OKLAHOMA TRANSPORTATION CENTER

ECONOMIC ENHANCEMENT THROUGH INFRASTRUCTURE STEWARDSHIP

COMMERCIAL VEHICLE ROUTE TRACKING USING VIDEO DETECTION

SAMUEL CHENG, PH.D.
ROBERT C. HUCK
PRAMODE VERMA, PH.D.
ANJAN GHOSH, PH.D.
JAMES SLUSS, JR., PH.D.

OTCREOS9.1-16-F

Oklahoma Transportation Center
2601 Liberty Parkway, Suite 110
Midwest City, Oklahoma 73110

Phone: 405.732.6580
Fax: 405.732.6586
www.oktc.org

DISCLAIMER

TECHNICAL REPORT DOCUMENTATION PAGE

1. REPORT NO.	2. GOVERNMENT ACCESSION NO.	3. RECIPIENTS CATALOG NO.
OTCREOS9.1-16-F		

4. TITLE AND SUBTITLE	5. REPORT DATE
Commercial Vehicle Route Tracking using Video Detection	October 31, 2010
	6. PERFORMING ORGANIZATION CODE

7. AUTHOR(S)	8. PERFORMING ORGANIZATION REPORT
Samuel Cheng, Robert C. Huck, Pramode Verma, Anjan Ghosh, James J. Sluss, Jr.	

9. PERFORMING ORGANIZATION NAME AND ADDRESS	10. WORK UNIT NO.
The University of Oklahoma 4502 E41st ST, #4403, Tulsa, OK 74135	11. CONTRACT OR GRANT NO. DTRT06-G-0016

12. SPONSORING AGENCY NAME AND ADDRESS		13. TYPE OF REPORT AND PERIOD COVERED
(Fiscal) Oklahoma Transportation Center 201 ARRC Stillwater, OK 74078 2	(Technical) Oklahoma Transportation Center 2601 Liberty Parkway, Suite 110 Midwest City, OK 73110	Final June 2009-October 2010
		14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

University Transportation Center

16. ABSTRACT

Interstate commercial vehicle traffic is a major factor in the life of any road surface. The ability to track these vehicles and their routes through the state can provide valuable information to planning activities. We propose a method using video cameras to capture critical information about commercial vehicles when they enter the state and store this information for later retrieval to provide tracking functions. As these vehicles continue on their routes, additional cameras will capture images that can be used for route tracking. By using these data, reports and highway utilization maps could be generated showing commercial vehicle routes and vehicle counts for state highways. Spurred by the competitive performance potential realized in face recognition via sparse representation, we treat the problem of vehicle identification with different video sources as signal reconstruction out of multiple linear regression models and use compressive sensing to solve this problem. By employing a Bayesian formalism to compute the l-1 minimization of the sparse weights, the proposed framework provides new ways to deal with three crucial issues in vehicle identification: feature extraction, online vehicle identification dataset build up, and robustness to occlusions and misalignment. For feature extraction, we use the simple down-sampled features which offer good identification performance as long as the features space is sparse enough. The theory also provides a validation scheme to decide if a newly identified vehicle has been included in the dataset. Moreover, unlike PCA or other similar algorithms, using down-sampling based features, one can easily introduce features of newly identified vehicles into the vehicle identification database without manipulating the existing data in the database. Finally, Bayesian formalism provides a measure of confidence for each sparse coefficient. We have conducted experiments to include different types of vehicles on the interstate highway to verify the efficiency and accuracy of our proposed system. The results show that the proposed framework cannot only handle the route tracking of commercial vehicles, but works well for all classes of vehicles.

17. KEY WORDS	18. DISTRIBUTION STATEMENT
Vehicle Tracking, Vehicle Identification	No restriction. This publication is available at www.oktc.org and from the National Technical Information Service.

19. SECURITY CLASSIF. (OF THIS REPORT)	20. SECURITY CLASSIF. (OF THIS PAGE)	21. NO. OF PAGES	22. PRICE
Unclassified	Unclassified	84 + covers	

ACKNOWLEDGMENTS

The authors express their appreciation to the following persons whose help and support have made the completion of this research project possible:

Shuang Wang, for the study and implementation of algorithms, the design of software system, the execution of experiments, the analysis of results, and the composition of the final report for the whole project.

Lijuan Cui, for providing constructive suggestions during the project and assisting the composition of the most part of this report.

Dianchao Liu, for his work editing parts of this report and the literature research in Appendix A.

Amin Roozgard, for his contributions to the literature research in Appendix A.

Bei Li, for his help describing the software design related concepts in Appendix B.

The authors also express their appreciation to the Oklahoma Transportation Center for funding for this project. The authors further thank the RITA University Transportation Center Program. Special thanks go to the OU-Tulsa Telecommunication Interoperability Laboratory, OU School of Electrical and Computer Engineering, the Vice Provost of OU-Tulsa, the OU Graduate College, and the OU Office of the Vice President for Research.

SI (METRIC) CONVERSION FACTORS

Approximate Conversions to SI Units

Symbol	When you know	Multiply by	To Find	Symbol
		LENGTH		
in	inches	25.40	millimeters	mm
ft	feet	0.3048	meters	m
yd	yards	0.9144	meters	m
mi	miles	1.609	kilometers	km
		AREA		
in^2	square inches	645.2	square millimeters	mm
ft^2	square feet	0.0929	square meters	m^2
yd^2	square yards	0.8361	square meters	m^2
ac	acres	0.4047	hectares	ha
mi^2	square miles	2.590	square kilometers	km^2
		VOLUME		
fl oz	fluid ounces	29.57	milliliters	mL
gal	gallons	3.785	liters	L
ft^3	cubic feet	0.0283	cubic meters	m^3
yd^3	cubic yards	0.7645	cubic meters	m^3
		MASS		
oz	ounces	28.35	grams	g
lb	pounds	0.4536	kilograms	kg
T	short tons (2000 lb)	0.907	megagrams	Mg
		TEMPERATURE (exact)		
°F	degrees Fahrenheit	(°F-32)/1.8	degrees Celsius	°C
		FORCE and PRESSURE or STRESS		
lbf	poundforce	4.448	Newtons	N
lbf/in^2	poundforce per square inch	6.895	kilopascals	kPa

Approximate Conversions from SI Units

Symbol	When you know	Multiply by	To Find	Symbol
		LENGTH		
mm	millimeters	0.0394	inches	in
m	meters	3.281	feet	ft
m	meters	1.094	yards	yd
km	kilometers	0.6214	miles	mi
		AREA		
mm^2	square millimeters	0.00155	square inches	in^2
m^2	square meters	10.764	square feet	ft^2
m^2	square meters	1.196	square yards	yd^2
ha	hectares	2.471	acres	ac
km^2	square kilometers	0.3861	square miles	mi^2
		VOLUME		
mL	milliliters	0.0338	fluid ounces	fl oz
L	liters	0.2642	gallons	gal
m^3	cubic meters	35.315	cubic feet	ft^3
m^3	cubic meters	1.308	cubic yards	yd^3
		MASS		
g	grams	0.0353	ounces	oz
kg	kilograms	2.205	pounds	lb
Mg	megagrams	1.1023	short tons (2000 lb)	T
		TEMPERATURE (exact)		
°C	degrees Celsius	9/5+32	degrees Fahrenheit	°F
		FORCE and PRESSURE or STRESS		
N	Newtons	0.2248	poundforce	lbf
kPa	kilopascals	0.1450	poundforce per square inch	lbf/in^2

COMMERCIAL VEHICLE ROUTE TRACKING USING VIDEO DETECTION

Final Report

October, 2010

Samuel Cheng

Principal Investigator

Samuel Cheng

Project Manager

Robert C. Huck, Pramode Verma, Anjan Ghosh, James J. Sluss, Jr.

Co-Principal Investigators

The University of Oklahoma

School of Electrical and Computer Engineering

4502 E 41st Street Room 4403

Tulsa, Oklahoma 74135

October 2010

TABLE OF CONTENTS

Chapter **Page**

LIST of FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

Interstate commercial vehicle traffic is a major factor in the life of any road surface. The ability to track these vehicles and their routes through the state can provide valuable information to planning activities. We propose a method using video cameras to capture critical information about commercial vehicles when they enter the state and store this information for later retrieval to provide tracking functions. As these vehicles continue on their routes, additional cameras will capture images that can be used for route tracking. By using these data, reports and highway utilization maps could be generated showing commercial vehicle routes and vehicle counts for state highways. Spurred by the competitive performance potential realized in face recognition via sparse representation [1], we treat the problem of vehicle identification with different video sources as signal reconstruction out of multiple linear regression models, and use compressive sensing to solve this problem. By employing a Bayesian formalism to compute the l^1 minimization of the sparse weights, the proposed framework provides new ways to deal with three crucial issues in vehicle identification: *feature extraction*, *online vehicle identification dataset build up*, and *robustness to occlusions and misalignment*. For feature extraction, we use the simple down-sampled features which offer good identification performance as long as the features space is sparse enough. The theory also provides a validation scheme to decide if a newly identified vehicle has been included in the dataset. Moreover, unlike PCA or other similar algorithms, using down-sampling based features, one can easily introduce features of newly identified vehicles into the vehicle identification database without manipulating the existing data in the database. Finally, Bayesian formalism provides a measure of confidence for each sparse coefficient. We have conducted experiments to include different types of vehicles on the interstate highway to verify the efficiency and accuracy of our proposed system. The results show that the proposed framework can not only handle the route tracking of commercial vehicles, but works well for all classes of vehicles.

CHAPTER 1
INTRODUCTION

States conduct traffic monitoring for many reasons, including highway planning and design, and motor vehicle enforcement. Traffic monitoring can be classified into two different types: flow monitoring and route monitoring. Flow monitoring will observe the amount traffic flow across an interested check point, whereas route monitoring will identify the route of an interested vehicle. Unlike flow monitoring, route monitoring generally needs to know the identity of the observed vehicle and is generally more difficult. This route monitoring capability will provide valuable information for freight logistics analysis, forecast modeling, and future transportation infrastructure planning.

1.1 VEHICLE IDENTIFICATION OVERVIEW

The main component of the proposed route monitoring system is an Automated Vehicle Identification (AVI) subsystem. AVI is the application of sensor technologies to automatically identify a vehicle of interest. It may be described as containing three components: the sensor, a signal processing device, and a data processing device. The sensor detects the passage or presence of a vehicle or its axles. The signal processing device typically converts the sensor output into an electrical signal. The data processing device usually converts the electrical signal to a unique signature of the vehicle to be stored in a dataset for later retrieval. A sensor is a key component of any AVI system. Depending on the installation location, sensors are categorized into two types, intrusive and non-intrusive [2]. Intrusive sensors are those that require the installation of the sensor directly onto or into the road surface, including inductive loops [3], magnetometers [4], microloop probes [5], pneumatic road tubes [6], piezoelectric cables [7], and other vehicle sensors. On the other hand, non-intrusive sensors are mounted overhead or on the side of the roadway. The video cameras, microwave radars, active and passive infrared sensors, and ultrasonic sensors fall in this category [8]. Not all aforementioned sensors can yield an accurate signature for vehicle identification. Video cameras that naturally gather a large amount of useful information potentially provide an ideal non-intrusive method for accurate vehicle identification. Even though Radio Frequency IDentification (RFID) based systems [9] are well-studied and can generally yield even higher identification accuracy, its use is restricted to a vehicle with a pre-installed tag. This greatly reduces the applicability of RFID based system for non-intrusive traffic monitoring. Moreover, thanks to the wide popularity of camera devices, the price of cameras has dropped rapidly while the technology has improved significantly in recent years in response to a strong consumer market. We envision that this trend will only continue and the cost of the proposed video based system will reduce over time. Infrastructure planning can benefit from advances in AVI technology, in particular, the video based techniques described in this report.

1.2 MAIN CONTRIBUTION

In this project, we proposed a video based vehicle identification system to track commercial vehicles and their routes through the state. The main contributions and accomplishments of our proposed method are in the following:

1. We use video cameras to capture the critical information of commercial vehicles for the purpose of vehicle tracking when they enter the state, and use additional video cameras to track the routes. Using these data, it is possible to record commercial vehicle routes and vehicle counts for state highways.
2. We treat the problem of vehicle identification from different video sources as signal reconstruction out of multiple linear regression

models, and use rising theories from an emerging signal processing area --- compressive sensing to solve this problem. By employing a Bayesian formalism to compute the l^1 minimization of the sparse weights, the proposed framework provides new ways to deal with three crucial issues in vehicle identification: *feature extraction*, *online vehicle identification dataset building*, and *robustness to occlusion and misalignment*. For feature extraction, we use the simple down-sampled features which offer good identification performance as long as the features space is sparse enough. The theory also provides a validation scheme to decide if an incoming vehicle has been already included into dataset. Moreover, by taking the advantages of down-sample based features, one can easily introduce features of newly entering vehicles into vehicle identification dataset without using any training algorithm. Finally, Bayesian formalism provides a measure of confidence of each sparse weight.

3. We conduct extensive experiments on different types of vehicles on the interstate highways to verify the efficiency and accuracy of our proposed system. The results show that the proposed framework can not only handle the route tracking of commercial vehicles, but also works well on all kind of vehicles.
4. Other sensing techniques (such as bluetooth and automatic license plate recognition) have also been studied. Some preliminary tests have been made to estimate the possible performance increase analytically if such modules are available. Moreover, a detail literature study has been employed to investigate the current ALPR technology.
5. The active server model is proposed. We have laid out the appropriate software model if the current module is to be incorporated into existing software framework (for example, the state traffic tracking system) in the future.

1.3 REPORT OUTLINE

This report is organized as follows. In Chapter 2, we explain the architecture of our proposed system. The video based vehicle identification via sparse representation and Bayesian formalism is presented in Chapter 3. Analysis and results are presented in Chapter 4. Cost information is given in Chapter 5 and in Chapter 6, we present a detail identification example. Chapter 7 we draw the concluding remarks. Finally, the appendices are given to summarize literature research for license plate detection technologies and software design concepts expected to be applied for full-scale deployment of the proposed technologies.

CHAPTER 2
SYSTEM ARCHITECTURE

In this section, we will introduce the architecture of our proposed vehicle identification system. First, the system overview is presented. Then the details of each system component are described in the following section. Finally, the process flow and development environment are explained in detail respectively.

2.1 SYSTEM OVERVIEW

Our commercial vehicle identification system is able to detect, track, and identify each commercial vehicle, and transmit vehicle information to a service center for further route tracking and other traffic monitoring tasks. The system includes three main components: the sensing stations, the service center, and the clients. The sensing stations mainly gather the vehicle information through video cameras. To further improve the identification accuracy, we propose to include an automatic bluetooth reader (ABR) and an automatic license plate reader (ALPR) as optional components (as shown in Figure 1). And we designate this system CBL, short for Camera, Bluetooth, and ALPR. The CBL can acquire video of traffic from video cameras, which will be further processed for vehicle detection, tracking, identification, and sample training; it can also capture Bluetooth device information carried on-board vehicles via any active Bluetooth device; it can capture images of vehicles, and identify the license plate numbers by character recognition in the ALPR. In addition, there are several CBL systems, and they are parallel to each other, as shown in Figure 2. This setup is used for data training and testing: one CBL can be used to gather training samples while another CBL can be used to employ the identification algorithm with the training dataset acquired from the previous CBL device, and vice versa. The service center component, the most critical part, collects the data from CBL stations, and employs our vehicle identification algorithm to achieve the identification results. The clients are terminals that query the identification results from the service center and produce reports of desired statistics and routing information.

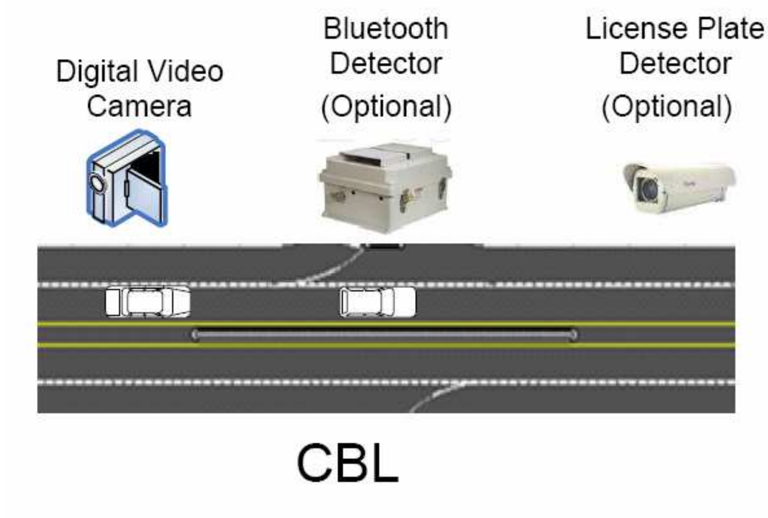


Figure 1. Proposed vehicle identification system, which include video camera, Bluetooth, and ALPR

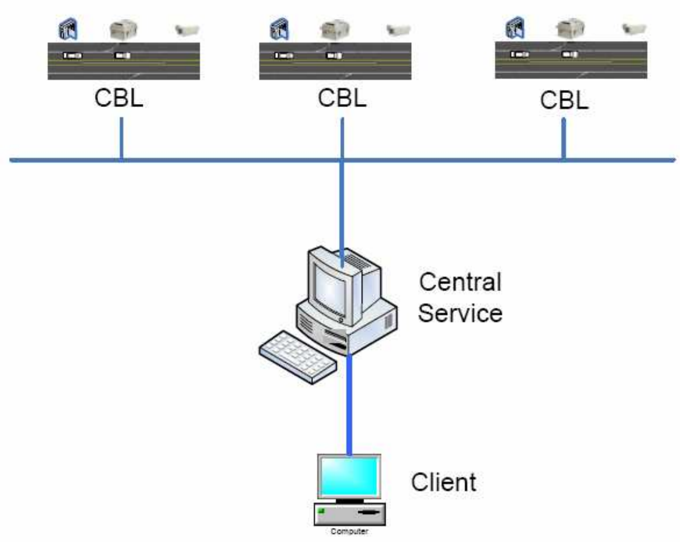


Figure 2. System Setup

2.2 DETAILED DESCRIPTION FOR SYSTEM COMPONENTS

In this section, the system components, including digital video camera, ABR, and ALPR, are described in terms of their properties.

