DATA FUSED URBAN MOBILITY APPLICATIONS

FOR SMART CITIES

By

Rebekah L. Thompson

Dr. Mina Sartipi
Professor of Computer Science
(Chair)

Dr. Dalei Wu
Professor of Computer Science
(Committee Member)

Dr. Farah Kandah
Professor of Computer Science
(Committee Member)

DATA-FUSED URBAN MOBILITY APPLICATIONS

FOR SMART CITIES

By

Rebekah L. Thompson

A Thesis Submitted to the Faculty of the University of
Tennessee at Chattanooga in Partial
Fulfillment of the Requirements of the Degree
of Master of Science: Computer Science

The University of Tennessee at Chattanooga
Chattanooga, Tennessee

August 2018

ABSTRACT

This thesis presents two applications that can be utilized by drivers, passengers, or pedestrians and allow a wider range of visibility during commutes. The first application uses the concept of see-through technology to assist the driver with a real-time augmented view of a traffic scene that in reality may be blocked by the vehicle in front. The second application presents a mobile application that uses two sources to gather location information, one using absolute location from a Global Positioning System (GPS) enabled device and the other from merging the concepts of computer vision, object detection, and mono-vision depth calculation, and place each instance of an identified object on the mapping application. Currently, mapping items such as stores, acci-dents, and traffic conditions are very common, but this application takes into account the location of individual users to give a holistic view of people instead of places.

ACKNOWLEDGMENTS

To begin, I would like to thank my family for all of their support throughout my life, especially throughout my college career. I want to thank all of the Computer Science faculty for teaching me so much during my time at UTC; I would not be where I am without the knowledge you all have shared. I also want to thank my committee members, Dr. Dalei Wu and Dr. Farah Kandah, for teaching me many topics that helped me understand aspects of the projects I was a part of during my time in the Smart Communications and Analysis Lab (SCAL) and that are presented in this thesis.

None of the projects presented in this thesis would have been possible without the help and support by the other members of the research group I was a part of during the time of these projects. So, a special thank you to Dr. Zhen Hu, Jin Cho, Jose Stovall, Austin Harris, Keith Hollingsworth, and Hector Suarez. I also want to thank our newest members for being there to support me during the time of writing this thesis: Pete Way, Cristina del Real, Jeremy Roland, Katie Rouse, and Dr. Khashayar Kotobi. Also, these projects would not have been possible without the support from Chattanooga's EPB, the city of Chattanooga, the Enterprise Center, Andrew Rodgers, and funding from US Ignite award number (1647161).

Finally, I would like to give a special thank you to the leader of the SCAL research group and my advisor, Dr. Mina Sartipi. She has been an invaluable person in my life, and I am so happy I have been able to learn from her from both an academia standpoint and personal life lessons. She has taught me so much and has played a large part in shaping me into the person I am today, and I cannot thank her enough for her advice and guidance.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation for Thesis Applications

According to a report by the National Safety Council [1], motor vehicle incidents took the life of 40,100 individuals in 2017, an average increase of 5 percent from 2015 and 2016. In addition to these deaths, approximately 4.57 million people required medical attention due to the severity of their injuries, costing approximately $413.8 billion in total. To break down the severity of motor vehicle incidents even further, the National Highway Traffic Safety Administration's National Center for Statistics and Analysis released a report in February 2018 [2] identifying motor vehicle traffic crashes as a leading cause of death in the United States in 2015. Among newborns to age 64, motor vehicle crashes are listed at the top ten leading causes of death; the number one cause for ages 8-24. Motor vehicle crashes are also listed at the top two leading causes of unintentional injury and death across all ages; the number one for ages 4-24.

The statistics shown are hard to ignore, and attempts have been made to decrease these numbers such as automatic braking, backup cameras, forward collision warning, blind spot assis-tance, and lane change assist [3]. Many of these technologies are now available by default on newer vehicle models, but this movement toward advanced safety features should not exclude those who own vehicles without these features. This thesis will discuss two applications using Vehicle-to-In-frastructure (V2I) and Vehicle-to-Vehicle (V2V) communication to help bridge the gap between these two generations of vehicles and improve the safety of individuals on the roadways with or without autonomous driving features. V2I and V2V communication is commonly included in the concept of Vehicle-to-Everything (V2X).

The move toward fully autonomous vehicles has gained momentum within the past decade but has not yet fully reached level 4 full autonomy; Figure 1.1 shows the five levels of autonomy. As the journey continues, new challenges will be faced and new solutions must be found. Currently companies such as Uber, Waymo, and Tesla has reached level 3 autonomy in which there is an autonomous mode, but the driver must take control of the vehicle in an emergency situation or when they choose [4].

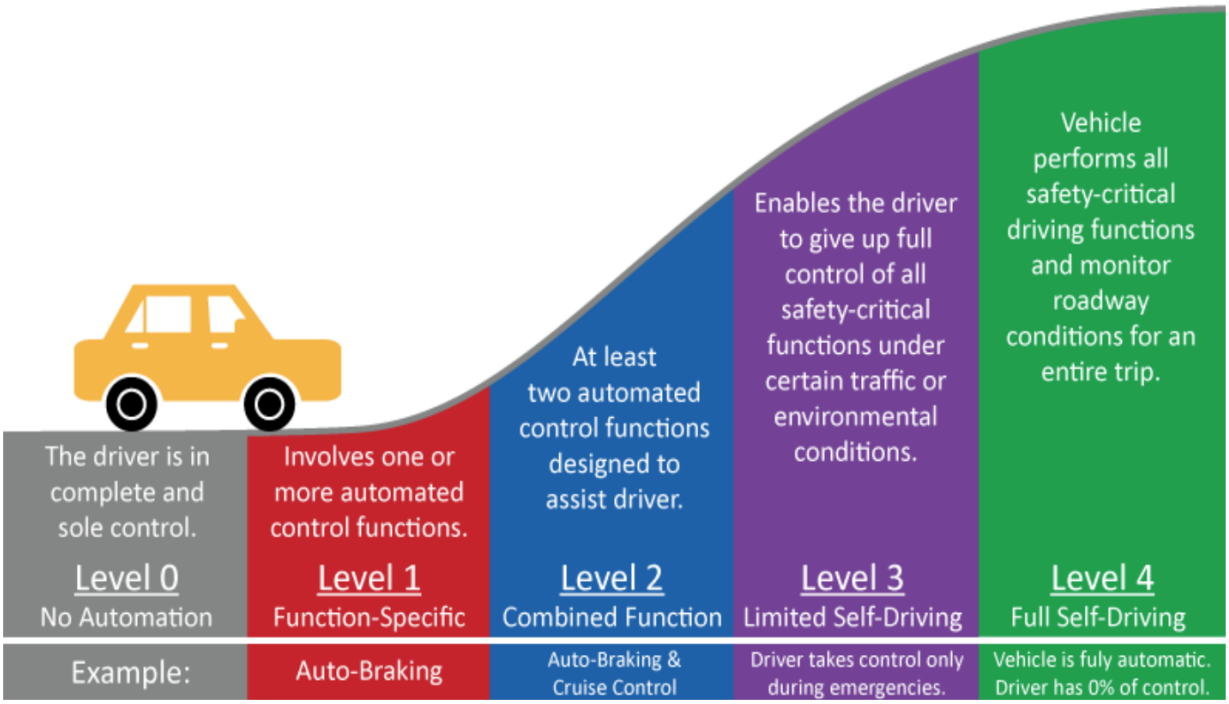


Figure 1.1 Levels of autonomy for vehicles

This thesis is divided into multiple chapters to assist the reader's understanding of the individual components that were involved in this research as well as to be able to see how each topic gradually affects the next. Chapter 2 will describe the testbed used to test our applications. Chapters 3 and 4 will each present one of the applications created during this research period, including real-world use cases for each application in each of the results section. Finally, Chapter 5 will conclude this thesis with my final thoughts about the experiments performed and future research that could be done using these applications or the technology utilized in these applications.

