**Final Research Report**
Agreement T2695, Task 55
CCTV Phase 3

# The Automated Use of Un-Calibrated CCTV Cameras as Quantitative Speed Sensors
## Phase 3

by

Daniel J. Dailey and Frederick W. Cathey
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
Eldon Jacobson, 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

January 2006

# TECHNICAL REPORT STANDARD TITLE PAGE

| 1. REPORT NO. | 2. GOVERNMENT ACCESSION NO. | 3. RECIPIENT'S CATALOG NO. |
|---|---|---|
| WA-RD 635.1 | | |

| 4. TITLE AND SUBTITLE | 5. REPORT DATE |
|---|---|
| THE AUTOMATED USE OF UN-CALIBRATED CCTV CAMERAS AS QUANTITATIVE SPEED SENSORS —PHASE 3 | January 2006 |
| | 6. PERFORMING ORGANIZATION CODE |
| 7. AUTHOR(S) | 8. PERFORMING ORGANIZATION REPORT NO. |
| Daniel J. Dailey and Frederick W. Cathey | |
| 9. PERFORMING ORGANIZATION NAME AND ADDRESS | 10. WORK UNIT NO. |
| Washington State Transportation Center (TRAC)<br>University of Washington, Box 354802<br>University District Building; 1107 NE 45th Street, Suite 535<br>Seattle, Washington 98105-4631 | 11. CONTRACT OR GRANT NO.<br>Agreement T2695, Task 55 |
| 12. SPONSORING AGENCY NAME AND ADDRESS | 13. TYPE OF REPORT AND PERIOD COVERED |
| Research Office<br>Washington State Department of Transportation<br>Transportation Building, MS 47370<br>Olympia, Washington 98504-7370<br>Doug Brodin, Project Manager, 360-705-7972 | Final 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

The Washington State Department of Transportation (WSDOT) has a network of several hundred closed-circuit television (CCTV) traffic surveillance cameras that are deployed for congestion monitoring on the freeways and arterials around Seattle. The goal of the first two phases of this project was to create *algorithms* that would allow these cameras to make continuous quantitative measurements of vehicle speed. In the first two phases, a number of algorithms were developed and tested; the most successful of these was chosen for implementation in this, Phase 3. The goal of this third phase was to implement the algorithms as *prototype software*.

| 17. KEY WORDS | 18. DISTRIBUTION STATEMENT |
|---|---|
| Closed-circuit television (CCTV), traffic surveillance, congestion monitoring, speed sensors, camera calibration | 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

The contents of this report reflect the views of the authors, who are responsible for the facts and accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the Washington State Transportation Commission, Department of Transportation, or the Federal Highway Administration. This report does not constitute a standard, specification, or regulation.

# Table of Contents

**Executive Summary** ..... **vii**

**1. Introduction** ..... **1**

**2. Review of Previous Work** ..... **3**

2.1 Phase 1 ..... 3

2.2 Phase 2 ..... 4

**3. Research Approach** ..... **7**

3.1 Overall Application Design ..... 7

3.2 Algorithmic View of the Application ..... 12

3.3 Calibration ..... 12

3.4 Speed Measurement ..... 21

**4. Summary** ..... **25**

**5. Conclusions/Recommendations** ..... **26**

**References** ..... **27**

## Table of Figures

Figure E-1: Video camera calibration and speed measurement application. viii
Figure 1: Overall steps in algorithm. 8
Figure 2: Application graphical user interface. 9
Figure 3: Calibration operations 13
Figure 4: Background image. 14
Figure 5: Gradient magnitude image. 15
Figure 6: Gradient angle map. 15
Figure 7: Binary image resulting from applying a threshold to the magnitude map. 16
Figure 8: Relationship between image and Hough coordinate systems. 17
Figure 9: Output of the Hough transform. 17
Figure 10: Background image with identified lines and vanishing point. 18
Figure 11: Background image and straightened projection of the roadway. 19
Figure 12: Autocorrelation of straightened image. 20
Figure 13: Autocorrelation along column with lane stripes. 20
Figure 14: Roadway and image distances. 21
Figure 15: Vehicle images on straightened roadway. 22
Figure 16: Cross-correlation between columns in the straightened image. 23
Figure 17: Traffic slowing to a stop at high density. 24
Figure 18: Comparison of loop data and video data for the slowing traffic scene. 24

# EXECUTIVE SUMMARY

The Washington State Department of Transportation (WSDOT) has a network of several hundred closed-circuit television (CCTV) traffic surveillance cameras that are deployed for congestion monitoring on the freeways and arterials around Seattle. The goal of the first two phases of this project was to create *algorithms* that would allow these cameras to make continuous quantitative measurements of vehicle speed. In the first two phases, a number of algorithms were developed and tested; the most successful of these was chosen for implementation in this, Phase 3. The goal of this third phase was to implement the algorithms as *prototype software*.

To use cameras to measure traffic speed, the cameras must first be calibrated. For the cameras under consideration, no camera calibration parameters are known *a priori*, and WSDOT operators dynamically change the focal length, pan angle, and tilt angle of these cameras. However, to measure vehicle speed from a series of video images, this report demonstrates that it is not necessary to completely calibrate the camera. Instead, by using the vanishing point of lines in the image, algebraic constraints on the calibration parameters are sufficient to straighten the image and compute a scale factor for estimating speed.

In surveillance situations with favorable geometry, the software developed can measure traffic speeds from a series of video images by first determining the calibration parameters and then using these parameters to estimate traffic speeds. The software is suitable for cameras that surveille approaching and/or receding traffic where the line of sight is directed slightly downward and more or less along the highway, rather than across it. The calibration step requires that highway lanes and lane boundaries be approximately straight in the region of surveillance near the camera.

Chapter Two provides details on the algorithms used and the calibration process. The three phases for implementing calibration are (1) highway line detection, (2) computation of the vanishing point and image straightening transformation, and (3) computation of the image-to-highway scale factor (feet/pixel). This calibration is implemented in a prototype application.

Chapter Three describes the prototype Java application created with the developed algorithms. The resulting application allows a user to select from a list of cameras, examine the camera view, calibrate the camera, and record speed data to a file. A screen image of the developed application is shown in Figure E-1.