2.2.1 Video Capture Component

As the key component of the CBL system, the video camera captures a large amount of the traffic information including environment conditions, illumination conditions, vehicle information such as color, shape, speed, and size. The cameras will be setup along the side of highway, they should be reliable and network accessible. The video cameras should be of high resolution, high speed, and real-time color web cameras. We propose to use Axis 223M network cameras, as shown in Figure 3 and Figure 4. They have many advantages: such as robust, wide range of temperature operation, day and night functionality, Mega-pixel resolution, and ease of installation. A typical frame from the video cameras is shown in Figure 5. We find this frame size captures environment and traffic information well. The web cameras can be easily accessed via the Internet as shown in Figure 6. These video cameras capture highway traffic videos at different locations, and the videos are transmitted to the service center for further processing. The data captured from each video camera can be used as a database for identification, and each record of the database can be used as the input for performing vehicle identification tasks. On the other hand, for route tracking of commercial vehicles, we will setup our systems among different highway routes. Therefore, the system requires several video cameras together.

Figure 3. Video camera I

Figure 4. Video camera II

Figure 5. An example of video frame

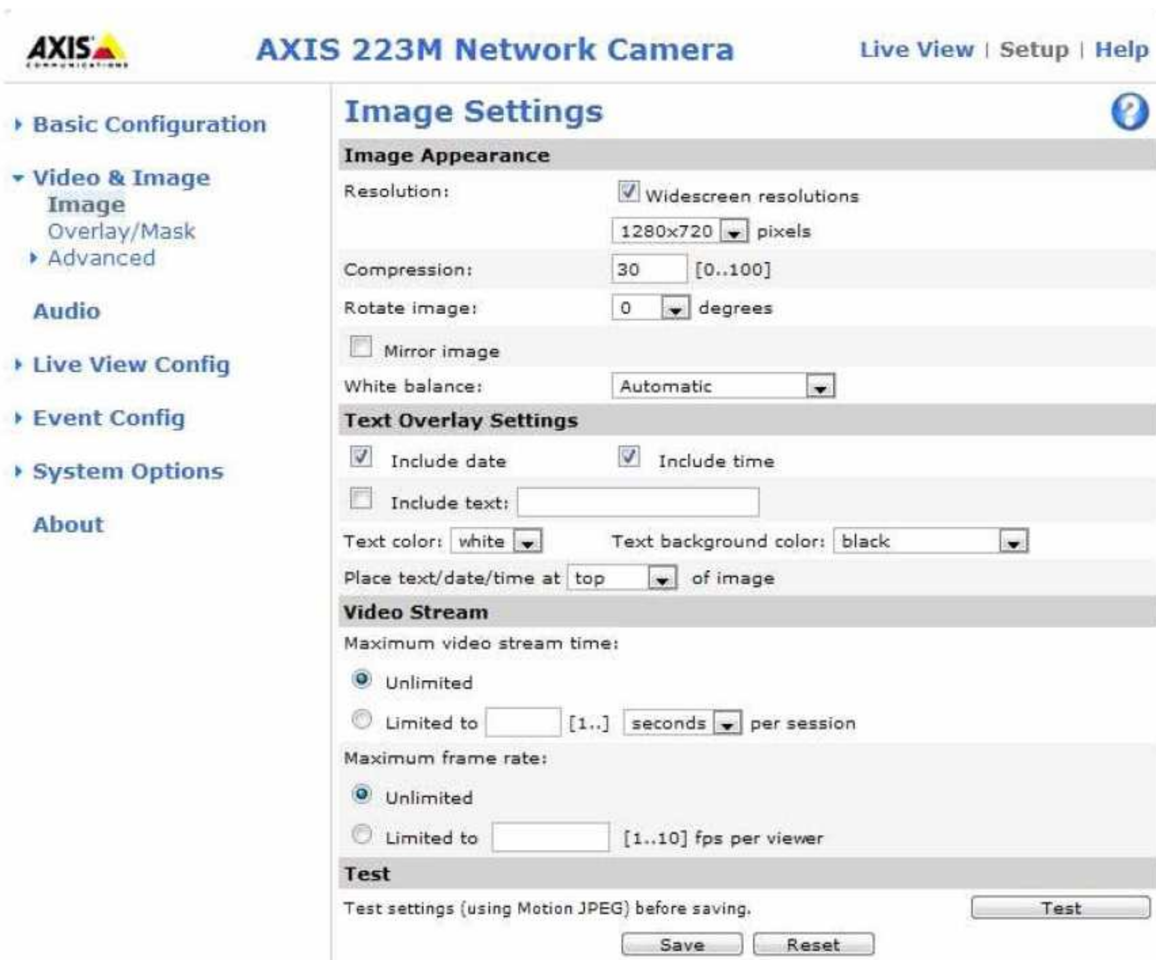


Figure 6. An example of video interface

2.2.2 Bluetooth Detection Component (Optional)

The Bluetooth detection component is used to provide additional supporting information from the vehicles on-board devices to help our system identify any specific vehicle. The Bluetooth devices consist of cellphones, headsets, laptops, and other electronic devices with Bluetooth capability. Once any vehicle passes by and is carrying a Bluetooth enabled device, the ABR captures Mac address and time interval and other information which can be applied to detect and identify that vehicle. Figure 7 shows the sample data that captured by the ABR.

The ABR in our CBL component runs the Fedora 2.6 operating system, the processor is a VIA Eden 1.2GHz, memory of DDR 533 1GB, and hard drive of 160G. see Figure 8 and Figure 9.

pkey	UnixTime	MacAdd	DevType	UnitSN
12737	1279817548.032	54:92:BE:7E:1F:F5	CellPhone	001-001-001
12738	1279818134.433	00:1E:B2:23:29:04	HandsfreeAudio/Video	001-001-001
12739	1279818933.228	00:1E:B2:24:EC:80	HandsfreeAudio/Video	001-001-001
12740	1279827886.566	00:1B:98:E1:D0:2F	CellPhone	001-001-001
12741	1279827886.566	00:1C:43:8B:41:B7	CellPhone	001-001-001
12742	1279827886.566	68:EB:AE:55:04:2D	CellPhone	001-001-001
12743	1279827894.282	00:05:4F:62:73:AD	HandsfreeAudio/Video	001-001-001
12744	1279827931.116	00:06:66:00:C3:19	Unknown Imaging	001-001-001
12745	1279827956.23	F4:0B:93:D3:AB:31	SmartPhone	001-001-001
12746	1279827963.964	00:23:D6:2F:CE:9C	CellPhone	001-001-001
12747	1279827962.034	00:00:A0:7C:9F:0E	CellPhone	001-001-001
12748	1279827979.449	00:21:06:07:AA:84	SmartPhone	001-001-001
12749	1279828000.869	00:18:C5:AB:61:16	CellPhone	001-001-001
12750	1279828006.657	00:26:5F:4E:19:A6	CellPhone	001-001-001
12751	1279828018.233	30:7C:30:46:2C:CE	SmartPhone	001-001-001
12752	1279828020.188	00:24:90:29:76:48	CellPhone	001-001-001
12753	1279828024.099	00:26:5D:CE:02:3A	CellPhone	001-001-001
12754	1279828033.77	00:26:5D:D5:D0:BB	CellPhone	001-001-001

Figure 7. Recorded data from ABR

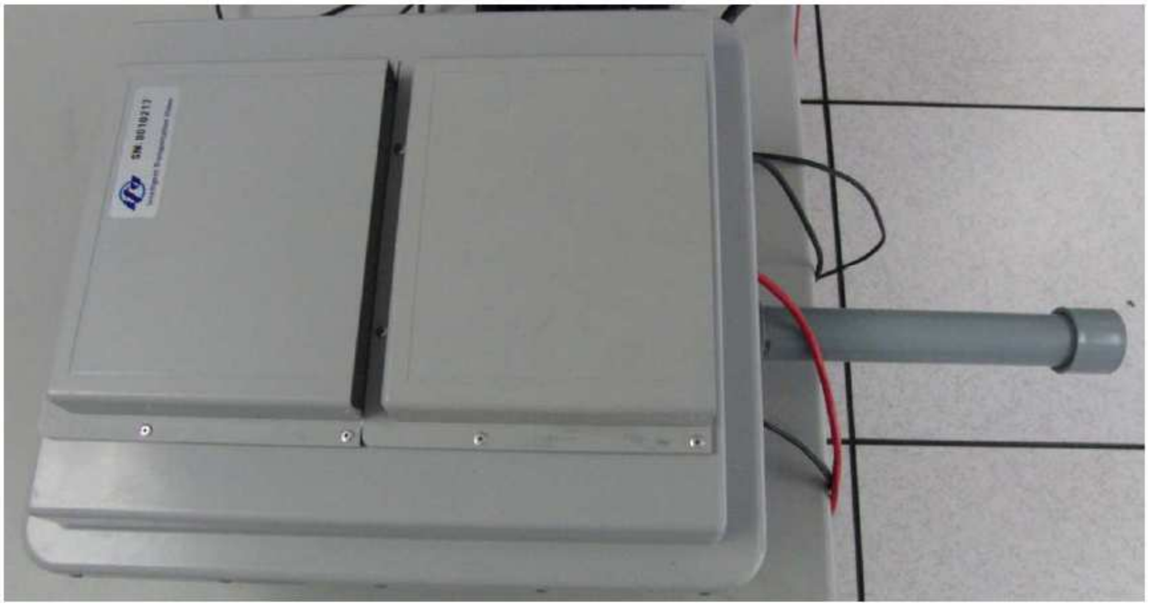


Figure 8. Automatic Bluetooth reader

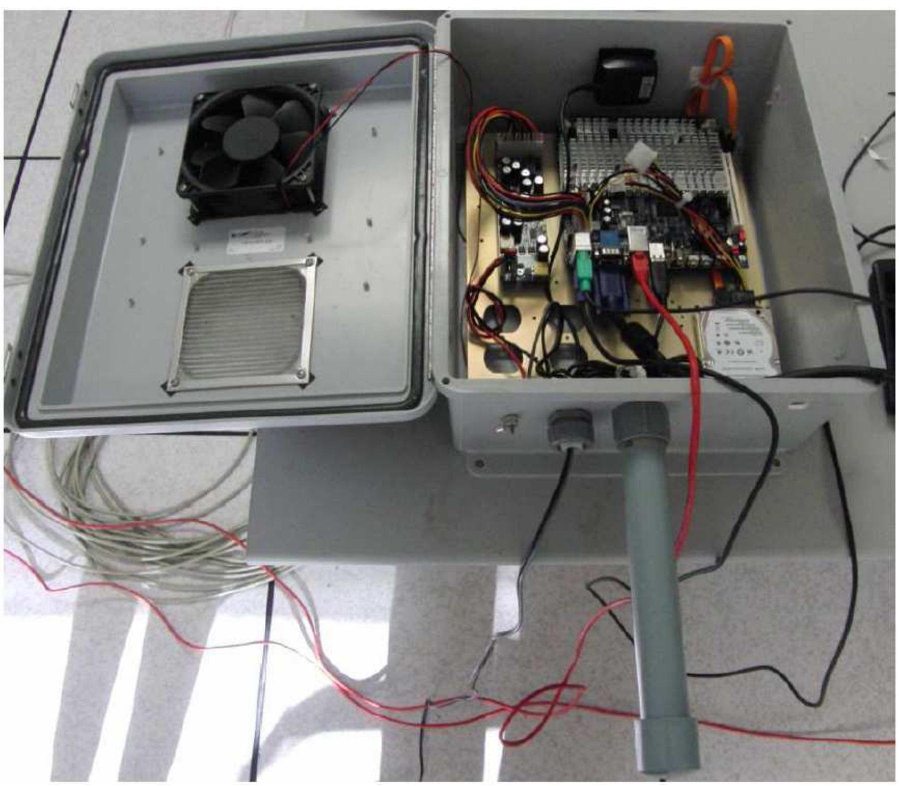


Figure 9. More details about ABR

2.2.3 License Plate Detection Component (Optional)

The ALPR (PIPS P372) used in our system is shown in Figure 10, which is manufactured by PIPS technology. By using strong infra-red illumination, it recognizes the license plate under a variety of lighting condition. In Figure 11, it demonstrates that the system works even under strong sunlight or in darkness. Figure 12 illustrates the vulnerability of the license plate recognition system that false and no-read images may occur. We suspect that alignment could be a culprit of the inability to capture the license plate. However, even under perfect alignment, an occlusion of the license by another vehicle can occur under heavy traffic. It is hence important to be able to identify a vehicle using other characteristics such as color, size, and shape.

Figure 10. The license plate device

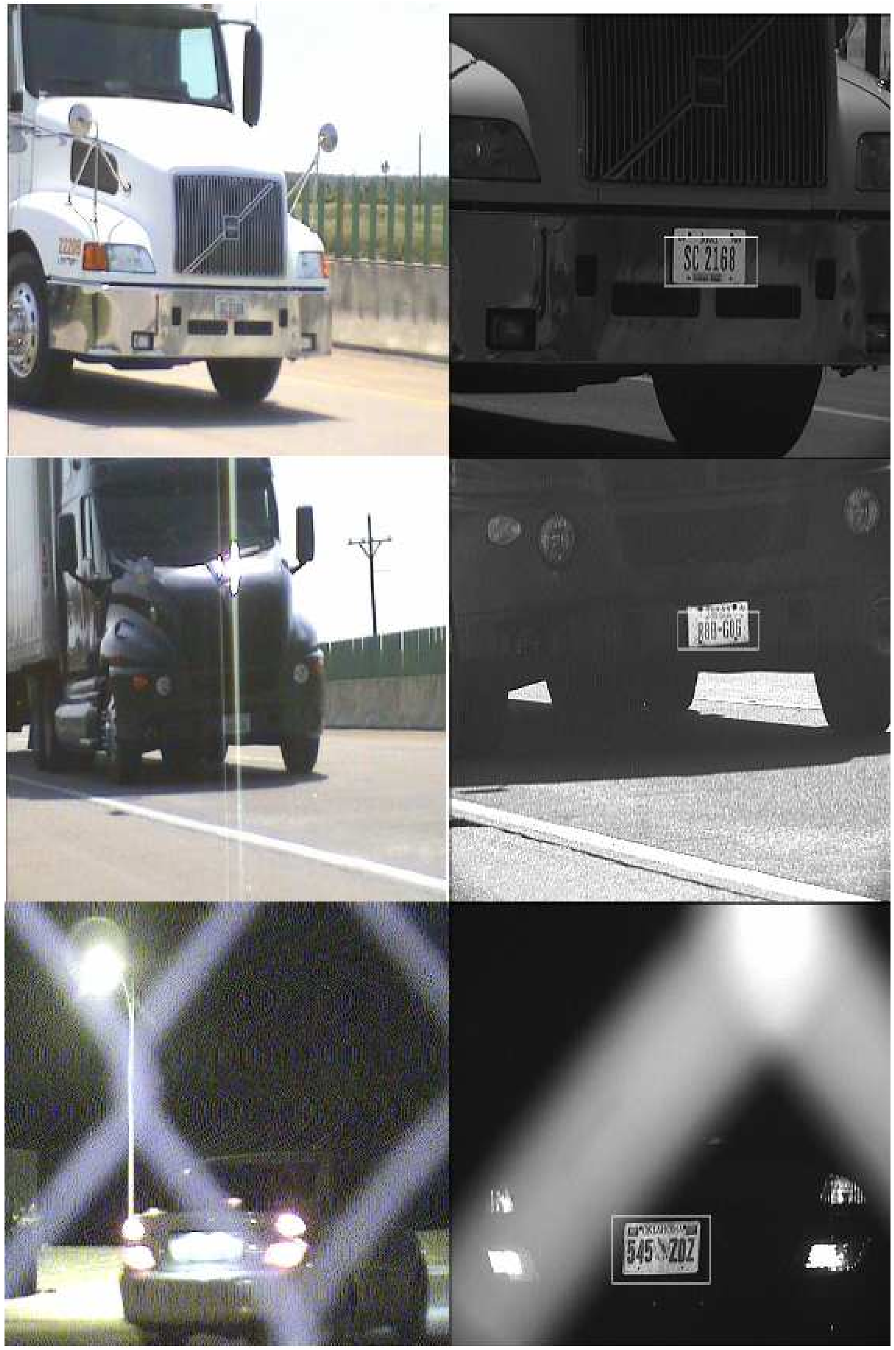


Figure 11. A view of the vehicle cab and license plate is shown in the upper photos for one vehicle and middle photos for another vehicle with lots of sun glare. The Infrared camera technology is very capable of eliminating sun glare to render a usable image. Nighttime images are also possible using the Infrared camera technology as show in lower photos.