1.2 Background Information

To better understand the content presented in this thesis, this section will explain key concepts used during the development of the applications presented in the next chapters. The primary concepts that will be covered in this section are: computer vision, object recognition, convolutional neural networks (CNN), and You Only Look Once (YOLO).

1.2.1 Computer Vision and Object Recognition

Computer vision is the process of interpreting and extracting information from a still image or video feed to achieve a particular goal. Just as humans use their eyes to obtain information from the world around them, computers can obtain information with the use of a camera and an algorithm. The algorithm is designed to process the image by "looking" at the image and extracting the necessary information to achieve the user's goal [5]. Unfortunately, the computer does not see the objects in an image as separate object forms such as a cat, dog, etc. Instead, the image is converted to greyscale, analyzed, and converted back to a red, blue, and green (RGB) image to more easily distinguish key differences between pixel values of the greyscale image. During the greyscale period, the computer sees the individual pixels that make up the image that range from 0 to 255 depending on the intensity of the pixel (see Figure 1.1). These values are used as building blocks for the computer to compare different sections of an image to find similarities between the input image and the image matrix data stored in its database from previously input images for the training set, which will be discussed further in the next section. By doing this, the computer can form a recognizable image comparable to the image we see as humans. This is the basis of how we are using computer vision as a primary tool for our applications. By using the information gained by the computer vision process, individuals can create algorithms to train the machine to recognize and identify an object, also known as object detection or object recognition, within an image [6–10]. Examples of this identification are shown in Figure 1.2 and Figure 1.3.

Object recognition is the process of identifying objects with the use of pattern recognition and matching algorithms based on unique features in an image or video [11]. A few feature-based matching algorithms include: histogram of oriented gradients (HOG) [12], Haar cascades [13], and canny edge detection [14].

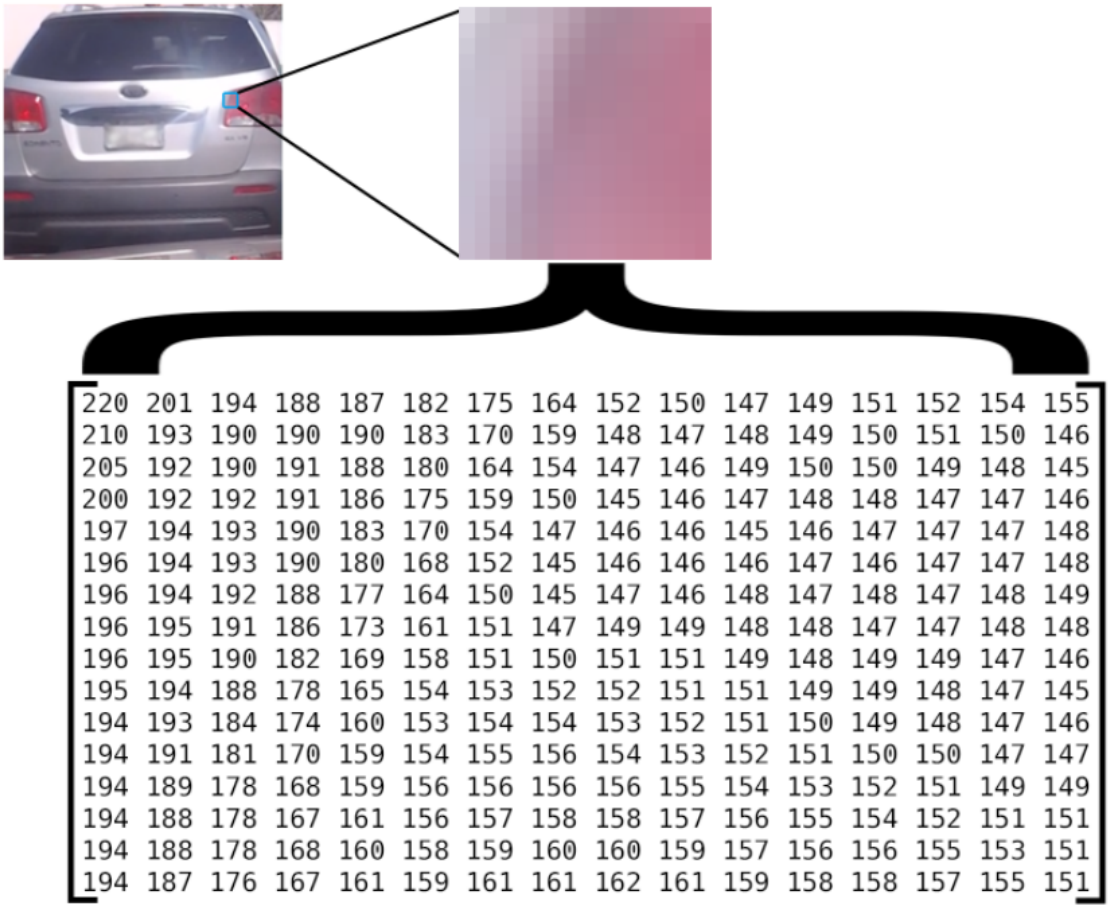


Figure 1.2 Excerpt of a pixel matrix from an image

Computer vision and object recognition areused across many disciplines. In one experiment, Töreyin, et al. created an algorithm to detect fire and flames in videos and real-time [15], Other research groups are using computer vision to detect the use of detecting falsified pharmaceuticals [16], the overall quality of food products [17], and for various applications for facial

recognition [18–20]. The examples listed are only a few instances in which computer vision has been used. In recent years, one of the largest uses of computer vision has been in the area of roadway and urban safety. By using computer vision in combination with an object detection al-gorithm, individuals have been researching how to better detect pedestrians, vehicles, and objects on the roadway to lessen the occurrence of roadway accidents [6, 8]. We have combined these two concepts to create an application to assist in vehicular and pedestrian safety called see-through technology. The next chapter will explain more about our version of see-through technology and how it is different from the version we found during our literature review on the topic.

1.2.2 Convolution Neural Networks

Convolutional neural networks (CNNs) are used as a more automatic feature recognition in comparison to the previously mentioned feature-based matching techniques. CNNs are trained by using hundreds to thousands of images that are input into an algorithm to find the primary object in the foreground of the image. When training the network, images containing the object and images not containing the object are entered into the algorithm. The algorithm then learns to recognize the object of interest in the foreground apart from the natural scene of the background. When the image containing the object is entered, it is entered with a label created by a human telling the computer what the object is. Once the network is trained with hundreds to thousands of labeled images, a test image can be put through the network to test the accuracy of the model.