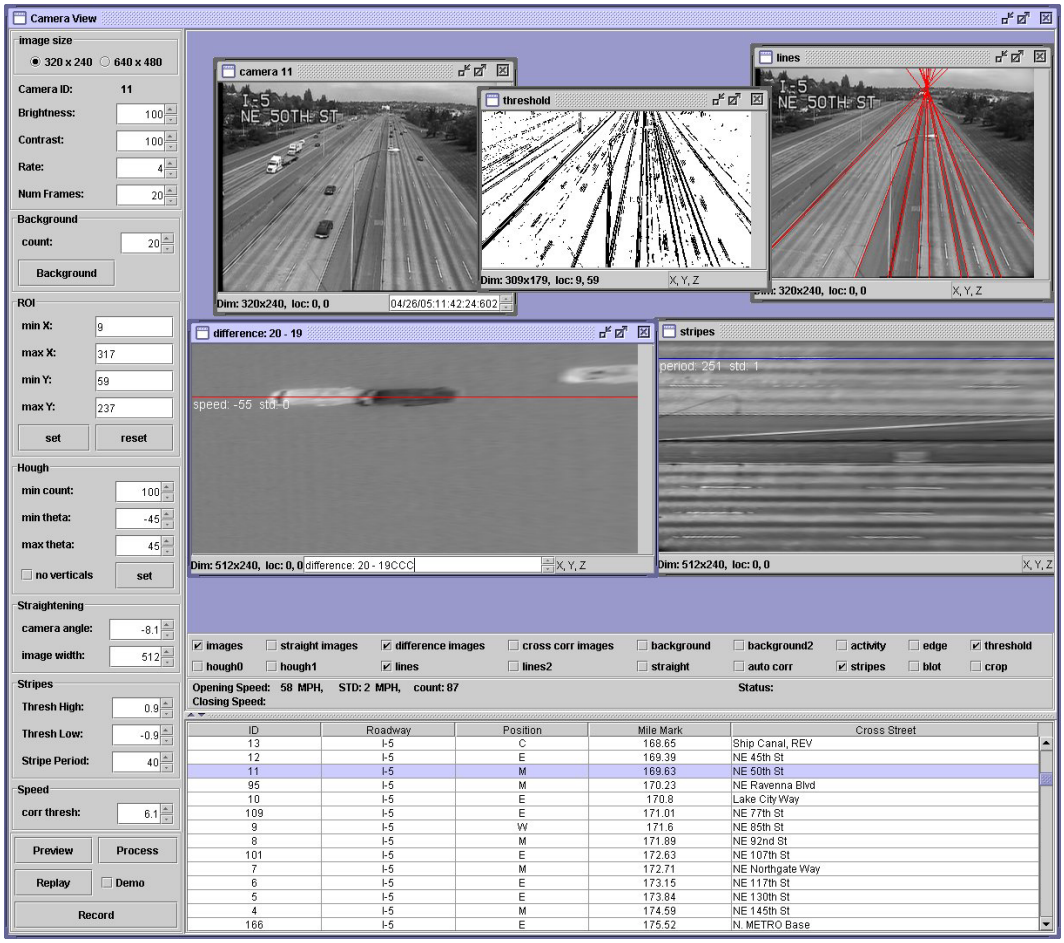


**Figure E-1: Video camera calibration and speed measurement application.**

This research created a prototype set of software and algorithms capable of producing speed estimates from cameras by using several frames per second. The prototype could be implemented in an operations center for use by DOT personnel and the application could be refined for traffic management tasks.

The presently installed set of cameras represents a large financial investment to all DOTs nationally. This research allows quantitative speed to be estimated from the existing camera pool and eliminates the need for additional, specially calibrated cameras and associated infrastructure. Furthermore, as cameras are deployed on arterial routes, this technology will allow quantitative traffic speed measurements to be estimated from those cameras; such measurements would otherwise be available only by installing additional, expensive traffic sensors (e.g., loops or radar).

The technology to produce the image streams required for this prototype is becoming cheaper year by year, and many DOTs have video cameras online capable of delivering such an image stream. Initially, the work will benefit both WSDOT operations and other ongoing research projects that require quantitative information from the roadways where loops do not exist. Long-term, this work will benefit any DOT operations that use un-calibrated cameras for surveillance. The number of DOTs using this technology is growing daily, and the potential to use the technology developed here exists in nearly every state.

# 1. INTRODUCTION

This report presents a new computer vision sensor approach to traffic speed estimation that uses un-calibrated highway surveillance cameras. Innovative image processing techniques that are presented here include (1) the use of perspective straightening for image linearization, (2) an autocorrelation technique for lane stripe detection and estimation of the linear pixel/foot scale factor, and (3) a cross correlation technique used to estimate mean traffic speeds and direction of flow. The approach is implemented for demonstration in prototype software.

The Washington State Department of Transportation (WSDOT) has a network of several hundred closed-circuit television (CCTV) traffic surveillance cameras that are deployed on the freeways and arterials around Seattle for congestion monitoring. Approximate location information is provided for each camera, such as highway name, cross-street, and position relative to the highway (north, south, east, west, median, or center), but no calibration parameters are provided. Camera height *(h)* above ground is unknown, and WSDOT operators can change the focal length (*f*), pan angle ($\theta$), and tilt angle ($\phi$) at will by remote control.

In surveillance situations with favorable geometry, the demonstration application can measure traffic speeds from a series of video images without actually determining the calibration parameters ($h, f, \theta$), or ($\phi$), assuming that they are constant for the series of images being used. The algorithms, implemented in software, are suitable for cameras that surveille approaching and/or receding traffic where the line of sight is directed slightly downward and more or less along the highway, rather than across it. The calibration step requires highway lanes and lane boundaries to be approximately straight

in the region of surveillance near the camera. Under these conditions, the highway lines will appear straight in a camera image and will have a recognizable vanishing point.

Given the highway vanishing point, a *straightening* transformation is defined that maps lines parallel to the highway to vertical image lines. Moreover, the highway-to-image scale factor is constant for each line, and it can be determined by measuring the vertical dimension of an image feature with known length along the highway, such as lane stripe period. Because traffic pixel speed can be estimated by cross correlating lines in successive pairs of straightened images, we can readily estimate speeds on the highway by using the scale factor.

This report covers two main topics. Image processing algorithms for highway vanishing point estimation and image straightening, as well as scale factor and speed estimation, are discussed at a high level. In addition, a software application of these concepts and algorithms that interfaces with the WSDOT camera system is presented.

# 2. REVIEW OF PREVIOUS WORK

## *2.1 Phase 1*

This section reports the results from the first phase, which applied the algorithms first developed to images in a variety of lighting conditions. The results indicated that the method is a viable alternative to calibrated cameras, when the region from the image used to estimate speed is appropriately constrained [Dailey et al. 2001]. This effort did not explicitly use a camera model.

Images from roadside cameras were used with appropriate constraints to obtain speed estimates suitable for comparison with both “ground truth” and inductance loop speed estimates. Speeds were estimated by using three practical constraints. The first constraint was that only vehicles whose projection was over seven-pixels in length were used. The second constraint was that only vehicles identified in the lower half of the image were used for scale factor and speed estimation. The third constraint was that the apparent angle of the roadway in the image had to be between 50 and 130.