Figure 12. Examples of the unsuccessful license plate recognition.

2.3 PROCESS FLOW

In this section, we describe the process flow for the CBL stations used for vehicle identification, see Figure 13.

Each CBL station mainly consists of three parts : video camera, ABR, and ALPR. The main job of a CBL station is to collect data from the above three parts and send this data to the service center for further processing, eg. the i-the sensing station $CBL(i)$ in Figure 13. At the service center, the data from the video camera is processed by the video processor module, and this process correlates a vehicle identification with the optionally supportive data from the ABR and ALPR. Here we take the data sent from the i-th CBL station as an example. At the video processor module, the first process is to perform blob detecting and blob tracking to get a unique vehicle ID, and saves it into a database for CBL station with index i. Then the obtained vehicle information of $CBL(i)$ will be compared with other CBL station databases $CBL(1),\ldots,CBL(m)$ except $CBL(i)$, where m denotes the total number of CBL stations. If a corresponding ID is found in $CBL(j), j \neq i$, it will report that this vehicle was captured in the j-th CBL station, otherwise, it will report -1, which means that this vehicle has not been captured by any CBL stations before.

To improve the system accuracy, we introduce the ABR and ALPR functions into the system. For the data sent from ABR (including Mac address, device type, and time) and ALPR (including license plate and time) in $CBL(i)$, similarly, the service center will save these records in a bluetooth database and a license plate database, respectively. Then for both databases, ABR, and ALPR, the service center will compare the current record from $CBL(i)$ with other $CBL(j), j \neq i$ in the database, respectively. If a corresponding record is found in $CBL(j), j \neq i$, it will report the corresponding record with a time stamp, otherwise, it will give a void report. With the supportive information from ABR and ALPR, the accuracy of the proposed vehicle identification system can be increased. Finally, the service center will report the identification result by combining the information from each component. Table 1 gives the final reports of all kinds of combinations based on the video processor and the additional ABR and ALPR.

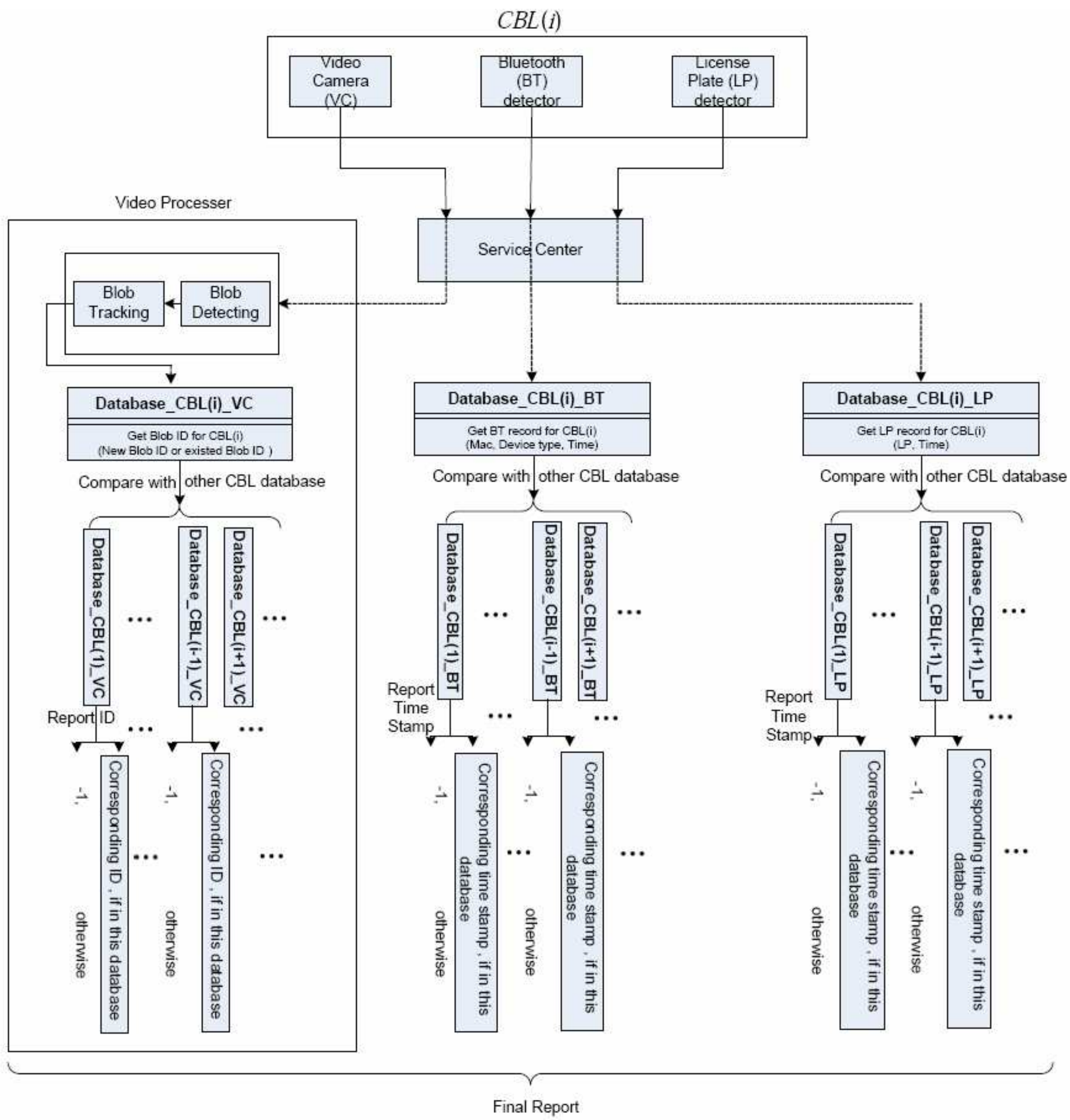


Figure 13. The Process flow.

Table 1. Final reports for service center

Video processor	ABR	ALPR	Final report
T	T	T	T
T	W	T	T
T	T	W	T
T	W	W	T
W	T	T	T
W	W	T	T
W	T	W	T
W	W	W	W

T: a corresponding vehicle ID or time stamp is found, and the time stamps from different components match each other;
W: a corresponding vehicle ID or time stamp is not found, or the time stamps do not match each other

2.4 DEVELOPMENT ENVIRONMENT

In this section, we will introduce the software development stage of our proposed system, which includes the development Operating System (OS), Language, Tools, dependent Libraries, and database.

Our proposed system is mainly developed under the Windows operating system and tested under both Windows (Windows XP/Windows Vista/WIndows 7) and Linux (Ubuntu/Fedora) operating systems to ensure the compatibility of the system.

We use three different programming languages, C++, Java, and Matlab to develop the proposed system, since they all support cross-platform operation.

1. The video capture, blob detection, and tracking modules are implemented by using OpenCV 2.1 library with C++ language under Microsoft Visual Studio 2008. Here, OpenCV is a free cross-platform computer vision library originally developed by Intel. It focuses mainly

on real-time image processing, which includes the basic image/video processing APIs, such as video capture from network camera/video file, blob detection/tracking etc. Then the developers can build their own processing modules very efficiently.

2. The GUI part of the proposed system is implemented mainly by using the Java platform under NetBeans 6.9 IDE. The Java language offers many outstanding characteristics, such as portability, cross-platform, object-oriented programming, easy GUI design, and so on. To achieve the portability and cross-platform operation, the developers use the Java development Kit (JDK) and end-users commonly use a Java Runtime Environment (JRE) installed on their own machine.
3. The video based identification algorithm is implemented by using Matlab 2009b since Matlab has the following advantages: Matlab is a numerical computing environment, which is developed by MathWorks. Matlab allows fast matrix manipulations, visualization of data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, such as Java, C++, etc.
4. The data representation, storage, and query in the proposed system are implemented by using folder based image datasets (for detected vehicle blobs and license plate images) and the SQLite database (for captured Bluetooth data).

CHAPTER 3
VIDEO BASED VEHICLE IDENTIFICATION

3.1 BACKGROUND

For vehicle detection from a video or image sequence, the most obvious approach has been first to compute the stationary background image, and then identify the moving vehicles as those pixels in the image that differ significantly from the background, which is named background subtraction [10]. In earlier work in the “Road watch” project at Berkeley, it showed that background subtraction can provide effective locating and tracking results for moving vehicles [11]. However, traffic shadows cause serious problems when doing subtraction and slow moving or stationary traffic is difficult to detect. This led to the emergence of the adaptive background methods. A standard adaptive method can create a background approximation which is similar to the current static scene and is adaptive to illumination changes, slowing moving traffic. The Gaussian Mixture Model (GMM) is also very effective for adaptive traffic detection. It was first used as a single Gaussian model for real-time tracking of the human body in [12], then proposed for background subtraction in [10]. One of the most famous approaches for updating GMM is presented in [13] and further elaborated in [14]. According to the computer vision literature, vehicle tracking from a video or image sequence can be classified into four categories [8]. First is model based tracking: the three dimensional model-based vehicle tracking system focuses on recovering trajectories and models with high accuracy [15], [16]. However, it is unrealistic to build detailed geometric models for all vehicles on the roadway. Second is region based tracking: this approach detects a connected region in the image, namely a blob, associated with each vehicle, and tracks it over time using different algorithms, such as cross-correlation [17], but this approach is not fit for congested traffic conditions. The third category is active contour based tracking: this approach builds a bounding contour of the vehicle and keeps updating it. Koller et al. [18] was based on this approach. But occlusion and initialization is still the difficult part of the problem. And the fourth category is feature based tracking: instead of tracking each vehicle as a whole, this approach tracks sub-features and shows effective tracking results with partial occlusions [8]. However, grouping features is still a difficult problem for this approach. For the vehicle identification problem, the state of the art methods can be mainly classified into two groups: template based and appearance based methods. The template based methods predefine patterns of vehicle, and perform correlation between the image and the template. Handmann et al. [19] proposed a U shaped template based on the front or rear view of each vehicle. These appearance based methods learn the characteristics of vehicle appearance from a set of training images, and each training image is represented as a set of local and global features. Wu and Zhang [20] used standard Principal Components Analysis (PCA) for feature extraction and the nearest neighbor classifier, reporting an 89 percent accuracy for vehicle detection. In [21], PCA was used together with Neural Networks (NNs) for vehicle recognition. Z. Sun at el. [22] employed Gabor filters for feature extraction and Support Vector Machines (SVM) to classify vehicles. Viola et al. [23] had introduced a rapid object detection scheme based on a boosted cascade of simple Haar like features. The Scale Invariant Feature Transform (SIFT) by Lowe [24] provided distinct features for object recognition. Recently, by using sparse representation, Wright et al. proposed a face recognition algorithm [1], which offers very competitive performance for face recognition. Based on the idea of sparse representation for objection classification and identification, we proposed a video based vehicle identification framework for this project.

3.2 VEHILCE TRACKING AND DETECTING

In this section, we will introduce our proposed vehicle tracking scheme. The four main components in our vehicle tracking scheme is shown in Figure 14, which contains Foreground/ Background (FG/BG) detecting, Blob detecting, Blob tracking, and moving direction and speed detecting. We will discuss the details of each component in the following subsections.

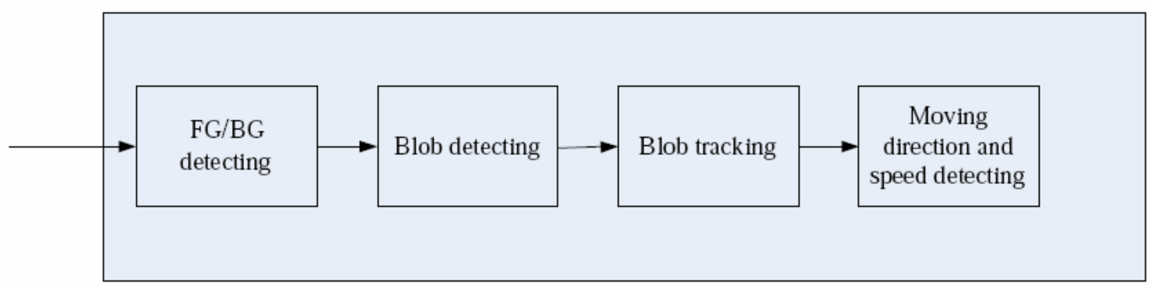


Figure 14. The workflow of the proposed vehicle tracking scheme

3.2.1 Foreground / Background Detecting

FG/BG detection is the first and most important step in our vehicle tracking scheme, since the accuracy of FG/BG estimation affects all subsequent steps.

Many FG/BG detection methods have been proposed in [25], [26], [27], [28], [29], [30], [12], [10], [13]. In [25], Ridder et al. use a Kalman Filter to model each pixel, which made their algorithm robust to light variance. However, Ridder's method does not consider adaptive thresholding. Pfinder [12] implemented a multi-class statistical model for the tracked objects, where the background is modeled as a single Gaussian per pixel. Nevertheless, this method is not working well for outdoor scenes. Recently, a pixel-wise framework for vehicle detection is proposed by Rriedman and Russell [10]. However, the pixels of this system were constrained by three separated predetermined distributions.

In this project, we adopt the approach proposed by Stauffer et al. in [13], which provides an adaptive background mixture model for real-time tracking by modeling the values of any pixel as a mixture of Gaussians. This method is robust for lighting changes, tracking through cluttered regions, slow-moving objects and so on. In Figure 15 (a), we present a video frame which contains a moving vehicle. Figure 15 (b) shows the background subtraction results by using the algorithm in [13].

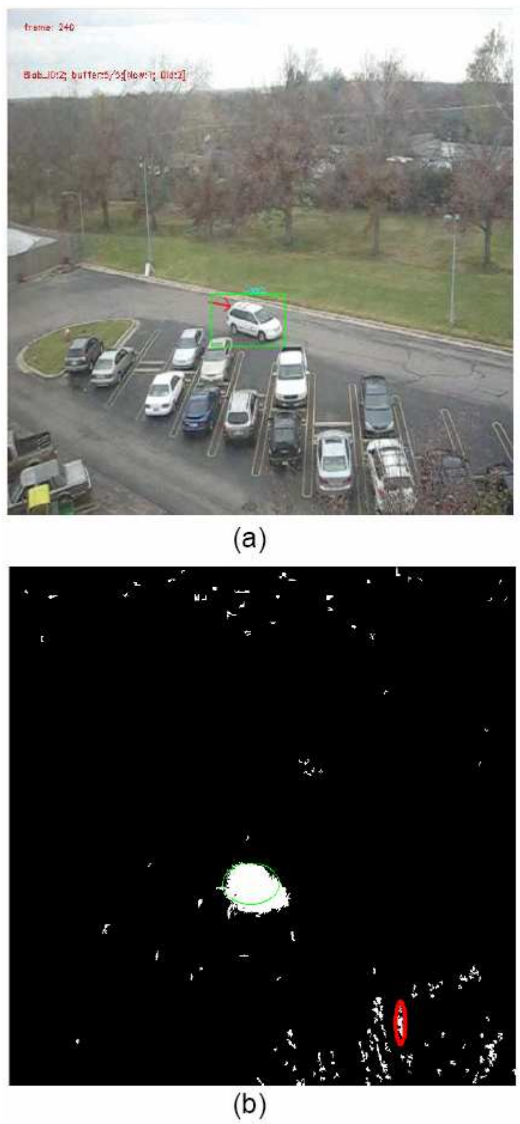


Figure 15. The example of the vehicle tracking algorithm.

3.2.2 Blob Detecting

Blob detection is employed to detect any newly entering object in each frame using the output from the FG/BG estimation module. Our blob detector is implemented based on [31]. First, neighboring pixels of the FG will be connected to generate blobs. Then for each successive frame, each blob is tracked. If a detected blob is not found previously, it will be added as a new blob in a list.

3.2.3 Blob Tracking

The blobs that have been found in the previous step will be traced by the blob tracking component. It consists of two subcomponents: one subcomponent creates a confidence map for the current frame based on the color histogram of the target object in the previous frame, and utilizes the mean shift algorithm [32] to find a peak in the confidence map close to the old position of the object; the other subcomponent uses a Kalman filter to predict the position of the blob in the next frame.

3.2.4 Moving Direction And Speed Detecting

The moving direction and speed detecting is accomplished by using optical flow estimation [33], which tries to calculated the motion between two video frames at times t and $t+\tau$. In our scheme, we use the blobs with the same index in different video frames to calculate the optical flow. In Figure 15 (a), the red arrow indicates the moving direction, and the length of the arrow indicates the speed of the vehicle, where a longer arrow means a faster moving vehicle.

3.2.5 Reduce False Positive Detection And Tracking

The aforementioned algorithms offer high sensitivity for blob detecting and tracking, however, false positive rate could be high due to clutters from the motions of leaves and grass. Moreover, we may only be interested in one direction of the traffic flow. To tackle these issues, we utilize the following filters to exclude these unwanted blobs.

