisfactory if the feature generation were part of a background process. Furthermore, unless we know the camera is zoomed in or has a large down-angle, we can simply calculate the features in the bottom half of the image, which will cut memory and processing requirements in half.

The algorithm for estimating u_1 could also be performed as a background process, if one decided to implement the secondary camera calibration methods we proposed. For each input frame, the computer must perform 13 X 1 and 1 X 13 convolutions, an image division, and calculate the arctangent of the image, all in the effort to obtain the image gradient. The computer must also perform three morphological dilations with a 7 X 2 kernel, one dilation with a 20 X 1 kernel, one morphological skeletonization, one connected components analysis, three thresholdings, seven logical AND’s, two logical NOT’s, several image searches for the coordinates of nonzero pixels, and a Hough-like accumulation for nonzero pixels in a very sparse binary image. This algorithm requires the computer to process the entire image for each of these operations.

Finally, we will wish to periodically analyze the past history of the Kalman filter to optimally select the covariance matrices using the algorithm of Bell [59]. This can also be performed in the background, so its complexity is not of great importance.

6.3.2 Scene analysis

After obtaining a sufficient number of frames to estimate the feature images, i.e., 1000 or more, the computer must perform the most complex processing required for our algorithm. However, since we only do these operations once, the computational complexity is not very important. The operations include the Canny edge detector and the Hough transform for estimating the vanishing point from the activity map. Extracting the lane masks from the activity map involves one bilinear image sampling and some one-dimensional signal processing that is computationally very inexpensive. Precisely locating the road boundaries and vanishing point involves one bilinear image sampling to generate the initial estimates. However, the search process is computationally expensive. A typical search involved two passes, the first with 251 function evaluations, and the second with 119 functional evaluations. Each function evaluation involves bilinear sampling of the pixels along two different lines in the image. Of course, the search algorithm overhead is also nontrivial. To estimate the confidence intervals for the parameters found required eight different searches, i.e., two for each parameter we located; this expense is optional. Each of these searches typically involved ten functional evaluations of the bilinear line pixel sampling. Typically, we could expect to process only the bottom half of all of these images.

Analyzing the scene for raindrops or other obstacles involves one bilinear image sampling and some one-dimensional signal processing. We anticipate that this set of algorithms could be automatically applied every 10-15 minutes, to either refine our calibration estimates or replace them. This would eliminate the need to determine whether or not the scene requires recalibration.

Estimating the lane marker interval requires one image maximum, two image multiplications, one image exponential, a bilinear resampling of the image with 3-4x upsampling, one morphological dilation with a 5 X 25 pixel (approximately) kernel, and calculation of the covariance function for all columns in the upsampled image. Again, we note that the computer performs all of these computationally expensive operations only once per scene or perhaps every 10-20 minutes.

6.3.3 Mean vehicle speed estimation

When estimating the mean vehicle speed, we only process the bottom one-third of a 320 X 240 image, which contains many fewer pixels than any of the images used in the operations described above. This estimation process requires one image convolution, several image thresholdings, several image AND’s, and a bunch of one-dimensional signal processing that should be computationally inexpensive. After a speed estimate has been obtained, the Kalman filtering process is quite inexpensive. Depending on the size of the history buffer for past signals, it may require the same amount of memory that an image would occupy.

6.3.4 Summary

The computational requirements for our algorithms should not prove prohibitive for implemention in a real-time system, particularly at the low frame rates the data lines can support. While the set of algorithms contains many nuances and areas of complexity, they build upon standard image and signal processing techniques for which optimal implementations are likely available in a library. Thus, the development effort should not prove overly burdensome, and the implementation should perform at a near-optimal level.

6.4 Applications

The algorithms described to this point perform best under certain assumptions for the task of average vehicle speed estimation. Assuming we use the recommended camera calibration method, these include daytime or dusk light conditions, a small pan angle ($\theta \in [0°,20°]$ degrees), a small down angle ($\phi \in [0°,12°]$), a camera height of 30 feet or more, and a relatively low focal length so that many cars are in view. We expect that the vehicle lanes will generally exit the bottom of the image. We also assume that the camera is positioned and oriented in such a way that it can get multiple samples of a vehicle as it travels through the range of the image. Camera poses with a large down angle are unusable because they can only capture a couple samples at the frame rate of 5 Hz before a vehicle has already traveled through the image. If overhead freeway lamps are present as an external light source besides the vehicle headlights, our algorithms should also be able to function at night as is or with some slight modifications. Since we recommend using the lane marker intervals to calibrate the camera, we do not anticipate road curvature restricting the application of our algorithms to vehicle speed estimation. Of course, we offer the raindrop and obstacle detection algorithms to label bad data or alert the operator that the camera must be repositioned so that a sign or overpass does not obscure the road. As for detecting when the camera should be recalibrated, we recommend either the algorithm by Pumrin and Dailey [57] or simply recalibrating every few minutes using the background process described in Section 6.3.