When the image enters the CNN algorithm, the image appears to the computer as a 2-dimensional (2D) matrix if the image is in greyscale or a 3-dimensional (3D) matrix if the image uses RGB channels. In the previous section on computer vision, Figure 1.2 shows an example of a 2D matrix because the image used was converted to greyscale before it was returned as a RGB image. At the beginning of the identification process, the algorithm will scan through the image section by section and learn key features of the image based on the distinct features of the object, such as edges. By analyzing the image in sections, usually as a matrix, the computer has a better chance of correctly identifying the attributes it is searching for, such as an eye to recognize a face

or a tail for an animal. By increasing the matrix from a 3x3 to a 4x4 to analyze more of the image at once, the process could be sped up, but the accuracy may decrease due overlooking details for the sake of time. Conversely, if the sliding window matrix was reduced to a 1x1 or 2x2 matrix, it could provide more accuracy in identification but cause the process to be longer due to having a smaller analysis area compared to the size of the overall image. Once the model is trained, the model will be able to distinguish the key features of the object from the background based on the value of the pixels in the matrix. When separating the interpretation of the foreground and background, the background will have a lower pixel value, if not a value of zero, and the foreground that contains a recognized image from its model will have a higher pixel value based on the intensity of the grey or RGB value depending on the type of color channel used. At the end of this algorithm, the features of the object will be determined by the largest numbers of pixel values in the matrix. The algorithm will then create probabilities based on the key features from the test image and the images containing an object and label in the model to determine which object label matches the input image the most. The one that has the highest probability will be selected and output as what item is detected in the image [21–24].

(a) Identification from back of the vehicle (b) Identification from side of the vehicle

Figure 1.3 Object detection and recognition using computer vision

For a more human-relatable example of the process of a CNN, consider the following: a person's electricity in their home goes out during the night and they need to find the electrical switch to turn their lights back on. They cannot see, but if they have a light, they can recognize certain objects to identify their location and navigate through their home. Using a flashlight, they shine the light from side to side in front of them to see different features of the room. In one section they see a desk, in the next a chest of drawers, and so on. Even though the light can only shine on a small section, the person is able to recognize and store the features of the room they had seen and identified in a previous section to piece each scene together and understand where they are. A CNN works in a similar way but using more mathematical values and a database of stored images, similar to the human brain storing images to identify objects or scenery at a later date.

Figure 1.3 shows an example of CNNs in action. The figure displays the CNN's ability to identify a vehicle from the rear, which will be the point-of-view primarily used during the identification process for the first application, using a mounted camera on a vehicle's windshield. Figure 1.3 also displays the CNN's ability to identify a vehicle from the side; this viewpoint allows the algorithm to seamlessly continue while the vehicles are making a turn if the driver is making the same turn as the vehicle in front. If the vehicle in front makes a turn while the driver continues straight, or the front vehicle moves out of the camera's view, the algorithm will simply redirect its focus onto the next vehicle matching the parameters of the see-through algorithm that will be explained in the next chapter.

Figure 1.3 also shows the results of the identification process by displaying a bounding box around the identified object. The bounding box placed around the vehicle displays a rough parameter surrounding the vehicle and gives the coordinates to use when a developer needs to manipulate or reference that particular section of the detected object. The bounding box also plays a role visually by giving the user a specific place on the video to focus and find the necessary information instead of overloading their brain by scanning the image to find where they need to focus.

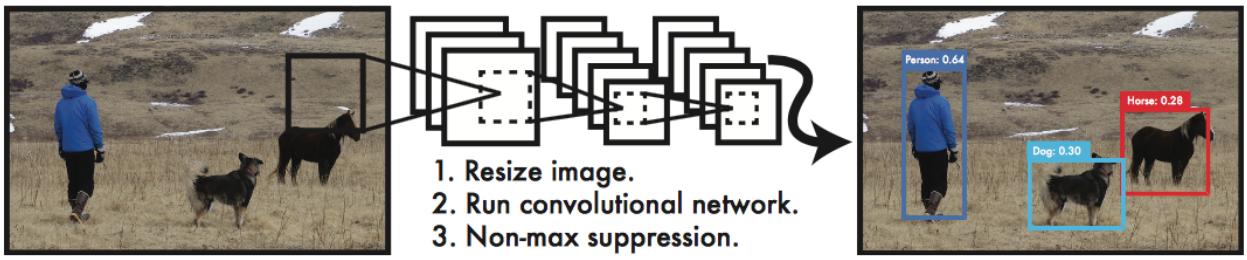


Figure 1.4 CNN YOLO process

1.2.3 You Only Look Once (YOLO)

You Only Look Once (YOLO) is an open source CNN that simultaneously predicts multiple object identifying probabilities while creating bounding boxes around the identified object, as seen in Figure 1.4. YOLO processes and analyzes frames at 45 frames per second (fps) with a relatively high accuracy. The original YOLO network is made up of 24 convolutional layers followed by 2 fully connected layers and trained using the ImageNet 1000-class competition. Though YOLO is a powerful CNN, it has limitations. The creators of YOLO mention in their scientific paper that their model struggles picking up on small objects that appear in groups and generalize objects in new or unusual aspect ratios or configurations [7].

1.2.4 Related Works

Given that this thesis covers two separate projects, related work attributed to each project will be discussed in their respective chapter.

CHAPTER 2

TESTBED

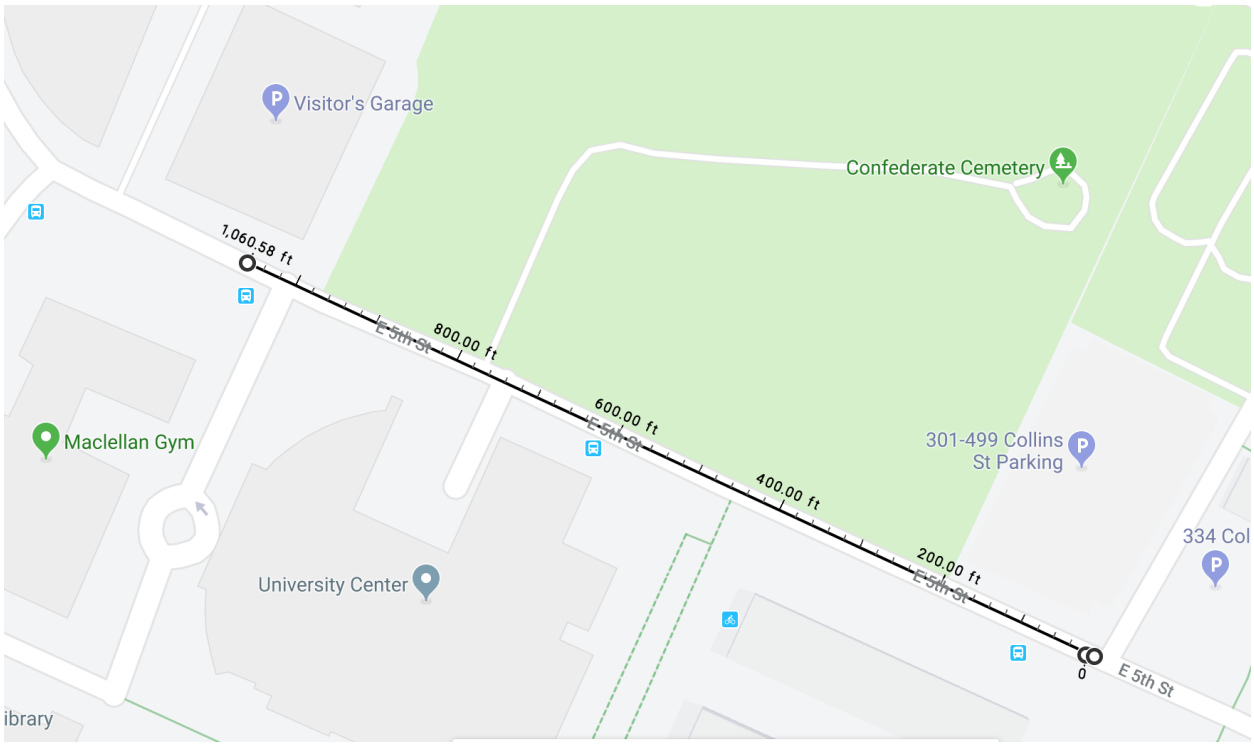


Figure 2.1 UTC's wireless testbed on 5th street from Google Maps

In order to test our applications in a real-world scenario, a testing location was needed. In 2017, a local testbed was created along one of the most densely populated streets at the University of Tennessee at Chattanooga (UTC), 5th Street. This chapter presents the details of the 5th Street testbed, including the hardware that was placed along the street to provide a wireless connection between the clients and the server.