1. Blob histogram (BH) filter: excludes blobs, where the number of observations from different video frames for each given blobs ID is less than τ_{BH} times, where τ_{BH} is a predetermined threshold.
2. Motion distance (MDs) filter: excludes blobs, whose moving distance are less than a given threshold τ_{MDs} (in pixels).
3. Motion direction (MDr) filter: excludes blobs, whose motion direction are not the same as the pre-assigned direction τ_{MDr} (right, left, up, down and etc.).

3.3 VEHICLE IDENTIFICATION VIA SPARSE REPRESENTATION AND BAYESIAN FORMALISM

A basic problem in vehicle identification is to determine if a newly entering vehicle has already been registered in a database or not, and to find a corresponding vehicle ID if such record exist. The core idea of the proposed vehicle identification algorithm is based on sparse representation, where similar idea was used in [1] for face recognition.

3.3.1 Sparse Representation Of A Vehicle

Before generating a sparse representation for a vehicle and finding its corresponding vehicle ID, we will first arrange the database into matrices, which is built using labeled training samples from M different vehicles. Here we assume that k_i denotes the number of $w \times h$ training images for the i-th vehicle ID, where $i = 1, \ldots, M$, and $k = k_1 + k_2 + \cdots k_M$ denotes the number of images in the database. Then, we reshape each $w \times h$ image into a column vector $v \in \mathbb{R}^c$, where $c = wh$; the k_i training images from the i-th vehicle ID constitute the columns of a matrix $\Phi_i = [v_{i,1}, v_{i,2}, \ldots, v_{i,k_i}] \in \mathbb{R}^{c \times k_i}$; all k images from the database are combined to form a new matrix $\Phi = [\Phi_1, \Phi_2, \ldots, \Phi_M] = [v_{1,1}, v_{1,2}, \ldots, v_{M,k_M}] \in \mathbb{R}^{c \times k}$.

For a newly entering vehicle $u \in \mathbb{R}^c$, if sufficient training samples in the database share the same feature as the incoming vehicle, (e.g. it happens when the incoming vehicle was captured previously, let say, with a vehicle ID i), then the vehicle can be approximately represented as the linear combination of the training samples in Φ_i

$$y = \Phi_i \theta = \theta_{i,1} v_{i,1} + \theta_{i,2} v_{i,2} + \ldots + \theta_{i,k_i} v_{i,k_i}, \tag{1}$$

where $\theta = [\theta_{i,1}, \theta_{i,2}, \ldots, \theta_{i,k_i}]^T$ and $\theta_{i,j} \in \mathbb{R}, j = 1,2,\ldots,k_i$.

However, we do not know the identity of the incoming vehicle at the beginning. Fortunately, we can instead represent the incoming vehicle $y \in \mathbb{R}^c$ using the entire set of images in the database with relatively small increase in computation complexity. The linear combination of all the training samples is written as

$$y = \Phi x_s = [\Phi_1, \Phi_2, \ldots, \Phi_M][0, \ldots, 0, \theta_{i,1}, \theta_{i,2}, \ldots, \theta_{i,k_i}, 0, \ldots, 0]^T, \tag{2}$$

where with a high probability, $x_s = [0, \ldots, 0, \theta_{i,1}, \theta_{i,2}, \ldots, \theta_{i,k_i}, 0, \ldots, 0]^T \in \mathbb{R}^k$ is a coefficient vector which just has nonzero entries for those associated with i-th vehicle ID.

In order to find x_s which can accurately determine the identity of the incoming vehicle, we need to solve the linear equation $y = \Phi x$. Nevertheless, this is a underdetermined equation, and it does not have a unique solution x_s. In particular, the problem could be solved as the following optimization problem.

$$\hat{x} = arg\min \|x\|_2 \quad \text{subject to} \quad \Phi x = y. \tag{3}$$

The optimization problem can be solved easily, but the solution will not be sparse. In other words, the nonzero entries of the solution will spread over many class IDs. This makes the later identification steps difficult to succeed. Instead, we could like to find the sparsest solution, ideally, by solving l^0-norm minimization:

$$\hat{x} = arg\min\|x\|_0 \quad \text{subject to} \;\; \Phi x = y. \tag{4}$$

Unfortunately, the above optimization problem is a difficult combinatory problem, and is NP-hard. However, if x_s is sparse enough, the l^1-norm minimization problem can provide an equivalent solution with an l^0-norm solution:

$$\hat{x} = arg\min\|x\|_1 \quad \text{subject to} \;\; \Phi x = y. \tag{5}$$

In general, measurement data may be noisy, so y may not be represented as the sparse combination of training samples exactly. Thus Eq. (2) will be rewritten as:

$$y = \Phi x_s + \Upsilon_z, \tag{6}$$

where $\Upsilon_z \in \mathrm{R}^c$ is noise and has a bounded energy $\|\Upsilon_z\|_2 < \varepsilon$. Then the l^1-minimization problem is modified as follows:

$$\hat{x} = arg\min\|x\|_1 \quad \text{subject to} \;\; \|\Phi x - y\|_2 \le \varepsilon. \tag{7}$$

3.3.2 Sparse Solution Via Bayesian Formalism

To find the sparse solution for the l^1-norm minimization problem, numerous methods have been proposed, such as Matching Pursuit (OMP) [34], LASSO [35], Interior-point Methods [36], SAMP [37], and Gradient Method [38]. However, above methods only provide approximate sparse solutions but do not tell how likely the given solutions are optimum. Therefore, we will use Bayesian formalism instead which returns both a sparse solution x and the probability information indicating the uncertainly of the solution from the actual sparse x. Our approach is based on [39] by extending Tipping's Relevance Vector Machine (RVM) theory [40].

First, we assume that x is the sum of two parts x_b and x_e (so, $x = x_b + x_e$), where $x_b \in \mathrm{R}^k$ is the vector composed of nonzero entries only at the L largest coefficients of x, and $x_e \in \mathrm{R}^k$ is the vector composed of nonzero entries only at the rest of the coefficients. Moreover, since we assume that measurements can be noisy as in Eq.(6), the vector corresponding to a vehicle y is rewritten as:

$$y = \Phi x + \Upsilon_z = \Phi x_b + \Phi x_e + \Upsilon_z = \Phi x_b + \Upsilon_e + \Upsilon_z = \Phi x_b + \Upsilon \tag{8}$$

where $\Upsilon_e = \Phi x_e$. Using Central-Limit Theorem [41], we assume that both Υ_e and Υ_z are zero mean and approximately Gaussian distributed, then $\Upsilon = \Upsilon_e + \Upsilon_z$ can be approximated as a Gaussian noise with zero mean and unknown variance σ^2. Then the Gaussian likelihood is given by

$$p(y \mid x_b, \sigma^2) = (2\pi\sigma^2)^{-c/2} \exp(-\frac{1}{2\sigma^2}\left\| y - \Phi x_b \right\|^2). \tag{9}$$

Given Φ and y, the problem now is to estimate the sparse vector x_b and the noise variance σ^2. By Bayes' rule, we have

$$p(x_b, \sigma^2 \mid y) = \frac{p(y \mid x_b, \sigma^2)\, p(x_b, \sigma^2)}{p(y)}. \tag{10}$$

Note that x_b is sparse and can be modeled by a Laplace distribution [42], [43]. However, the Laplace prior is not conjugate to the Gaussian likelihood, and thus the inference problem can not be written in closed-form [42], [44]. Thus instead of Laplace prior, we will perform a hierarchical sparseness prior [40] which has similar properties as the Laplace prior and thus allows convenient conjugate exponential analysis on x_b.

To perform the hierarchical sparseness prior, we define a zero mean Gaussian prior distribution on x_b with:

$$p(x_b \mid \alpha) = \prod_{i=1}^{k} N(x_{bi} \mid 0, \alpha_i^{-1}), \tag{11}$$

where $\alpha \in \mathrm{R}^k$ is hyperpriors, which themselves are modeled by Gamma distributions

$$p(\alpha \mid a,b) = \prod_{i=1}^{k} \Gamma(\alpha_i \mid a,b). \tag{12}$$

Based on Eq.(11) and Eq.(12), the conditional probability x_b given a and b is obtained by marginalizing over the hyper-parameters α:

$$p(x_b \mid a,b) = \prod_{i=1}^{k} \int_0^{\infty} N(x_{bi} \mid 0, \alpha_i^{-1}) \Gamma(\alpha_i \mid a,b) d\alpha_i. \tag{13}$$

where the integral $\int_0^{\infty} N(x_{bi} \mid 0, \alpha_i^{-1}) \Gamma(\alpha_i \mid a,b) d\alpha_i$ obeys a Student-t distribution [45]. The Student-t distribution has a strong peak at $x_b = 0$ with appropriate values of a and b, so that most entries of x_b are zeros with the prior distribution Eq.(13).

Moreover, we model the inverse of the noise variance $\beta = \sigma^{-2}$ with a Gamma distribution:

$$p(\beta \mid c,d) = \Gamma(\beta \mid c,d), \tag{14}$$

where c and d are also hyperpriors. Here we fix the value of the parameters a,b,c and d to zero, and thus α and β are uniform hyper-priors over a logarithmic scale.

Now, based on the priors which have been defined above, the posterior can be decomposed as:

$$p(x_b, \beta, \sigma^2 \mid y) = p(x_b \mid y, \alpha, \sigma^2) p(\alpha, \sigma^2 \mid y). \tag{15}$$

For the first term $p(x_b \mid y, \alpha, \sigma^2)$, since we can compute the integral $p(y \mid \alpha, \sigma^2) = \int p(y \mid x_b, \sigma^2) p(x_b \mid \alpha) dx_b$ based on Eq.(9) and Eq.(11), the posterior distribution of x_b can be expressed analytically as:

$$\begin{aligned} p(x_b \mid y, \alpha, \sigma^2) &= \frac{p(y \mid x_b, \sigma^2) p(x_b \mid \alpha)}{p(y \mid \alpha, \sigma^2)} \\ &= (2\pi)^{-(k+1)/2} |\Sigma|^{-1/2} \exp\left\{ -\frac{1}{2} (x_b - \mu)^T \Sigma^{-1} (x_b - \mu), \right\} \end{aligned} \tag{16}$$

where the covariance and mean of the posterior distribution are

$$
\begin{aligned}
\Sigma &= (\sigma^{-2}\Phi^T\Phi + A)^{-1}, \\
\mu &= \sigma^{-2}\Sigma\Phi^T y,
\end{aligned} \tag{17}
$$

respectively, with $A = diag(\alpha_0, \alpha_1, \ldots, \alpha_k)$.

For the second term $p(\alpha, \sigma^2 \mid y)$, maximizing $p(\alpha, \sigma^2 \mid y)$ is the same as maximizing $p(y \mid \alpha, \sigma^2) p(\alpha) p(\sigma^2)$. Since $p(\alpha)$ and $p(\sigma^2)$ are uniform hyper-priors, we only need to maximize the term $p(y \mid \alpha, \sigma^2)$, which is given by:

$$
\begin{aligned}
p(y \mid \alpha, \sigma^2) &= \int p(y \mid x_b, \sigma^2) p(x_b \mid \alpha) dx_b \\
&= (2\pi)^{-(k)/2} \mid \sigma^2 I + \Phi A^{-1} \Phi^T \mid^{-1/2} \exp\left\{ -\frac{1}{2} y^T (\sigma^2 I + \Phi A^{-1} \Phi^T)^{-1} y \right\}
\end{aligned} \tag{18}
$$

where this quantity is called marginal likelihood, and its maximization is known as the type-II maximum likelihood. The associated maximization problem becomes one of seeking the hyperpriors α and σ^2. Estimation of these hyperpriors can be achieved by employing the Type-II maximum likelihood method which is also referred to as the “evidence for the hyper-parameters” [40].

3.3.3 Identification Based On Sparse Representation

Before identifying an incoming vehicle, first, we need to use the information of sparse representation to decide if the test object is a vehicle, and if the entering vehicle corresponds to one of the vehicle IDs in the dataset.

For each estimated sparse representation $\hat{x}$, the entries of coefficient vector $\hat{x}$ distribute in two different ways: a). most nonzero entries concentrate on one vehicle ID, and b). all nonzero entries spread widely among multiple vehicle IDs or the entire database. The first case implies that the incoming vehicle is likely to correspond to the vehicle ID on which non-zero entries concentrate; whereas the second case indicates that the feature information of the incoming vehicle is not in the database. Based on the above observation, we adopt the sparsity concentration index (SCI) to determine if a vehicle has been captured before or not [1]. The SCI of a coefficient vector

$x \in \mathrm{R}^k$ is defined as

$$SCI(x) = \frac{M \cdot \max_i \|\delta_i(x)\|_1 / \|x\|_1 - 1}{M - 1} \in [0,1], \tag{19}$$

where $\delta_i(x) \in \mathrm{R}^k$ is a vector whose coefficients are only associated with the i-th vehicle ID of vector x.

Hence, for an estimated sparse representation $\hat{x}$ solved in Section 3.1, if $SCI(\hat{x}) = 1$, nonzero entries only concentrate on one vehicle ID, and if $SCI(\hat{x}) = 0$, nonzero entries are spread uniformly among all classes. Given a threshold $\varepsilon \in (0,1)$, if $SCI(\hat{x}) \geq \varepsilon$, the test vehicle will be considered as a "known" vehicle, otherwise an "unknown" vehicle will be reported. For the former case, we still need to determine the identity of the vehicle. We can achieve this by comparing the residual errors corresponding to different vehicle IDs. More precisely, denote $\hat{y}_i = \Phi\delta_i(\hat{x}_1)$ which is the approximate representation obtained by using only the entries associated with the i-th vehicle ID. Intuitively, we assign the incoming vehicle y to the ID with the best approximation. This corresponds to the minimum residual error between y and $\hat{y}_i$ given by

$$\min_i r_i(y) = \|y - \Phi\delta_i(\hat{x}_1)\|_2 \tag{20}$$

Now based on the sparse representation, SCI method for validation, and residual identification, the algorithm procedure for vehicle identification can be summarized in the following:

1. Input: an arranged matrix of dataset with M vehicles $\Phi = [\Phi_1, \Phi_2, \ldots, \Phi_M] \in \mathrm{R}^{c \times k}$, an incoming vehicle represented by $y \in \mathrm{R}^c$, error tolerance $\varepsilon > 0$ and SCI threshold $\varepsilon \in (0,1)$.
2. Normalize the columns of Φ to have unit l^2-norm.
3. Solve the l^1-minimization problem:
   $\hat{x} = arg\min_x \|x\|_1$ subject to $\|\Phi x - y\|_2 \leq \varepsilon$

4. Compute $SCI(\hat{x}) = \dfrac{M \cdot \max_i \|\delta_i(x)\|_1 / \|x\|_1 - 1}{M-1} \in [0,1]$. If $SCI(\hat{x}) \geq \varepsilon$, go to step 5, otherwise return a report that the incoming vehicle is not in the dataset.
5. Compute the residuals errors $r_i(y) = \|y - \Phi\delta_i(\hat{x})\|_2$, for $i = 1, \ldots, M$.
6. Output : identity $(y) = arg\min_i r_i(y)$.

3.4 IDENTIFICATION BASED ON MULTIPLE FRAMES

As stated in Section 3.3, for further identification, $SCI(\hat{x})$ works like a filter for each frame to sift through any unknown vehicle. However, in the vehicle identification problem, sometimes the constraint $SCI(\hat{x}) \geq \varepsilon$ can not differentiate the known and unknown vehicle accurately. Fortunately, it is possible to take advantage the information from multiple frames to further improve the identification accuracy. In this project, we propose an additional rule using the identification concentration index (ICI), which is based on multiple frame validation, to improve the vehicle identification accuracy.

Here we assume that there are F number of frames which include an incoming vehicle and $d \in \mathrm{R}^F$ is a vector to save the identified IDs from F frames. An average $\overline{SCI(\hat{x})}$ is obtained by $\overline{SCI(\hat{x})} = \frac{1}{F}\sum_{l=1}^{F} SCI(\hat{x}_l)$ and assuming that out of the F frames, D number of unique identified IDs is found. Then, considering the information from all the F frames, the proposed ICI is defined as:

$$ICI(d) = \frac{D \cdot \max(\rho_j(d))/\|d\|_0 - 1}{D-1} \in [0,1], \tag{21}$$

where $\rho_j(d)$ counts the existing number of j-th unique ID in d, $j = 1, \ldots, D$; if $D = 1$, $ICI(d) = 1$, since there is only one vehicle ID. Moreover, if $ICI(d) = 1$, F frames only concentrate on one vehicle ID, and if $ICI(d) = 0$, F frames are spread among all D number of IDs. Then given a threshold $\zeta \in (0,1)$, an incoming vehicle is considered as "known" if $ICI(d) \geq \zeta$, otherwise it is considered as "unknown".