Many of our scene analysis algorithms should prove useful when performing other tasks involving traffic monitoring or vehicle tracking. For example, extracting the lane masks from the road greatly constrains the tracking problem and should prove advantageous for vehicle tracking algorithms. As mentioned before, our method of estimating u_1 from the vehicle lines should eliminate the need for a camera calibration tool in some cases, e.g., see Gupte et al. [22]. Our raindrop/obstacle detector should also prove useful in making other traffic monitoring systems more robust.

Under the proper conditions, our camera calibration Method 1 involving perpendicular and parallel line extraction is worth trying in applications such as vehicle navigation in corridors. Researchers [60] have already used similar techniques to solve such a problem, but they leveraged the pre-existing rectangular pattern present on office ceilings. In our framework, since the system produces four measurements (slope and intercept for two parallel lines on the ground plane), it could obtain the focal length, lateral position, tilt angle, and pan angle by viewing the corridor itself, assuming a fixed, known camera height and corridor width. Such an approach would be particularly tenable if many estimates could be continuously gathered to reduce the variability of the estimate. Using our approach in this situation could either reduce the number of cameras on the vehicle or allow it to focus its attention forward and downward rather than regularly pausing to gaze at the ceiling.

Airport surface surveillance [61][62][63] represents another transportation problem that might benefit from some or all of the camera calibration and tracking techniques presented here. In brief, the Federal Aviation Administration (FAA) has begun working with industry and academia to meet the needs of large airports to run more efficiently. One key to these improvements involves providing air traffic controllers with precise knowledge of the position of every aircraft and surface vehicle. In an evaluation of airport surface surveillance technologies [63], employees of the Rannoch Corporation (the primary industry partner of the FAA in this venture) included infrared cameras used to measure vehicle position as an essential part of their proposed surveillance solution. Infrared cameras prove advantageous over traditional visible-spectrum cameras in the stormy or nighttime conditions often present at major airline hubs. Since the cameras must locate vehicles with fairly high accuracy, they also require reasonably accurate calibration. Although painting parallel lines visible in the infrared spectrum all over the airport surface might prove impractical for calibrating roving cameras, the methods presented here might prove useful when calibrating a fixed camera to avoid the expense of a surveying crew. Depending on the camera perspective, certain aspects of the tracking techniques presented here might apply, as well as the 2-D to 3-D equations due to Lai and Yung that enable 3-D tracking on a ground plane.

Video surveillance of people could also benefit from the self-calibration methods described above. For example, Jaynes [64] used the parallel and perpendicular structure in a scene to calibrate a camera. In fact, he developed a multi-camera system capable of extracting the trajectory of people as they move through a surveillance area. However, he followed Haralick’s analysis of a rectangle in 3-D [65] to obtain the pose of the camera, which needlessly assumes knowledge of the focal length. Bradshaw, Reid, and Murray [66] spent a great deal of effort developing a computer vision system with complex system dynamics capable of detecting and actively tracking an object on a ground plane in simple scenes and extracting its 3-D trajectory. However, they assumed that four points or four lines lie within view of the camera at all times whereby the system can calibrate itself and thereby identify the object’s real-world location. Clearly, both these systems could benefit from the methods described above that appreciably reduce the information required to dynamically calibrate the camera. In fact, from this brief survey, one can imagine a variety of surveillance applications for actively tracking vehicles, equipment, and people at delivery centers, in parking lots, outside building entrances, or inside warehouses.

The strongest contribution of our work is the mean vehicle speed estimation. It does an excellent job at estimating the mean speed of a collection of objects moving in a constrained fashion through an image, i.e., objects with an exponentially distributed arrival time and normally distributed speed. Situations that might benefit include assembly line processes and trains. However, the true power of our approach is obvious when we consider nonlinear motion for which we sample the image in a special way. In particular, angular velocity is not particularly easy to track using a camera (particularly at low frame rates) but could be very amenable to a modified version of our algorithm if the path of motion could be estimated by a technique like our activity map.

In summary, while our algorithms are most directly applicable to traffic monitoring, they may very well prove useful in other applications as well.

6.5 Future work

As with any endeavor, there remains work that would enhance our algorithms, as well as new research directions that could prove fruitful. One of the biggest needs is for our algorithms to be implemented in a real-time system to characterize their performance under various conditions. Specifically, we would be interested in the system's performance in free-flowing, congested, and stop-and-go situations. This would allow us to determine realistic values for the time resolution of the output and reveal any weaknesses in these situations. It would probably require fairly tight synchronization between inductance loops or even laser range pointers to obtain accurate reference data. Furthermore, research should also undertake the challenge of determining how rain on the ground, dusk lighting conditions, and extremely bright sunlight statistically affect the tracking accuracy, if at all.

