Development of a Video-Based Traffic Data Collection System

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Hai Quang Dinh

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

Hua Tang

February 2011

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my thesis advisor Dr. Hua Tang for the unlimited support he has given me throughout my graduate course work and thesis research. I would also like to thank my committee members Dr. Eil Kwon and Dr. Mohammed Hasan for their interest and input on this research. Furthermore, I would like to thank the Northland Advanced Transportation Systems Research Laboratories (NATSRL) for providing financial support for this research. I would like to thank the entire University of Minnesota Duluth Electrical and Computer Engineering faculty for all the help they have given me throughout my graduate degrees. Finally, I would like to thank my friends and family for their continuous support in all aspects of life.

ABSTRACT

Automatic traffic data collection can significantly save labor work and cost compared to manual data collection. The collected traffic data is necessary for traffic simulation and modeling, performance evaluation of the traffic scene, and eventually (re)design of the traffic scene. However, automatic traffic data collection has been one of the challenges in Intelligent Transportation Systems (ITS). This thesis presents the development of a single camera based video system for automatic traffic data collection for roundabouts and intersections. The system targets roundabouts and intersections because no mature data collection systems exist for these traffic scenes yet in contrast to highway scenes. The developed system has mainly three steps of processing. First, the camera is calibrated for the traffic scene of interest and a novel circle-based calibration algorithm is proposed for roundabouts. Second, the system tracks vehicles from the video by incorporating powerful imaging processing techniques and tracking algorithms. Finally, the resulting vehicle trajectories from vehicle tracking are analyzed to extract the interested traffic data, which includes vehicle volume, vehicle speed (including acceleration/de-acceleration behavior), travel time, rejected gaps, accepted gaps, follow-up time and lane use. Practical tests of the developed system show that it can reliably track vehicles and provide reasonably accurate traffic data in most cases.

Table of Contents

LIST OF TABLE

LIST OF FIGURES

Chapter 1: Introduction

1.1) Traffic surveillance

The increasing traffic demand in conjunction with the limited construction of new roads has caused recurring congestion in the U.S. While building new facilities is still needed, a much less expensive additional measure is utilizing the use of existing roads, which is one of the important goals in Intelligent Transportation System (ITS). Using current facilities wisely could help reduce travel time and pollutant emissions, ease delay and congestion, and improve safety. To enhance efficient usage and capacity of current transportation networks, detailed traffic data are needed. However, manual collection of traffic data is very laborious and costly. Therefore, automatic traffic data collection has been one of the active research areas. In the past, various approaches to automatic traffic data collection based on inductive loops, microwave radar, cameras, etc were proposed.

Each method has its strength and weakness. For example, whereas the inductive loop technique is mature and well understood, its installation requires pavement cut and the accuracy decreases when a large variety of vehicle classes are to be detected. Microwave radar, whereas it is easier to install and could measure speed directly, has difficulty when detecting stopped vehicles. Another emerging technology is vehicle-based sensor networks, which collects data by locating vehicles via mobile phones or Global Positioning System (GPS) devices over the entire road network. But this technology has raised many concerns about drivers' privacy and it also requires devices installed in all vehicles.

The type of data that can be collected also varies between these methods. Whereas most methods could produce vehicle count and speed, not all of them could classify vehicle type or be employed for multiple lane detection. Table 1-1 shows traffic output data and communication bandwidth of typical technologies. Their equipment costs and

life time can be found in Table 1-2. Those data gives us a glimpse of how to choose appropriate technology to collect desired traffic data. As it is seen from Table 1-1, regarding the most variety of traffic data that could be collected, video image processor is the best approach. This is a very important advantage to traffic data collection for some complex traffic scenes such as a roundabout or an intersection for which there are more traffic data specifications. For example, traffic performance measurements for highway target mostly vehicle volume, vehicle speed and lane use, but for roundabouts or intersections, other measurements such as origin-destination pairs, waiting time and gap size are needed as well.

Sensor technology	Count	Presence	Speed	Output data	Classification	Multiple lane, multiple detection zone data	Communication bandwidth
Inductive loop	✓	✓	✓[b]	✓	✓[c]		Low to moderate
Magnetometer (two axis fluxgate)	✓	✓	✓[b]	✓			Low
Magnetic induction coil	✓	✓[d]	✓[b]	✓			Low
Microwave radar	✓	✓[e]	✓	✓[e]	✓[e]	✓[e]	Moderate
Active infrared	✓	✓	✓[f]	✓	✓	✓	Low to moderate
Passive infrared	✓	✓	✓[f]	✓			Low to moderate
Ultrasonic	✓	✓		✓			Low
Acoustic array	✓	✓	✓	✓		✓[g]	Low to moderate
Video image processor	✓	✓	✓	✓	✓	✓	Low to high[h]

Table 1- 1. Traffic output data, communications bandwidth of available sensors [1]

Unit Cost Element	Lifetime (years)	Capital Cost ($1000)	Cost Date	O&M Cost ($1000)	Cost Date
Inductive Loop Surveillance on Corridor	5	3-8	2001	0.4-0.6	2005
Inductive Loop Surveillance at Intersection	5	8.6-15.3	2005	0.9-1.4	2005
Machine Vision Sensor on Corridor	10	21.7-29	2003	0.2-0.4	2003
Machine Vision Sensor at Intersection	10	16-25.5	2005	0.2-1	2005
Passive Acoustic Sensor on Corridor		3.7-8	2002	0.2-0.4	1998
Passive Acoustic Sensor at Intersection		5-15	2001	0.2-0.4	2002
Remote Traffic Microwave Sensor on Corridor	10	9-13	2005	0.1-0.58	2005
Remote Traffic Microwave Sensor at Intersection	10	18	2001	0.1	2001
Infrared Sensor Active		6-7.5	2000		
Infrared Sensor Passive		0.7-12	2002		
CCTV Video Camera	10	9-19	2005	1-2.3	2004
CCTV Video Camera Tower	20	4-12	2005		

Table 1- 2. Equipment cost of some detectors [2].

1.2) Video-based system

Video-based systems have been applied for traffic data collection since 1970s. Figure 1-1 shows an Autoscope camera by Image Sensing System Inc., a video-based system which was initiated at the University of Minnesota in 1984 [3] and has been installed in more than 55 countries [4]. However, the main application of the Autoscope video system is vehicle presence detection or monitoring while producing limited performance measurements such as vehicle count. Also, the system uses running average based background estimation for segmentation. Therefore, there is no explicit global threshold for extracting vehicles. Instead, the empirical database is used to adjust the vehicle presence detection by the selection of particular algorithms or merely parameter values. Furthermore, the system was developed for highways and intersections, where the vehicle movements are simpler than that of roundabout.

In this thesis, we present the development of a video-based traffic data collection system for roundabouts and intersections. The system targets roundabouts and intersections because no mature data collection systems exist for these traffic scenes yet in contrast to highway scenes. The developed system has mainly three steps of processing. First, the camera is calibrated for the traffic scene of interest and a novel circle-based calibration algorithm is proposed for roundabouts. Second, the system tracks vehicles from the video by incorporating powerful imaging processing techniques and tracking algorithms. Finally, the resulting vehicle trajectories from vehicle tracking are analyzed to extract the interested traffic data, which includes vehicle volume, vehicle speed (including acceleration/de-acceleration behavior), travel time, rejected gaps, accepted gaps, follow-up time and lane use.

Figure 1- 1. Autoscope Solo Terra video detection system.

In following chapters, we describe in detail the developed video-based system for traffic data collection of roundabouts and intersections. In chapter two, we present a novel approach to calibrate the camera for roundabout traffic scenes. In chapter three, vehicle segmentation and tracking will be presented in detail. Algorithms for traffic data collection will be presented in chapter four. Finally, conclusions are made in chapter five.

Chapter 2: Camera calibration

In this chapter, we present an approach to estimate intrinsic and extrinsic parameters of the camera for roundabout traffic scenes. We first review previous works on camera calibration for roadway scenes based on the parallel line method, and then present our proposed method for camera calibration of roundabout scenes. The proposed method can estimate tilt angle, focal length, and camera height by matching the ellipse equation extracted from an image with the perspective-transformed equation of the corresponding real-world circle. The pan angle is not required in our method and only one image is needed for camera calibration. The method is validated with real-world roundabout traffic scenes and the calibration results are reasonably accurate compared to ground truth measurements.

2.1) Introduction

Camera calibration is to estimate the camera's perspective information from images of real-world objects and it is a well-grown field in computer vision. Due to widespread use of camera-based vision systems in Intelligent Transportation Systems (ITS), camera calibration has also become an important topic in this area. The extracted camera parameters are typically necessary for estimation of vehicle speed [5], object classification and tracking [6], etc. In [7], camera calibration is also used for resolving vehicle occlusions. To fully extract the camera's perspective information including both intrinsic and extrinsic parameters involves very complex calculations using a set of images taken from well-arranged objects such as a check board [8][9]. However, in ITS applications, objects or landmarks may not be easily arranged for camera calibration like in computer vision applications. Besides, the computation complexity and the requirement for multiple images make those methods unsuitable for applications in practical traffic scenes. Therefore, in the past years simplified methods to perform

camera calibration in ITS applications have been proposed. In the following, we briefly review some previous work and then motivate our work.

2.2) Roadway Camera Calibration

Most previous works on camera calibration have targeted roadway scenes such as highways, in which vehicle moving directions are known. For roadway scenes, the geometry setup of the camera for calibration is typically described as shown in Figure 2-1 [10]. The camera height and the lane width are known a priori for camera calibration. Some works attempt camera calibration with landmarks like points, lines, poles, or objects with known shape, while other works apply techniques that detect moving vehicles and use mean vehicle dimension.

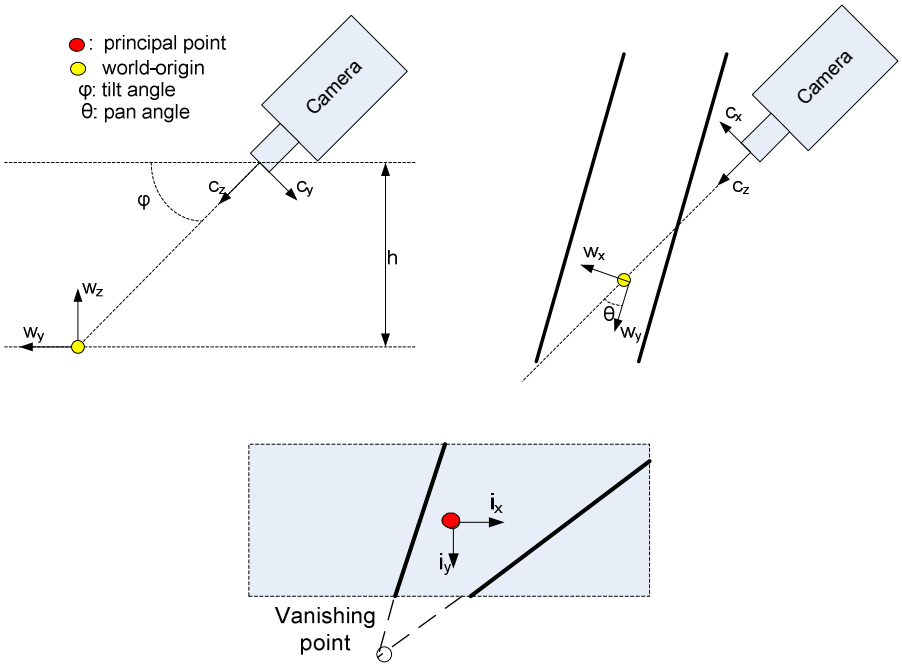


Figure 2- 1. Side view and top view of the camera setup for roadway scenes and projection of real-world traffic lanes in the image coordinate.