Figure 2.1 shows a Google Maps screenshot of 5th Street at UTC with the testbed area highlighted. The testbed extends approximately 1,060 feet (323 meters) along 5th Street. Along the testbed, wireless access points (AP's) were mounted on five electrical poles. Each of the access points were distributed on the same side of the street and approximately 3.3 meters from the ground. The access points used were able to utilize either a 2.4Ghz or 5Ghz frequency. Each of the access points were placed approximately 120 meters apart, which allowed for an adequate range for communication as well as the ability to overlap and allow a seamless handover to each other AP's during experimentation. Dedicated short range communication (DSRC) was tested during our research to determine whether it would be beneficial to our applications or not. From our findings, we determined that the range was too small for the type of data we were sending and receiving via V2V and V2I. Therefore, we chose to use the IEEE 802.11ac standard 5Ghz frequency to allow for a faster data transfer rate compared to 2.4Ghz that will be needed to ensure a near real-time exchange of data for each client. Each of the AP's are connected to Chattanooga's Electric Power Board (EPB)'s fiber optic internet so that a higher bandwidth and transfer speed can be achieved when transferring the data from the client to the server across components and vice versa.

One feature that was recently added to the testbed was a 1080p infrastructure camera to assist in the localization and mapping of individuals who are not using the mobile version of the mapping application which will be presented later in this paper. To help ensure the privacy of detected individuals, the camera does not record or save any data; it is only for real-time streaming and analysis purposes. A graphical representation of the testbed can be seen in Figure 2.2.

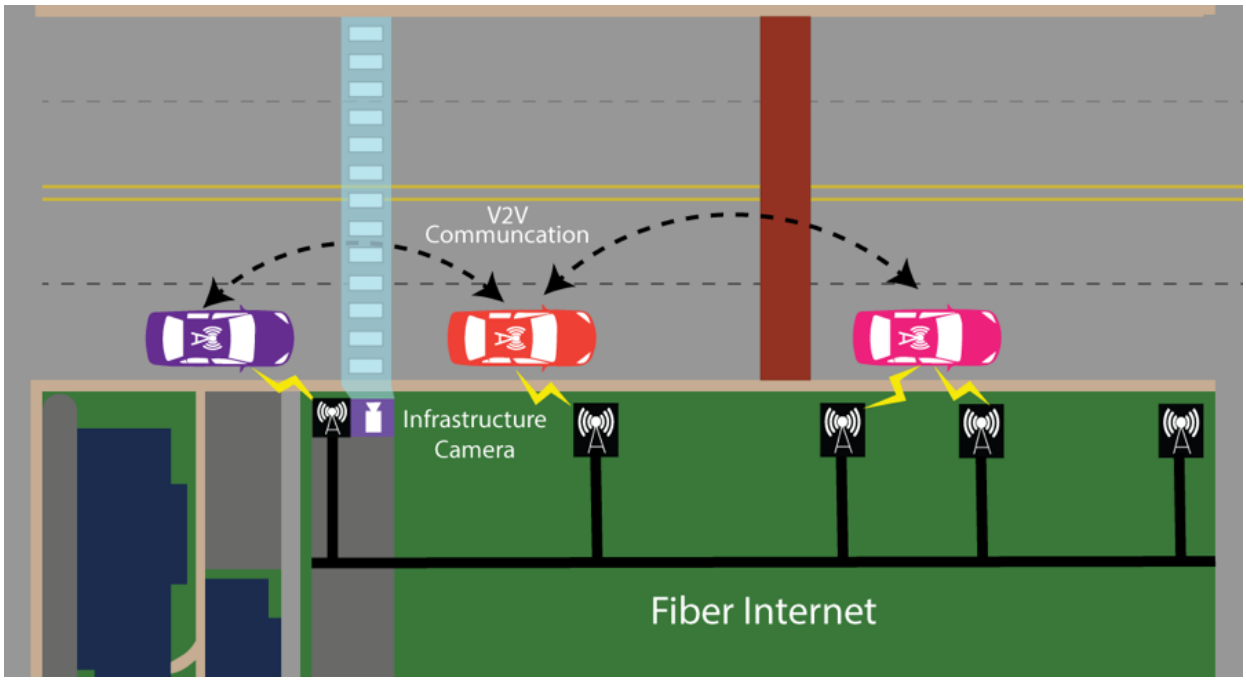


Figure 2.2 Graphical representation of UTC's 5th Street testbed

CHAPTER 3

SEE-THROUGH TECHNOLOGY

3.1 Introduction

The concept of see-through is not new, but previous attempts at making the concept a reality was either theoretical or was accomplished using stereo-vision, such as research from [8, 25, 26, 26]. See-through is the process of overlapping and stitching, or combining, images from one or more cameras onto one or more related images from another camera by transferring the images through V2X communication, which currently includes V2V and V2I. The application pre-sented here is able to achieve see-through using mono-vision, a single camera view, on a moving vehicle compared to stereo-vision. The application itself enables the driver to essentially "see through" the vehicle in front and give an extended view of the driver's immediate environment while proceeding forward in their vehicle. This chapter will present the motivation behind at-tempting the see-through application specifically, the overall setup for the experiments to test this application, results from our experiments, and challenges we faced during the development process.

3.2 Motivation for the See-Through Technology Application

From calls, texts, e-mails, and even mobile games, distracted driving is prevalent on roadways today. One of the simplest solutions to decrease distracted driving and increase driver awareness would be to stop using mobile phones while driving, but in reality, a request such as this seems highly unlikely and difficult to be effectively enforced. In 2015, the National Highway Traffic Safety Administration's National Center for Statistics and Analysis published a

report of statistical analysis related to distracted driving [27]. It was reported that of the 32,166 vehicle crashes that occurred, 3,196 (10 percent) were distraction-affected. Of those 3,196 distracted-based crashes, 442 were documented due to the use of mobile phones. The same document also reports that of the 35,092 fatalities reported, 3,477 (10 percent) were distraction-affected and 476 were documented due to the use of mobile phones.

According to the Centers of Disease Control and Prevention [28], there are three categories of distracted driving: visual, manual, and cognitive. Visual distractions occur during an event where the driver takes their eyes off of the road, manual distractions occur when the driver takes their hand(s) off of the steering wheel, and cognitive distractions when the driver takes their mind off of the actions taking place on the roadway around them. Any one of these distractions can increase the potential for an incident and combining more than one can increase the rate even more. A recent campaign to raise awareness and attempt to decrease distracted driving is the "It Can Wait" campaign commissioned by AT&T [29]. To date, the campaign has had approximately 21 million pledges to stop using smartphones while driving. Another campaign created by the Ad Council and National Highway Traffic and Safety Administration (NHTSA) is "Stop Texts. Stop Wrecks" [30]. As the name states, the main purpose of this campaign is used to promote awareness of the dangers of texting and driving. In relation to the three types of distracted driving listed above, texting is one of the most dangerous because it combines all three of the distracted driving categories. Thinking about the process of texting and driving, the driver takes their eyes off the road and hand off the steering wheel to access the text and begins using their thought process to perform these tasks plus the action of reading and trying to comprehend the message. The texting example combined each of the three driving distraction categories into the 1-2 second of reading the text; it did not include the extra time it would take for the driver to respond to the text. Triggs and Harris [31] tested the reaction times of drivers in different simulations. According to their findings of the 85th percentile reaction time values, the range of driver reaction times were between 1.26 and 3.6 seconds, varying with the type of simulation. Combining the time used to perform a simple task such as texting with a value from the range of reaction times concluded by

Triggs and Harris, the driver could potentially be distracted from their driving environment for at least three to ten seconds. On roadways, this time could result in a collision that could potentially injure the driver or others.