In addition to characterizing our algorithms' performance in difficult conditions, we believe they can also be enhanced to the point where they can operate reliably in the dark and even when raindrops are on the camera lens. We believe that if a human being can hand-calibrate the scene and draw lane masks, then we should be able to design an algorithm to perform the same task. The scene analysis relies heavily on the quality of the feature images. Future research can look into unsharp masking, the amplification of high frequencies, nonlinear diffusion, and other techniques to enhance the average top-hat image in dark scenes. In addition, it might be useful to accumulate the moving edges in the activity map under dark conditions rather than the frame difference because the moving edges will probably give a stronger response relative to the background than will the moving pixels themselves. Because we can distinguish dark scenes and raindrop scenes from normal ones (via histogram analysis and our rain detector, respectively), future research can develop specialized algorithms to build an expert-style system to handle all the different cases.

In the current work, we used the RGB average to do our image processing, i.e., the V component of the HSV color space. However, researchers [67] have recently shown that the Y component in the YCbCr color space is most effective at distinguishing the foreground, shadows, and objects when traffic is monitored. While this may not specifically apply to our feature images (i.e., the activity map and the top-hat image), it would be useful to quantitatively evaluate the performance of these two color spaces (at a minimum).

The decreased performance of the activity map as the pan angle θ increases is a somewhat disappointing limitation of our system, though it is not insurmountable. This limitation is due to the nonzero vehicle heights that overlap the adjacent lanes. It would also be useful to determine whether this limitation impinges on the practical usefulness of the algorithm, and if so, how to overcome it.

As illustrated by their heavy application throughout our algorithms, the autocovariance and cross-covariance functions are extremely useful. Although we leveraged these methods well in analyzing the vertical portion of the road (after removing the perspective effects), we failed to apply them to the horizontal portion of the road. This would have provided us with a simple means of estimating the lane widths and could constrain our search for the lane markers. Furthermore, we note that the autocovariance functions for adjacent columns in the lane marker images are very similar, whereas those for the rest of the road are highly variable. This would be another useful criterion to use when identifying image columns containing the lane markers.

As we have emphasized previously, one of the strengths of our mean speed estimation algorithm is that it only attempts to estimate one parameter: mean vehicle speed. As such, it is able to integrate information throughout the image in a very natural way. One of the other great applications for vehicle tracking is vehicle counting. Although we fundamentally avoid identifying individual vehicles, we hypothesize that we can derive parameters that are just as useful as counting the number of vehicles contained in an area of the image. Specifically, we propose to calculate the autocovariance function of each of the signals used to calculate the cross-covariance function for a lane of traffic. The first peak in the autocovariance function should give us an idea of the average spacing between vehicles, which would lead to estimates of traffic occupancy and density. The binary image obtained by thresholding the edge features could prove particularly useful, since we anticipate that any blank spots in the data would aid the specificity of this method.

In summary, we have established a useful set of algorithms for performing traffic monitoring, including a camera model, a method for camera calibration, methods of generating scene features, and several creative uses for the auto- and cross-covariance functions. However, many more opportunities exist to enhance the performance of the proposed algorithms or to apply our research to new areas.

REFERENCES

1. U. S. Dept. of Transportation, “The Future of transportation starts here,” http://www.itsdocs.fhwa.dot.gov/jpodocs/brochure/2kk01!.pdf.

2. D. Shrank and T. Lomax, “The 1999 annual mobility report: information for urban America,” http://www.itsdocs.fhwa.dot.gov/jpodocs/rept_mis/9dv01!.PDF.

3. U. S. Dept. of Transportation, “Intelligent transportation systems benefits: 2001 update,” http://www.itsdocs.fhwa.dot.gov/jpodocs/repts_te/13463.pdf.

4. Puget Sound traffic cameras. http://www.wsdot.wa.gov/PugetSoundTraffic/cameras/

5. D. J. Dailey, F. W. Cathey, and S. Pumrin, “An algorithm to estimate mean traffic speed using uncalibrated cameras,” *IEEE Trans. Intelligent Transportation Sys.*, vol. 1, pp. 98-107, 2000.

6. S. Pumrin, “A framework for dynamically measuring mean vehicle speed using un-calibrated cameras,” Technical report UWEETR-2002-0005, *Univ. Washington Dept. of Elect. Engr.*, 2002.

7. A. Fusiello, “Uncalibrated Euclidean reconstruction: a review,” *Image and Vision Computing*, vol. 18, pp. 555-563, 2000.

8. R. Mohr and B. Triggs, “Projective geometry for image analysis,” tutorial at *Int. Symp. of Photogrammetry and Remote Sensing*, 1996. http://www.inrialpes.fr/movi/people/Triggs/p/Mohr-isprs96.ps.gz.

9. S. Maybank, O. Faugeras, “A theory of self-calibration of a moving camera,” *Int. J. Comp. Vis.*, vol. 8, pp. 123-151, 1992.