Now we will introduce how to combine SCI and ICI to increase the accuracy of vehicle identification. Based on the previous algorithm for vehicle identification and the proposed ICI, the algorithm procedure can be rewritten as follows:

1. Input: an arranged matrix of dataset with M vehicles $\Phi = [\Phi_1, \Phi_2, \ldots, \Phi_M] \in \mathbb{R}^{c \times k}$, an incoming vehicle $y \in \mathbb{R}^c$, error tolerance $\varepsilon > 0$, SCI threshold $\varepsilon \in (0,1)$, and ICI threshold $\zeta \in (0,1)$.
2. Normalize the columns of Φ to have unit l^2-norm.
3. For all F frames which include an incoming vehicle, solve the l^1 -minimization problem: $\hat{x}_l = arg\min_{x_l} \|x_l\|_1$ subject to $\|\Phi x_l - y_l\|_2 \leq \varepsilon$, $l = 1, \ldots, F$.
4. Compute $SCI(\hat{x}_l) = \dfrac{M \cdot \max_i \|\delta_i(x_l)\|_1 / \|x_l\|_1 - 1}{M - 1} \in [0,1]$ for each frame, and then get $\overline{SCI(\hat{x})} = \dfrac{1}{F} \sum_{l=1}^{F} SCI(\hat{x}_l)$ for F multiple frames.
5. Compute $ICI(d) = \dfrac{D \cdot \max(\rho_j(d)) / \|d\|_0 - 1}{D - 1} \in [0,1]$.
6. If $\overline{SCI(\hat{x})} < \varepsilon$, and $ICI(d) < \zeta$, where $\varepsilon \in (0,1)$ and $\zeta \in (0,1)$, return a report that the incoming vehicle is not in the dataset, otherwise go to step 7. Note that the value of ε and ζ are tuned by using the algorithm of SVM.
7. Compute the residuals errors $r_i(y) = \|y - \Phi \delta_i(\hat{x})\|_2$, for $i = 1, \ldots, M$.
8. Output : identity $(y) = arg\min_i r_i(y)$.

CHAPTER 4
ANALYSIS

In this Chapter, we present the experimental results based on the proposed system setup and algorithms for vehicle identification. In the experiment, we use two video cameras (C_1 and C_2) to capture video data from highway I-44 in Tulsa.

Figure 16. The vehicle dataset sample

4.1 EXPERIMENT SETUP

First, we separate the video data captured from two cameras into two parts, one for training and the other for testing. During the training phase, we use the video from C_1 to build the vehicle database, which included 291 captured vehicles and in total 13,931 images (about 48 frames per vehicle). Then another 601 captured vehicles from C_2 (about total 23,920 images), where 291 vehicles were also captured by C_1 and the remaining 310 vehicles were not, were registered into the database using our proposed algorithm. Then the SCI and ICI values are obtained after the registration, and a ground truth is generated manually. This information is used to train the SVM classifier, which will be used to identify unknown vehicles based on SCI and ICI values. During the testing phase, we use another 10 minute length of video from C_1 to build the vehicle database, which includes 13,573 images for 287 vehicles. Then, another 23,838 images for 608 vehicles from C_2 are used to test the identification accuracy of our proposed algorithm. In this testing phase, 287 vehicles appeared in the database, and 321 vehicles were out of the database.

Figure 17. Blob detection result I

Figure 18. Blob detection result II

4.2 Vehicle Tracking And Detecting

In Figure 17 and Figure 18, we present the blob detection results by using our proposed vehicle detecting and tracking algorithms. The green box in the video frame indicates the location and the size of a moving vehicle. Then each detected blob will be saved into the database for registration, see Figure 16. In this experiment, we achieved more than 90% accuracy of the blob detection.

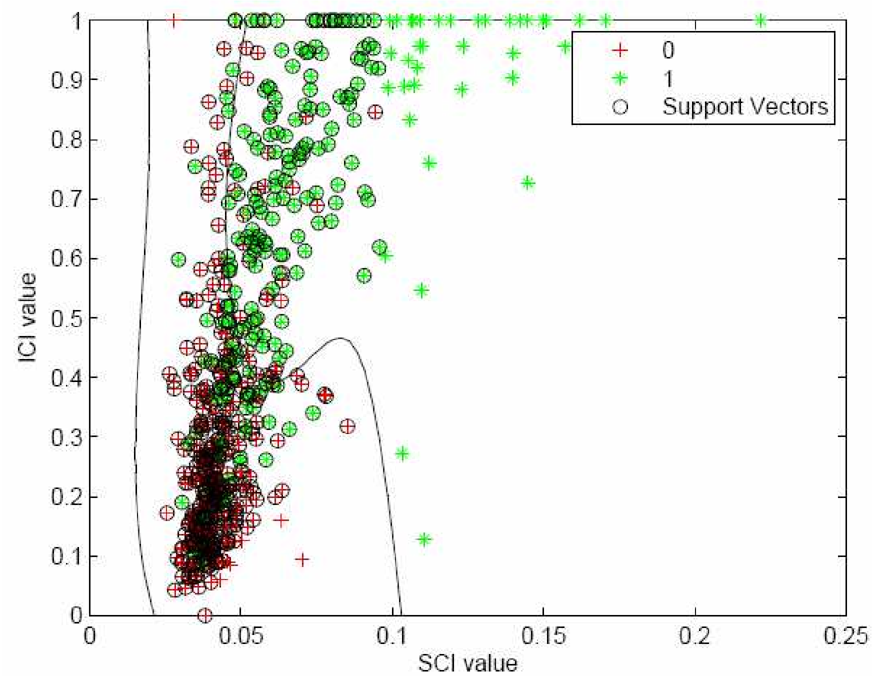


Figure 19. Trained SVM classifier using SCI and ICI, where 1 indicates correctly identified vehicles and 0 indicates incorrectly identified vehicles

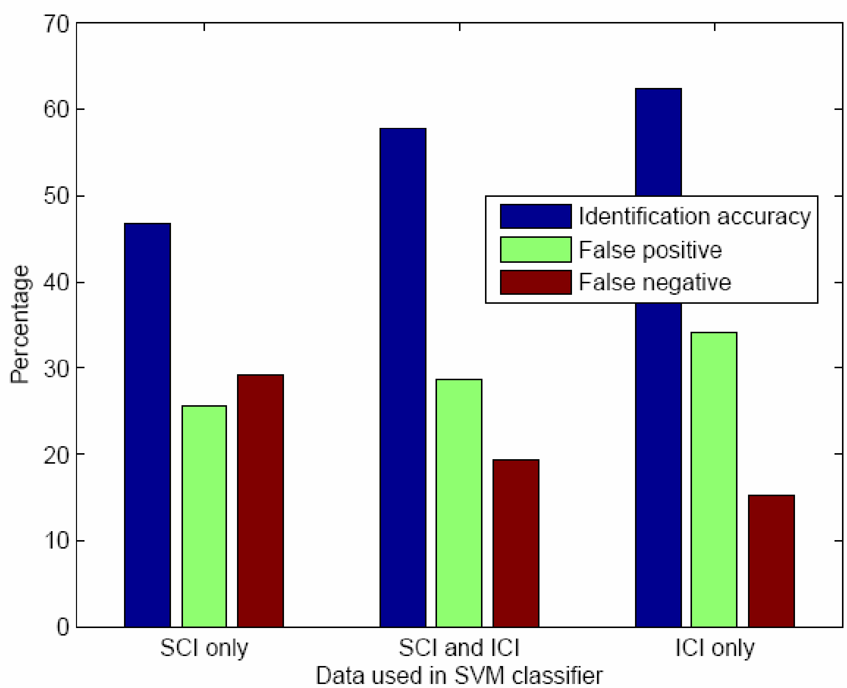


Figure 20. Classification performance by using different kinds of data for the SVM classifier, such as, SCI only, ICI only, and the combination of SCI and ICI

4.3 Video Based Vehicle Identification

In this section, results of the SVM classifier, identification with different feature sizes, and a detailed example of vehicle identification are presented.

4.3.1 SVM Classifier

In our experiment, we applied a sparse representation-based Identification (SRI) algorithm to each testing vehicle image by solving the optimization problem in Eq. (7) with the RVM. Two dimensional training data is used in the SVM classifier which includes the SCI and the ICI. Moreover, a 4th order polynomial kernel function is used in the SVM classifier. Figure 19 shows the trained SVM classifier using the manually labeled data (SCI and ICI) obtained from the training part, where the (red) cross means that vehicles only appeared in both C_1 and C_2 cameras, the (green) star means that vehicles only appeared in the C_2 camera, and the solid (black) line is the classification boundary obtained from SVM classifier.

We compare the classification performance by using different kinds of data for SVM classifier, such as, SCI only, ICI only, and the combination of SCI and ICI. Before showing the results, we will define some terminologies first. In this project, "$\#\text{ of acceptance}$" is the number of vehicles that were accepted by the SVM classifier and "$\#\text{ of rejection}$" is the number of vehicles that were rejected by the SVM classifier. The identification accuracy (IA) is defined as $\frac{\#\text{ of positive reports}}{\#\text{ of vehicles in the database}}$, where "$\#\text{ of vehicles in the database}$" means vehicles appeared in both C_1 and C_2 cameras. The false positive rate (FPR) is defined as $\frac{\#\text{ of false positive reports}}{\#\text{ of vehicles in the database}}$. The false negative rate (FNR) is defined as $\frac{\#\text{ of false negative reports}}{\#\text{ of vehicles out of the database}}$, where "$\#\text{ of vehicles out of the database}$" means vehicles only appeared in C_2 camera (see Table 4 as an example). Figure 20 shows IA, FPR, and FNR by using different kinds of data in the SVM classifier. We can see that using ICI in the SVM obtains the best IA and lowest FNR. However, the FPR of using ICI is also the highest. Among SCI, ICI, and the combination of SCI and ICI, SCI yields the worst performance in terms of lowest IA and highest FNR. Now, we introduce the following definitions to explain why we use the combination of SCI and ICI in the SVM classifier. The relative identification accuracy (RIA) was defined as $\frac{\text{\# of positive reports}}{\text{\# of acceptance}}$. The relative false positive rate (RFPR) was defined as $\frac{\text{\# of false positive reports}}{\text{\# of acceptance}}$. The relative false negative rate (RFNR) was defined as $\frac{\text{\# of false nagetive reports}}{\text{\# of rejection}}$. Then, in Figure 21, we can see that the one using the combination of SCI and ICI in the SVM classifier yields the best RIA, the lowest RFPR and an acceptable RFNR, which means the combination of SCI and ICI can best leverage RIA, RFPR, and RFNR. Further, the detailed classification performance datum of using SCI, ICI, and the combination of SCI and ICI are listed in Table 2, Table 3 and Table 4.

Table 2. Identification accuracy using SCI with a feature size of 500

SP + SVM classifier output with SCI and ICI			
# of acceptance (# of vehicles in dataset)		# of rejection (# of vehicles out of dataset)	
208 (287)		400(321)	
# of positive	# of false positive	# of negative	# of false negative
134	74	306	94
Identification accuracy	False positive rate	False negative rate	
46.69%	25.7%	29.28%	

Table 3. Identification accuracy using ICI with a feature size of 500

SP + SVM classifier output with SCI and ICI			
# of acceptance (# of vehicles in dataset)		# of rejection (# of vehicles out of dataset)	
277 (287)		331(321)	
# of positive	# of false positive	# of negative	# of false negative
179	98	282	49
Identification accuracy	False positive rate	False negative rate	
62.37%	34.14%	15.26%	

Table 4. Identification accuracy using SCI and ICI with a feature size of 500

SP + SVM classifier output with SCI and ICI			
# of acceptance (# of vehicles in dataset)		# of rejection (# of vehicles out of dataset)	
248 (287)		360(321)	
# of positive	# of false positive	# of negative	# of false negative
166	82	298	62
Identification accuracy	False positive rate	False negative rate	
57.84%	28.57%	19.31%	

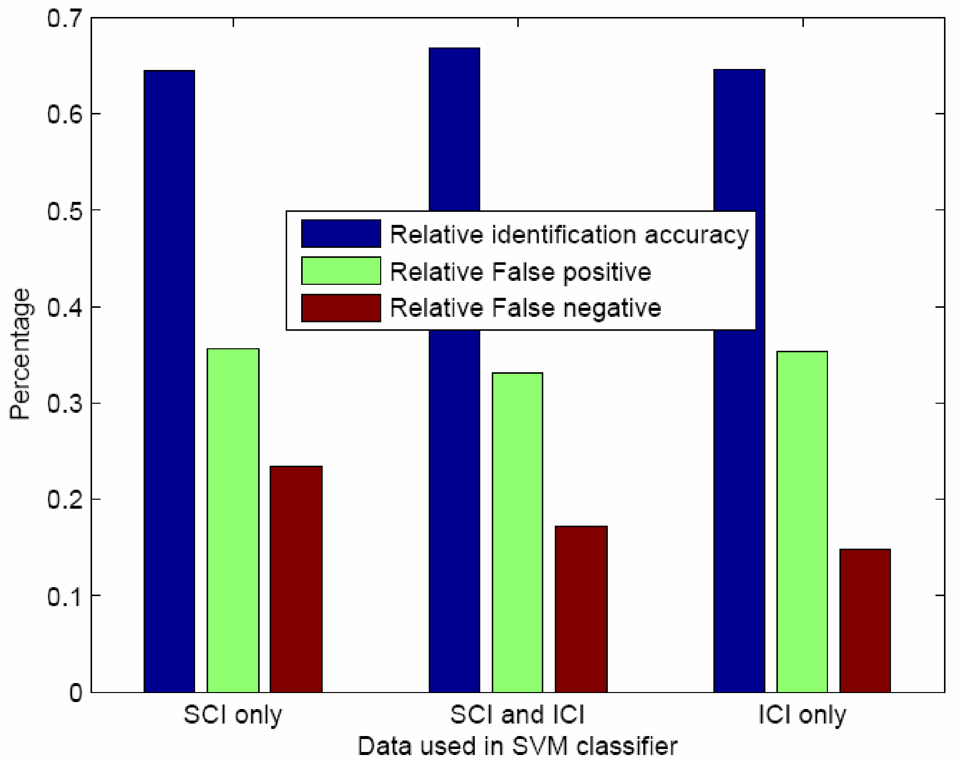


Figure 21. Relative identification accuracy, relative false positive and relative false negative rate by using different kinds of data for the SVM classifier, such as, SCI only, ICI only, and the combination of SCI and ICI

Figure 22. The example of different feature sizes

Table 5. Identification accuracy using SCI and ICI with a feature size of 30

SP + SVM classifier output with SCI and ICI			
# of acceptance (# of vehicles in dataset)		# of rejection (# of vehicles out of dataset)	
207 (287)		401(321)	
# of positive	# of false positive	# of negative	# of false negative
40	167	301	100
Identification accuracy	False positive rate	False negative rate	
13.94%	58.19%	31.15%	

Table 6. Identification accuracy using SCI and ICI with a feature size of 120

SP + SVM classifier output with SCI and ICI			
# of acceptance (# of vehicles in dataset)		# of rejection (# of vehicles out of dataset)	
221 (287)		387(321)	
# of positive	# of false positive	# of negative	# of false negative
115	106	302	85
Identification accuracy	False positive rate	False negative rate	
40.07%	36.93%	26.48%	

4.3.2 Identification with Different Feature Sizes

In this section, we compare IA, FPR, and FNR with different feature sizes of 30, 120, and 500. Here, we implement a down-sampling scheme for each detected vehicle to get the feature image. The advantage of using down-sampled feature image is that each feature image can be generated independently and requires less computation. Figure 22 shows an example of different feature sizes. Furthermore, in Figure 23, we can see the IA increases as the feature size increases, while the FPR and FNR decrease as the feature size increases. Thus, to obtain a higher IA, we need a large feature image, so that enough information of a given vehicle can be provided in the given feature image. Moreover, the detailed datum of IA, FPR, FNR with different feature sizes were listed in the Table 4, Table 5 and Table 6.