In an effort to improve safety in various situations, such as distracted driving, researchers have produced many publications and applications featuring the concept of computer vision (de-tailed in section 1.2) [7, 8, 26, 32–35]. Being able to incorporate a concept such as this can rev-olutionize the safety software we have today. With the combination of computer vision and the CNNs used during the analysis process, we can gain time and accuracy while identifying objects. Imagine if this combination was utilized in a way to allow an extra two to three seconds of added time between a driver being made aware of an obstacle and having that extra time to react to the situation. The driver may be able to think more clearly about what actions to take to avoid or prepare for the obstacle; this can also create a buffer between a distracted driver and the real-time action they need to take pertaining to the obstacle ahead. One such way to give the driver a larger time span to react to an obstacle is by utilizing the concepts of object detection and computer vision to create a see-through application.

In the next section, the goal and concept of see-through technology using V2X communi-cation is explored in the context of improving driver awareness; one with the use of V2I and V2V communication and one with only V2V communication. The ultimate goal of the experiments was to allow the rear vehicle to be able to "see-through" the middle vehicle (see Figure 3.1) and increase the driver's knowledge of the overall driving environment.

3.3 See-Through Technology Experiments

3.3.1 Vehicular Setup

In this subsection, two experiments pertaining to see-through will be discussed. The first will use V2I communication by utilizing the 5th Street testbed at UTC, as described in

Chapter 2, to send and receive video streams. To achieve see-through during these experiments, three vehicles were aligned linearly along 5th Street so that each were being visually blocked by the vehicle in front. The second will solely be using V2V wireless communication via APs placed within the middle vehicle. A graphical representation of this setup can be found in Figure 3.1. The same vehicular setup was used during the first V2X experiment, excluding the AP located within the middle vehicle.

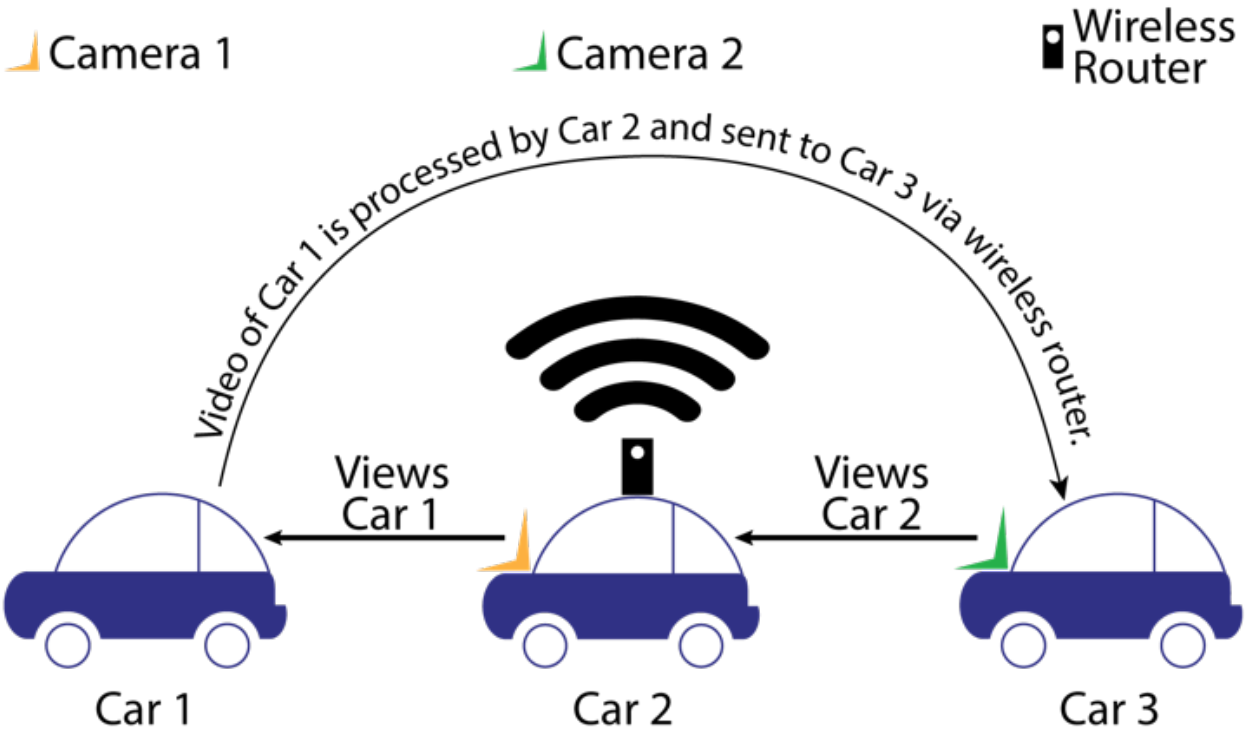


Figure 3.1 Vehicular setup for V2V and V2I see-through experiments

3.3.2 Experiment 1: V2I

While the imaging program was running, each of the cameras captured a continuous series of images and sent them to an offsite server via the connection between the clients and APs along the street. The server then incorporates its graphics processing unit (GPU) for the use of multithreading to analyze each of the image feeds simultaneously to check for any objects matching or related to the image data found in the CNN model. If there is an object found, the program will create a visible bounding box around the object for the driver's visual reference. If the rear vehicle is close enough to the middle vehicle for the see-through to

be activated, the server will overlay the image from the middle vehicle onto the center of the object bounding box detected by the rear vehicle. The overlapping of the images result in a continuous image feed as long as the rear vehicle is behind the vehicle it is connected to via the infrastructure connection. This continuous overlapping allows the rear vehicle to "see-through" the middle car and expand the driver's field of view. Figure 3.2 shows a representation of the image transfer and overlaying process. At a larger scale with multiple vehicles with the capability to use this application, see-through would not activate until it is close enough to connect to the vehicle in front. Ideally, this distance would be approximately two car lengths away to give the driver enough time to acknowledge the new visual without being too close to the vehicle in front to cause an accident if the vehicle in front was to suddenly stop.

Ultimately, with the combination of the wireless communication connected to fiber and the enhancement of the see-through algorithm used, the overall latency was one second or less. Figure 3.3 shows images taken from a recorded video of our experiment from the view of the rear car. These images would be for the use of the driver to broaden their field of vision to better understand their full driving environment.

To minimize the distraction of the see-through visualization presented, the feature itself could be viewed as an optional assist feature using a visual or auditory alert when the driver should be notified. The visualization would not be active throughout the duration of driving time, but it could be activated when the driver is within a set range that was previously mentioned between itself and the vehicle in front. This range can be a manually set distance or activated once the vehicles have been wirelessly connected and ready to receive data from each other. Other than the driver needing to understand how to interpret the image shown, the see-through component would be completely passive and would need no further interaction from the user once activated.