The accuracy of the algorithm was evaluated in two ways. First, a ground truth estimate of vehicle speed was compared to the algorithm results, and second, the time average algorithm result was compared to equivalent inductance loop speed measurements. The ground truth speed was obtained by placing calibration lines on the highway and measuring “by hand” the time for individual vehicles to pass the sequential highway markings in sequential images. This ground truth individual vehicle speed could then be compared to the estimates derived from the algorithm. The algorithm presented depended on using the mean of a distribution of vehicle dimensions. A single vehicle represented a realization from this distribution and not the mean. As a result, there would be errors in the estimate of any individual vehicle speed. If speed estimates were made

for a number of vehicles, the errors in these estimates would have a distribution, and the mean of this distribution would be zero if the algorithm were performing well.

Traffic speeds derived from inductance loops were also compared with the algorithm estimates. The inductance loop speeds came from the Traffic Data Acquisition and Distribution (TDAD) data mine (http://www.its.washington.edu/tdad), which contains data from the Seattle metropolitan region.

The mean of the difference between the estimates was 4 miles per hour, indicating a bias in one of the estimators. However, the deviation for that mean was relatively symmetric, indicating that the underlying process creating the estimates, the variability of speed on the freeway, was the same.

### *2.2 Phase 2*

In this phase, a new set of algorithms, based on a quantitative camera model, was developed to calibrate a roadside traffic management camera and process the images to estimate the mean vehicle speed [Schoepflin and Dailey 2003]. Past work that had used a camera and camera model to estimate traffic conditions had generally postulated that either the focal length or 2-D homography must be available *a priori*, or that a human operator had to move the camera to an assigned and calibrated position. For the WSDOT cameras, *a priori* calibration information is not available. The equipment for this effort consists of monocular roadside cameras controlled by traffic management center operators who can pan, tilt, and zoom them at will. The algorithm developed had the role of a passive viewer of the available images and had to be able to calibrate the camera by 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 low frame rate (three to five frames/second). Thus, this problem presents major obstacles in both camera calibration and tracking.

A simplified camera model designed to convert distance measurements in the image to estimates of distances along the road was developed. 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. The 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.

A morphological top-hat operator, a well-known technique in the image-processing community, was introduced as an important technique for generating feature images with which to analyze traffic scenes. After the images have been manipulated appropriately, the lane marker interval is estimated by using the ensemble of auto-covariance sequences for the image. This method yields an estimate of the lane marker interval that is nearly identical to the hand-calibrated result in a typical scene.

The average spatial shift of the vehicles in a lane of traffic is estimated by 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, as compared to sparse traffic conditions. Furthermore, the method is computationally inexpensive and is suitable for implementation at high frame rates (e.g., 30 Hz).

After a Kalman filter was applied to the output of the spatial shift estimator, the mean speed estimation method presented in the report functioned well, even with raindrops on the camera lens, under dark conditions, when traffic was 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, the algorithm estimated space mean speed, which has better theoretical properties than the time mean speed estimated by inductance loops.

The authors 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, and this should be the next stage of the research.

# 3. RESEARCH APPROACH

This section provides a high-level discussion of the application and algorithms that were produced by this research effort. Mathematical details associated with each step or sub-step can be found in the companion Technical Report, which provides detailed mathematical proofs for each of the steps in the algorithms [Dailey and Cathey 2005]. The goal of this section is to provide an overall description of the application and each of the steps in the calibration and estimation of speed that can be understood by a practicing engineer without being bogged down in mathematical details. The algorithms are described in a combination of narratives and figures so that the reader can understand the operation and limitations of the algorithms/application, and references are provided to the Technical Report as appropriate.

## *3.1 Overall Application Design*

A distributed application has been developed in Java for traffic video image acquisition, camera calibration, and speed estimation. The server program, “Capture,” (running in the WSDOT local network) has access to the CCTV camera video switch that grabs and serves video image sequences. A client application program, “AutoCalSpd,” requests and processes the image sequences. Communication between server and client is through a “controlled proxy” program, which prevents unauthorized users from accessing the server.

This section discusses the implementation of the image processing client application, AutoCalSpd. There are two main parts to the application: the image processor that performs the calculations for calibration and speed estimation and a graphical user interface that allows camera selection, parameter tuning, and visibility for

the various image processing stages. An interactive mode of operation allows the user to supervise the calibration, and an automatic mode allows speed data to be continuously recorded into a file.

The steps necessary to estimate traffic speed from video images are shown in Figure 1 and consist of (1) selecting a camera, (2) acquiring images, (3) calibrating the selected camera in the present position, if possible, and if the camera calibration is successful, (4) making speed estimates of oncoming and receding traffic. Each of these steps is realized in a pipeline-like operation; in fact, the application is implemented by extending the Java Advanced Imaging package from SUN to create a pipeline member for each of the steps.

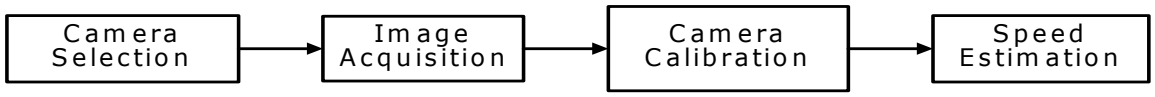


**Figure 1: Overall steps in algorithm.**

The first step is camera selection. The user interface is implemented by using the Java Swing graphical user interface (GUI) toolkit. It consists of three principle components: a control panel, an image display desktop panel, and a camera selector panel. A screenshot of the GUI is shown in Figure 2.The present implementation is based on use of WSDOT's CCTV cameras distributed over the metropolitan Seattle area. A computer is placed in the Traffic Systems Managements Center (TSMC) and is connected to (1) the video output of the switch and (2) the RS232 port on the switch that allows camera selection and the data network to communicate the images and commands between the University of Washington (UW) and TSMC. Associated with each candidate camera is a small database. The database includes acquisition parameters such as image

size, color model (we currently use grayscale), capture rate, brightness, contrast, number of images to capture, and data compression type. It also includes an image processing region of interest (ROI) and any ad-hoc processing rules, as well as some notation about the quality of the location for this application. In the bottom of Figure 2, the client displays the camera database. The user selects a camera by clicking on a row, and the application fetches images from WSDOT TSMC for that camera. The top right panel displays the image obtained for I-5 and 45th Street. At the top left are controls for image size, frame rate, and number of images to be obtained.