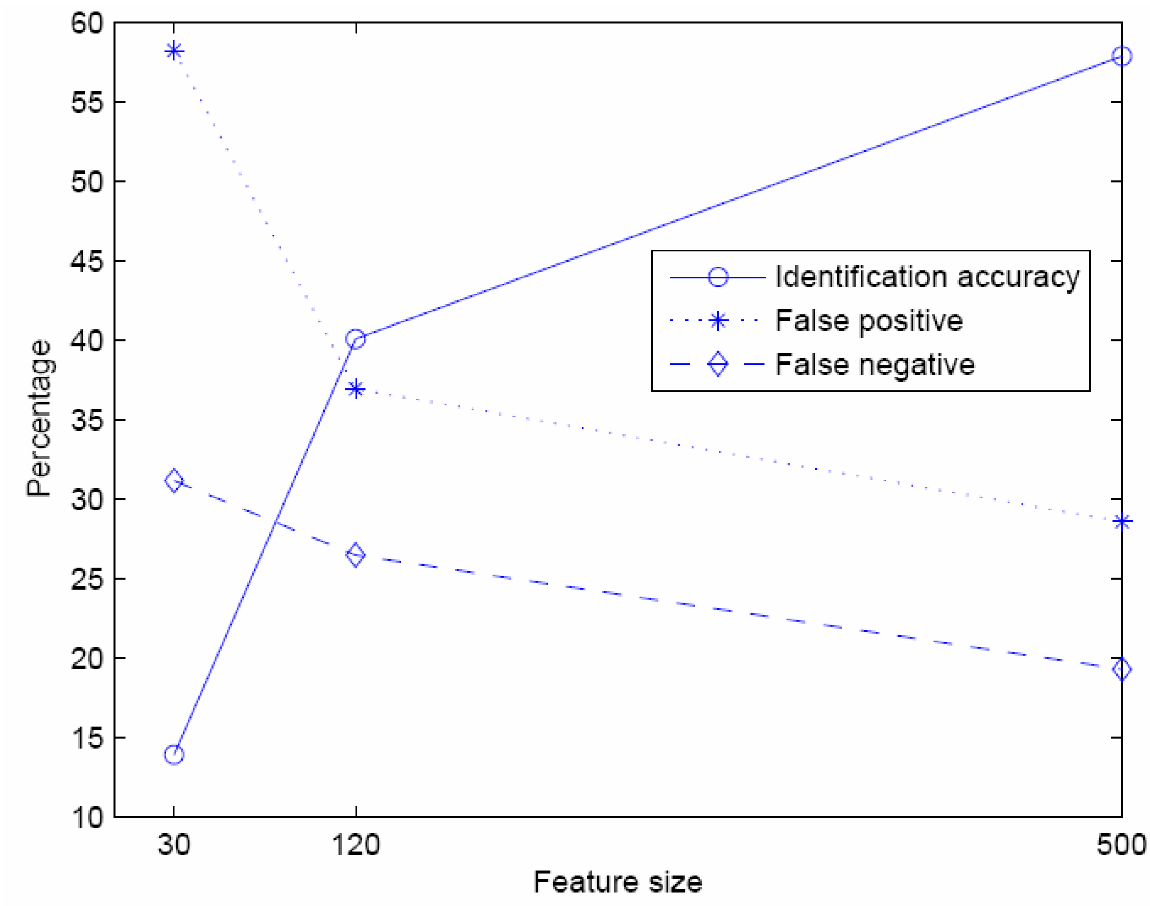


Figure 23. Identification accuracy, false positive and false negative rate with different feature sizes.

4.4 BLUETOOTH AND ALPR AS SIDE INFORMATION FOR IDENTIFICATION

In this section, the results of using side information from bluetooth and ALPR to improve the identification accuracy are presented. We tested the bluetooth and ALPR on highway I-44 in Tulsa with four-lanes and two-directions of traffic. During a one hour time slot, there were a total of 3568 vehicles passing by our detectors, and 698 bluetooth records were captured. Thus we obtain an approximated identification accuracy of using ABR as 19.56%. Moreover, the identification accuracy for ALPR is about 32.63%. The accuracy for the ALPR is rather low. We suspect that it may be due to malfunction of our equipment. Since we already obtained the identification accuracy of the video based method, we can conclude that the overall identification accuracy of our proposed system is about $1-(1-0.5784)(1-0.1956)(1-0.3263)=77.15\%$, which are also listed in Table 7.

CHAPTER 5
COST INFORMATION

The source of matching is listed in the following table. Tulsa Campus Vice Provost and two of the co-PIs (Dr. Sluss and Dr. Verma) had allocated $26,348 for the projects. This had mainly gone to the salary, fringe benefit and tuition remission of one of the two graduate assistants. The remaining was used for material and supplies, travel expense, and the salaries and benefits of the PIs. The School of ECE and the Graduate College had allocated $27,847 and $5,472, respectively. The funding from the School of ECE was spent completely on the salaries and fringe benefits of the PIs. The funding from the Graduate College was used for the tuition remission of the graduate assistants.

	Tulsa Grad College	Dr. Sluss	Dr. Verma (TCOM)	ECE	Grad College	OVPR	Totals
Faculty Salary			$1,000	$20,985			$21,985
Faculty FB			$327	$6,862			$7,189
GRA Salary (1 GRA @$14,400)	$8,460	$5,000	$940				$14,400
GRA FB	$271	$160	$30				$461
GRA Tuition-15%	$1,269	$750	$141		$5,472		$7,632
Travel		$3,000					$3,000
Material & Supplies		$1,090	$3,910				$5,000
IDC on CS						$26,018	$26,018
TOTAL MATCH	$10,000	$10,000	$6,348	$27,847	$5,472	$26,018	$85,685

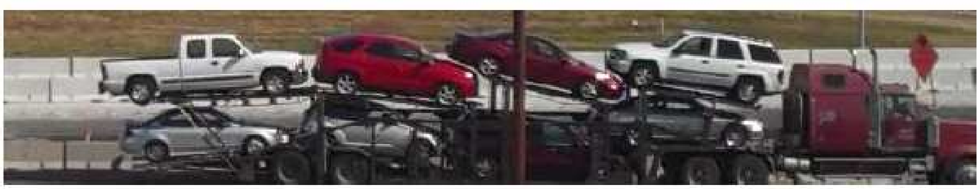
Legend: FB--Fringe Benefit; GRA--Graduate Research Assistant; IDC--Indirect Cost; CS-- Cost Share; TCOM--Telecommunications; ECE--Electrical and Computer Engineering; OVPR--the Office of the Vice President for Research

The funding from OTC had greatly enhanced the project by doubling the available resources. The funding was used to double the time available for the PIs and to hired more graduate assistants to work on the project. The OTC funding was also be used to cover a part of the material cost and travelling expense.

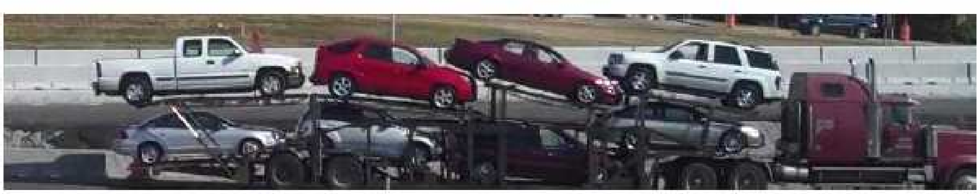
CHAPTER 6
DISCUSSION

6.1 A DETAILED EXAMPLE OF IDENTIFICATION

In this section, a detailed identification example using a sparse representation-based algorithm is presented. Figure 24 and Figure 25 show two correct identification results, where the dataset was built by using the data from camera 1 and the data of the testing vehicle was obtained from camera 2. In Figure 24, although an occlusion (a pole) exists in the image from the dataset, the proposed algorithm can obtain a correct identification result. Moreover, Figure 25 shows the case of a correct identification using misaligned images from camera 1 (dataset) and camera 2 (testing data). The above two cases demonstrate that the proposed algorithm is robust with occlusion and misalignment.



Vehicle from camera 1



Vehicle from camera 2

Figure 24. An example of correct identification I

Vehicle from camera 1

Vehicle from camera 2

Figure 25. An example of correct identification II

Vehicle from camera 1

Vehicle from camera 2

Figure 26. An example of incorrect identification

Figure 26 shows an incorrect identification result, where the two vehicles from camera 1 and camera 2 are too similar to be discriminated by the proposed algorithm. However, almost any video based identification algorithm suffers from this difficulty. Thus, in this case, some additional information, such as the license plate, can be used to increase the identification accuracy.

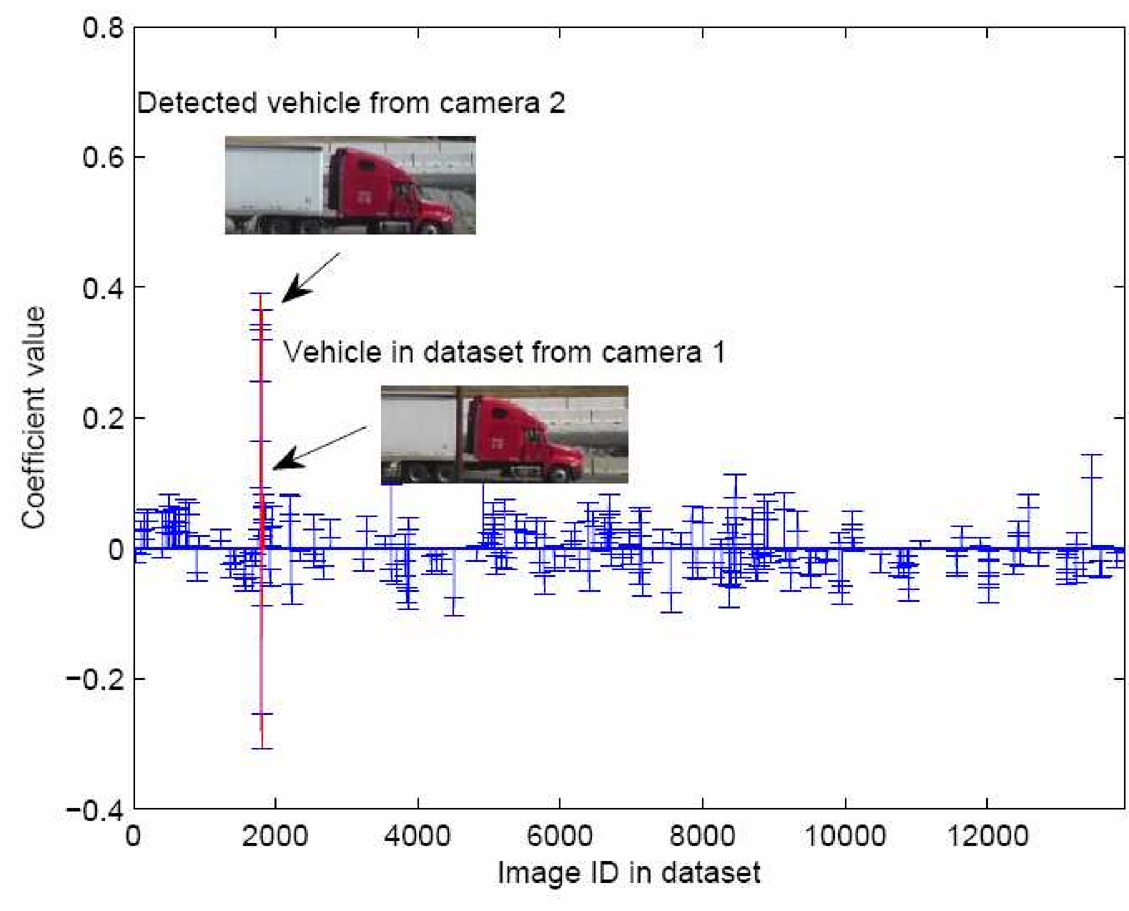


Figure 27. Sparse coefficients for a testing vehicle

Figure 27 and Figure 28 show the sparse coefficients and residuals for a given testing vehicle, respectively. In Figure 27, we can see that the magnitudes of some non-zero coefficients are much larger than others. Moreover, the corresponding image IDs of these large non-zero coefficients belong to the same vehicle in the dataset, which is identical to the testing vehicle. The error bar, another output of RVM, is also shown in Figure 27, which can be used to measure the confidence of each coefficient. In Figure 28, we can see that the residual between a testing vehicle and a vehicle in dataset reach the minimum value, when the testing vehicle and the vehicle in the dataset are identical.

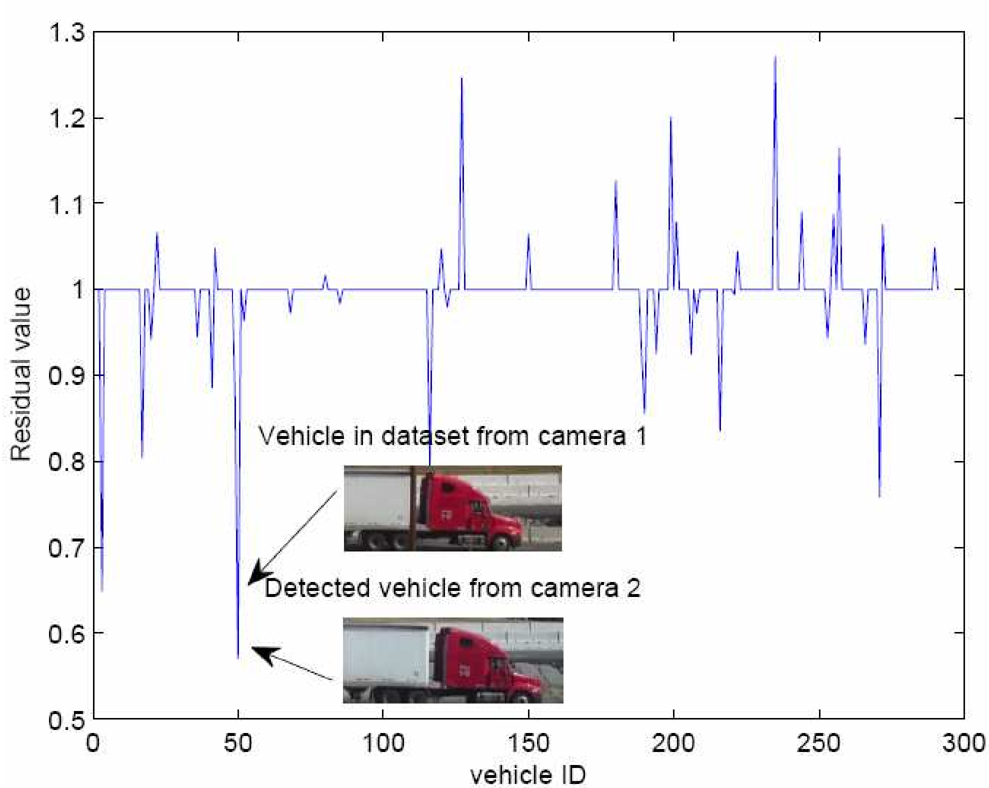


Figure 28. Residuals for a testing vehicle

Table 7. Identification accuracy using video, Bluetooth, and license plate

Video only	Bluetooth only	License plate only	Overall accuracy
57.84%	19.56%	32.63%	77.15%

CHAPTER 7
CONCLUSIONS

In this project, we mainly consider the problem that the major factor influencing the lifetime of the road surface is the commercial vehicle traffic. It is necessary to understand their activities by tracking commercial vehicles' routes and counting their numbers. In this report, we proposed a video based vehicle identification system to track commercial vehicles and their routes through the state. Here we treat the problem of vehicle identification from different video sources as signal reconstruction out of multiple linear regression models, and use rising theories from compressive sensing to solve this problem. By employing a Bayesian formalism to compute the l^1 minimization of the sparse weights, the proposed framework provides new ways to deal with three crucial issues in vehicle identification: *feature extraction*, *online vehicle identification dataset building*, and *robustness to occlusion and misalignment*. For feature extraction, we utilize a simple down-sampled features. This theory also provides a validation scheme to decide if an incoming vehicle has been already included into the dataset. Moreover, unlike PCA or other similar algorithms, using down-sample based features, one can easily introduce features of an incoming vehicle into the vehicle identification dataset without manipulating the existing data in the database. Finally, the Bayesian formalism provides a measure of confidence of each sparse weight. We have conducted experiments to include different type of vehicles on the interstate highways to verify the efficiency and accuracy of our proposed system. The results show that the proposed framework can not only handle the route tracking of commercial vehicles, but it also works well for any kind of vehicles.

BIBLIOGRAPHY

[1] Wright, J. and Yang, A.Y. and Ganesh, A. and Sastry, S.S. and Ma, Y. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, :210--227, 2008.

[2] Mimbela, L.E.Y. and Klein, L.A.et al. Summary of vehicle detection and surveillance technologies used in intelligent transportation systems. *Federal Highway Administration, Intelligent Transportation Systems Joint Program Office*, 2000.

[3] Koerner, R.J. Inductive loop vehicle detector. 1976. US Patent 3,989,932.

[4] Scully, M.O. and Fleischhauer, M. High-sensitivity magnetometer based on index-enhanced media. *Physical review letters*, 69(9):1360--1363, 1992.

[5] Uesaka, M. and Hakuta, K. and Miya, K. and Aoki, K. and Takahashi, A. Eddy-current testing by flexible microloop magnetic sensor array. *IEEE Transactions on Magnetics*, 34(4 Part 2):2287--2297, 1998.

[6] Gardner, M.P. Highway traffic monitoring. *Transportation in the New Millennium, Transportation Research Board--National Research Council, National Academy Press, Washington, DC*, 2000.

[7] Chatigny, J.V. and Thompson, M. and Radice, P.F. and Halvorsen, D.L. Traffic sensor having piezoelectric sensors which distinguish lanes. 1996. US Patent 5,486,820.

[8] Coifman, B. and Beymer, D. and McLauchlan, P. and Malik, J. A real-time computer vision system for vehicle tracking and traffic surveillance. *Transportation Research Part C*, 6(4):271--288, 1998.