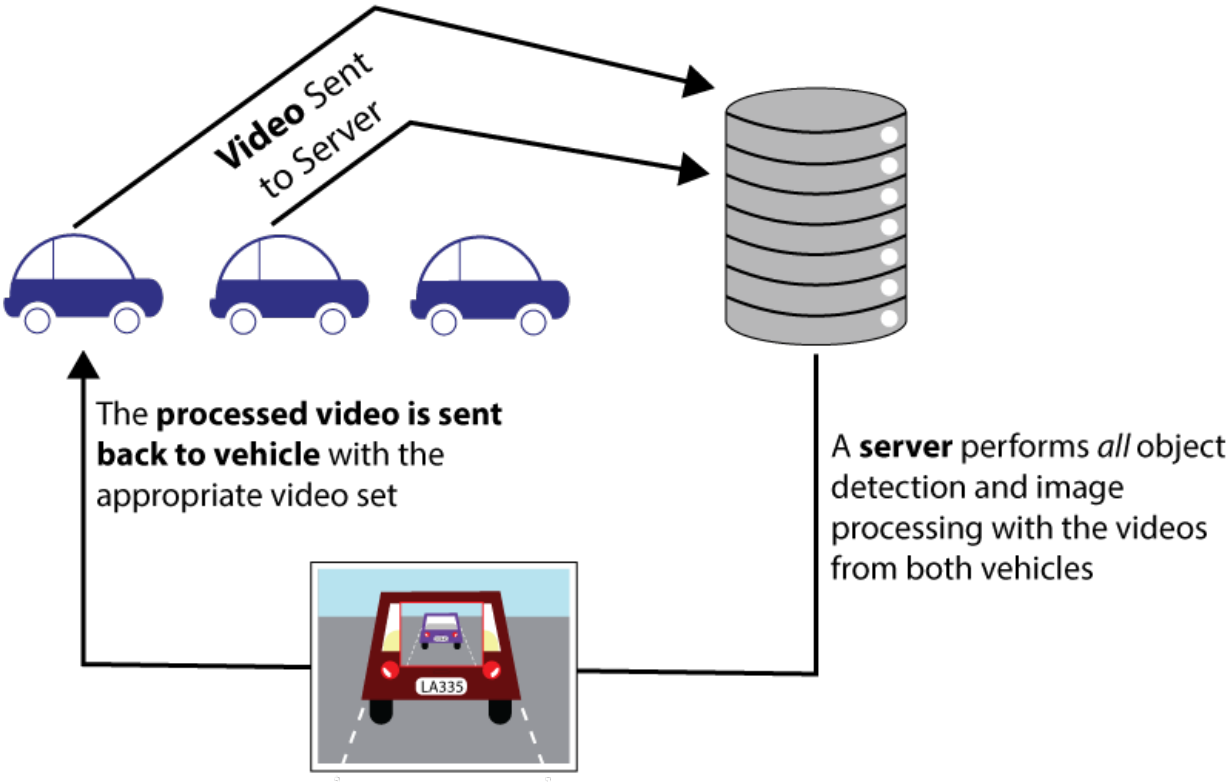


Figure 3.2 Graphical representation of image transfer and processing

3.3.3 Experiment 2: V2V

To achieve the goal of identifying the vehicle in front of the driver, our algorithm consisted of a thirty-two-layer pre-trained CNN called YOLO, explained in further detail in Chapter 1, was used to identify the objects of interest in the images. In the case of this experiment the objects of interest were vehicles. To further explore the concept of see-through technology for the use in roadway safety, the differences in a driver's ability to see an object in their field of vision with and without the see-through algorithm presented was tested. The likelihood of continuously being located in an area with APs connected to fiber is not always plausible. To account for this, the next section describes a second experiment where APs connected to fiber and a group of APs for separate connections are unavailable. In this experiment V2V communication is the focus of the wireless communication system.

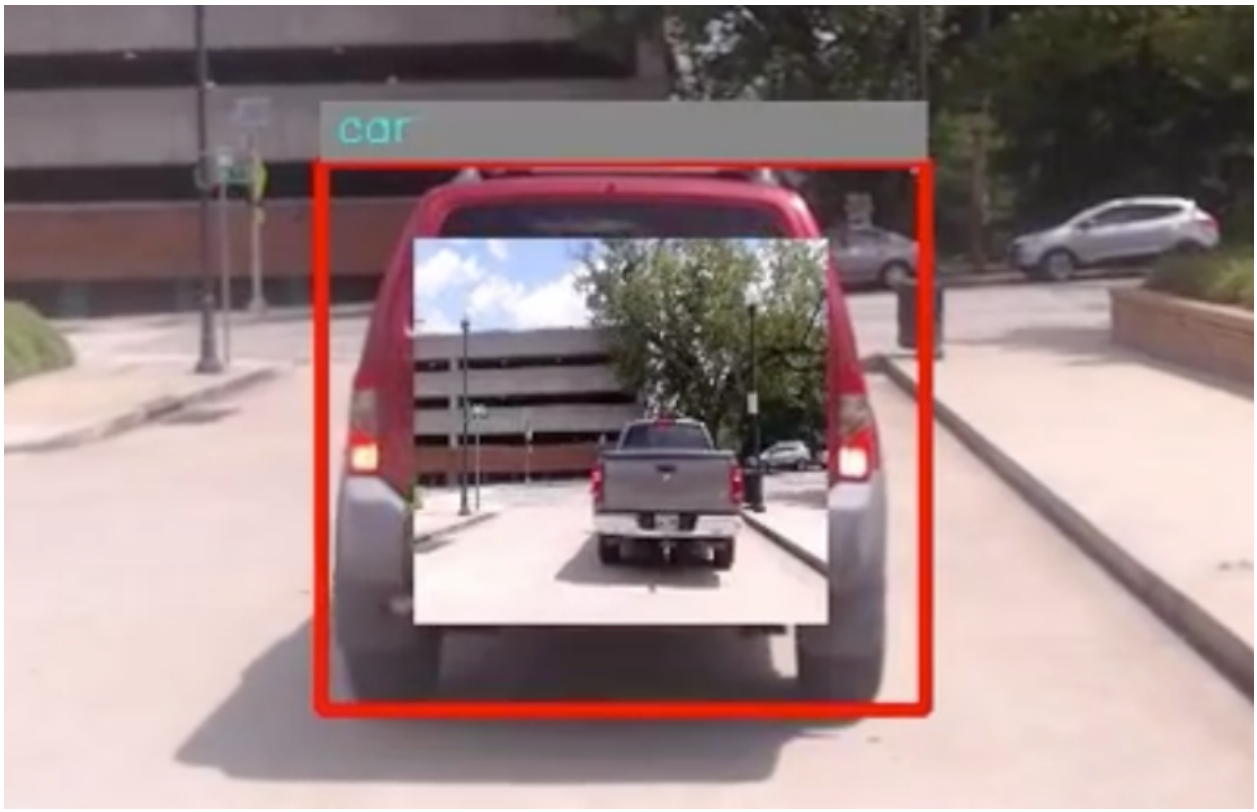


Figure 3.3 Example of see-through algorithm from rear vehicle's perspective

In the event of no V2I access points, how would see-through technology still benefit driver awareness while using V2V communication instead of both V2V and V2I communication? This experiment takes this question into account as well the case of having no offsite server for collecting, processing, and returning the images with a multi-threaded system. The image processing for this experiment was done by the laptop that is receiving the image from the vehicle in front; in this case, the rear vehicle acts as the server and processes the images from the front vehicle. This experiment was performed to examine the potential benefits see-through can give drivers in the case of V2V communication and local computation only.