The landmarks used in [10-12] are traffic stripe lanes, which are supposed to be parallel in real world but generally do not appear parallel in the image frames. In some cases, these traffic lanes can be automatically extracted from the images by various image processing algorithms such as the Hough-transform technique [25]. Once the traffic lanes are extracted, the coordinate of their intersection point in the image is computed. This intersection point is called a vanishing point, which can be used to estimate intrinsic and extrinsic parameters of the camera. The vanishing point is often computed along the vehicle moving direction.

Extracting traffic lanes to estimate the vanishing point in images may require expensive computational processing. Besides, in practice, traffic lanes may not appear clearly in images depending on the specific traffic scene or road condition. To address this concern, some works propose to track vehicles to find vehicle trajectories [4][13-14], which can be regarded as the equivalents of traffic lanes, and then estimate the vanishing point from the intersection of multiple vehicle trajectories. [13] can calculate focal length, tilt, pan angle and camera height but requires mean dimensions of vehicles (width, length) to calculate scale factors on vertical and horizontal axes. These techniques can be automated but need to collect data from a large number of image frames.

In some works, two vanishing points are used for camera calibration. The second vanishing point is the intersection of lines perpendicular to the traffic lanes. There are also various techniques to estimate the second vanishing point. [5] uses bottom edges of vehicles with the assumption that the camera is oriented properly. This technique may not work well if bottom edges of vehicles are distorted or of abnormal shape in case of shadow. [14] improves accuracy by choosing edges of vehicle windows, which seem to be more homogenous in vehicles.

In [15-16], the authors suggest a simplified form of camera calibration using a scale factor. This camera model with a reduced number of calibration parameters is

claimed to be adequate for making accurate mean speed estimations. But one of the critical underlying assumptions is that the vehicle's moving direction is known.

In summary, previous works on camera calibration of roadway scenes are mostly based on the parallel line method in which vanishing points are estimated. However, these methods are not suitable to traffic scenes such as roundabouts. On one hand, parallel lines in roundabouts are seldom available, as traffic lanes in roundabouts are not parallel. On the other hand, it is rarely possible to derive parallel lines from vehicle trajectories as vehicles move in a circular fashion. Therefore, we need a different technique for camera calibration of roundabout traffic scenes.

2.3) Conic-Based Camera Calibration

Instead of using parallel lines, [17-20] propose methods to calibrate the camera using circles. A circle in the real world becomes an ellipse in the image after projection. However, in general the projected circle-center is not the ellipse-center in the image, except the case in which the optical axis of the camera points exactly to the circle center (in that case the projected circle-center becomes the center of the image). If the coordinate of the projected circle-center in the image is found, one can extract the camera's parameters. For example, [19] applies a technique that asymptotically locates the projected circle-center in the image using two concentric circles in the real world. This method requires a well-arranged scene that may only be available in computer vision applications. Another disadvantage of this method is that it requires the complete circles in the scene. [17] uses both a circle and a few lines that go through the circle-center in the real world to compute the vanishing line, and then estimates the camera's parameters. This method also requires an arranged scene and three or more images of the scene to calibrate the camera. Also, the lines that go through the circle-center are seldom available in practical roundabout traffic scenes. Another method is proposed in [18] to use two arbitrary coplanar circles for camera calibration and it requires only one image

frame. The unit normal vector and the focal length can be estimated. [20] uses only one circle and one image frame for calibration, but it requires an additional right angle.

In summary, one common disadvantage of the above methods is that they mostly require well-arranged scenes. The other drawback is that computation complexity of these methods is usually very high in spite of good accuracy. Hence, these methods are not suitable to real-world traffic scenes in ITS applications.

2.4) Proposed method for calibration at roundabout

The proposed method uses one common landmark, a circle, which is usually available and can be identified, to calibrate the camera for roundabout traffic scenes. We first extract the equation of the ellipse in the image, and then match it with the projected equation of the corresponding real-world circle to find out the camera parameters. We also simplify the algorithm to calibrate the camera with only one image frame.

In the following, we first describe geometry setup of the camera and the projection matrix between the image coordinate and the world coordinate. Then, we discuss the projection of a circle in the world coordinate. Subsequently, we present fitting of the ellipse in the image. Finally, the method to calibrate the camera is described.

2.4.1) Geometry Setup of the Camera

Cameras used for traffic surveillance are generally Pan-Tilt-Zoom cameras and mounted nearby the traffic scene [12]. Figure 2-2 shows the side view and top view of the camera setup used in our calibration method. Let (w_x, w_y, w_z) denote the unit vector of the world coordinate, (c_x, c_y, c_z) the unit vector of the camera coordinate, (i_x, i_y) the unit vector of the image coordinate, φ the tilt angle, and finally P the origin of the world coordinate that is projected into the center (or the principal point) of the image.

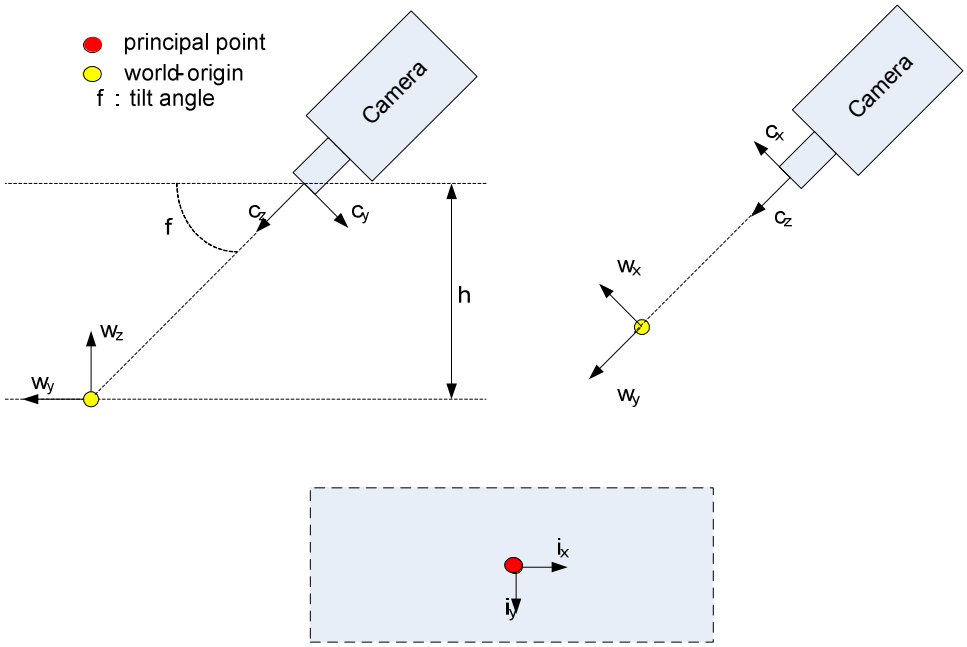


Figure 2- 2. Side view, top view of the camera setup used in our calibration method and image coordinate.

The camera's intrinsic parameters in general can be represented by the following matrix *K* [20-21]

$$K = \begin{bmatrix} f & s & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

where parameter *f* refers to the camera's focal length with unit in pixel. The point (p_x, p_y) is the principal point in the image coordinate. In most cases in traffic surveillance applications, we can assume that $(p_x, p_y) = (0,0)$ [12] and the error incurred is typically very small. The skew factor *s* is the amount by which the angle between the horizontal axis and vertical axis differs from 90 degrees. This parameter is often admitted to zero because true CCD cameras have x and y axes perpendicular [24].

The rotation matrix, representing the orientation of the camera coordinate with respect to the world coordinate, can be written as

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & sin\varphi & cos\varphi \\ 0 & -cos\varphi & sin\varphi \end{bmatrix} \quad (2)$$

With $C = \begin{bmatrix} w_{cx0} \\ w_{cy0} \\ w_{cz0} \end{bmatrix} = \begin{bmatrix} 0 \\ -h/tan\varphi \\ h \end{bmatrix}$ representing the camera's optical center in the world coordinate and the transformation vector $t = -RC$, perspective transformation from the world coordinate to the image planar coordinate can be described as

$$\begin{bmatrix} u \\ v \\ g \end{bmatrix} = K[R|t] \begin{bmatrix} w_x \\ w_y \\ w_z \\ 1 \end{bmatrix}; \quad \begin{bmatrix} i_x \\ i_y \end{bmatrix} = \frac{1}{g} \begin{bmatrix} u \\ v \end{bmatrix} \quad (3)$$

where $[u\ v\ g]$ is the homogenous vector. Substituting *K*, *R* and *t*, (3) can be rewritten as:

$$\begin{bmatrix} u \\ v \\ g \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f.sin\varphi & f.cos\varphi & 0 \\ 0 & -cos\varphi & sin\varphi & -h/sin\varphi \end{bmatrix} \begin{bmatrix} w_x \\ w_y \\ w_z \\ 1 \end{bmatrix} \quad (4)$$

In typical traffic scenes, the depth of images is large because the camera is often mounted high above the ground to have a wide view. For that reason, the height of objects (vehicles or some landmarks) can be neglected to simplify the calculation. With $w_z = 0$, from (3) and (4) we can obtain the transformation equations between the image coordinate and the world coordinate as follows

$$i_x = -\frac{f w_x}{w_y.cos\varphi + \frac{h}{sin\varphi}} \quad (5)$$

$$i_y = -\frac{f sin\varphi . w_y}{w_y . cos\varphi + \frac{h}{sin\varphi}} \quad (6)$$

Solving (w_x, w_y) from (5) and (6), the image coordinate can be mapped back to the world coordinate as follows

$$w_x = -\frac{h.i_x}{f.sin\varphi + i_y.cos\varphi} \quad (7)$$

$$w_y = -\frac{h.i_y}{\sin\varphi.(f.sin\varphi + i_y.cos\varphi)} \quad (8)$$

Equation (7) and (8) reveal three important parameters for camera calibration, which are focal length *f*, tilt angle φ and camera height *h*.

2.4.2) Projection of a Circle

General equations of a circle and an ellipse can be written as follows

$$(w_x - a)^2 + \left(w_y - b\right)^2 = R^2 \quad (9)$$

$$i_x{}^2 + 2.H.i_x.i_y + B.i_y{}^2 + 2.G.i_x + 2.F.i_y + E = 0 \quad (10)$$

The point *(a,b)* is the circle-center in the world coordinate and *R* refers to the radius of the circle. The ellipse is characterized by a set of parameters *(H, B, G, F, E)* in the equation. When a circle in the real world is projected to the image plane with perspective projection, it becomes an ellipse in the image [24-25]. As discussed, the projection depends on camera parameters such as focal length, camera height, and tilt angle. The key is that after projection, a circle originally characterized by (9) in the world coordinate should match exactly the ellipse characterized by (10) in the image coordinate. Based on this principle, substituting (7) and (8) in (9), (9) can be re-written as in (11)

$$h^2.{i_x}^2 + 2h.a.cos\varphi.i_x.i_y + 2.h.a.f.sin\varphi.i_x - f^2sin^2\varphi(R^2 - a^2 - b^2) +$$

$$[2.h.b.f - 2.f.sin\varphi.\cos\varphi.(R^2 - a^2 - b^2)].i_y +$$

$$\left[\frac{h^2}{sin^2\varphi} + 2.h.b.cot\varphi - cos^2\varphi.(R^2 - a^2 - b^2)\right].{i_y}^2 = 0 \qquad (11)$$

Then, by matching (10) and (11) we can compute the camera parameters.