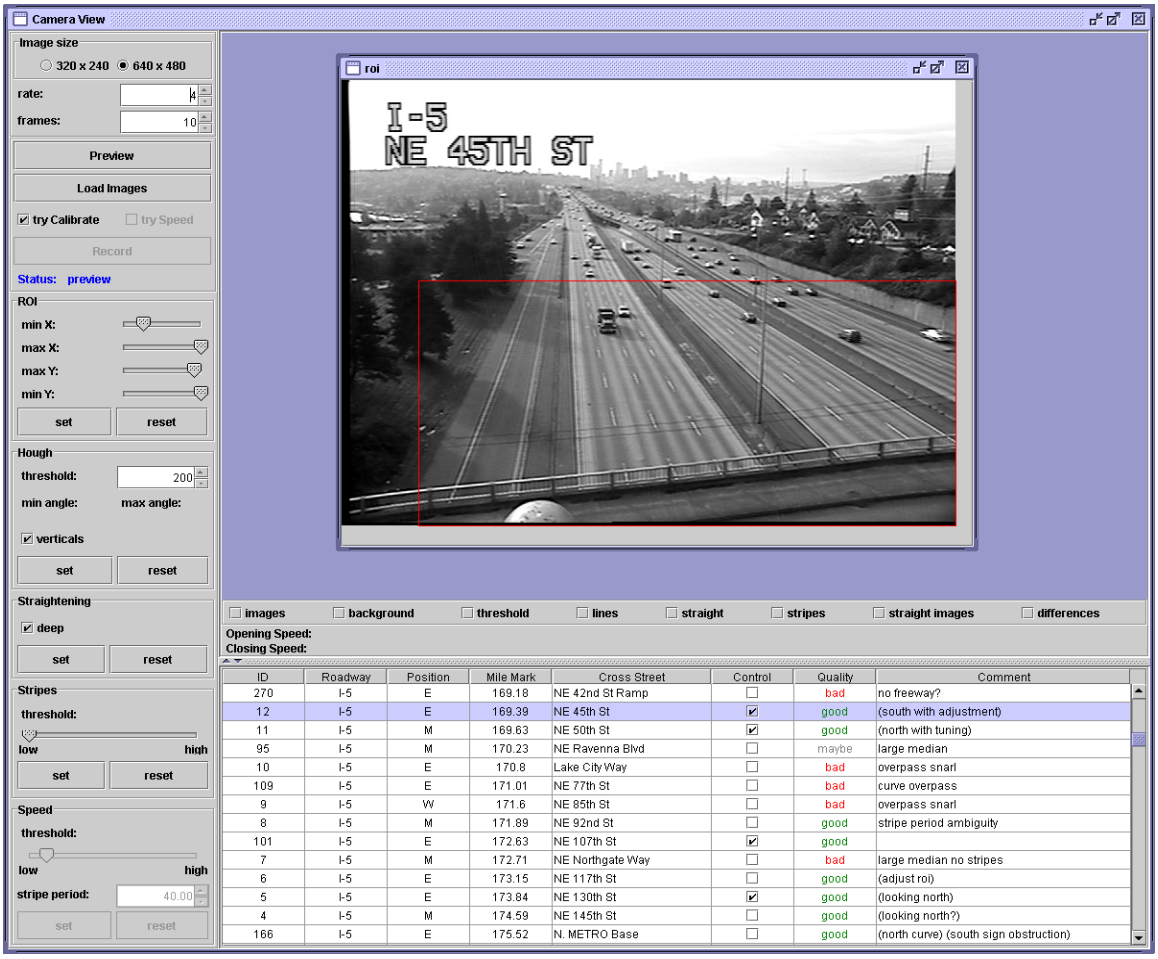


**Figure 2: Application graphical user interface.**

Once a camera has been selected, the image server is instructed to capture images by using the camera-specific acquisition parameters. All calibration parameters are set at their default values, and the “try calibrate” toggle is set, but no calibration is attempted until the “load images” button has been pressed. The “try speed” toggle is disabled because no image-to-highway scale factor is known yet. The user may switch cameras, increase the image size to 640 × 480, or preview images at any time. For calibration purposes, a camera should be selected with the following properties:

- The camera should survey approaching and/or receding traffic; that is, it should point downward and along the highway, rather than across it.
- The view should show straight highway lines in the bottom half of the image as well as lane stripes.
- The view should be mostly unobstructed by overpasses, large overhead signs, divergent lanes of traffic, etc.

Under these conditions, a successful calibration is likely.

To begin a calibration, the user selects the image size, number of images to acquire, and the frame rate and then presses the “load images” button. A request is sent to the Capture server program with the three aforementioned parameter values as well as other (currently uneditable) values for brightness, contrast, compression type, and color model (grayscale). The response from the server is a sequence of time-tagged images, or frames, the first of which is presented in a “captured frames” viewer in the image display desktop. The user can cycle through the images by using “spinner” buttons attached to the bottom of viewer. Because the “try calibrate” toggle button is selected, a calibration process is attempted. The user may view images produced at various stages of the process by pressing labeled buttons under the desktop. This is useful for confirming the validity

of a calibration or diagnosing and perhaps correcting a failure. If the calibration fails, an error message pops up indicating where the failure occurred: either no vanishing point was found or no stripes were detected. The desktop may be used as a diagnostic tool in these cases. Visibility into the image processing stages prior to the failure point may suggest parameter changes that could lead to a successful calibration. For example, if the vanishing point cannot be found, it is useful to view the “lines” image which shows detected lines superimposed on the background image. If too many or too few lines are shown, the user may edit the Hough parameters and force a reactivation of the calibration process starting at the Hough stage. If stripes cannot be found, it is useful to view the “straightened background” image. This may show that not enough stripes are present, in which case the length of the straightened image may be increased and the calibration process reactivated beginning with the straightening step. If the stripes are faint, the stripe detection thresholds may be lowered and the calibration process reactivated beginning with the stripe detection step.

If the calibration process succeeds, a status message under the desktop indicates “successful calibration,” and the “try speed” toggle button is enabled. However, before activating a speed computation, the user should view the “stripes” image to double check that road stripes were actually found rather than some other periodic structure such as construction barrels. (Also, if an insufficient number of image frames is collected, traffic can appear as a periodic structure in the background.)

Selecting “try speed” will start a speed estimation process by using default values for the correlation threshold and the stripe period (40 feet). When this process completes, a status message under the desktop indicates “speeds computed,” and the current approaching and receding traffic speed estimates are displayed also just below the

desktop. If no speeds are shown, then either there is no traffic (which can be verified by spinning through the “captured frames” viewer) or the cross correlation threshold is too high. The user can edit this parameter and reactivate the speed computation. If the speeds appear unreasonable, then the default stripe period may be wrong (some highway stripes are spaced at 12- and 15-foot intervals).