The vehicular setup for this experiment is similar to that of experiment one, but instead of using three vehicles for the experiment, only two were used. See Figure 3.1 for a representation of the vehicular setup involving three vehicles. A wireless router

was placed on the rear vehicle to allow a local connection by the laptops within the vehicles. The router was set to a 5Ghz frequency to allow improved communication between the vehicles and laptops. Using 5Ghz provided sufficient bandwidth to support continuous transfer of the data in the case of an object blocking the direct path between the router and the laptop from the front vehicle. The see-through algorithm of capturing the image stream from both vehicles, analyzing each of them with YOLO, and sending the overlapped image to the rear vehicle (depicted in Figure 3.2) used was the same algorithm used in experiment one, with the exception of sending the collected images to the offsite server. Instead, a laptop placed in two vehicles were running a custom-written Python script to capture video with the current time based of the local time from the operating systems. After these videos were captured, they were later processed using the same see-through algorithm as experiment one. Compared to experiment one, this process was done without multithreading and also used a less powerful GPU than experiment one. Though the GPU was less powerful, similar results in time differences between the frame in which the rear vehicle can see the obstacle in its frame versus the see-through frame was found. In the next section, we will take a look at the results from each experiment.

3.4 Challenges Faced During the See-Through Experiments

During the development of our see-through algorithm we face a few challenges:

1. The first challenge we faced was a result of YOLO detecting and identifying any object in the frame that did not relate to an object we were interested in visually identifying with a bounding box. Before resolving this, YOLO would create a bounding box around objects we had no use for such as boats, potted plants, etc. instead of only vehicles and pedestrians. To solve this issue, we edited the labels in the source code to only detect objects related to our experiment such as vehicles and pedestrians rather than sofas, televisions, etc.

2. The second challenge we faced was YOLO detecting multiple instances of the identified

objects. In a real-world scenario this could potentially be the appropriate action needed, depending on the application, but for our purposes we only needed to detect the vehicle directly in front of the rear vehicle. To resolve this, we set a margin within the image frame so that YOLO would only detect objects within the set frame. Once the program was running with the new specifications, we were able to successfully detect the single vehicle in front.

3. The third challenge we faced was regarding latency. Prior to this change, we were processing each image that was received from both vehicles one frame at a time; first the middle vehicle's image and then the rear vehicle's image. This caused a large latency time between the video feed that was returned to the rear vehicle and the real-time feed that was being sent to be processed. To resolve this, we implemented a multi-threading aspect to our server. This allowed us to process both vehicles' image feed simultaneously and decreased the latency of the output to under 2 seconds. While the video feeds were being processed, we used time stamps based on the time on the client machine to be sure that the video times were synchronized. If the time stamps were one second older than the current time, they were discarded. By discarding the frames that were older than one second, this assured that we would be getting the appropriate frame for object analysis. Once we implemented this process on 5th Street where fiber was available, the latency and ability to keep the videos synchronized at the analysis time was enhanced even more.

3.5 Results and Discussion

The following results are divided into three sections based on overall data collected through-out the development process and results related to each see-through experiment performed. The first section of results will be pertaining to experiment one. The second section of results will be pertaining to experiment two. The experiments are divided into two separate sections to compare each case individually. In the Discussion section, all scenarios presented in this section will be discussed as a whole. In each results section, the results shown

were implemented and recorded by screen recording software in real-time. The times states in the descriptions are based on the time that particular event appeared in the frame compared to the overall time of the recorded video.

3.5.1 Image Transfer Delays Using YOLO

5GHz 25ft Arrival Propogation	5GHz 25ft YOLO Propogation	5GHz 50ft Arrival Propogation	5GHz 50ft YOLO Propogation	5GHz 25ft Combined	5GHz 50ft Combined
560.618639	1.865148544	530.1229954	1.2114048	562.4837875	531.3344002
516.1170959	1.234054565	507.2193146	1.201868057	517.3511505	508.4211826
514.1203403	1.355886459	516.9949532	1.161813736	515.4762268	518.1567669
558.0322742	1.43289566	523.64254	1.231908798	559.4651699	524.8744488
603.5499573	1.334905624	575.8140087	94.41065788	604.8848629	670.2246666
599.8637676	1.375675201	730.894804	1.184225082	601.2394428	732.0790291
585.2963924	95.05724907	738.9295101	1.220226288	680.3536415	740.1497364
706.7625523	1.263141632	755.0137043	1.194000244	708.0256939	756.2077045
708.5609436	1.168966293	723.4563828	1.177310944	709.7299099	724.6336937
713.259697	1.501083374	701.4462948	107.2494984	714.7607803	808.6957932
706.3038349	1.193523407	833.3408833	1.235485077	707.4973583	834.5763683
674.0033627	1.145601273	818.9330101	1.169204712	675.1489639	820.1022148
671.2448597	1.522541046	816.7173862	1.554489136	672.7674007	818.2718754
657.2704315	1.271247864	603.1386852	1.360416412	658.5416794	604.4991016
651.6201496	101.8424034	598.8967419	1.525878906	753.462553	600.4226208
769.0122128	1.419305801	612.8387451	1.155853271	770.4315186	613.9945984
761.2164021	1.228094101	608.880043	1.19137764	762.4444962	610.0714207
777.4398327	99.46656227	616.9266701	1.16276741	876.906395	618.0894375
855.1638126	1.335382462	614.866972	95.70240974	856.4991951	710.5693817
848.4518528	1.184463501	775.8078575	1.175165176	849.6363163	776.9830227
846.3270664	1.358747482	772.803545	1.220703125	847.6858139	774.0242481
659.2662334	1.276731491	771.6112137	1.227617264	660.5429649	772.8388309
660.9313488	1.20472908	710.1764679	1.172780991	662.1360779	711.3492489
655.7035446	1.207590103	685.8713627	1.150608063	656.9111347	687.0219707
650.5720615	1.190662384	699.1496086	1.197814941	651.7627239	700.3474236
645.9245682	100.7976532	701.8022537	1.282453537	746.7222214	703.0847073
798.6254692	1.192569733	707.4687481	1.189470291	799.8180389	708.6582184
793.5891151	1.153469086	706.0031891	95.76845169	794.7425842	801.7716408
789.5569801	1.174211502	818.7773228	1.1947155	790.7311916	819.9720383
654.3409824	1.206874847	821.1433887	1.295566559	655.5478573	822.4389553

Figure 3.4 Table screenshot of time taken to fully execute the image transfer

During the development process of the see-through application, we performed an analysis of the time to determine the time delay during the image transfer from vehicle to vehicle. To find this time, we used a V2V style connection and sent a series of sequential frames from one vehicle to the other that were sent through the YOLO pipeline to detect objects in the frame. The data set comprised of 450 frames transferred at the 5GHz frequency. To get a better understanding the maximum range at which we could effectively transfer the images, we ran an image transfer test at 25ft (7.62m) and 50ft (15.24m). To record the times, we added a line of code into our program

to begin and end a timer at the beginning and end of the image transfer code and also around the section where YOLO was processing the image. Figure 3.4 shows the times for the first 30 frames with each column labeled based on the times taken. The times shown are the time taken in milliseconds at the arrival time of the frame post-processing to the rear vehicle and the time taken in milliseconds for YOLO to complete its analysis at 25ft and 50ft. The final two columns show the combination of the two times at each distance to provide an overall delay time from the beginning to end of the image transfer process. The average time found from all 450 frames was approximately 696ms at 25ft and 733ms at 50ft (see Table 3.1). A charted result from the overall delay time over all 450 frames can be seen in Figure 3.5, where the x-axis represents the number of frames and the y-axis represents the delay in milliseconds; at both distances, the delay was less than one second.