2.4.3) Ellipse Fitting

In this subsection, we briefly describe how to obtain equation (10). The ellipse in the image can be mathematically characterized by a set of five parameters *(H, B, G, F, E)*. We apply ellipse fitting techniques to find these parameters. Ellipse fitting, or ellipse detection, uses two common techniques in data fitting, clustering such as Hough-based methods [26] and maximum-likelihood such as expectation-maximization or the least-square algorithm [27]. The least-square method chooses the best fit for which the sum of squared residuals has the least value. Let us denote $F(i_x,i_y)$ the distance of a point (i_x,i_y) to the ellipse in the image, which is calculated by substituting (i_x,i_y) into equation (10), the fitting can be approached by minimizing the sum of squared distances from *N* known points on the ellipse perimeter as follows:

$$Sum = \sum_{k=1}^{N} F(i_{x_k}, i_{y_k})$$

The ellipse equation has five degrees of freedom; hence *N* should be at least equal to five.

2.4.4) Overall Camera Calibration

With what is described in the above three sub-sections, we can now outline the proposed method for camera calibration of roundabouts. First, we extract the ellipse equation using ellipse fitting [27] from selected points in the image on the ellipse

perimeter. Then, we compute the camera parameters by the direct solving method (based on matching of (10) and (11)) or the optimization method.

For direct solving method, we divide equation (11) by h^2 and match it with the ellipse equation in (10), then the tilt angle and focal length can be computed as follows

$$\varphi = \sin^{-1}\left(sqrt\left(\frac{1}{B + \frac{E.H^2}{G^2} - \frac{2.H.F}{G}}\right)\right) \tag{12}$$

$$f = \frac{G}{H}\cot(\varphi) \tag{13}$$

If the radius of the circle, denoted as *R*, is known which is usually the case, then the height of the camera can be solved as

$$h = \frac{R.f.G.cos\varphi}{sqrt(H^2.f^2.(E - G^2) - (F.G - H.E)^2.cos^2\varphi)} \tag{14}$$

From (12)-(14), we can extract intrinsic and extrinsic camera parameters required for calibration. It should be noted that the circle-center *(a,b)* in the world coordinate does not appear in the above equations, and in fact they can be solved as well.

To evaluate the reliability of the direct solving method, we performed camera calibration 10 times on a test image with different points on the ellipse perimeter manually picked each time. Then, we evaluated the ratio of standard deviation over mean for each parameter. The results show that *f*, calculated from (13), varies up to ±15% throughout the trials. On the other hand, tilt angle is very stable between the trials.

To address the possible concern that focal length may be sensitive to the results of ellipse fitting due to inaccuracy incurred in manual selection of points from the ellipse perimeter, we propose a more robust method based on optimization. Assuming that the

radius of the circle R and camera height h are known, we first calculate tilt angle from (12) (as it is stable from direct solving), then use the least-square algorithm to optimize f. Denoting $F(f, p_k)$ the algebraic distance of a point $p_k(w_x, w_y)$ to the circle as follows

$$F(f, p_k) = (w_x - a)^2 + \left(w_y - b\right)^2 - R^2$$

then the error term to be minimized with N projected points is

$$Error = \sum_{k=1}^{N} F(f, p_k)$$

It was evaluated that the resulting focal length f was very stable throughout the trials based on the optimization method.

2.5) Result

We applied the proposed method to many artificial lab scenes and real-world roundabout traffic scenes and the results for a few of them are shown below.

2.5.1) Experiment 1: lab scenes

For comparison purpose, the lab scene has two parallel lines, a check board and a dish plate to represent a circle (denoted in white) as in Figure 2-3. The radius of the plate is 13.5cm. The camera is set up 70cm above ground. For the proposed calibration method, 10 to 15 points are manually selected from the ellipse perimeter in the image, and then fitted to obtain the ellipse equation which is matched against the circle equation to derive focal length, tilt angle, and camera height. The calibration results are shown in Table 2-1, together with those from the parallel line method [11] and Bouguet's calibration toolbox [28] for comparison. Note that in the parallel line method, camera height h can not be computed and is assumed to be known. It can be seen that the results

from the three methods are comparable. Compared to Bouguet's method, the proposed method has 95% accuracy for focal length and 90% accuracy for distance estimations.

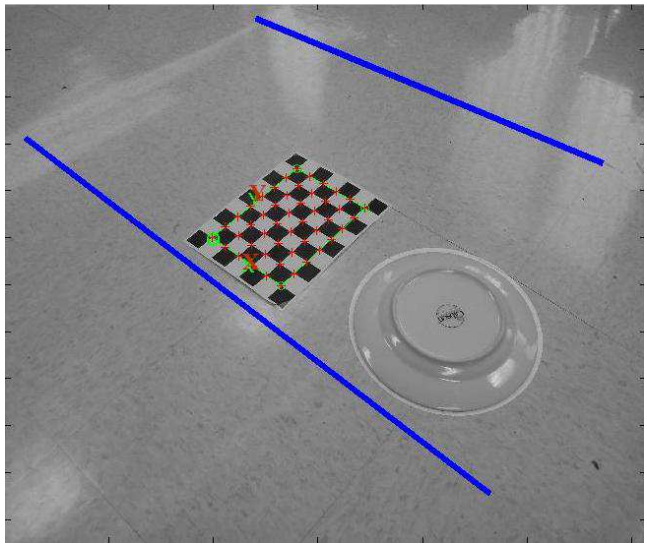


Figure 2- 3. Lab images for experiments.

	f (pixel)	*φ (rad)*	*h (cm)*	*X (cm)*	*Y (cm)*
Parallel line method	3632	0.8	-	12.3	17.3
Bouguet's method	3200	-	-	15	21
Proposed method (manual)	3359	0.8	66.8	13.5	18.8
Proposed method (automatic)	3075	0.78	72.4	15.5	21.9

Table 2- 1. A Comparison of the calibration results from three methods.

2.5.2) Experiment 2: real-world roundabouts

A real-world roundabout is shown in Figure 2-4 and the landmark of the circle is shown in red. The images were captured by a surveillance camera nearby the scene. This roundabout is located in Cottage Grove, Washington County in Minnesota. Traffic speed limit, radius of the roundabout, and camera height are respectively 25 miles per

hour, 110 feet, and 51 feet. Using the proposed method for camera calibration, the estimated tilt angle, focal length, and camera height are shown in Table 2-2. .

The estimated camera height (52.3 feet) is very close to the measured camera height (51 feet) with an error less than 2.5%. With these parameters and the known frame rate of the video, we estimate the speed of each vehicle (see Table 2-4) in this roundabout to be in the range of 10-35mph, which agrees with the speed limit well, though in this case no ground speed measurements are available for comparison. Also, we measured several marked distances in the real world and compared them with the distances estimated using the calibration results. The comparison results are shown in Table 2-3 and the accuracy is about 93%.

Roundabout	*φ (rad)*	*f (pixel)*	*h (feet)*
Figure 2-4	0.318	240	52.3
Figure 2-5(a)	0.286	118	0.646**R*
Figure 2-5(b) (manual)	0.271	213	0.523**R*
Figure 2-5(b) (automatic)	0.256	250	0.588**R*

Table 2- 2. Calibration results for the roundabouts in Figure 2-4 and Figure 2-5.

Distance	AB	CD	EF
Designed	32 ft.	14 ft.	8 ft.
Estimated	31.6 ft.	13 ft.	7.7 ft.

Table 2- 3. Estimation of distances in Figure 2-4.

Vehicle #	1	2	3	4	5	6	7	8
Speed (mph)	35	12.1	24.6	27.8	29.8	22.3	15.2	15.2

Table 2- 4. Estimation of vehicles' speed in Figure 2-4.

Figure 2- 4. An image of the roundabout.

Two more roundabout traffic scenes are shown in Figure 2-5 (a) and (b). They are both located in Minnesota and can be found in the video posted at the website in [29].

(a) (b)

Figure 2- 5. Images of two other roundabouts.

As of the time this thesis was written, we have not obtained the original design data, such as the radius of the roundabout. Therefore, we show the calibration results as a function of the radius *R* for the two roundabouts in Table 2-2. It should be noted that the

tilt angle and focal length are independent of R as shown in equation (12) and (13) whereas the camera height is.

Finally, note that in the above experiments, the points on the ellipse perimeter have been manually picked for ellipse fitting. As in previous works for traffic lane extraction, ellipse can also be automatically extracted. In the automatic mode, we used the Chord-Tangent method in [30]. First, we convert the color image to grayscale and perform Sobel edge detection. The result is a binary image with edge points. Tangents to the ellipse at a random pair of edge points are constructed, and then a line is formed that connects the intersection point of tangents and the midpoint of the line connecting the pair of edge points. All points on this line are accumulated in the parameter space because the ellipse-center lies on this line. We repeat the process with all pairs of edge points. Then, peak location proceeds and we have the candidate centers of the ellipse. After that, for each candidate center of the ellipse located at (x_i, y_i) found in the previous step, we rewrite the ellipse equation in the new (u, v) coordinate as $u^2 + 2Huv + Bv^2 + E' = 0$ with u= x-x_i and v= y-y_i. Now for each edge point we accumulate in the (H, B, E') parameter space and again search for the peak location to derive the final ellipse parameters. For the experiment on a lab scene, the calibration results in the automatic mode are shown in Table 2-1 for comparison. Also, for the real-world roundabout shown in Figure 2-5(b), the calibration results in the automatic mode are shown in Table 2-2 for comparison. It can be seen the calibration results in the manual mode and the automatic model agree well. However, for the automatic mode of camera calibration, the disadvantage is that low-resolution cameras, noise, and unclear ellipse perimeters may have negative effect on the accuracy of ellipse fitting. This actually happened to the real-world roundabout in Figure 2-4 and the other one in Figure 2-5(a). Currently, we provide both manual mode and automatic mode in our software implementation and let the user decide which option to use.

In case the ellipse is not clear in the image (as the landmark of the circle is not clear in real-world roundabout scenes), we may track the vehicles in the roundabout scene to obtain the vehicle trajectories, which can then be regarded as the equivalents of circles. This idea has been also used in previous works on camera calibration for roadway scenes when traffic lanes are not clear in the image [5][13-14].

In our algorithm, we assume that principal point is image origin (p_x=0, p_y=0). To evaluate the stability of results regarding principal point's position, we recomputed camera parameters in case of (p_x=10, p_y=10). The result show that tilt angle, focal length, and camera height vary by 1%, 5%, and 4% respectably. Those variations are relatively small and in acceptable ranges.

2.6) Summary

This chapter proposes a new and simple method for camera calibration of roundabout traffic scenes in ITS applications by using a landmark of a circle. The proposed method extracts the ellipse equation from the image and matches it with the projected equation of the corresponding circle in real-world traffic scenes. The proposed method is validated using images of both lab scenes and real-world roundabout scenes. The results are very comparable to measurements and those from the parallel line method previously used in roadway scenes. Another advantage of the proposed method is that it requires only one image for camera calibration. Finally, note that the proposed method does not require the complete ellipse (equivalently the landmark of the complete circle in real world), but only a number of points on the ellipse perimeter.

Chapter 3: Vision-based tracking system

Vision-based tracking systems have been used widely due to their advantage to provide the most comprehensive information about the vehicles compared to loop-detectors or radars. Though videos captured could be manually inspected for traffic performance measurements, it is very costly and laborious. Recent research work focuses on automatically collecting traffic performance measurements from video processing. There are relatively mature data acquisition technologies available for highways, but automatic data collection from roundabouts and intersections presents unique challenges because of more complex traffic scenes, data specifications and vehicle behavior. In this chapter, we propose a tracking-based automated traffic data collection system dedicated to roundabouts. This system could also be applied to intersections and highways with slight modifications.