[9] Blythe, P. RFID for road tolling, road-use pricing and vehicle access control. *IEE Colloquium on RFID Technology (Ref. No. 1999/123)*, pages

8/1 - 816, 1999. IET.
[10] Friedman, N. and Russell, S. Image segmentation in video sequences: A probabilistic approach. *The Thirteenth Conference on Uncertainty in Artificial Intelligence(UAI), Proceedings*, pages 175--181, 1997.
[11] Koller, D. and Weber, J. and Huang, T. and Malik, J. and Ogasawara, G. and Rao, B. and Russell, S. Towards robust automatic traffic scene analysis in real-time. *ICPR, Israel*, 1:126--131, 1994.
[12] Wren, C.R. and Azarbayejani, A. and Darrell, T. and Pentland, A.P. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):781, 1997.
[13] Stauffer, C. and Grimson, W.E.L. Adaptive background mixture models for real-time tracking. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 246--252, 1999. Citeseer.
[14] Hayman, E. and Eklundh, J.O. Statistical background subtraction for a mobile observer. *Ninth IEEE International Conference on Computer Vision, Proceedings*, pages 67--74, 2003.
[15] Baker, KD and Sullivan, GD. Performance assessment of model-based tracking. *IEEE Workshop on Applications of Computer Vision, Proceedings*, pages 28--35, 1992.
[16] Roller, D. and Daniilidis, K. and Nagel, H.H. Model-based object tracking in monocular image sequences of road traffic scenes. *International Journal of Computer Vision*, 10(3):257--281, 1993.
[17] Karmann, K.P. and von Brandt, A. Moving object recognition using an adaptive background memory. *Time-varying image processing and moving object recognition*, 2:289--296, 1990.
[18] Koller, D. and Weber, J. and Malik, J. Robust multiple car tracking with occlusion reasoning. *Computer Vision ECCV'94*, :189--196, 1994.
[19] Handmann, U. and Kalinke, T. and Tzomakas, C. and Werner, M. and

Seelen, W. An image processing system for driver assistance. *Image and Vision Computing*, 18(5):367--376, 2000.

[20] Wu, J. and Zhang, X. A PCA classifier and its application in vehicle detection. *International Joint Conference on Neural Networks, Proceedings*, pages 600--604, 2001. IEEE.

[21] Matthews, ND and An, PE and Charnley, D. and Harris, CJ. Vehicle detection and recognition in greyscale imagery. *Control Engineering Practice*, 4(4):473--479, 1996.

[22] Sun, Z. and Bebis, G. and Miller, R. On-road vehicle detection using gabor filters and support vector machines. *International Conference on Digital Signal Processing*, pages 1019--1022, 2002. Citeseer.

[23] Viola, P. and Jones, M. Rapid object detection using a boosted cascade of simple features. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 511--518, 2001. Citeseer.

[24] Lowe, D.G. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91--110, 2004.

[25] Ridder, C. and Munkelt, O. and Kirchner, H. Adaptive background estimation and foreground detection using kalman-filtering. *International Conference on recent Advances in Mechatronics, Proceedings*, pages 193--199, 1995. Citeseer.

[26] Martin, F. and Horaud, R. Multiple-camera tracking of rigid objects. *The International Journal of Robotics Research*, 21(2):97, 2002.

[27] Drummond, T. and Cipolla, R. Real-time visual tracking of complex structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):932--946, 2002.

[28] Tan, T. and Sullivan, G. and Baker, K. Pose determination and recognition of vehicles in traffic scenes. *Computer Vision ECCV'94*, :501--506, 1994.

[29] Marchand,E. and Bouthemy, P. and Chaumette, F. A 2D-3D

model-based approach to real-time visual tracking. *Image and Vision Computing*, 19(13):941--955, 2001.
[30] Jurie, Frederic. Tracking objects with a recognition algorithm. *Pattern Recogn. Lett.*, 19(3-4):331--340, 1998.
[31] Senior, A. and Hampapur, A. and Tian, Y.L. and Brown, L. and Pankanti, S. and Bolle, R. Appearance models for occlusion handling. *Image and Vision Computing*, 24(11):1233--1243, 2006.
[32] Ramesh, D.C.V. and Meer, P. Real-time tracking of non-rigid objects using mean shift. *IEEE Conf. Computer Vision and Pattern Recognition, Proc.*, pages 142--149, 2000. Citeseer.
[33] Verri, A. and Poggio, T. Motion field and optical flow: Qualitative properties. *IEEE Transactions on pattern analysis and machine intelligence*, :490--498, 1989.
[34] Tropp, J.A. and Gilbert, A.C. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655--4666, 2007.
[35] Zhao, P. and Yu, B. On model selection consistency of Lasso. *The Journal of Machine Learning Research*, 7:2541--2563, 2006.
[36] Potra, F.A. and Wright, S.J. Interior-point methods. *Journal of Computational and Applied Mathematics*, 124(1-2):281--302, 2000.
[37] Do, T.T. and Gan, L. and Nguyen, N. and Tran, T.D. Sparsity adaptive matching pursuit algorithm for practical compressed sensing. *42nd Asilomar Conference on Signals, Systems and Computers*, pages 581--587, 2008. Citeseer.
[38] Tropp, J.A. and Wright, S.J. Computational methods for sparse solution of linear inverse problems. *Proceedings of the IEEE*, 98(6):948--958, 2010.
[39] Ji, S. and Xue, Y. and Carin, L. Bayesian compressive sensing. *IEEE Transactions on Signal Processing*, 56(6):2346--2356, 2008.

[40] M.E. Tipping. Sparse Bayesian Learning and the Relevance Vector Machine. *Machine learning research*, :211--244, 2001.
[41] Barron, A.R. Entropy and the central limit theorem. *The Annals of probability*, 14(1):336--342, 1986.
[42] Bernado, JM and Smith, A.F.M. Bayesian theory. *John Wiley and Sons, New York. AFM Smith, eds,'Bayesian Statistics*, 4:169--193, 1994.
[43] Figueiredo, M.A.T. Adaptive Sparseness using Jeffreys' Prior. *Advances in neural information processing systems*, 1:697--704, 2002.
[44] Kruschke, J.K. Bayesian data analysis. *Wiley Interdisciplinary Reviews: Cognitive Science*, 1(5):658--676, 2010.
[45] Grosswald, E. The Student t-distribution of any degree of freedom is infinitely divisible. *Probability Theory and Related Fields*, 36(2):103--109, 1976.
[46] Hung, K.M. and Chuang, H.L. and Hsieh, C.T. License plate detection based on expanded Haar wavelet transform. *Fourth International Conference on Fuzzy Systems and Knowledge Discovery(FSKD)*, pages 415--419, 2007.
[47] Anagnostopoulos, C.N.E. and Anagnostopoulos, I.E. and Loumos, V. and Kayafas, E. A license plate-recognition algorithm for intelligent transportation system applications. *IEEE Transactions on Intelligent transportation systems*, 7(3):377--392, 2006.
[48] Dlagnekov, L. and San Diego, UC. License Plate Detection Using AdaBoost. *Computer Science & Engineering Dpt., San Diego, La Jolla.[Online]. Available: http://www. cse. ucsd. edu/classes/fa04/cse252c/projects/louka. pdf*, 2004.
[49] Kim, S. and Kim, D. and Ryu, Y. and Kim, G. A robust license-plate extraction method under complex image conditions. *International Conference on Pattern Recognition*, pages 216--219, 2002.
[50] Jia, W. and Zhang, H. and He, X. and Piccardi, M. Mean shift for

accurate license plate localization. *IEEE Intelligent Transportation Systems. Proceedings*, :566--571, 2005.
[51] Wu, Q. and Zhang, W. and Chen, T. and Kumar, B.V.K.V. Prior-based Vanishing Point Estimation Through Global Perspective Structure Matching. *ICASSP*, :2110--2113, 2010.
[52] Matas, J. and Zimmermann, K. Unconstrained licence plate and text localization and recognition. *IEEE Intelligent Transportation Systems. Proceedings*, :225--230, 2005.
[53] Chang, S.L. and Chen, L.S. and Chung, Y.C. and Chen, S.W. Automatic license plate recognition. *IEEE Transactions on Intelligent Transportation Systems*, 5(1):42--53, 2004.
[54] Deb, K. and Lim, H. and Kang, S.J. and Jo, K.H. An Efficient Method of Vehicle License Plate Detection Based on HSI Color Model and Histogram. *Next-Generation Applied Intelligence*, :66--75, 2009.
[55] Rahman, C.A. and Badawy, W. and Radmanesh, A. A real time vehicle's license plate recognition system. *IEEE Conference on Advanced Video and Signal Based Surveillance, Proceedings. Miami, Florida*, pages 163--166, 2003.
[56] Kahraman, F. and Kurt, B. and Gokmen, M. License plate character segmentation based on the Gabor transform and vector quantization. *Computer and Information Sciences-ISCIS 2003*, :381--388, 2003.
[57] Broumandnia, A. and Fathy, M. Application of pattern recognition for Farsi license plate recognition. *ICGST International Conference on Graphics, Vision and Image Processing*, number 2, pages 25--31, 2005.
[58] Nabaee, M. and Hossein-Zadeh, G.A. License Plate Detection in Complex Scenes based on Improved Gradient and Match Filtering. *IEEE International Conference on Signal Processing and Communications (ICSPC)*, pages 564--567, 2007.
[59] Li, B. and Zeng, Z.Y. and Dong, H.L. and Zeng, X.M. Automatic License

Plate Recognition System. *Applied Mechanics and Materials*, 20:438--444, 2010.

[60] R. T Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, Citeseer, 2000.

[61] Agha, G. Actors: A Model of Concurrent Computation in Distributed Systems, Series in Artificial Intelligence. *MIT Press*, 11(12):12, 1986.

[62] Eric Evans. *Domain-driven design: tackling complexity in the heart of software*. Addison-Wesley, 2004.

[63] Martin Odersky and Philippe Altherr and Vincent Cremet and Burak Emir and Sebastian Maneth, Stéphane Micheloud and Nikolay Mihaylov and Michel Schinz and Erik Stenman and Matthias Zenger. An Overview of the Scala Programming Language. *Technical Report IC/2004/64,EPFL Lausanne, Switzerland*, 2004.

[64] Baudis, P. Git-fast version control system. URL: http://git. or. cz/., Accessed on: Nov 10th 2007.

APPENDIX A

LITERATURE RESEARCH FOR LICENSE PLATE DETECTION

We have also investigated the current research into license plates detection, as a supporting approach to the commercial vehicle identification. Table 8 and Table 9 show the selected license plate detection approaches from our literature review and some of those approaches have been implement for comparisons.

Table 8. License plates (LP) detection approaches I

Approaches	LP Detection Methods	LP Recognition Methods	Overall Rate	Implementation	Drawbacks
Haar Wavelet Transform based [46]	Haar Wavelet features and vertical sub-band edges	NA	Detection rate 97.31%	Haar Wavelet with Gaussian filter and edge statistics, achieved 85% accurate rate, (98.5% detection rate with 13.5% false rate)	License plate size
Sliding Concentric Window [47]	SCW and "Local" irregularity	Probabilistic Neural Network (PNN)	LP segmentation 96.5%, Recognition 89.1% and overall 86.0%	First part SCW	Not size adaptive
Haar like feature with AdaBoost [48]	Harr like feature and AdaBoost training	NA	Detection rate 95.6% (5.7% false positive)	NA	NA
Gradient features [49]	Gradient variance, density of edges and density variance	NA	Detection rate 90%	Feature extractions	NA
Mean Shift [50]	Rectangularity, aspect ratio and edge density	NA	NA	Mean shift	NA
Vanishing points [51]	Pre-knowledge of size and location	NA	NA	Similar to lane detection	Camera setup restrictions
Local extremals [52]	Color intensity	NA	Detection rate 95%	NA	NA

Table 9. License plates (LP) detection approaches II

Approaches	LP Detection Methods	LP Recognition Methods	Overall Rate	Implementation	Drawbacks
Color edge [53]	Color model transform / HIS, fuzzy aggregation	Self-organized Neural Networks (SONN)	Detection rate 97.9%, Recognition rate 95.6% and overall rate 93.7%	Extract color features	Color edges varies
HIS color [54]	HSI color system and color segmentation	NA	Detection rate 92%	Color detection	NA
Real time recognition [55]	NA	Histogram matching (Gray level after segmentation)	Real time	NA	NA
Gabor Transform and Vector Quantization [56]	2D Gabor filter	NA	Detection rate 98%	Gabor features	NA
Multiple interlacing [57]	Multiple Interlacing method	Neural Network	Overall rate 95%	NA	NA
Gradient and match filtering [58]	Gradient features and matching filter	NA	NA	Gradient features	NA
Blob growing [59]	Blob growing	Template matching and Neural network	Overall time 2.41 s (Celeron 850 mHz)	NA	NA

APPENDIX B

SOFTWARE DESIGN RELATED CONCEPTS

This section describes the software part of the project. First some core technologies employed are introduced. Then the technology stack is described. At the end of the section is the discussion of the detailed design of each component.

THE SYSTEM AND ITS COMPONENTS

The whole system consists mainly three parts: the central service, sensing stations and clients.

The central service is the key component of the system. Data collected by the sensing station are sent to the central service for further processing. The algorithm for car identification resides here. If a client desires to query car identification result, they need to query from the central service as well.

Client workstations are terminals which query the identification result from the central service. A GUI desktop application is installed, and end users can perform any query operations through the GUI.

Sensing stations comprise bluetooth sensors, ALPR, and video cameras for collecting vehicle data.

These components are illustrated in Figure 29.

CORE TECHNOLOGIES

RESTful Web Service

The inter-process communication (IPC) between the server, sensors, and clients are through web services. In this section, we briefly introduce the web service technology, and especially the relatively new RESTful-style web service which we adopted for the implementation.

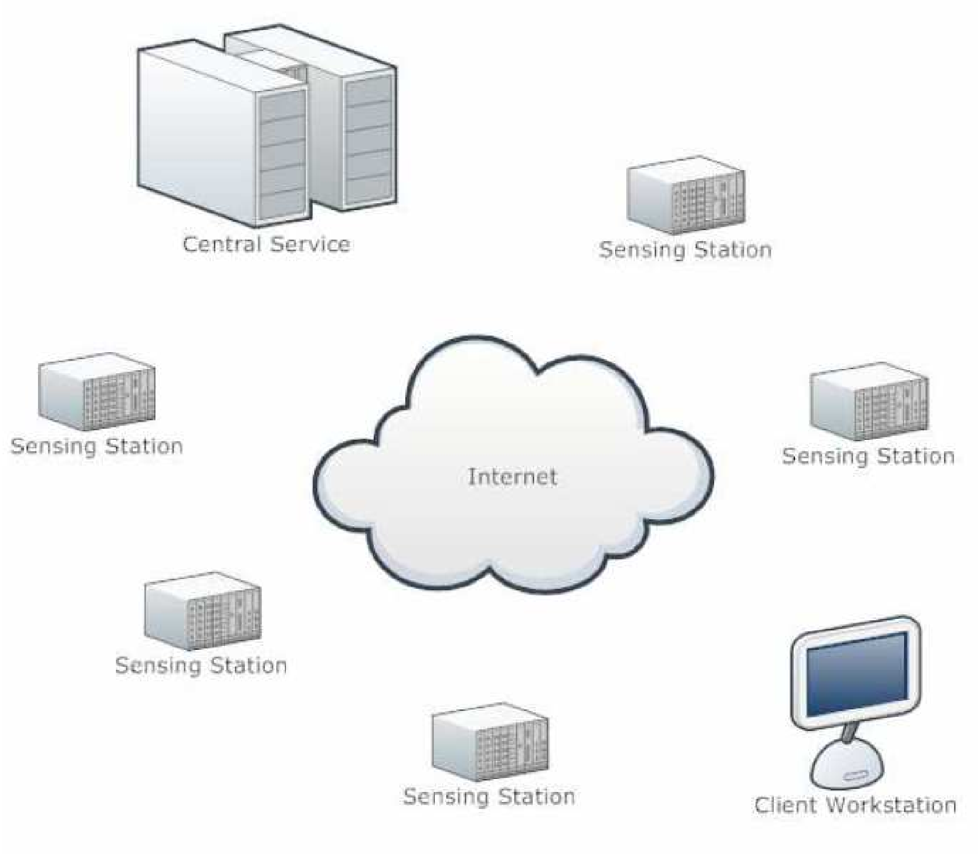


Figure 29. System Architecture

Web Service

Web services are typically application programming interfaces (API) or web APIs that are accessed via Hypertext Transfer Protocol (HTTP) and executed on a remote system hosting the requested services. Web services tend to fall into one of two camps: Big Web Services and RESTful Web Services.

RESTful Web Service

Representational State Transfer (REST) [60] attempts to describe architectures which use HTTP or similar protocols by constraining the interface to a set of well-known, standard operations (like GET, POST, PUT, DELETE for

HTTP). Here, the focus is on interacting with stateful resources, rather than messages or operations.

REST defines a set of architectural principles by which you can design Web services that focus on a system's resources, including how resource states are addressed and transferred over HTTP by a wide range of clients written in different languages. If measured by the number of Web services that use it, REST has emerged in the last few years alone as a predominant Web service design model. In fact, REST has had such a large impact on the Web that it has mostly displaced SOAP- and WSDL-based interface design because it's a considerably simpler style to use.

A REST Web service follows four basic design principles:

- Use HTTP methods explicitly.
- Be stateless.
- Expose directory structure-like URIs.
- Transfer XML, JavaScript Object Notation (JSON), or both.

By these rules, the clarity of the interface of the web services is significantly improved, and development is easier as well.

Actor Model

In each sensing station, A thread consistently reads data stream from video camera and detects vehicles. After the detection, it triggers the station to read license plate number, bluetooth records. These data together with the visual features of the vehicle will be send back to the server. So here we need a concurrency mechanism.

Shared-memory Multiplethreading: there are different models for concurrency. One most popular model is shared-data model. In this model, the threads of a process share the latter's instructions (its code) and its context (the values the various variables have at any given moment). To give an analogy, multiple threads in a process are like multiple cooks reading off the same cook book and following its instructions, not necessarily from the same page.