10. L. de Agapito, R. I. Hartley, and E. Hayman, “Linear self-calibration of a rotating and zooming camera,” in *IEEE Comp. Soc. Conf. on Comp. Vis. and Pattern Recog.* (Ft. Collins, CO), vol. 1, pp. 15-21, Dec. 1999.

11. H. Kim and K. S. Hong, “Practical self-calibration of pan-tilt cameras,” *IEE Proc. on Vis., Image, and Signal Process.*, vol. 148, pp. 349-355, 2001.

12. M. Pollefeys, “Tutorial on 3D modeling from images,” *2001 European Conf. Comp. Vis.* http://www.esat.kuleuven.ac.be/~pollefey/SMILE2/tutorial.html.

13. W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge, UK: Cambridge University Press, 1988.

14. R. Szeliski and H. Shum, “Creating full view panoramic image mosaics and environment maps,” in *Int. Conf. Comp. Graphics and Interactive Techniques*, pp. 251-258, 1997.

15. F. Chausse, R. Aufrere, R. Chapuis, “Recovering the 3D shape of a road by on-board monocular vision,” in *15th Int. Conf. on Pattern Recognition*, vol. 1, pp. 325-328, 2000.

16. J. Yang and S. Ozawa, “Recover of 3-D road plane based on 2-D perspective image analysis and processing,” *IEICE Trans. Fund. Elect.,Comm., and Comp. Sci.*, vol. E79-A, pp. 1188-1193, 1996.

17. H. Takada, M. Momozawa, and H. Kawano, “A study of the measurement of velocity by image sensors,” in *2001 IEEE Conf. Comm., Comp., and Sig. Proc.*, vol. 1, pp. 251-254.

18. Y. K. Jung and Y. S. Ho, “Traffic parameter extraction using video-based vehicle tracking,” in *Proc. 1999 IEEE/IEEJ/JSAI Int. Conf. on Intelligent Transportation*, pp. 764-769, 1999.

19. Z. Zhu, G. Xu, B. Yang, D. Shi, and X. Lin, “VISATRAM: A real-time vision system for automatic traffic monitoring,” *Image and Vision Computing*, vol. 18, pp. 781-794, 2000.

20. M. Yu, G. Jiang, and B. Yu, “An integrative method for video based traffic parameter extraction in ITS,” in *2000 IEEE Asia-Pacific Conf. on Circuits and Sys.*, pp. 136-139, 2000.

21. S. Bouzar, F. Lenoir, J. M. Blosseville, and R. Glachet, “Traffic measurement: image processing using road markings,” in *Eighth Int. Conf. on Road Traffic Monitoring and Control*, pp. 105-109, 1996.

22. S. Gupte, O. Masoud, R. F. K. Martin, and N. P. Papanikolopoulos. “Detection and classification of vehicles,” *IEEE Trans. Intelligent Transportation Sys.*, vol. 3, pp. 37-47.

23. D. Beymer, P. McLauchlan, B. Coifman, and J. Malik, “A real-time computer vision system for measuring traffic parameters,” in *1997 IEEE Comp. Soc. Conf. on Comp. Vision and Pattern Recognition*, pp. 495-501, 1997.

24. J. Kim, C. Lee, K. Lee, T. Yun, and H. Kim, “Wavelet-based vehicle tracking for automatic traffic surveillance,” in *Proc. IEEE Region 10 Int. Conf. Electrical and Electronic Tech.*, vol. 1, pp. 313-316, 2001.

25. A. Giachetti, M. Cappello, and V. Torre, “Dynamic segmentation of traffic scenes,” in *Proc. Intelligent Vehicles ’95 Symposium*, pp. 258-263, 1995.

26. 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 *Proc. 33rd IEEE Conf. Decision and Control*, vol. 4, pp. 3776-3781, 1994.

27. C. Ooi and P. Liatsis, “Co-evolutionary-based active contour models in tracking of moving obstacles,” in *Int. Conf. Advanced Driver Assistance Sys.*, pp. 58-62, 2001.

28. Z. Fan, J. Zhou, D. Gao, and G. Rong, “Robust contour extraction for moving vehicle tracking,” in *Proc. Int. Conf. Image Proc.*, pp. 625-628, 2002.

29. S. T. Tseng and K. T. Song, “Real-time image tracking for traffic monitoring,” in *IEEE 5th Int. Conf. Intelligent Transportation Sys.*, pp. 1-6, 2002.

30. H. Kollnig and H. Nagel, “3D pose estimation by fitting image gradients directly to polyhedral models,” in *Proc. Fifth Int. Conf. Comp. Vision*, pp. 569-574, 1995.

31. C. Setchell, E. Dagless, “Vision-based road-traffic monitoring sensor,” in *IEE Proc. Vision, Image and Sig. Proc.*, vol. 148, pp. 78-84, 2001.