The proposed system has three main steps of processing. First, the system uses an enhanced Mixture of Gaussian algorithm with shaking removal for video segmentation, which can tolerate repeated camera displacements and background movements. Next, Kalman filtering, Kernel-based tracking and overlap-based optimization are employed to track the vehicles occluded and to derive the complete vehicle trajectories. The resulting vehicle trajectory of each individual vehicle gives the position, size, shape and speed of the vehicle at each time moment. Finally, a data mining algorithm is used to automatically extract the interested traffic data from the vehicle trajectories.

In our work so far, there are totally seventy-two videos processed by our system. These videos were captured from July 2009 to October 2010 for the same roundabout and the main traffic scene of interest is an entrance of the roundabout, not the roundabout itself. An image of the traffic scene can be found in Figure 3-3 in page 37. It can be seen that there entrance way has a slope, therefore does not satisfy the ground-plane constraints. Therefore, the camera calibration algorithm introduced in chapter 2 can not

be used for this traffic scene. Instead, the camera calibration would be used later to process videos that target the whole roundabout to derive origin-destinations for example. Due to lack of camera calibration, the proposed tracking system derive vehicle states in 2-dimensional (2D) image, instead of the real 3D world. In spite of that, the proposed system is still valuable and can measure a broad variety of traffic data such as vehicle volume, travel time, rejected gaps, accepted gaps and follow-up time. The overall traffic data collection system has been implemented in software. The extracted traffic data has been compared to manual measurements and an accuracy of up to 90% has been achieved.

3.1) Introduction

Traffic data collection is a very important task in transportation applications as it provides data necessary for traffic simulation, modeling and performance evaluation. While traffic data can be manually collected, automating this task is vital in reducing cost and improving efficiency. In literature, there has been a significant amount of work on automated traffic data collection for highways. Those systems developed for highway scenes are relatively mature and successful thanks to relatively simple vehicle behavior and simple traffic data specifications. Some representative work can be found in references [31-36]. We will not go into detail for the developed systems for highways. Instead, we focus on systems that are developed for roundabouts, as these traffic scenes are more challenging due to complex scene characteristics, data specifications and vehicle behavior. For example, camera calibration techniques have not been well developed for roundabout traffic scenes while mature for highway scenes. Also, existing traffic data collection for highways target mostly vehicle volume, vehicle speed and lane use, but for roundabouts other measurements such as origin-destination pairs, waiting time and gap size are needed as well. Besides, vehicle behavior at roundabouts tends to be more complicated, as it involves frequent acceleration/de-acceleration, waiting and turning.

A variety of methods have been proposed to use sensors, loop inductors, radars or cameras for traffic data collection. Recently, in [37] an approach using wireless sensor networks is proposed for traffic data estimation for intersections. For each way of the intersection, a sensor is placed in each lane. When a vehicle drives through the sensor, its timing is simply recorded in the sensor's log. Then, all the logs of the sensors are analyzed offline to estimate the vehicle volumes of each turning direction. While the detection part could be accurate using sensors, however, a number of issues exist with this approach. First, there is ambiguity in the process of analyzing the logs. As shown in [37], depending on the specific timings of the vehicles, some vehicle trajectories could not be correctly classified. Second, the major limitation of the system is that it can only estimate traffic data like vehicle turning volumes and waiting time. To estimate the vehicle speed, more sensors are needed in each lane. However, even in that case, it is not possible to track the acceleration/deceleration behavior of the vehicle. Other traffic data, such as gap size and lane use, cannot be handled either. A very similar project and developed system based on detection using sensors is also reported in [38]. Another commercial system, Miovision [39], is available today for traffic data collection for intersections; however it is restricted to vehicle count. It should be noted that the Miovision system uses cameras instead of sensors.

While various sensor technologies are available, visual information provided by cameras can potentially provide comprehensive and accurate traffic data with non-intrusiveness, easier management, lower cost and higher efficiency. In this chapter, we present a tracking-based data collection system that can address the above issues with the existing data acquisition technologies.

3.2) System overview

The proposed traffic data collection system is implemented as software that runs on a regular PC. It incorporates powerful image processing algorithms to allow accurate

traffic data collection. The data collection system has mainly three processing steps, segmentation to identify vehicles, tracking vehicles to derive the trajectory of each vehicle, and the final step of data-mining to extract traffic data.

Vehicle segmentation is the first processing step in the data collection system. There are many existing methods for vehicle segmentation, such as background subtraction, running average, texture-based method and Eigen-background method [41-47]. However, one main disadvantage of these methods is that they do not cope well with camera shaking and background movements, which happen quite often in practice. Another method to build background models is to model each pixel with a Gaussian distribution. To cope with repeated camera shaking, the proposed system applies a mixture of Gaussian distributions for each pixel [48-49]. To further remove false detections from camera shaking, a shaking-removal algorithm is applied to refine the segmented objects.

The resulting outputs from vehicle segmentation are binary objects that are potentially vehicles. After segmentation, tracking is the next step to detect and track vehicles in each time frame. The goal is to obtain the complete vehicle trajectory, as it gives comprehensive information about the state of the vehicle at each time moment, which is more accurate for traffic data collection than without trajectories. A number of methods exist for tracking. Some popular approaches are region-based tracking [31], contour-based tracking [50], model-based tracking [51], and feature-based tracking [52]. Our approach is region-based tracking with combined Kalman filtering, Kernel-based tracking and overlap-based optimization techniques for handling vehicle occlusions.

Finally, in the last step a data mining algorithm is applied to extract all interested traffic data from the vehicle trajectories. This step does not incur any accuracy loss on traffic data given vehicle trajectories. Such a traffic data collection system relieves traffic

engineers from the laborious job of manual data collection as in [22-23], thus can significantly reduce cost and improve efficiency.

The next section describes the implementation of the proposed data collection system in detail, followed by results and conclusions.

3.3) Vehicle Segmentation

Segmentation is the first step for traffic surveillance. The better the result of this step is, the easier and more accurate vehicle tracking is. We first briefly review some previous methods and the introduces a segmentation method based on background modeling in which each pixel is consider as a mixture of Gaussian.

3.3.1) Existing approaches

There are many methods for background modeling. The beginning idea is calculating background based on the history of pixel values. The average background simply takes the average of N previous frame, and when a new frame comes, it will replace the oldest frame. Other similar approaches take median or mode value instead of average. Background of those approaches can be defined as

$$B_N = \text{average, median or mode } \{I_k | \, k = 0, 1 \ldots N\}$$

where I_k is the intensity at frame k. Those methods run very fast, but they also require a lot of memories: N * size (frame). To reduce memory consumption, the running average method simply adapts background based on incoming frame with a certain learning rate:

$$B_N = (1 - \alpha) * B_{N-1} + \alpha * I_N$$

where B_{N-1} is the background at frame N-1, I_N is pixel intensity value at frame N, and α a learning rate. When a new frame comes, we take α percent of its intensity and (1-α) percent of existing background as the new background. A large value of α means that the

model will update faster, typically α= 5% (0.05). An example of pseudo code for this algorithm could be found in Table 3-1.

```
%initialize
read the first frame
create a matrix with the same size as the frame to represent background
for all pixels in the frame
    background = current frame;
end for

%processing
for all frames in the video
    read new frame
    for all pixels in the frame
        if |current frame - background| > threshold
            set the pixel as foreground;
        end if
        background = (1-alpha)*background + alpha*I_N;%update
    end for
end for
```

Table 3- 1. Pseudo code for running average algorithm

There is also a combined method like in [58], choosing between running mode and running average base on a scoreboard algorithm. The mechanism to update background can be modified from blind update to selective update:

$$B_N = (1 - \alpha) * B_{N-1} + \alpha * I_N * M$$

with M=1 if the pixel is classified as background, and 0 if foreground.

[47] uses another advanced approach, eigenspace, to form the background. M eigenvectors corresponding to M largest eigenvalues are kept when applying Principle Component Analysis (PCA) on a sequence of N images. Background pixels, which appear frequently in frames, will significantly contribute to this model while moving objects do not. [59] is an modified version of eigenbackground modeling, which run faster because the decomposition step is eliminated in updating procedure.

Texture property of the image also could be used for segmentation. [60] uses autocorrelation difference between two image blocks to compare their similarities. This method also requires an empirical threshold which is not explicit and varies through videos. Moreover, the computational workload for autocorrelation is high.

The above-discussed methods have some common disadvantages. They do not provide an explicit way to choose the threshold for segmentation. Also, they do not cope well with camera shaking (repeated or occasional shaking due to wind) and background movements (such as trees, grass).

3.3.2) Mixture of Gaussian

In our implementation, the variation of each pixel across time is modeled by a mixture of 5 Gaussian distributions (MoG) and each distribution has its own mean, standard deviation, and weight [48-49]. The MoG actually models both foreground and background. The first B (B<=5) distributions are chosen as the background model. A threshold T is defined to represent the portion of the data that should be accounted for the background model:

$$Background = min_B\left(\sum_{k=1}^{B} w_k > T\right)$$

where w_k is the weight of distribution k (k=1,2,3,4,5). We normalize the weights such that:

$$\sum_{k=1}^{5} w_k = 1$$

For a new image frame, each incoming pixel will be subject to a matching test to see whether it belongs to any existing distribution of that pixel or not. In our project, we

apply confidence intervals for matching. We choose the confidence interval to be 98%, so an incoming pixel *X* will belong to the distribution D(μ,σ) if:

$$p = \frac{|X - \mu|}{\sigma} \leq 2.5$$

Background model update is an important step to keep the background models up-to-date with environment changes such as illumination and scene changes. If none of 5 Gaussian distributions matches the incoming pixel, the update will simply replace the distribution with the lowest weight by a new distribution with the same weight, a large standard deviation and the mean equal to the incoming pixel. If there is one distribution matching the pixel, its weight, mean and standard deviation will be updated as follows [48]:

$$w_k = w_{k-1} + \alpha * (1 - w_{k-1})$$

$$\mu_t = (1 - \alpha)\mu_{t-1} + \alpha * X_t$$

$$\sigma_t^2 = (1 - \alpha)\sigma_{t-1}^2 + \alpha * (X_t - \mu_t)^2$$

whereas the mean and standard deviation for other distributions are kept the same except their weights which are reduced as follows:

$$w_k = w_{k-1} + \alpha * w_{k-1}$$

where α is a learning rate. In case there are more than one distributions that match the pixel, the distribution which has smallest *p* will be chosen for update.

MoG can model fast illumination changes in the scene. Another advantage of MoG is its robustness against repeated camera shaking, which is one very practical problem that affects tracking accuracy. As the obtained images are not stable, large regions of noisy strips are segmented from the image, which severely affects the accuracy of object extraction. MoG can fight with repeated camera shaking very well, as the pixel

changes caused by repeated camera shaking are recorded in the background distributions and are therefore not identified as false segmentation regions any more.

On the other hand, when the camera shakes to specific positions which have not yet been observed before, many background pixels may get pixel values not modeled before and be falsely classified as foreground while they are actually from nearby background. Those false detections could also happen if the camera shakes only occasionally. In that case, even the pixel values are modeled in some distributions before, but their rare observations make their weights too small to be considered in background models. Therefore, to suppress these false detections and further refine the detection results, we employ a shaking-removal step. This is based on the observation that the false detection regions will have a high probability to be a part of the background distributions at its original locations. Hence, we can decide whether a detected region is caused by a real foreground object or the background by considering the background distributions in a small neighborhood of the detection region. If the detection pixel matches with the background distributions of pixels in the neighborhood, it is highly possible that this detection is false caused by camera shaking. When considering color images with the RGB (Red, Green, Blue) space, this technique proves to be very effective as the probability of mismatching is very small [53]. In our implementation, we choose a square 5x5 window for shaking removal.