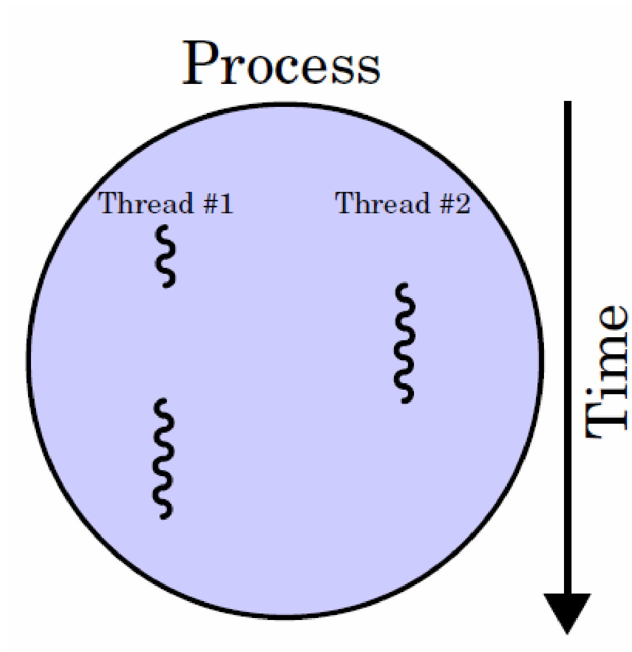


Figure 30. Shared-memory Multiplethreading

This model has been very successful, and it's widely used in many products. But it has its own problem. Since multiple threads sharing the same state, if dealt with inappropriately, the shared data can be corrupted, or the system can run into deadlock.

Actor Model: the Actor model [61] is another mathematical model of concurrent computation that treats "actors" as the universal primitives of concurrent digital computation: in response to a message that it receives, an actor can make local decisions, create more actors, send more messages, and determine how to respond to the next message received. The Actor model originated in 1973. It has been used both as a framework for a theoretical understanding of concurrency, and as the theoretical basis for several practical implementations of concurrent systems.

The Actor model adopts the philosophy that everything is an actor. This is similar to the everything is an object philosophy used by some object-oriented programming languages, but differs in that object-oriented software is typically executed sequentially, while the Actor model is inherently concurrent.

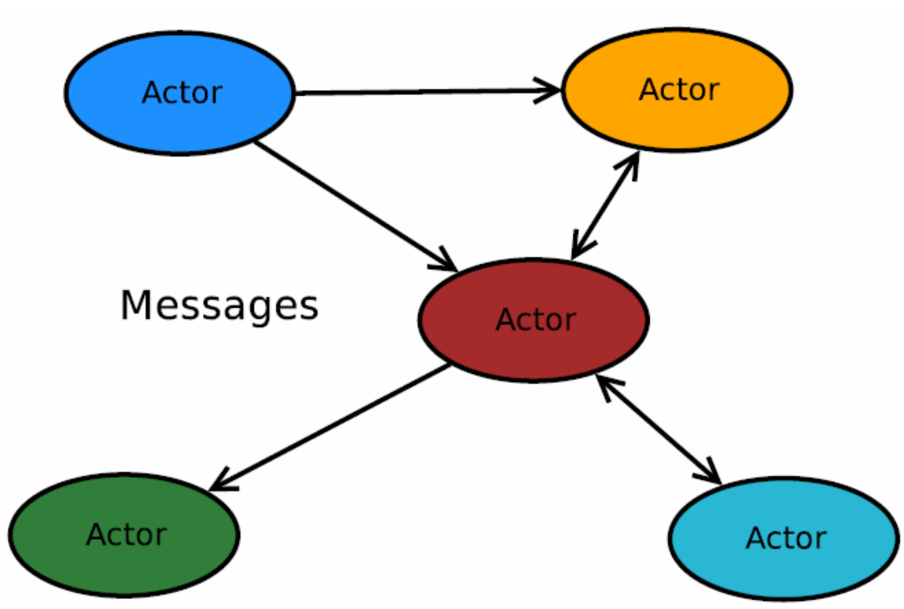


Figure 31. Message Passing

The advantage of the actor model over shared-memory models is that, each actor operates only on its own, private data. Since all data is private, in theory you won't need any locks at all. Without locks, you're obviously immune to problems like deadlock. Without shared data to modify, race conditions are impossible (since no two threads compete over it).

Domain Driven Design: Domain-driven design (DDD) [62] is an approach to developing software for complex needs by deeply connecting the implementation to an evolving model of the core business concepts.[1] The premise of domain-driven design is the following:

- Placing the project's primary focus on the core domain and domain logic
- Basing complex designs on a model
- Initiating a creative collaboration between technical and domain experts to iteratively cut ever closer to the conceptual heart of the problem.

Domain-driven design is not a technology or a methodology. DDD provides a structure of practices and terminology for making design decisions that focus and accelerate software projects dealing with complicated domains.

TECHNOLOGY STACK

Scala as the programming language

Scala [63] is a general purpose programming language designed to express common programming patterns in a concise, elegant, and type-safe way. It smoothly integrates features of object-oriented and functional languages, enabling Java and other programmers to be more productive. Code sizes are typically reduced by a factor of two to three when compared to an equivalent Java application.

Maven as the build tool

Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

ScalaTest as the unit testing library

ScalaTest is designed to facilitate different styles of testing in Scala programming language. ScalaTest provides several traits that you can mix together into whatever combination makes you feel the most productive. For instance, the above example illustrates a Behavior-Driven Development (BDD) style.

GIT as a source code repository

GIT [64] is a distributed version control system. Its highlights include: distributed development, strong support for non-linear development, efficient handling of large projects, and cryptographic authentication of history.

DETAILED DESIGN

Central Service

The central service is the key component of the system. Data collected by the sensing station are all sent to the central service for further processing. The algorithm of car identification resides here. Also if a client wants to query car identification result, it needs to query from the central service as well.

The user interface of the central service is a RESTful web service. With this interface, sensing stations can push data collected to the service. The central service then parses the data, generates domain models, and with the help from repositories saves the data into database. With these data, the car identification algorithm can perform computation and get the result.

The car identification algorithm is behind another RESTful web service. This web service is designed specifically for clients to query the result. The querying event triggers a identification action. As described in the algorithm sections, this event puts a test data into the classification algorithm and returns the result.

Whenever new data is captured by the sensing station, it is pushed back to the central server and added to the training dataset. The algorithm performs learning on this dataset to output a classifier, which is used to do car identification.

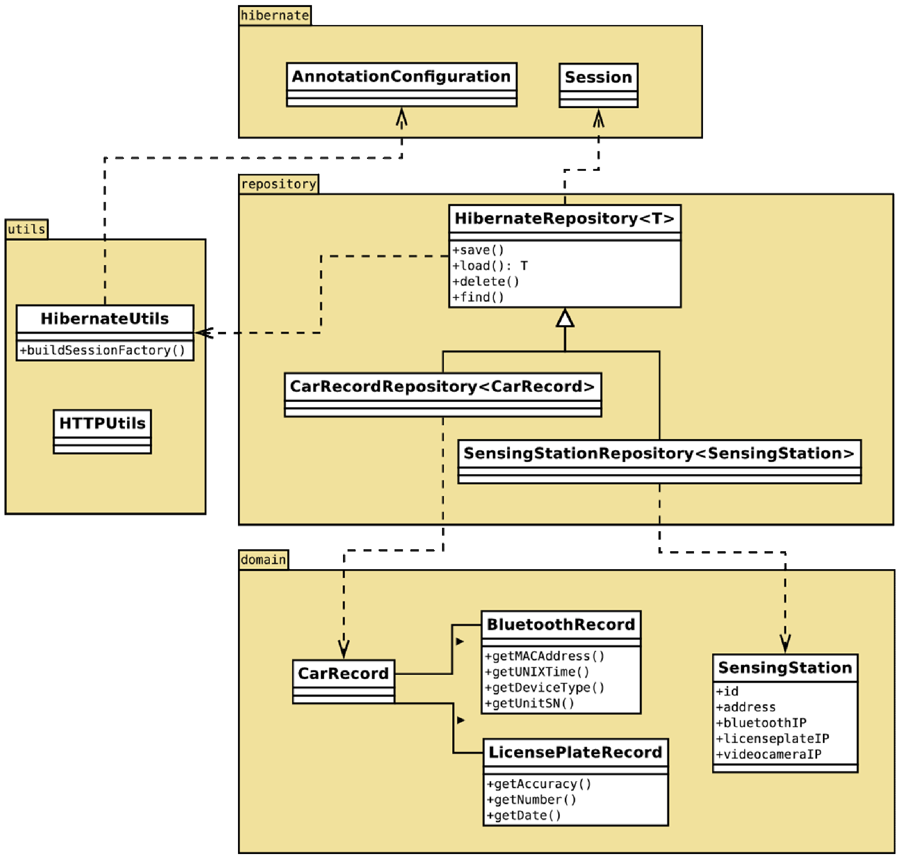


Figure 32. Central Service Domain Class Diagram

Bluetooth Sensor

In the bluetooth sensor, records captured are stored in a mysql database. To expose the data so that other devices and query the captured data, a RESTful web service is installed on the bluetooth sensor.

There are multiple RESTful web service frameworks on Java platform like Restlet, JBoss RESTEasy, Jersey, Apache CXF, NetKernel, Apache Sling, and Restfulie. They all provide complete RESTful web service support, but need a certain amount of work to setup and consumes considerable resource on the device. In the case of bluetooth sensor, the only operation is query, thus these full-blown frameworks are overkilling.

Here we chose simple Java Servlet as the RESTful web service framework.

The servlet captures the incoming URL request, reads get parameters, querie database, and returns the result to the client.

The format of URL is:

http://server:host/servletname?start=ssss&end=eeee

where ssss and eeee specify the range within which records were captured.

After parsing the query string, the servlet uses JDBC to query database and generates the XML in the following format:

```
<records>
    <record>
        <unixTime>***</unixTime>
        <macAddress>***</macAddress>
        <deviceType>***</deviceType>
        <unitSN>***</unitSN>
    </record>
    <record>
        <unixTime>***</unixTime>
        <macAddress>***</macAddress>
        <deviceType>***</deviceType>
        <unitSN>***</unitSN>
    </record>
    <record>
    ...
    </record>
    ...
</records>
```

The servlet pass the XML back to the client.

The class diagram of the RESTful web service is illustrated in Figure 33.

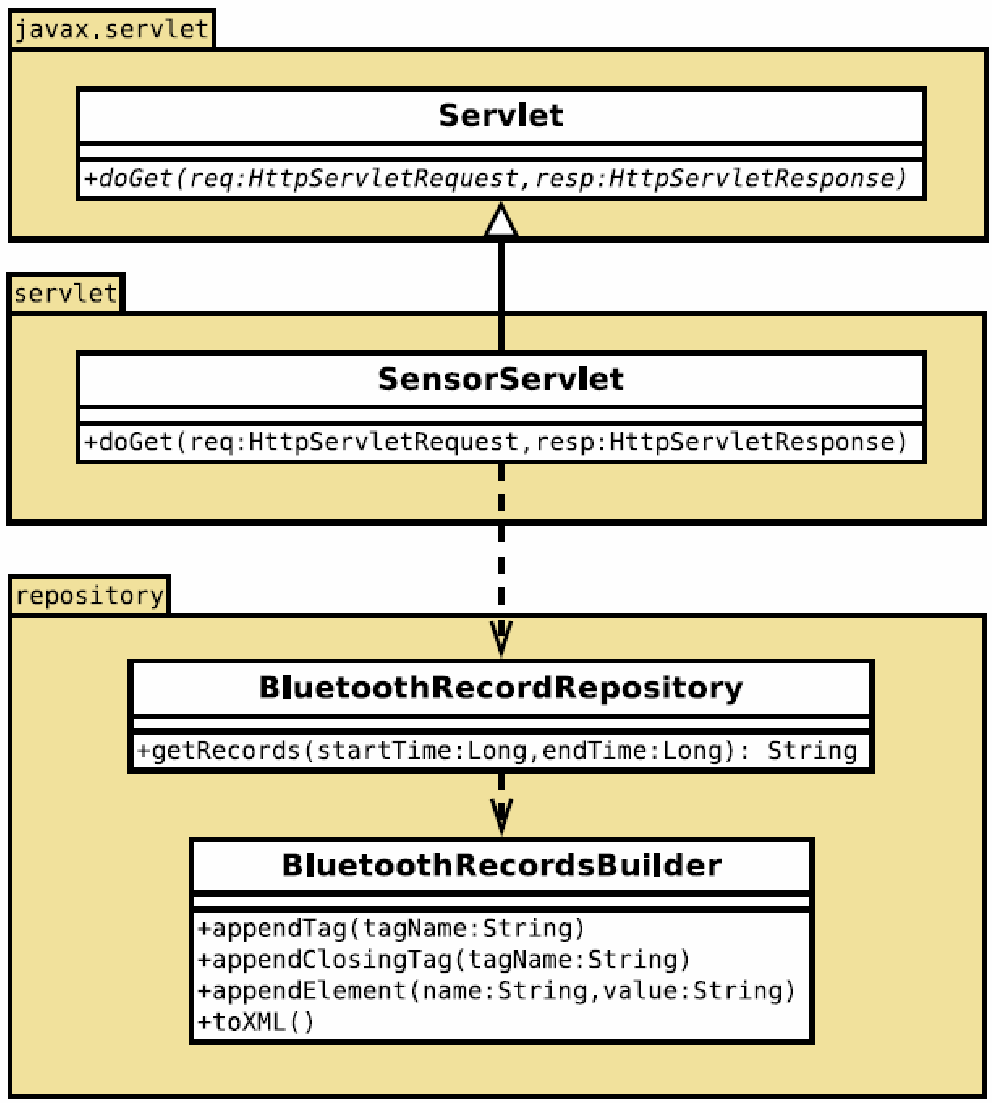


Figure 33. Bluetooth Class Diagram

Clients

Client workstations are terminals which query the identification result from the central service. It might need a GUI desktop application installed and end users can perform any query operations through the GUI. Again it connects to the central service through the RESTful web service. Development of clients are mainly focuses on the user interfaces design and communication with the central service, thus not described in detail here.

Sensing Station

A sensing station comprise a blutooth sensor, a ALPR, and a video camera. The video camera acts like a trigger, which detects the vehicle and triggers read operation to the bluetooth sensor and the ALPR. Since vehicles pass the sensing station in a fast speed, the processing needs to be real-time, so the code reading data stream from the video camera and detecting passing vehicles has to be running on a separate thread.

Here we employ the actor model to do concurrency. The class VideoCameraActor pulls data from the video camera and does analysis. Once a vehicle has been detected, it sends a VehicleIdentifiedMsg to the Mediator, which in turn read data from bluetooth sensor and the ALPR. Putting all these data together, features of the car are captured. These data will be then send to the central server for further processing.

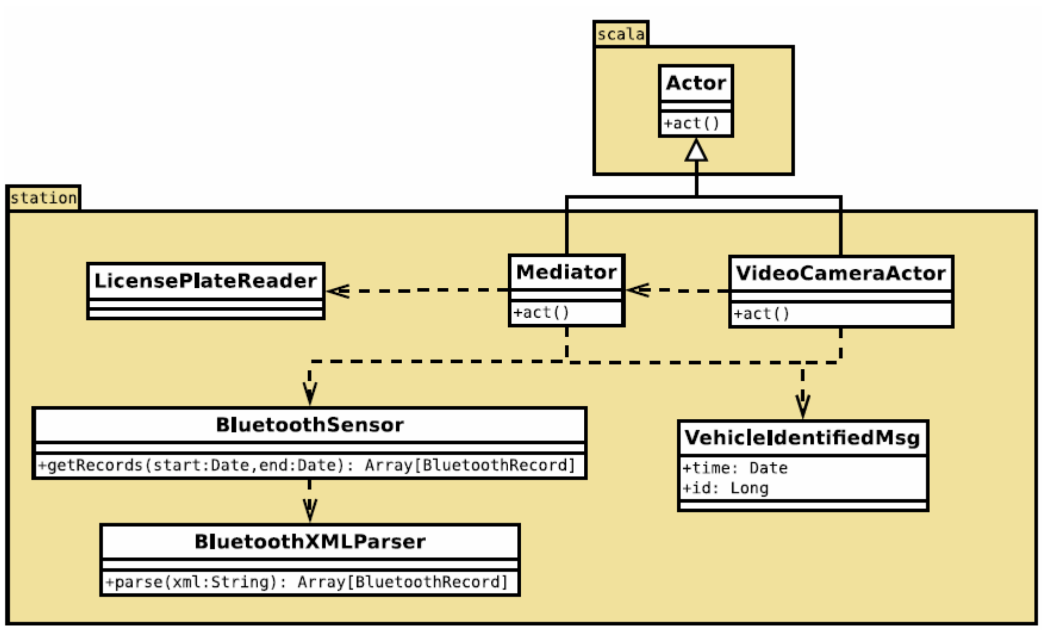


Figure 34. Station Class Diagram

As stated above, the bluetooth sensor sends back data requested as a XML string. To parse the data, class BluetoothXMLParser is implemented. It goes through the XML string, picks up each value and populate the data into a BluetoothRecord object.

The Actor class of the Actor Model framework is from Scala programming language.