32. J. Ferryman, A. Worrall, G. Sullivan, K. Baker, “Visual surveillance using deformable models of vehicles,” *Robotics and Autonomous Sys.*, vol. 19, pp. 315-335, 1997.

33. J. Badenas, J. Sanchiz, and F. Pla, “Using temporal integration for tracking regions in traffic monitoring sequences,” in *Proc. 15th Int. Conf. Pattern Recog.*, vol. 3, pp. 1125-1128, 2000.

34. M. Fathy and M. Y. Siyal, “An image detection technique based on morphological edge detection and background differencing for real-time traffic analysis,” *Patt. Recog. Letters*, vol. 16, pp. 1321-1330, 1995.

35. J. Kato, T. Watanabe, S. Joga, J. Rittscher, and A. Blake, “An HMM-based segmentation method for traffic monitoring movies,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, pp. 1291-1296, 2002.

36. H. Taniguchi, T. Nakamura, H. Furusawa, “Methods of traffic flow measurement using spatio-temporal image,” in *Proc. Int. Conf. Image Proc.*, vol. 4, pp. 16-20, 1999.

37. C. Li, K. Ikeuchi, M. Sakauchi, “Acquisition of traffic information using a video camera with 2D spatio-temporal image transformation technique,” in *Proc. IEEE/IEEJ/JSAI Int. Conf. Intelligent Transportation Sys.*, pp. 634-638, 1999.

38. G. Garibotto, P. Castello, E. Del Ninno, P. Pedrazzi, G. Zan, “Speed-vision: speed measurement by license plate reading and tracking,” in *IEEE 5th Int. Conf. Intelligent Transportation Sys.*, pp. 585-590, 2002.

39. T. W. Pai, W. J. Juang, and L. J. Wang, “An adaptive windowing prediction algorithm for vehicle speed estimation,” in *IEEE 5th Int. Conf. Intelligent Transportation Sys.*, pp. 901-906, 2002.

40. B. D. Stewart, I. Reading, M. S. Thomson, et. al, “Adaptive lane finding in road traffic image analysis,” in *Seventh International Conf. on Road Traffic Monitoring and Control*, pp. 133-136, 1994.

41. A. Lai and N. Yung, “Lane detection by orientation and length discrimination,” *IEEE Trans. Sys., Man, and Cybernetics—Part B: Cybernetics*, vol. 30, pp. 539-548, 2000.

42. U.S. Department of Transportation. *Manual on Uniform Traffic Control Devices*, Chapter 3a, Section 6. *http://mutcd.fhwa.dot.gov/.* December, 2000.

43. J. Canny, "A computational approach to edge detection." *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 8, pp. 679-698, 1986.

44. R.O. Duda and P.E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Comm. ACM*, vol. 15, pp. 11-15, 1972.

45. R. C. Gonzalez and R. E. Woods. *Digital Image Processing*, 2nd ed. Upper Saddle River, New Jersey: Prentice-Hall, 2002.

46. C. G. Broyden, "The convergence of a class of double-rank minimization algorithms," *J. Inst. Math. Applic.*, vol. 6, pp. 76-90, 1970.

47. R. Fletcher, "A new approach to variable metric algorithms," *Computer Journal*, vol. 13, pp. 317-322, 1970.

48. D. Goldfarb, "A family of variable metric updates derived by variational means," *Mathematics of Computing*, Vol. 24, pp. 23-26, 1970.

49. D.F. Shanno, "Conditioning of quasi-Newton methods for function minimization," *Mathematics of Computing*, Vol. 24, pp. 647-656, 1970.

50. T.F. Coleman and Y. Li, "An interior, trust region approach for nonlinear minimization subject to bounds," *SIAM J. Optim.*, vol. 6, pp. 418-445, 1996.

51. T.F. Coleman and Y. Li, "On the convergence of reflective Newton methods for large-scale nonlinear minimization subject to bounds," Mathematical Programming, vol. 67, pp. 189-224, 1994.

52. M. Forbus, Engineer, Washington State Department of Transportation, personal communication, June 2003.

53. M. Mahmoud, Engineer, Washington State Department of Transportation, personal communication, July 2003.

54. J. Lundstrom, Project Manager, Washington State Department of Transportation, personal communication, June, 2003.

55. Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation*. New York: John Wiley & Sons, 2001, pp. 268-270.

56. B. Tseng, C. Lin, and J. Smith, "Real-time video surveillance for traffic monitoring using virtual line analysis," in *Proc. 2002 IEEE Int. Conf. Multimedia and Expo*, pp. 541-544, 2002.

57. S. Pumrin and D. J. Dailey, “Roadside camera motion detection for automated speed measurement,” in *IEEE 5th Int. Conf. Intelligent Transportation Sys.*, pp. 147-151, 2002.