Once satisfied with the camera calibration and speed parameter settings, the user may continue interactively to load images, and as long as “try speed” is selected, speeds will be computed. The automatic mode of operation may be enabled by pressing the “record” button. Then the program will repeatedly load images, compute speed reports, and append them to a file. A test is performed on each cycle to determine whether the camera calibration (scale factor and straightening transformation) is still valid. If not, a popup alerts the user that action needs to be taken, either by selecting “try calibrate” or by selecting a new camera and starting over.

### *3.2 Algorithmic View of the Application*

The camera calibration step is the process for which the most development and intellectual energy was spent. This section provides a high-level description of the detailed mathematical algorithms. Readers who would like to see the detailed math and proofs are referred to the accompanying Technical Report.

### *3.3 Calibration*

To automatically calibrate the camera, the image needs to be “straightened” or warped, so that pixels have the same effective size throughout, and scaled, so that the size of each pixel can be expressed in feet. The straightening operation involves finding a vanishing point in the image, and the scaling operation requires finding lane striping. The

three phases for implementing calibration are (1) highway line detection, (2) computation of the vanishing point and image straightening transformation, and (3) computation of the image-to-highway scale factor (feet/pixel). The algorithm is implemented as a series of operations that are shown in Figure 3. Once the camera has been calibrated, a cross correlation between sequential images is used to estimate traffic speed.

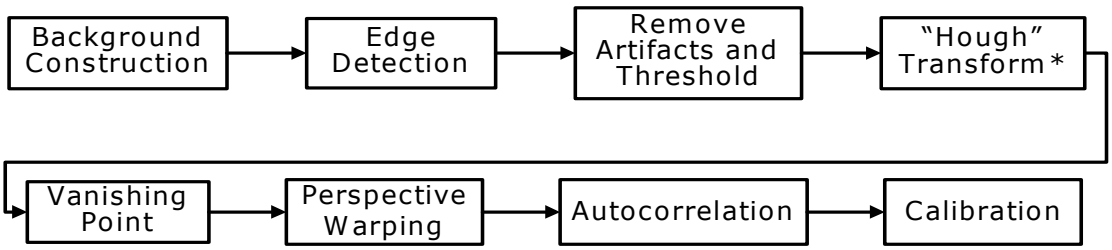


**Figure 3: Calibration operations**

In this discussion, all image processing is restricted to a specified region of interest. We typically use a rectangle with its top edge on the center line of the image, since imagery above this line is typically not usable, and with sides inset from the image sides to eliminate vertical border artifacts produced by the image capture card.

The first phase of calibration, line detection, is implemented by using the four operations found in the top of Figure 3. To obtain lines in the image for calibrating the camera, a background scene of the roadway that does not contain vehicles is needed. This "background map" is constructed by summing a series of images, where

$$B = \frac{1}{n} \sum I_j \ . \qquad \text{Equation 1}$$

Averaging has the effect of suppressing the traffic and enhancing the road boundary curves and stripes. An example background image is shown in Figure 4.


**Figure 4: Background image.**

The next step in identifying lines in the image is edge detection. Edge detection is accomplished by creating a gradient image and applying a threshold to that image. The gradient image, *G*, is constructed by using Sobel kernels, $\partial$, applied to the background image, *B*, in both the horizontal (*u)* direction and the vertical *(v)* direction independently

$$G = (\partial_u B,\ \partial_v B)\ . \qquad \text{Equation 2}$$

The magnitude and angle map images of this gradient are created by using

$$\mathbf{M} = |G| = ((\partial_u B)^2 + (\partial_v B)^2)^{1/2} \qquad \mathbf{\Theta} = \arctan\left(\frac{\partial_v B}{\partial_u B}\right), \qquad \text{Equation 3}$$

where $\mathbf{\Theta}$ is the angle that the gradient vector at (u, v) makes with the u-axis, as shown in Figure 5, and $\mathbf{M}$ is the magnitude map, as shown in Figure 6.

The magnitude map is used to find lines in the image, and the angle map is used to separate closely spaced lines in the image. To do this, a threshold is applied to the magnitude map that changes it from grayscale to binary, as shown in the third step in the top of Figure 3. The value for this threshold is obtained by using the method described by

Otsu [1979]. When this threshold is applied to the magnitude map in Figure 5, the result as is shown in Figure 7, where lines in the image are now clearly emphasized.


**Figure 5: Gradient magnitude image.**

**Figure 6: Gradient angle map.**

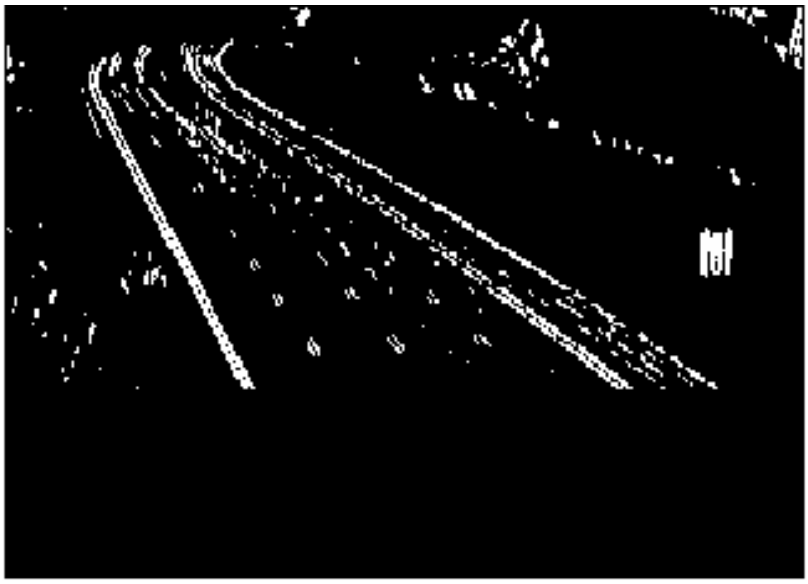

**Figure 7: Binary image resulting from applying a threshold to the magnitude map.**

To quantify the identification of lines in the image, a Radon Transform, sometimes called a Hough transform, is applied to the binary image. This transformation attempts to identify the locations of lines in the image by counting the number of points that are on each possible line and expressing the line location in a range and angle coordinate system (see Figure 8). The Radon Transform output can be expressed as integrals along lines in image *f(u,v)* ,

$$H(\theta, p) = \int_{L_{\theta p}} f(u,v)$$

Equation 4

This effectively provides a function, like that shown in Figure 9, in which the highest points are the locations, in $(\theta, p)$ coordinates, for lines in the image.