The combined MoG modeling of the background and the shaking-removal algorithm turn out to be very valuable as video cameras are subject to shakings in practice, which would significantly affect segmentation quality if not treated. The proposed method is very general and does not involve manual intervention or rely on any prior scene knowledge. The pseudo code for proposed method using MoG could be found in Table 3-2.

```
%Initialize
read the first frame from the video to get the size of images
```

```
create a matrix M with the same size as the frame
for all elements in M
    for all distributions
        mean=initial mean;
        variance=initial variance;
    end for
end for
%Processing
for all frames of the video
    %segmenting
    for all pixel in the frame
        for all distributions
            t=sum((mean-newcoming_pixel)^2-(2.5*var)^2);%matching
            if t<0
                the new pixel matches one or more distributions;
            end if
        end for
        if there is no match
            the pixel is recognize as foreground;
        end if
        if there is one or more matches
            find the match that has smallest t value;
            check whether that match belongs to the background model;
            if no
                set pixel as foreground;
            end if
        end if
    end for
    %shaking remove
    for all pixel in the frame
        if foreground
            for neighbor pixels
                take the distribution that has largest weight;
                t=sum((mean-newcoming_pixel)^2-(2.5*var)^2);%matching
                if t<0
                    match found, reset pixel as background;
                end if
            end for
        end if
    end for
    %tracking
    tracking %see more detail in tracking pseudo code
    %updating
    for all pixel in the frame
        if not belong to a vehicle
            update mean, variance, weight;
        end if
    end for
end for
```

Table 3- 2. Pseudo code for MoG segmentation

3.4) Vehicle Tracking

3.4.1) Object and vehicle model

After segmentation, results are binary objects in the image. Objects are then extracted by the connected component algorithm [31], and classified as vehicles if their sizes are large enough. To well represent objects, each object is characterized by both a rectangular box with width and length to bind the object, and a contour with the detailed shape of the object. An object with above descriptions would be associated with one or more vehicles. A vehicle is detailed by states, predicted states, Kalman parameters, a start frame, the frame when the vehicle is first detected, and an end frame, the last frame when vehicle still could be detected. A state vector, denoted as [x y v_x v_y] is associated with each claimed vehicle, where (x,y) gives the center position of the vehicle and (v_x v_y) the velocity along the image coordinates (x,y). Figure 3-1 shows how an object and vehicle will be represented in our system.

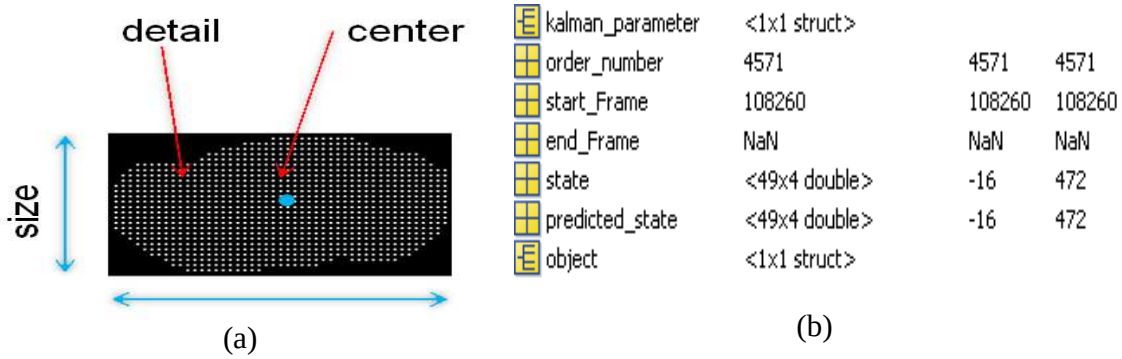


Figure 3- 1. (a) Object's description (b) vehicle's hierarchy.

To relate objects to vehicles, we compare the areas of overlap between the objects and vehicles in the image frame. Vehicles' speed or camera's frame-rate affects the areas of overlap. For example, a fast-moving vehicle or low frame-rate camera could cause

segmented objects in the current and previous frame of the same vehicle to have no overlap at all. And then the system will create new vehicle description for the current segmented object while report the previous vehicle missing. To address this problem, we use Kalman filter [54] to estimate the state of vehicles in the next frame based on their current states and Kalman parameters, and then compare the areas of overlap between the segmented objects and predicted vehicles in the current image frame.

3.4.2) Kalman filter

The Kalman filter is a set of mathematical equations that provides a recursive solution of estimations of past, present, and future states. We use the Kalman filter to predict the state of a vehicle in the next time frame and save it in predicted state vector. The state vector is then updated and tracking is repeated in the next image frame. The state vectors at all times are recorded to derive the complete vehicle trajectory since its detection until exit.

Due to prediction errors, objects and predicted vehicles could slightly overlap even though they are just close to each other. In this case, one object would be associated with multiple vehicles or vice verse. To accurately track vehicles, we will break those associations by considering the percent of overlap. If an object and a vehicle overlap more than a certain threshold, currently is set at 80%, this association is recognized as a "strong match". Otherwise, they have a "loose match". The priority is given to strong matches. If an object and a vehicle has a strong match, all loose matches associated with this object and this vehicle will be broken. Then, we try to break all multiple-object to multiple-vehicle associations by breaking the association that has the lowest overlap percentage. In this way, object-to-vehicle associations are either one-to-one, multiple-to-one or one-to-multiple, which simplify the tracking process.

Moreover, when vehicles have a long stop, the system will gradually model the vehicle itself in the background. This leads to the case that many scattered segmented

objects are associated with one vehicle. A simple solution is grouping all scattered objects to create a new object associated with that vehicle. A pseudo code for overall tracking system can be found in Table 3-3.

```
%Initialize
create an empty list of vehicle and object

for all frame in the video
    segment foreground using MoG %see more details in MoG pseudo code
    apply median filter, holes filter;
    label and group connected components to objects;
    %check overlap
    for all objects
    for all vehicles
        check overlap between objects and vehicles;
        if percent of overlap > 0.8
            set a strong match and a loose match;
        else set only a loose match
        end if
    end for
    end for
    %break some associations which have multiple matches
    for all objects
    for all vehicles
        if there is a strong match
            break all other matches for the current vehicle and object;
        else
            if there are multiple matches
                if multiple objects match with one vehicle K
                    for all those objects
                        if there is another match with other vehicles
                            break the match with vehicle K;
                        end if
                    end for
                end if
                if multiple objects match with multiple vehicles
                    break the match with the least overlap percentage;
                end if
            end if
        end if
    end for
    end for
    %update vehicles' state
    for all objects
    for all vehicles
        if the match is one-one
            update the vehicle with the object;
        end if
```

```
        if multiple objects match with one vehicle k
            group all objects together to create a new object;
            update the vehicle with the new object created;
        end if
        if multiple vehicles match with one object
            apply kernel-based tracking to locate each vehicle;
            create a new object for vehicle;
            update vehicles with corresponding new objects;
        end if
    end for
    end for
    %update vehicles/objects that have no associations
    for all objects
    for all vehicles
        if there is no loose match
            compute potential match;
            if potential match > threshold
                    update vehicles with corresponding objects;
            else
                    update vehicle as missing;
                    create new vehicles for objects;
            end if
        end if
    end for
    end for
    create a current vehicle list
;
end for
```

Table 3- 3. Pseudo code for tracking system

3.4.3) Kernel-based tracking in joint feature-spatial spaces

In a complex or crowded traffic scene, vehicles can partially or fully occlude each other. The occlusion makes it difficult to track the individual vehicle. Kalman filtering could help when vehicles are occluded for a short time, but if vehicles are occluded for a long time and have dramatic velocity changes while occluded, then errors could happen. To deal with this problem, we apply Kernel-based tracking in joint feature-spatial spaces [55-56] and overlap-based optimization when occlusion is detected. Given sample points $\{x_i, u_i\}_{i=1}^{N}$centered at $\hat{x}$ in the model image, and $\{y_j, v_j\}_{j=1}^{N}$ centered at $\hat{y}_0$ in the current target image, the kernel-based tracking with the Gaussian kernel is [56]:

$$\hat{y} = \frac{\sum y_i f(y_i)}{\sum f(y_i)}$$

where

$$f(y_i) = \sum e^{-(x_i - \hat{x})^2/\theta^2} e^{-(u_i - v_j)^2/h^2} e^{-(y_j - \hat{y}_0)^2/\theta^2}$$

where σ and h are the bandwidths in the spatial and feature spaces.

The model of the vehicle is taken before occlusion. The initial center to be applied in Kernel-based tracking is the predicted position of the vehicle provided by Kalman filtering.

To further improve tracking accuracy in case of occlusion, we also employ overlap-based optimization. The purpose of this optimization is to maximize covering the area of the objects by occluding vehicles. For example, considering a simple case of two occluding vehicles A and B associated with an object C, we perform the following operations:

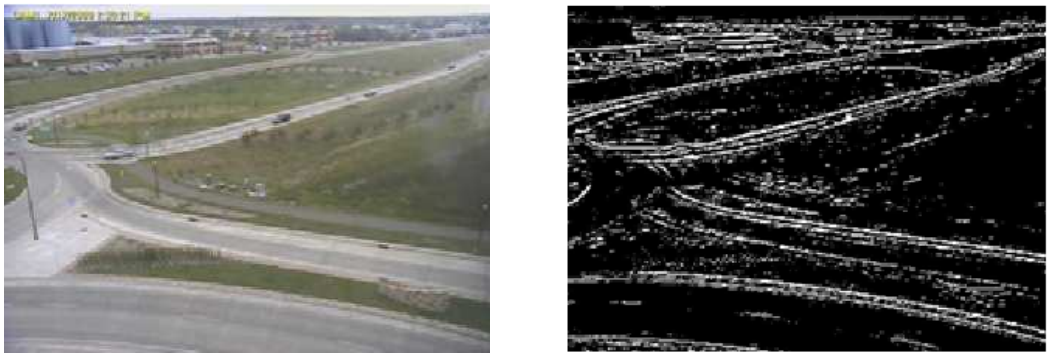
$$F = (A \cap C) \cup (B \cap C) \; and \; D = F \oplus C$$

where D represents the area of the object that is not covered by the vehicles. We compute the center of D, and then iteratively move the vehicles toward that center to cover the overall area of the object as much as possible until no improvement is achieved. Finally, the pseudo code for the overall tracking system can be found in Table 3-3.

3.5) Result

We first show the results from vehicle segmentation. Figure 3-2 (a) shows the image frame that has encountered significant camera shaking with respect to the previous image frame, and Figure 3-2 (b) shows segmentation results in traditional background subtraction methods, and finally Figure 3-2 (c) gives segmentation results with the

proposed method. It is noted the proposed mixture-of-Gaussian background modeling and shaking removal algorithm are robust to false detections from camera shaking in comparison to other methods.