58. Transportation Research Board, “Traffic flow theory: a revised monograph on traffic flow theory,” *http://www.tfhrc.gov/its/tft/tft.htm*, Chapter 2.

59. B. Bell, “The marginal likelihood for parameters in a discrete Gauss-Markov process,” *IEEE Trans. Sig. Proc.*, vol. 48, pp. 870-873, 2000.

60. Z. Yang and W. Tsai, “Viewing corridors as right parallelepipeds for vision-based vehicle localization,” *IEEE Trans. Industrial Electronics*, vol. 46, pp. 653-661, 1999.

61. J. Herrero, J. Portas, F. Rodriguez, and J. Corredera, “Surface movement radar data processing methods for airport surveillance,” *IEEE Trans. Aerospace Electronic Sys.*, vol. 37, pp. 563-585, 2001.

62. R. Castaldo, C. C. Franck, and A. Smith, “Evaluation of FLIR/IR camera technology for airport surface surveillance,” *Proc. of the SPIE*, vol. 2736, pp. 64-74, 1996.

63. A. Smith, C. Evers, R. Cassell, “Evaluation of airport surface surveillance technologies,” *1996 CIE Int. Conf. of Radar Proc.*, pp. 535-538, 1996.

64. C. Jaynes, “Multi-view calibration from planar motion for video surveillance,” in *Proc. Second IEEE Workshop Visual Surveillance*, pp. 59-66, 1999.

65. R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision, Vol. 2.* Reading, MA: Addison-Wesley, 1993.

66. K. Bradshaw, I. Reid, and D. Murray, “The active recovery of 3D motion trajectories and their use in prediction,” *IEEE Trans. PAMI*, vol. 19, pp. 219-234, 1997.

67. P. Kumar, K. Sengupta, and A. Lee, “A comparative study of different color spaces for foreground and shadow detection or traffic monitoring system,” in *IEEE 5th Int. Conf. Intelligent Transportation Sys.*, pp. 100-105, 2002.

BIBLIOGRAPHY

J. Badenas, J. Sanchiz, and F. Pla, "Using temporal integration for tracking regions in traffic monitoring sequences," in *Proc. 15th Int. Conf. Pattern Recog.*, vol. 3, pp. 1125-1128, 2000.

B. Bell, "The marginal likelihood for parameters in a discrete Gauss-Markov process," *IEEE Trans. Sig. Proc.*, vol. 48, pp. 870-873, 2000.

D. Beymer, P. McLauchlan, B. Coifman, and J. Malik, "A real-time computer vision system for measuring traffic parameters," in *1997 IEEE Comp. Soc. Conf. on Comp. Vision and Pattern Recognition*, pp. 495-501, 1997.

S. Bouzar, F. Lenoir, J. M. Blosseville, and R. Glachet, "Traffic measurement: image processing using road markings," in *Eighth Int. Conf. on Road Traffic Monitoring and Control*, pp. 105-109, 1996.

K. Bradshaw, I. Reid, and D. Murray, "The active recovery of 3D motion trajectories and their use in prediction," *IEEE Trans. PAMI*, vol. 19, pp. 219-234, 1997.

C. G. Broyden, "The convergence of a class of double-rank minimization algorithms," *J. Inst. Math. Applic*., vol. 6, pp. 76-90, 1970.

J. Canny, "A computational approach to edge detection." *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 8, pp. 679-698, 1986.

R. Castaldo, C. C. Franck, and A. Smith, "Evaluation of FLIR/IR camera technology for airport surface surveillance," *Proc. of the SPIE*, vol. 2736, pp. 64-74, 1996

F. Chausse, R. Aufrere, R. Chapuis, "Recovering the 3D shape of a road by on-board monocular vision," in *15th Int. Conf. on Pattern Recognition*, vol. 1, pp. 325-328, 2000.

T.F. Coleman and Y. Li, "An interior, trust region approach for nonlinear minimization subject to bounds," *SIAM J. Optim.*, vol. 6, pp. 418-445, 1996.

T.F. Coleman and Y. Li, "On the convergence of reflective Newton methods for large-scale nonlinear minimization subject to bounds," Mathematical Programming, vol. 67, pp. 189-224, 1994.

D. J. Dailey, F. W. Cathey, and S. Pumrin, "An algorithm to estimate mean traffic speed using uncalibrated cameras," *IEEE Trans. Intelligent Transportation Sys.*, vol. 1, pp. 98-107, 2000.

L. de Agapito, R. I. Hartley, and E. Hayman, "Linear self-calibration of a rotating and zooming camera," in *IEEE Comp. Soc. Conf. on Comp. Vis. and Pattern Recog.* (Ft. Collins, CO), vol. 1, pp. 15-21, Dec. 1999.

R.O. Duda and P.E. Hart, “Use of the Hough transformation to detect lines and curves in pictures,” *Comm. ACM*, vol. 15, pp. 11-15, 1972.