In Figure 10 the lines found in the image are augmented with a color that is based on the angle from the angle map at those locations. Note that many features, such as fog lines, become two colored lines with a leading and a trailing edge. This is important because if the points from these two lines are mixed when the program tries to accurately

fit lines to the sets of points, a poor individual line estimate results and a poor vanishing point estimate is made. This observation, that there is a need and a method to separate closely placed parallel lines when lines are fit to find a vanishing point, is a significant contribution to the image processing component of camera calibration.

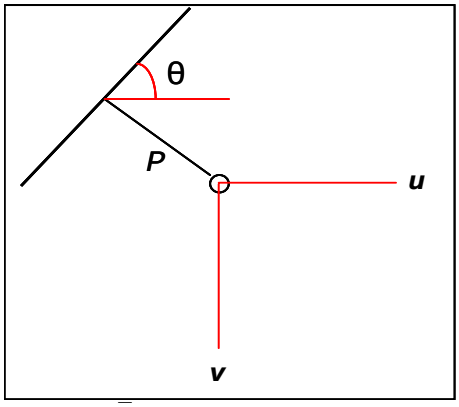


**Figure 8: Relationship between image and Hough coordinate systems.**

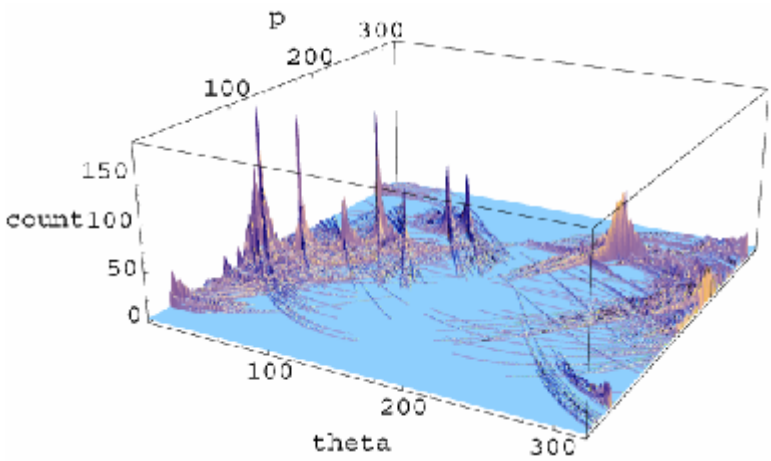


**Figure 9: Output of the Hough transform.**


**Figure 10: Background image with identified lines and vanishing point.**

Now that clear lines can be found in the image, a vanishing point is found by calculating least squares. This completes the functions found in the first box on the bottom of Figure 3. This vanishing point is used with a set of equations presented both in the Phase 2 report and in the accompanying Technical Report to estimate the parameters for the straightening transformation that effectively turns the image into a top down view of the roadway. Figure 11 shows the background image and the resulting straightened image. This completes the function of the second box from the right at the bottom of Figure 3.

The next step in the calibration process uses information in the image to obtain a scale factor to relate distance on the ground to pixels in the image. This is done by searching for the lane stripes in the image. The auto correlation function,

$$C(k, j) = \sum_j S_B(i, j) S_B(i + k, j) \qquad k = 0 \ldots k_{\max} \qquad \text{Equation 5}$$


**Figure 11: Background image and straightened projection of the roadway.**

of the straightened image, shown at right in Figure 11, is constructed and is shown in Figure 12. This function has peaks where repeated features are found in the image, and the lane stripes are a strong repeated feature. The values for the normalized autocorrelation function,

$$NC(k, j) = \frac{C(k, j) - \min_j C(i, j)}{\max_j C(i, j) - \min_j C(i, j)}, \qquad \text{Equation 6}$$

for the column where the peaks appear in Figure 12 is shown in Figure 13. The periodicity of this function is obvious, and the period is easily measured between the peaks. This period is a good estimate of the length, in the image, from the beginning of a lane stripe to the beginning of the next lane stripe. The scale factor, required for calibration, is constructed as the ratio of the length, in feet, on the roadway from the beginning of one stripe to the beginning of the next lane stripe (*L)* and the length in pixels as seen in the image ($\ell$),

$$\kappa = \frac{L}{\ell}. \qquad \text{Equation 7}$$

Figure 14 shows this relationship pictorially. Now that the camera has been calibrated, it can be used to measure traffic speed.