(a) (b)

(c)

(a): original image

(b): segmentation result using traditional subtraction

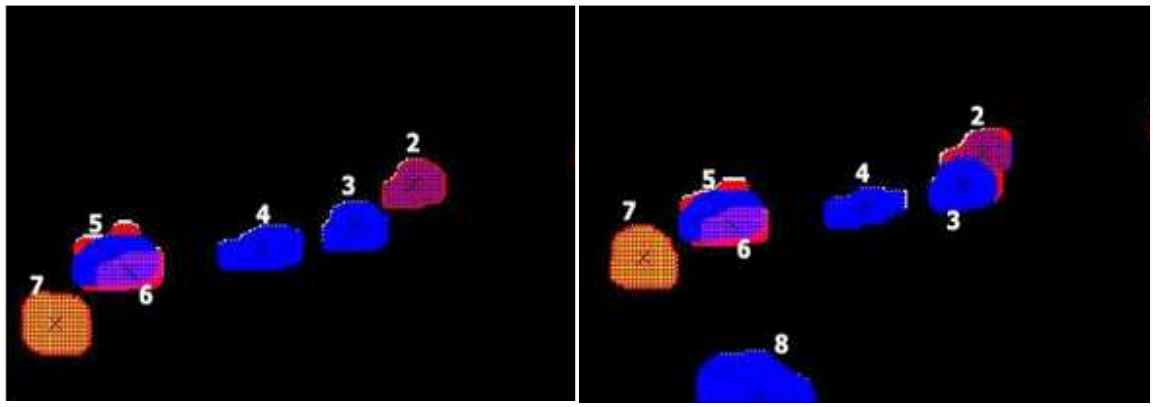
(c): segmentation result using the proposed method

Figure 3- 2. Segmentation results in case of camera shaking.

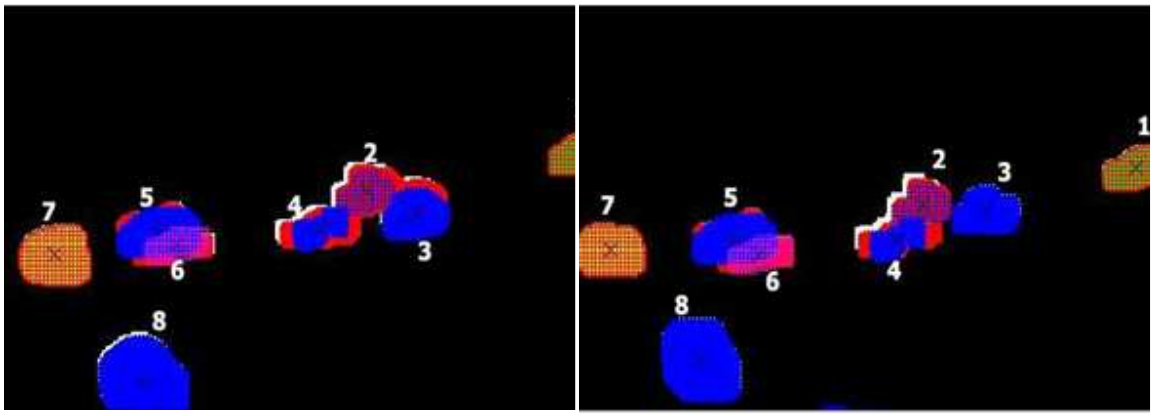
Figure 3-3 shows the overlay of some sample vehicle trajectories for a 2-hour video (4-6pm) collected on July 17th 2009 for a roundabout entrance. All tracked vehicle trajectories have been checked for correctness and those that do not have complete vehicle trajectories from ramp entrance to exit are excluded.

Figure 3- 3. Overlay of vehicle trajectories (one line represents one vehicle trajectory).

The Kernel-based tracking in joint feature-spatial spaces and overlap-based optimization are applied when two or more vehicles encounter occlusions. Figure 3-4 gives an example which shows how the system identifies the positions of individual vehicles involved in occlusions, when one vehicle (#2) is passing the other two (#3 and #4) leading to a merging of three of them. Also notice the two waiting vehicles (#5 and #6) with significant occlusions on the left-hand side of Figure 3-4, due to incoming vehicles (#7 and #8) from the other ramp entrance that has right-of-way. Note that in the figure, objects are represented in red while vehicles in other various colors.



Frame 250 Frame 255



Frame 260 Frame 265

Figure 3- 4. Tracking with vehicle occlusions.

Chapter 4: Data collection

The output of the proposed tracking system is a list of vehicles with their details such as the start and end frame, and their position at each specific frame while they're still in the scene. Figure 4-1 shows the hierarchy of the output from all above steps.

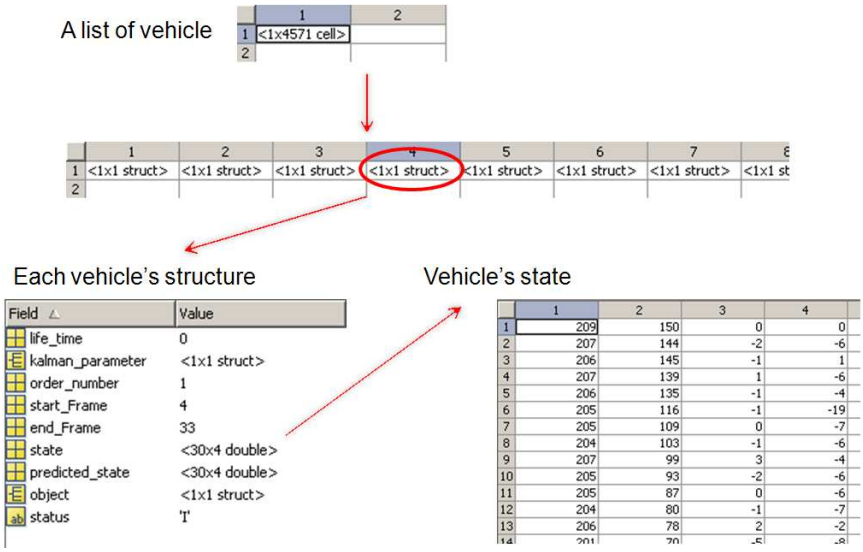


Figure 4- 1. The hierarchy of the result from tracking.

To provide meaningful data, all information will be processed in data collection step. This step is separated from the tracking system discussed in chapter 3.

4.1) Vehicle count and travel time

In traffic data collection, the number of vehicles is a basic data to acquire. All typical traffic surveillance system shown in Table 1-1 could obtain this data type. In our system, every new vehicle detected will be given an ID (Identification Number). From the number of IDs given, we know the number of vehicle in the scene.

On the contrary, travel time is not available in all surveillance systems. Many of them use vehicles' average speed to indirectly compute the travel time. In the proposed

system, travel time could be measured directly from trajectory. For example, as in Figure 4-2, we specify two lines (line1 and line2) for the roundabout entrance and the time vehicles travel between those lines are derived as travel time. Those customized lines ensure that the travel time is consistently measured for all vehicles. Currently, the user needs to select a number of image points to define the set of lines for data collection purpose. Future work will eliminate this requirement of manual input.

Figure 4- 2. Travel time definition

4.2) Accepted, rejected, and follow-up gaps

We collect accepted and rejected gaps by using manually added lines as in Figure 4-3. We consider a vehicle from the roundabout entrance has entered the roundabout for the moment it passes line 2. If this happens while there are ther vehicles in the other roundabout entrance or the roundabout itself, we will group them as accepted-gap couple.

And the time from that moment to when the other vehicle in the couple passes line 4 will be considered the accepted gap.

Figure 4- 3. Accepted and rejected gaps collection

To collect rejected gaps, we consider vehicles from the roundabout entrance in waiting mode when they cross line 3 but have not entered roundabout yet, which means that they have not passed line 2. If there are vehicles in the other entrance or in the roundabout at this moment, we group them as a rejected-gap couple. And the time from when the vehicle crosses line 3 to when the other vehicle in the couple passes line 4 is defined as the rejected gap.

The follow-up gap is more complex to define. We consider two consecutive vehicles from the roundabout entrance to form a follow-up gap if they freely enter the roundabout. This means there should not be vehicles from the other entrance or the

roundabout itself during the time when two consecutive vehicles cross line 2. Figure 4-4 shows how we define the follow-up gap. In practice, the follow-up gap in the case of roundabouts is more meaningful when the two consecutive vehicles enter the roundabout after long waiting and the roundabout frees up. Future work will take this into consideration.

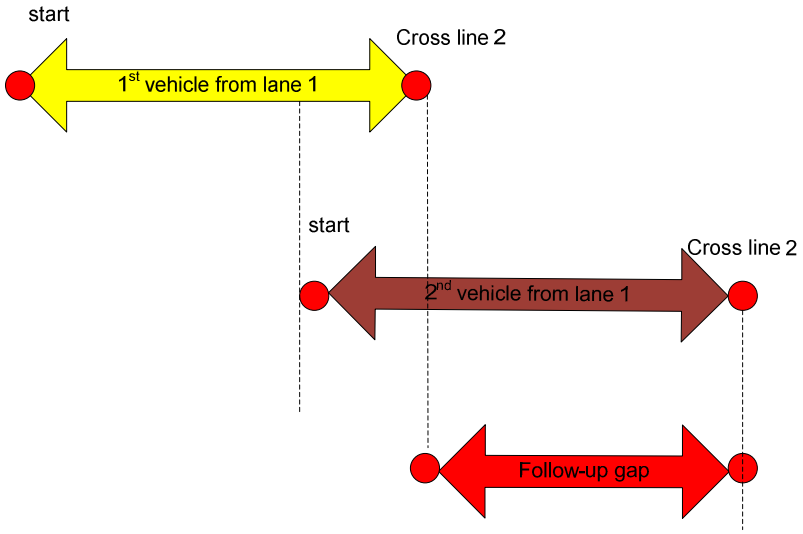


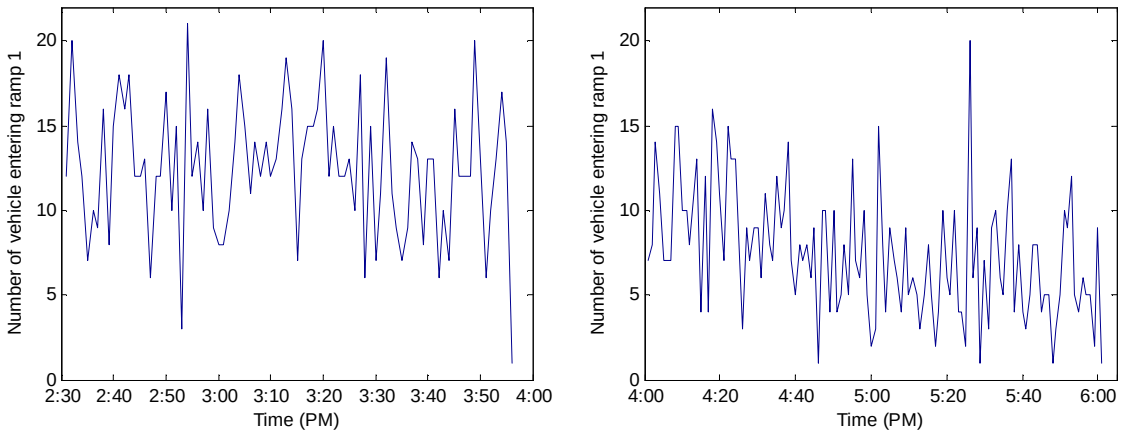
Figure 4- 4. Follow-up time definition

4.3) Result

The overall data collection system has been implemented in C and Matlab and runs on a 2.33GHz Xeon PC. We tested the system on a total of 72 videos captured from typical surveillance cameras installed by MnDoT [57] on different days (mostly in the afternoon) and the average video length is 3 hours. The processing time for a 3-hour video is currently 6 hours in the PC. All the 72 videos are for the same traffic scene, a roundabout entrance (see the above Figures 4-2 or 4-3). The roundabout is located in Cottage Grove, Washington County in Minnesota.