Z. Fan, J. Zhou, D. Gao, and G. Rong, “Robust contour extraction for moving vehicle tracking,” in *Proc. Int. Conf. Image Proc.*, pp. 625-628, 2002.

M. Fathy and M. Y. Siyal, “An image detection technique based on morphological edge detection and background differencing for real-time traffic analysis,” *Patt. Recog. Letters*, vol. 16, pp. 1321-1330, 1995.

J. Ferryman, A. Worrall, G. Sullivan, K. Baker, “Visual surveillance using deformable models of vehicles,” *Robotics and Autonomous Sys.*, vol. 19, pp. 315-335, 1997.

R. Fletcher, “A new approach to variable metric algorithms," *Computer Journal*, vol. 13, pp. 317-322, 1970.

A. Fusiello, “Uncalibrated Euclidean reconstruction: a review,” *Image and Vision Computing*, vol. 18, pp. 555-563, 2000.

G. Garibotto, P. Castello, E. Del Ninno, P. Pedrazzi, G. Zan, “Speed-vision: speed measurement by license plate reading and tracking,” in *IEEE 5th Int. Conf. Intelligent Transportation Sys.*, pp. 585-590, 2002.

A. Giachetti, M. Cappello, and V. Torre, “Dynamic segmentation of traffic scenes,” in *Proc. Intelligent Vehicles ’95 Symposium*, pp. 258-263, 1995.

D. Goldfarb, “A family of variable metric updates derived by variational means," *Mathematics of Computing*, Vol. 24, pp. 23-26, 1970.

R. C. Gonzalez and R. E. Woods. *Digital Image Processing*, 2nd ed. Upper Saddle River, New Jersey: Prentice-Hall, 2002.

S. Gupte, O. Masoud, R. F. K. Martin, and N. P. Papanikolopoulos. “Detection and classification of vehicles,” *IEEE Trans. Intelligent Transportation Sys.*, vol. 3, pp. 37-47.

R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision, Vol. 2.* Reading, MA: Addison-Wesley, 1993.

J. Herrero, J. Portas, F. Rodriguez, and J. Corredera, “Surface movement radar data processing methods for airport surveillance,” *IEEE Trans. Aerospace Electronic Sys.*, vol. 37, pp. 563-585, 2001.

C. Jaynes, “Multi-view calibration from planar motion for video surveillance,” in *Proc. Second IEEE Workshop Visual Surveillance*, pp. 59-66, 1999.

Y. K. Jung and Y. S. Ho, “Traffic parameter extraction using video-based vehicle tracking,” in *Proc. 1999 IEEE/IEEJ/JSAI Int. Conf. on Intelligent Transportation*, pp. 764-769, 1999.

J. Kato, T. Watanabe, S. Joga, J. Rittscher, and A. Blake, “An HMM-based segmentation method for traffic monitoring movies,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, pp. 1291-1296, 2002.

H. Kim and K. S. Hong, “Practical self-calibration of pan-tilt cameras,” *IEE Proc. on Vis., Image, and Signal Process.*, vol. 148, pp. 349-355, 2001.

J. Kim, C. Lee, K. Lee, T. Yun, and H. Kim, “Wavelet-based vehicle tracking for automatic traffic surveillance,” in *Proc. IEEE Region 10 Int. Conf. Electrical and Electronic Tech.*, vol. 1, pp. 313-316, 2001.

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 *Proc. 33rd IEEE Conf. Decision and Control*, vol. 4, pp. 3776-3781, 1994.

H. Kollnig and H. Nagel, “3D pose estimation by fitting image gradients directly to polyhedral models,” in *Proc. Fifth Int. Conf. Comp. Vision*, pp. 569-574, 1995.

P. Kumar, K. Sengupta, and A. Lee, “A comparitive study of different color spaces for foreground and shadow detection or traffic monitoring system,” in *IEEE 5th Int. Conf. Intelligent Transportation Sys.*, pp. 100-105, 2002.

A. Lai and N. Yung, “Lane detection by orientation and length discrimination,” *IEEE Trans. Sys., Man, and Cybernetics—Part B: Cybernetics*, vol. 30, pp. 539-548, 2000.

C. Li, K. Ikeuchi, M. Sakauchi, “Acquisition of traffic information using a video camera with 2D spatio-temporal image transformation technique,” in *Proc. IEEE/IEEJ/JSAI Int. Conf. Intelligent Transportation Sys.*, pp. 634-638, 1999.

S. Maybank, O. Faugeras, “A theory of self-calibration of a moving camera,” *Int. J. Comp. Vis.*, vol. 8, pp. 123-151, 1992.

R. Mohr and B. Triggs, “Projective geometry for image analysis,” tutorial at *Int. Symp. of Photogrammetry and Remote Sensing*, 1996. http://www.inrialpes.fr/movi/people/Triggs/p/Mohr-isprs96.ps.gz.