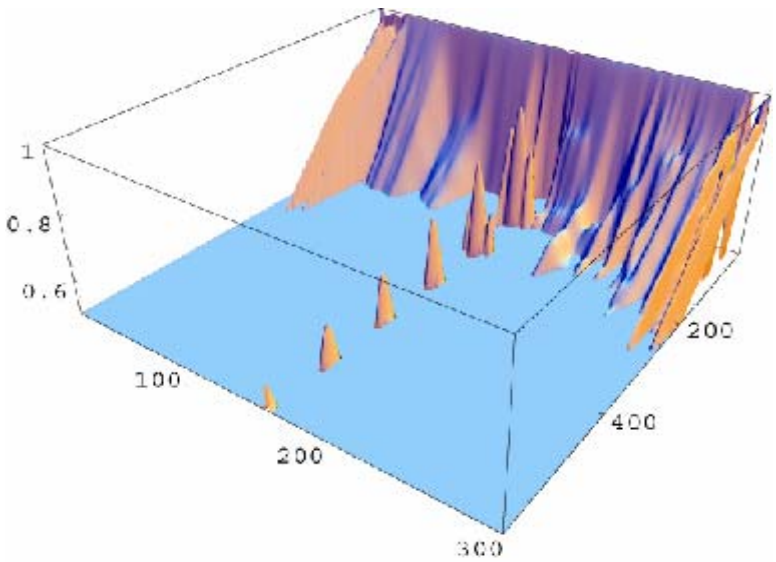


**Figure 12: Autocorrelation of straightened image.**

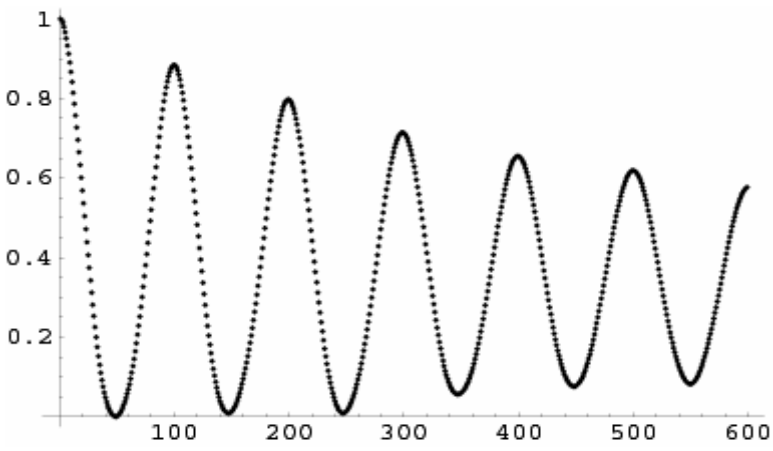


**Figure 13: Autocorrelation along column with lane stripes.**

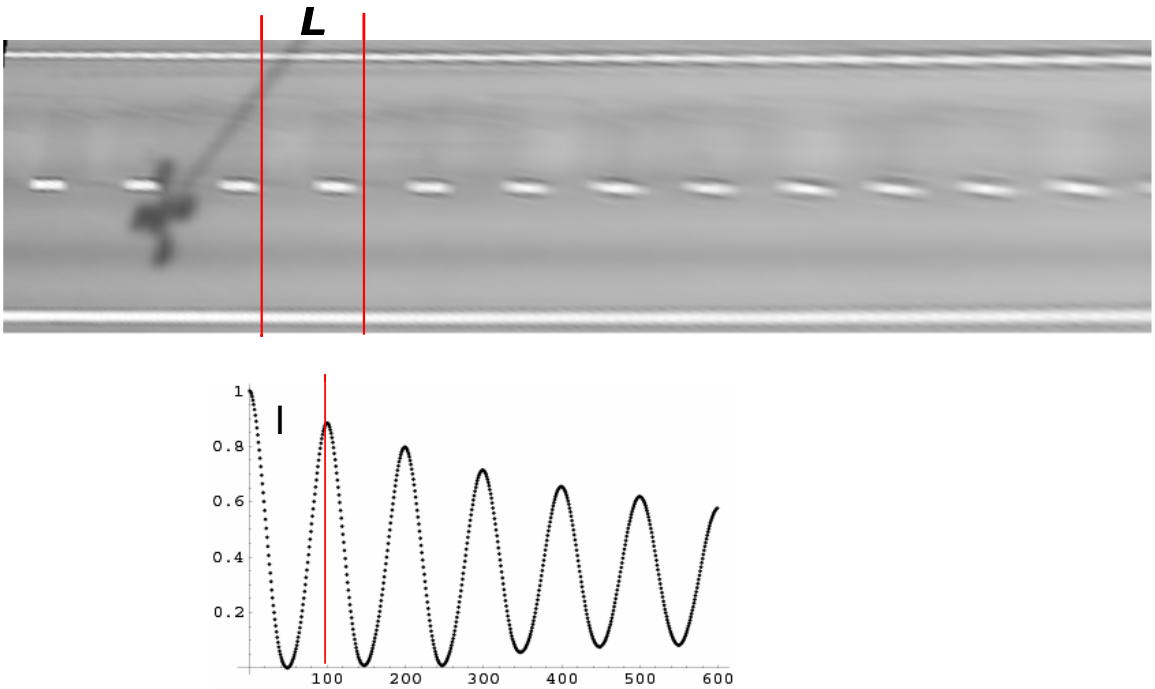


**Figure 14: Roadway and image distances.**

### *3.4 Speed Measurement*

Traffic speed is measured with pairs of sequential images. The images are straightened, and the cross-correlation between the images provides a quantitative measure of the distance traveled in image coordinates, pixels [Schoepflin and Dailey 2003; Schoepflin and Dailey 2004] This pixel distance is then scaled to feet on the roadway by using the scale factor developed just above. The distance traveled is divided by the time between the image frames to obtain a speed measure.

Two sequential straightened images are shown in Figure 15. When these two images are cross-correlated, the cross-correlation function has peaks for repeated features. In this case, the strongest repeated feature is the image of the vehicle. The column from the cross-correlation function that has the peak with the largest value is shown in Figure 16. The location of the peak, at 275 pixels, represents the distance

traveled by the vehicle in the image. The clearly identifiable peak from Figure 16 provides a very reliable estimate for travel distance.

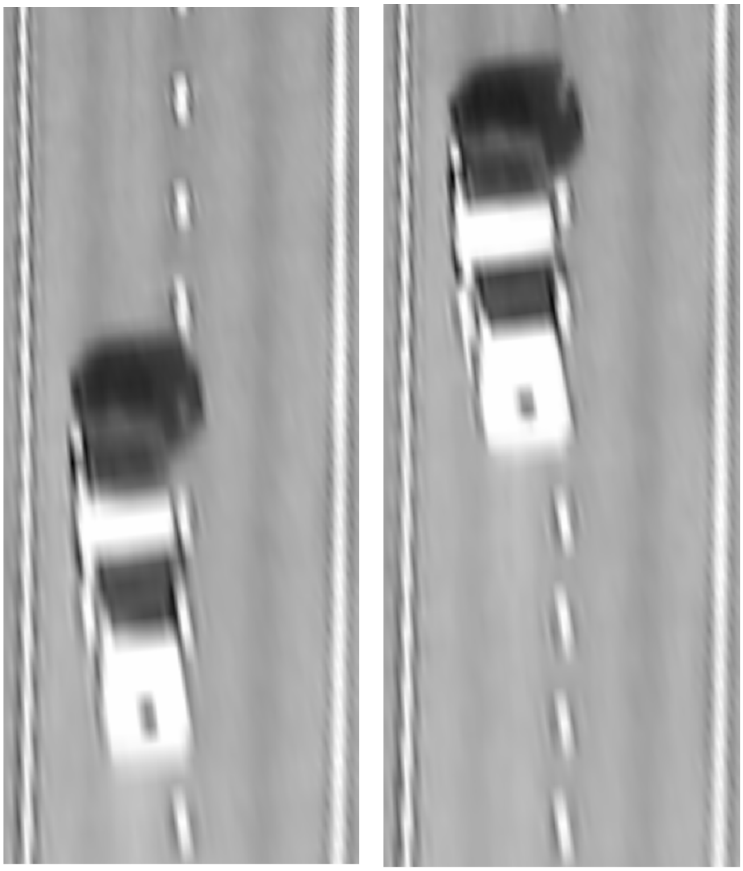

**Figure 15: Vehicle images on straightened roadway.**

The application uses a selectable number of frames in a series to make a robust estimate of speed, and the default number of frames is ten. From these ten frame, quite a number of columns will contain cross-correlations like that shown in Figure 16. When the application reports a traffic speed, it also reports the number of cross-correlations used. In an image with good geometry, the ten frames may produce on the order of 800 cross-correlation peaks.