In this thesis, we show results from two videos collected on July 17th 2009 (one video for 2:30-3:55pm and the other 4-6pm). Figure 4-4 shows the number of vehicles that enters and exits the ramp 1 (denoted with the red line in Figure 4-2) every 60 seconds. Note that exiting the ramp 1 means the vehicle has entered the roundabout. In addition, Figure 4-5 shows the waiting time it takes for the vehicles to turn into the roundabout in ramp 1 every 60 seconds.

Figure 4-6 shows the accepted gap sizes for the two videos. The main error for gap size is incurred when there are significant occlusions. First, the vehicle did not turn into the roundabout but was taken so by the system due to poor detections sometimes, which caused false gap sizes. Second, the vehicle turned into the roundabout but was not taken so, causing some gap size misses. Considering the number of false gap sizes and gap size misses, the accuracy of gap size in terms of the number of correct entries is 95%. However, it should be noted that the accuracy of gap size in terms of actual time value is close to 100% when averaged over the number of entries.



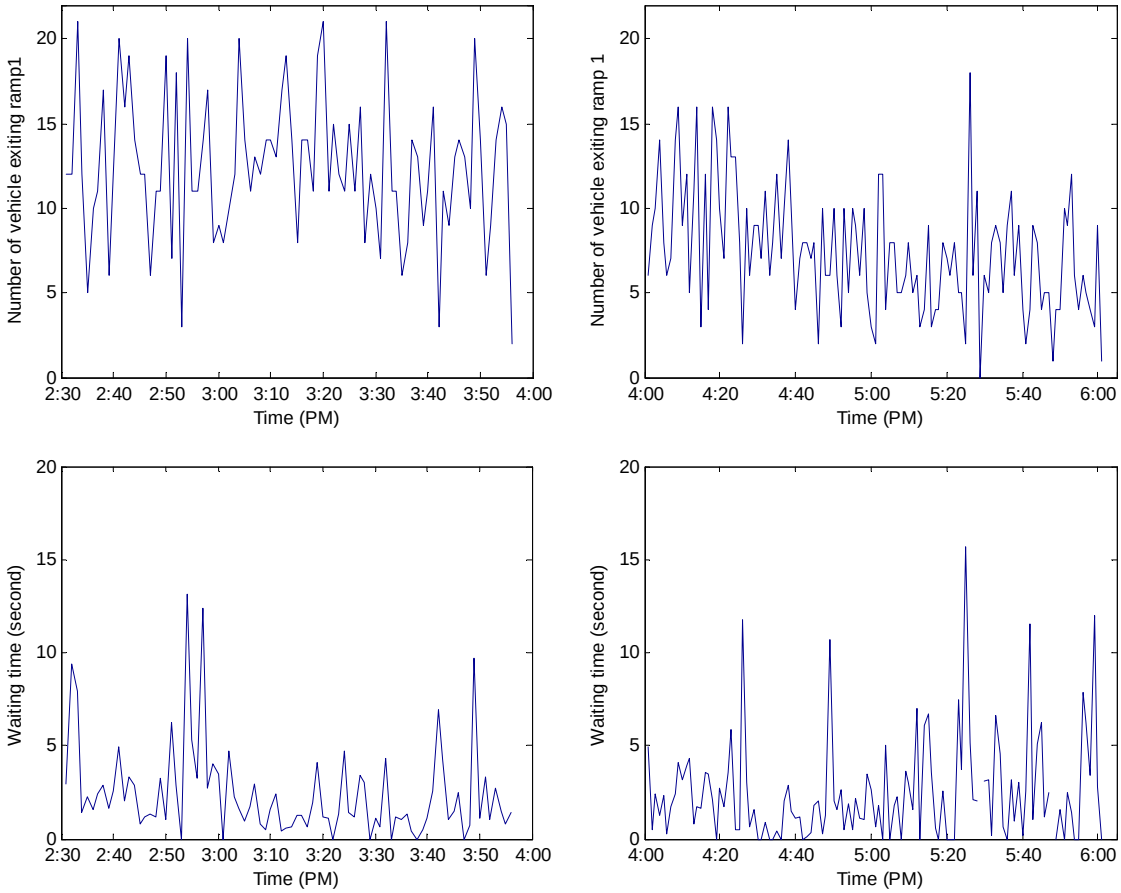


Figure 4- 5. Vehicle count and waiting time for video 1 (2:30-3:55pm) on left-hand side and video 2 (4-6pm) on right-hand side.

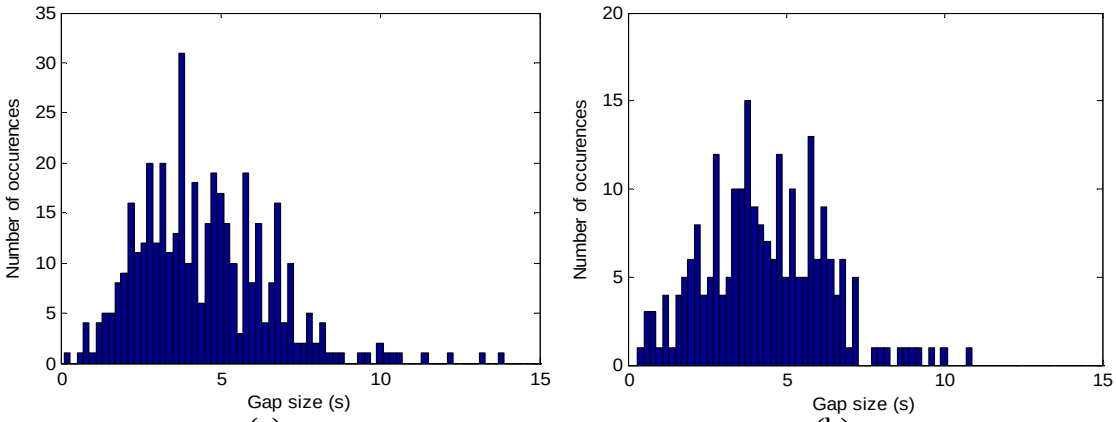
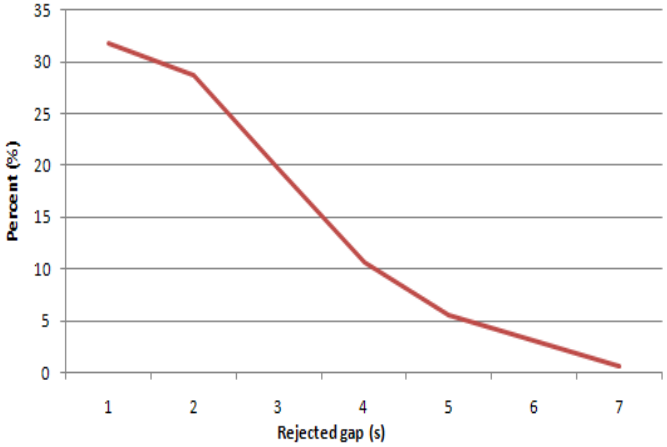


Figure 4- 6. Accepted gap sizes for (a) video 1 (2:30-3:55pm) and (b) video 2 (4-6pm).

We also collected rejected and follow-up gaps for video captured by the same camera on September 7th 2010. Data are shown in Figure 4-7 (horizontal axis refers to gap sizes in seconds and vertical axis percents of vehicle).



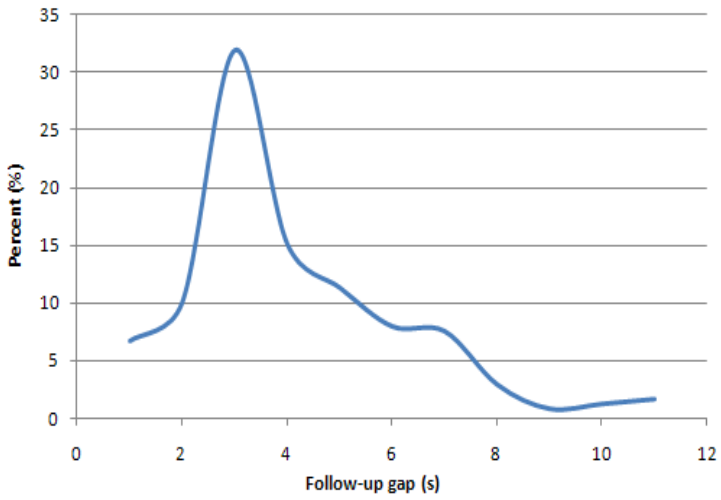


Figure 4- 7. Rejected and follow-up gap sizes for video on September 7th 2010

In summary, the proposed system is capable of producing a variety of traffic data. When compared to ground-truth measurements obtained from manual inspection of the videos, it is found that the accuracy on the traffic data is in the range of 70% to 90%, which is encouraging. As previous mentioned, the data collection step is separated from vehicle tracking itself and does not incur any accuracy loss given vehicle trajectories from the vehicle tracking. Therefore, the error is strictly from vehicle tracking, for example, poor detection and vehicle occlusion.

Chapter 5: Conclusion

In this thesis, we have developed a tracking-based traffic data collection system, which is dedicated to roundabout traffic scenes. We first propose a novel circle-based method for camera calibration of roundabout traffic scenes. Then, the system takes pre-recorded videos as inputs, applies the mixture-of-Gaussian background modeling method and a shaking-removal algorithm to segment vehicles, subsequently Kalman filtering, Kernel-based tracking and overlap-based optimization to derive complete vehicle trajectories, and finally a data mining algorithm to extract interested traffic data. Extensive experiments on many videos of a real-world roundabout traffic scene have shown that the proposed data collection system can provide traffic data, such as vehicle count, waiting time (or travel time), rejected gaps, accepted gaps and follow-up gaps, with up to 90% accuracy. Compared to previous methods using sensors or loop inductors for traffic data collection of intersections [37-38], the proposed data collection system derives the complete vehicle trajectories, giving enough details to collect all types of interested traffic data. The main benefit of the proposed system is that it can automatically collect traffic data with minimum requirements of manual inputs so that significant labor work and cost can be saved. Also, it can be used for traffic data collection for intersections, as traffic behavior at intersections is similar to those at roundabouts.

A few specific innovations achieved in this research are as follows:

First of all, to the best of our knowledge, this paper reports the first work on camera calibration of roundabouts. However, we did not use the calibration results in our tracking system because the sloping roundabout entrances in the videos. The significant slope could cause some inaccuracy in the tracking results.

Second, from algorithm point of view, the proposed system employs a shaking-removal algorithm to address false detections from camera shaking.

Third, we propose to apply Kernel-based tracking and overlap-based optimization to address vehicle occlusions, which has not been explored before.

The proposed system is similar to Miovision in concept [39]. However, as mentioned before Miovision is restricted to vehicle count for intersections so far [39], whereas the propose system can collect all types of interested traffic data for roundabouts and can be readily extended for intersections.

At the same time, we are also aware of the limitations of the developed system, which will be addressed in future work. First, the proposed system has not addressed night-time vehicle tracking, as vehicles are more difficult to be detected at night. Though there are methods proposed to use the headlights or taillights for vehicle tracking, it is not found to be very reliable in our experiments. Second, vehicle shadow is also difficult issue. Vehicle shadows could be recognized as part of the objects, which lead to tracking errors. In particular, when the Sun moves to some specific angles, the sunlight significantly blocks the view of the camera. One way to deal with vehicles' shadows is to take advantage of the color information, which is not very reliable in our experiments. Third, a single camera in the developed system may not provide adequate view coverage of the whole roundabout and this prevents deriving original-destination pairs of the roundabout. If the single camera is zoomed out to cover the whole roundabout, the vehicle may be too small to be differentiated from environmental noise during image processing. The other challenge with a single camera is vehicle occlusion, which is a notorious problem again due to limited view of one camera. Future work plans to explore the use of multiple cameras to improve tracking accuracy.