C. Ooi and P. Liatsis, “Co-evolutionary-based active contour models in tracking of moving obstacles,” in *Int. Conf. Advanced Driver Assistance Sys.*, pp. 58-62, 2001.

T. W. Pai, W. J. Juang, and L. J. Wang, “An adaptive windowing prediction algorithm for vehicle speed estimation,” in *IEEE 5th Int. Conf. Intelligent Transportation Sys.*, pp. 901-906, 2002.

M. Pollefeys, “Tutorial on 3D modeling from images,” *2001 European Conf. Comp. Vis.* http://www.esat.kuleuven.ac.be/~pollefey/SMILE2/tutorial.html.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge, UK: Cambridge University Press, 1988.

Puget Sound traffic cameras. *http://www.wsdot.wa.gov/PugetSoundTraffic/cameras/*

S. Pumrin, “A framework for dynamically measuring mean vehicle speed using un-calibrated cameras,” Technical report UWEETR-2002-0005, *Univ. Washington Dept. of Elect. Engr.*, 2002.

S. Pumrin and D. J. Dailey, “Roadside camera motion detection for automated speed measurement,” in *IEEE 5th Int. Conf. Intelligent Transportation Sys.*, pp. 147-151, 2002.

T. Schoepflin and D. Dailey, “A correlation technique for estimating traffic speed from cameras,” *Transportation Research Board Annual Meeting*, 2002.

T. Schoepflin and D. Dailey, “Dynamic calibration of roadside traffic management cameras,” in *IEEE Int. Conf. Intelligent Transportation Systems*, pp. 25-30, 2002.

C. Setchell, E. Dagless, “Vision-based road-traffic monitoring sensor,” in *IEE Proc. Vision, Image and Sig. Proc.*, vol. 148, pp. 78-84, 2001.

D.F. Shanno, “Conditioning of quasi-Newton methods for function minimization," *Mathematics of Computing*, Vol. 24, pp. 647-656, 1970.

D. Shrank and T. Lomax, “The 1999 annual mobility report: information for urban America,” http://www.itsdocs.fhwa.dot.gov/jpodocs/rept_mis/9dv01!.PDF.

A. Smith, C. Evers, R. Cassell, “Evaluation of airport surface surveillance technologies,” *1996 CIE Int. Conf. of Radar Proc.*, pp. 535-538, 1996.

B. D. Stewart, I. Reading, M. S. Thomson, et. al, “Adaptive lane finding in road traffic image analysis,” in *Seventh International Conf. on Road Traffic Monitoring and Control*, pp. 133-136, 1994.

R. Szeliski and H. Shum, “Creating full view panoramic image mosaics and environment maps,” in *Int. Conf. Comp. Graphics and Interactive Techniques*, pp. 251-258, 1997.

H. Takada, M. Momozawa, and H. Kawano, “A study of the measurement of velocity by image sensors,” in *2001 IEEE Conf. Comm., Comp., and Sig. Proc.*, vol. 1, pp. 251-254.

H. Taniguchi, T. Nakamura, H. Furusawa, “Methods of traffic flow measurement using spatio-temporal image,” in *Proc. Int. Conf. Image Proc.*, vol. 4, pp. 16-20, 1999.

Transportation Research Board, “Traffic flow theory: a revised monograph on traffic flow theory,” *http://www.tfhrc.gov/its/tft/tft.htm*.

S. T. Tseng and K. T. Song, “Real-time image tracking for traffic monitoring,” in *IEEE 5th Int. Conf. Intelligent Transportation Sys.*, pp. 1-6, 2002.

United States Dept. of Transportation, “The Future of transportation starts here,” *http://www.itsdocs.fhwa.dot.gov/jpodocs/brochure/2kk01!.pdf*.

United States Dept. of Transportation, “Intelligent transportation systems benefits: 2001 update,” *http://www.itsdocs.fhwa.dot.gov/jpodocs/repts_te/13463.pdf*.

J. Yang and S. Ozawa, “Recover of 3-D road plane based on 2-D perspective image analysis and processing,” *IEICE Trans. Fund. Elect.,Comm., and Comp. Sci.*, vol. E79-A, pp. 1188-1193, 1996.

Z. Yang and W. Tsai, “Viewing corridors as right parallelepipeds for vision-based vehicle localization,” *IEEE Trans. Industrial Electronics*, vol. 46, pp. 653-661, 1999.

M. Yu, G. Jiang, and B. Yu, “An integrative method for video based traffic parameter extraction in ITS,” in *2000 IEEE Asia-Pacific Conf. on Circuits and Sys.*, pp. 136-139, 2000.

Z. Zhu, G. Xu, B. Yang, D. Shi, and X. Lin, “VISATRAM: A real-time vision system for automatic traffic monitoring,” *Image and Vision Computing*, vol. 18, pp. 781-794, 2000.