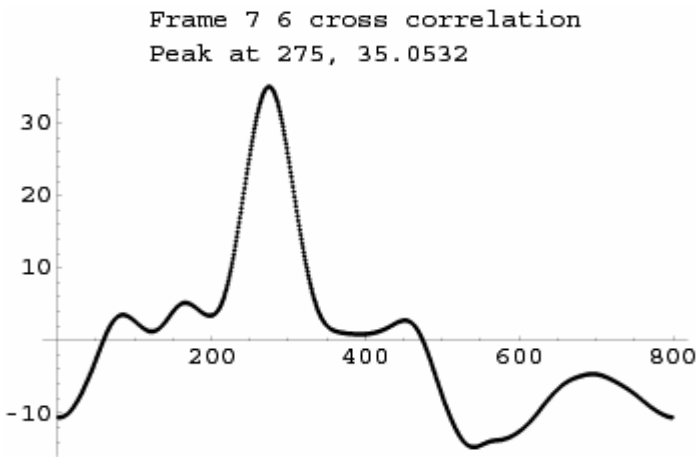


**Figure 16: Cross-correlation between columns in the straightened image.**

An analysis of the errors inherent in the estimates of speed produced with this methodology are presented in detail in the accompanying Technical Report. There, it is demonstrated with statistics, differential calculus, and four assumptions that the overall error in speed is less than 3 miles per hour. The four assumptions are that (1) the error in the location peak in the cross-correlation function is one pixel, (2) the error in measuring stripe length is one pixel, (3) the error in the estimate of the stripe length on the ground is 1 foot, and (4) the inter-frame time is 0.25 seconds. In the actual application, estimates of the errors just mentioned are made continuously, and the application reports both the present speed estimate and a standard deviation for that estimate (e.g., 64 MPH 4 STD).

As an example of the operation of this application, a day was chosen in which traffic became congested and slowed dramatically. Most of the published algorithms that estimate vehicle speed attempt to track individual vehicles. In very dense congestion with a camera relatively high above the roadway, as most freeway cameras are, tracking individual vehicles becomes impossible as a result of occlusion. However, the correlation techniques used here can function in this type of situation. Figure 17 shows two images

of the increasing congestion, and Figure 18 shows both loop data and camera data tracking the traffic speed from freeway speeds to a stopped condition.


**Figure 17: Traffic slowing to a stop at high density.**

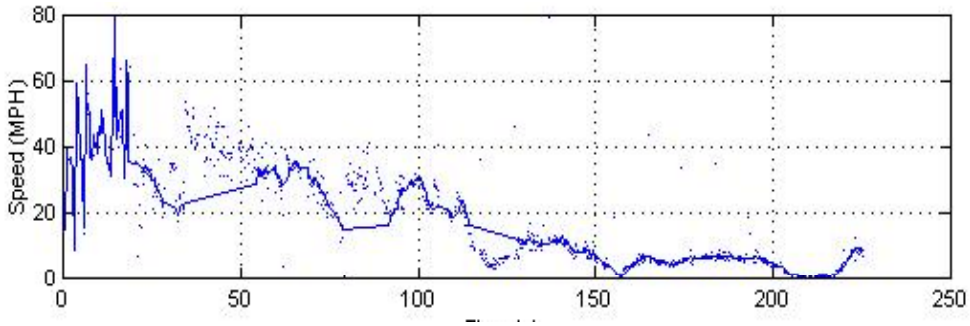


**Figure 18: Comparison of loop data and video data for the slowing traffic scene.**

The result of this effort is that an un-calibrated camera looking at a roadway, with a reasonable geometry, can be used to estimate traffic speed accurately. Moreover, the application that implements the algorithms described provides both a speed estimate and an accurate assessment of the error in that estimate.

# 4. SUMMARY

This report presents the results of efforts to reuse un-calibrated cameras as quantitative roadway sensors. A set of image processing and optimization steps to dynamically calibrate a movable camera are presented. The camera calibration is used to "straighten" the images and to compute an image to roadway scale factor. A cross-correlation method is used with the straightened images to estimate travel distance. The peak in the cross-correlation function is used to identify the distance traveled between frames. This cross-correlation method is robust in the face of high density traffic with much occlusion of vehicles, a situation in which other camera-based algorithms fail. This method has analytical bounds on the accuracy of the speed measurement. An application that uses this these techniques and that has a graphical user interface has been developed and tested on freeways in the Puget Sound region.

# 5. CONCLUSIONS/RECOMMENDATIONS

It is possible to construct an application that can control camera selection, calibrate the images from un-calibrated cameras, and produce a speed estimate along with an error estimate for that speed. These data can be continuously recorded into a file for downstream uses. The application further tests the calibration each time it obtains images and automatically identifies when a new calibration is necessary.

The image frame rate necessary to support this application is well within the rates that can be obtained by modern digital Internet cameras. So while the application and algorithms are developed for use with CCTV cameras, they should generalize to inexpensive, networked digital cameras.

The application was tested and demonstrated by using freeway cameras placed in what WSDOT TSMC personnel have indicated is the default position to which the cameras are returned when not in active use by traffic management personnel. To extend this application to arterials, either the arterial cameras would have to have similar geometries and unobstructed views or the application would need to be extended. The extensions would include (1) templates other than the lane stripes for calibration (e.g., turning arrows) and (2) the ability to mask out obstructions, such as bushes and wires (presently the user can set the region of interest to avoid obstructions and to avoid using vertical lines like those from poles).

Testing by WSDOT operations would provide useful feedback for future upgrades.

## REFERENCES

D.J. Dailey, F.W. Cathey, and S. Pumrin. “The Use of Uncalibrated Roadside CCTV Cameras to Estimate Mean Traffic Speed,” Washington State Transportation Center - TRAC/WSDOT, Final Technical Report, WA-RD 527.1, 26 pages, December 2001.

D.J. Dailey and F.W. Cathey. “CCTV Phase 3 Technical Report,” Washington State Transportation Center - TRAC/WSDOT, Final Technical Report, WA-RD 635.2 44 pages, September 2005.

N. Otsu. “Threshold selection method from gray-level histograms,” IEEE Transactions Systems, Man, and Cybernetics, vol. 9, 1979.

T.N Schoepflin and D.J. Dailey. “Algorithms for Estimating Mean Vehicle Speed Using Uncalibrated Traffic Management Cameras,” Washington State Transportation Center - TRAC/WSDOT, Final Technical Report, WA-RD 575.1, 230 pages, October 2003.

T.N. Schoepflin and D.J.Dailey. “Dynamic Camera Calibration of Roadside Traffic Management Cameras,”, IEEE Transactions on Intelligent Transportation Systems, Vol. 4: No. 2, pp. 90-98, June 2003.

T.N. Schoepflin and D.J. Dailey. “A Cross-Correlation Tracking Technique for Extracting Speed from Cameras Under Adverse Conditions,” Transportation Research Record, 1867, 36-45, 2004.