References:

[1] *Traffic Detector Handbook*, 3rd Edition, Vol. 1, pp 1-10, Oct. 2006, US DoT

[2] *Intelligent Transportation System Unit Costs Database*, Oct. 2007, US DoT.

[3] Michalopoulos, P.G.; , "Vehicle detection video through image processing: the Autoscope system," *Vehicular Technology, IEEE Transactions on* , vol.40, no.1, pp.21-29, Feb 1991.

[4] Image Sensing Systems, Inc., Autoscope, http://autoscope.com/.

[5] T. N. Schoepflin, D. J. Daily. Dynamic Camera Calibration of Roadside Traffic Management Cameras for Vehicle Speed Estimation. *IEEE Trans. on Intelligent Transportation System*, Vol. 4, No. 2, June 2003, pp. 744-759.

[6] S. Gupte, O. Masoud, R. F. K. Martin, N. P. Papanikolopoulos. Detection and Classification of Vehicles. *IEEE Transaction on Intelligent Transportation Systems*, Vol. 3, No. 1, March 2002, pp. 37-47.

[7] C. C. C Pang, W.W.L Lam, N.H.C Yung. A Novel Method for Resolving Vehicle Occlusion in a Monocular Traffic-Image Sequence. *IEEE Transaction on Intelligent Transportation System*, Vol. 5, No. 3, Sep. 2004, pp. 129-141.

[8] O. Masoud, N. P. Papanikolopoulos. Using Geometric Primitives to Calibrate Traffic Scenes. *Proc. of IEEE Conf. on Intelligent Robots and System*, Sep. 2004.

[9] Y. Liu, Y. Wu, M. Wu, Xi. Hu. Planar Vanishing Points Based Camera Calibration. *Proc. of IEEE International Conf. on Image and Graphics*, 2004.

[10] E. K. Bas, J. D. Crisman. An Easy to Install Camera Calibration for Traffic Monitoring. *Proc. of IEEE Intelligent Transportation System Conf.* , Nov. 1997.

[11] Y. Li, F. Zhu, Y. Ai, F. Wang. On Automatic and Dynamic Camera Calibration based on Traffic Visual Surveillance. *Proc. of IEEE Intelligent Vehicles Symposium*, June 2007.

[12] K. Song, J. Tai. Dynamic Calibration of Pan-Tilt-Zoom Cameras for Traffic Monitoring. *IEEE Transaction on System, Man, and Cybernetics*, Vol. 36, No. 5, Oct. 2006, pp. 1091-1103.

[13] T. H. Thi, S. Lu, J. Zhang. Self-Calibration of Traffic Surveillance Camera using Motion Tracking. *Proc. of IEEE Conf. on Intelligent Transportation System*, October 12-15 2008.

[14] N. K. Kanhere, S. T. Birchfield, W. A. Sarasua. Automatic Camera Calibration Using Pattern Detection for Vision-Based Speed Sensing. *Transportation Research Record*, No. 2086, 2008 pp. 30-39.

[15] S. Pumrin, D. J. Dailey. Dynamic Camera Calibration in Support of Intelligent Transportation Systems. *Transportation Research Record*, No. 1804, 2002, pp. 77-84.

[16] F.W. Cathey, D.J. Dailey. One-Parameter Camera Calibration for Traffic Management Cameras. *Proc. of IEEE Conf. on Intelligent Transportation System*, Oct. 2004.

[17] X. Meng, H. Li, Z. Hu. A New Easy Camera Calibration Technique Based on Circular Points. *British Machine Vision Conf.*, 2000.

[18] Q. Chen, H. Wu, T. Wada. Camera Calibration with Two Arbitrary Coplanar Circles. *Springer Berlin / Heidelberg*, Vol. 3023, 2004, pp. 521-532.

[19] G. Jiang, L. Quan. Detection of Concentric Circles for Camera Calibration. *Proc. of IEEE Conf. on Computer Vision*, 2005, pp. 71290C.1-71290C.6.

[20] H. Zhong, F. Mai, and Y. S. Hung. Camera Calibration Using Circle and Right Angles. *Proc. of IEEE Conf. on Pattern Reconigzation*, 2006.

[21] S. Mensah, S. Eshragh, A. Faghri. A Critical Gap Analysis of Modern Roundabouts. *Transportation Research Board 2010 Annual Meeting*.

[22] S. Mandavilli. Study of Operational Performance and Environmental Impacts of Modern Roundabouts in Kansas. *Transportation Research Board National Roundabout Conference,* 2005.

[23] K. Parma. *Going Around the Neighborhood – a Roundabout Case Study in Texas.* http://www.leeengineering.com/roundabouts/.

[24] R. Hartley, A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2nd ed. 2003, Ch. 6.

[25] E.R. Davies. *Machine Vision: Theory, Algorithm, Practicalities.* Morgan Kaufmann, 3rd ed. Ch. 12.

[26] V.F. Leavers. Shape Detection in Computer Vision Using the Hough Transform. *Springer Berlin / Heidelberg*, 1992.

[27] Fitzgibbon, M. Pilu, R. B. Fisher. Direct Least Square Fitting of Ellipses. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 5, May 1999, pp. 476-480.

[28] Jean-Yves Bouguet, *Camera Calibration Toolbox for Matlab* http://www.vision.caltech.edu/bouguetj/

[29] *Roundabouts in Minnesota*, http://www.dot.state.mn.us/roundabouts/

[30] H.K. Yuen, J. Illingwoth, and Kittler. Ellipse Detection using Hough Transform. *Proc. of 4th Alley Vision Conf.*, 1988, pp. 265-271.

[31] C. Smith, C. Richards, S. Brandt, N. P. Papanikolopoulos. Visual Tracking for Intelligent Vehicle Highway Systems. *IEEE Transactions on Vehicular Technology*, Vol. 45, No. 4, Nov. 1996, pp. 744-759.

[32] S. Gupte, N. P. Papanikolopoulos. Algorithms for Vehicle Classification. *Minnesota Department of Transportation,* Final report, MN/RC-2000-27.

[33] D. Dailey. CCTV Technical Report: Phase III. *Washington Department of Transportation, Technical report*, WA/RD-2006-635.2.

[34] Y. Wu, F. Lian, C. Huang, T. Chang. Traffic Monitoring and Vehicle Tracking using Roadside Camera. *Proc. of IEEE International Conference on Systems, Man and Cybernetics*, 2006.

[35] R, Ervin, C. MacAdam, J. Walker, S. Bogard, M. Hagan, A. Vayda, E. Anderson. System for Assessment of the Vehicle Motion Environment (SAVME). *Final Report, UMTRI-2000-21-1*.

[36] Next Generation of Simulation Program (NGSIM). ngsim-community.org.

[37] T. Kwon. Portable Cellular Wireless Mesh Sensor Network for Vehicle Tracking in an Intersection. *Minnesota Department of Transportation,* Final report, CTS 08-29.

[38] Abdel-Rahim, B. Johnson. An Intersection Traffic Data Collection Device Utilizing Logging Capabilities of Traffic Controllers and Current Traffic Sensors. ***National Institute for Advanced Transportation Technology,* Final report.**

[39] Automated Video Traffic Studies. *Miovision Technologies Inc.*

[40] H. Dinh, H. Tang. Camera Calibration for Roundabout Traffic Scenes. *paper submitted to TRB Annual Meeting*, 2011.

[41] S. Gupte, O. Masound, R. F. K Martin, N.P. Papanikolopoulos. Detection and Classification of Vehicles. *IEEE Transaction on Intelligent Transportation System*, Vol.3 No.1, March 2002, pp. 37-47.

[42] L. Wang, N.H.C. Yung. Extraction of Moving Objects From Their Background Based on Multiple Adaptive Thresholds and Boundary Evaluation. *IEEE Transaction on Intelligent Transportation System*, Vol. 11, No. 1, March 2010, pp. 40-51.

[43] J.W. Hsieh, S.H Y, Y.S. Chen. Automatic Traffic Surveillance System for Vehicle Tracking and Classification. *IEEE Transaction on Intelligent Transportation System*, Vol. 7, No. 2, June 2006, pp.179-186.

[44] W. W. L. Lam, N.H.C. Hung. Highly accurate texture-based vehicle segmentation method. *Optical Engineering*, Vol.43, No. 3, March 2004, pp. 591-603.

[45] M. Heikkila, M. Pietikainen. A texture-based method for modeling the background and detecting moving objects. *IEEE Transaction on Pattern Analysis and Machine Intelligent*, Vol. 28, No. 4, April 2006, pp. 657-662.

[46] H.S Lai, N. H.C Yung. A fast and Accurate Scoreboard Algorithm for Estimating Stationary Backgrounds in an Image Sequence. *Proc. IEEE Intl. Symp. Circuit Syst.*,Vol. 4, 1998, pp.241-244.

[47] N. M. Oliver, B. Rosario, A. P. Pentland. A Bayesian Computer Vision System for Modeling Human Interactions. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 8, August 2000, pp.831-843.

[48] C. Stauffer, W.E.L Grimson. Adaptive background mixture models for real-time tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, 1999, pp. 246-252.

[49] Pavlidis, V. Morellas, P. Tsiamyrtzis, S. Harp. Urban Surveillance Systems: From the Laboratory to the Commercial World. *Proc. IEEE,* 2001, pp. 1478-1496.

[50] B. Yiu, K Wong, F. Chin, R. Chung. Explicit Contour Model for Vehicle Tracking with Automatic Hypothesis Validation. *IEEE Intl. Conference on Image Processing,* 2005.

[51] T. N. Tan, G. D. Sullivan, K. D. Baker. Model-based Localization and Recognition of Road Vehicles. *International Journal of Computer Vision,* Vol. 27, No.1, 1998, pp. 5-25.

[52] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. A real-time computer vision system for measuring traffic parameters. *IEEE Conference on Computer Vision and Pattern Recognition,* 1997.

[53] Elgammal, D. Harwood, L. Davis. Non-parametric Model for Background Subtraction. *IEEE Conference on Computer Vision and Pattern Recognition,* 2004.

[54] G. Welch, G. Bishop. *An Introduction to the Kalman filter*. University of North Carolina- Chapel Hill. http://www.cs.unc.edu/~welch/kalman/kalmanIntro.html. Accessed Feb. 5, 2008.

[55] D. Comaniciu, V. Ramesh, P. Meer. Kernel-based object tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence,* Vol. 25, No. 5, May 2003, pp.564-577.

[56] C. Yang, R. Duraiswami, A. Elgammal, L. Davis. Real-time Kernel-Based Tracking in Joint Feature-Spatial Spaces. *IEEE Conference on Computer Vision and Pattern Recognition,* 2005.

[57] Minnesota Department of Transportation, www.dot.state.mn.us.

[58] Andrew H.S Lai and Nelson H.C. Yung, “A Fast and Accurate Scoreboard Algorithm for Estimating Stationary Backgrounds in an Image Sequence”, Proc. IEEE Intl. Symp. Circuit Syst. 4, 241-244 (1998).

[59] Ruonan Li, Yu Chen, and Xudong Zhang, “Fast Robust Eigen-Background Updating for Foreground Detection”, ICIP 2006.

[60] William Wai Leung Lam, Clement Chun Cheong Pang and Nelson H. C. Yung, "Highly accurate texture-based vehicle segmentation method", Opt. Eng. 43, 591 (2004);