Table 3.1 Average Time for Image Transfer with YOLO

Distance Between Vehicles	**Average Time for Complete Transfer (ms)**
25 Feet	696.42
50 Feet	733.07

In Figure 3.6, a scenario of a lane blocked by another vehicle is shown. In this case, the lane was blocked by a truck owned by an electrical company. In the recorded video, the event of the truck appearing within the rear vehicle frame was at 7:14 (Figure 3.6 b) whereas with see-through in effect, the driver of the rear vehicle was able to determine something was blocking its lane at 7:11 (Figure 3.6 a); this is a 3 second time difference. If the rear vehicle had been driving too close to the vehicle in front or had see-through not been activated, the driver of the rear vehicle would have had less time to react to the situation and potentially have caused an accident that would have endangered themselves and the workers that were in their line of sight. The extra 3 seconds given to the driver could have been the difference between endangering lives and the ability to make a decision to keep all parties safe.

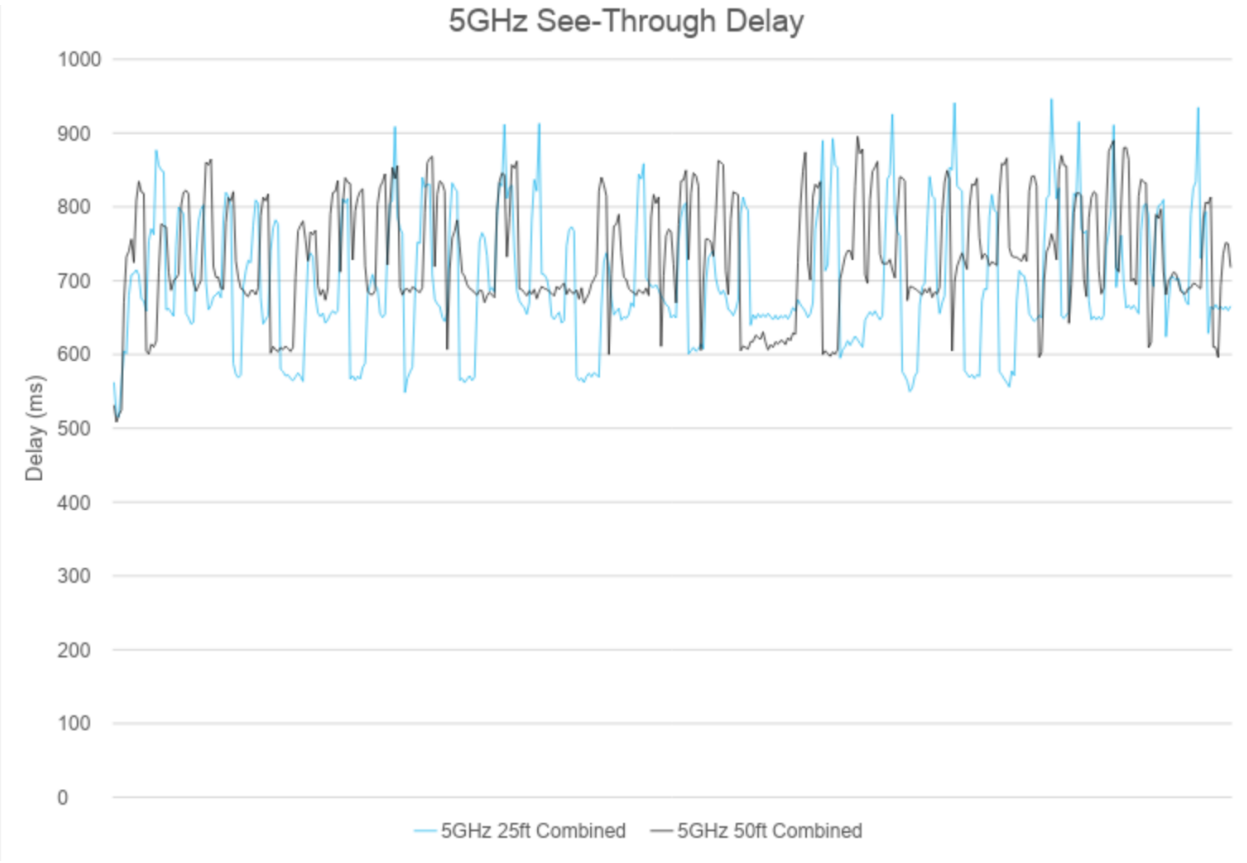


Figure 3.5 Line graph of time taken to fully execute the image transfer

3.5.2 Experiment 1 Results

Blocked Lane

Another observation based on this scenario is that by using see-through not only did the driver of the rear vehicle gain an early view of the electrical workers, they also gained a view of the lane to the left of the truck that would have been blocked by the vehicle in front had see-through not been available. Having this extended view before the truck was within the rear vehicle's viewpoint allowed the rear vehicle to smoothly pass the truck along with the car in front instead of having to stop to make sure no vehicles were driving in the opposite direction toward them before proceeding.

3.5.3 Experiment 2 Results

Road Debris

In Figure 3.7, an object was placed in the center of a parking lot lane. To avoid this object, each of the vehicles would have had to shift to the opposite side of the lane or stop, in the event of a vehicle traveling in the opposite direction in the same lane. In the recorded video, the time at which the object was in the frame of the rear vehicle without see-through was at 0:32 (Figure 3.7 b). The time at which the object was in the frame of the rear vehicle with see-through was at 0:30 (Figure 3.7 a). This gave the driver of the rear vehicle an extra two seconds to decide what action to take to avoid the object. As mentioned in a previous example where an electrical company's truck was blocking the lane, the see-through capability also gave the driver of the rear vehicle an opportunity to see whether another vehicle was driving towards them while the vehicle in front is actively avoiding the object by switching lanes and act accordingly.

Pedestrian Crossing

In many cases of a pedestrian crossing the street, the vehicle furthest away will be able to see the pedestrian before the vehicle in front of them, but what if they are distracted by something? What if they do not see the pedestrian beginning to cross, decide the car in front of them is stopping for no reason, and go around them? That driver has now put a life in danger because of their lack of awareness to their driving environment. By using see-through, the driver in the rear vehicle is able to see a pedestrian crossing the street for the duration of the time the pedestrian is present both in the first vehicle and the rear vehicle's line of sight. This gives them the opportunity to see that there is someone crossing the road regardless of whether they were distracted by a sign, person, or even a text. In the case of the experiment shown in Figure 3.8, a three second difference was recorded from the time the pedestrian entered the field of vision for the see- through image at 1:01 (Figure 3.8 a) of the recorded video and the field of view of the rear vehicle at 1:04 (Figure 3.8 b).

In the results section, each of the scenarios were observed using a real-time implementation of the see-through algorithm presented in this thesis and a screen recorder to film each experiment. In each scenario, a time difference between the point in which the rear vehicle could see the obstacle in its primary frame without see-through versus the time it could see the obstacle in its frame with see-through was stated. Each of the times was based on the time the scenario appeared on the recorded video from the experiment. Table 3.2 shows each of the times individually from each scenario presented and the difference between the two times to show the consistency see- through has with giving the driver of the rear vehicle an early awareness of the obstacle. This excess time gives the driver more opportunity to react to the situation and be able to think more clearly about what actions need to be taken to avoid a collision or accident. Table 3.3 shows the absolute worst, best, and average time difference taken overall from the experiments performed. The time shown is the time difference in seconds that the driver of the rear vehicle was able to see an object in the road and react using see-through compared to not using see-through. Table 3.3 can be interpreted similarly to Table 3.2 with the exception that Table 3.3 encompassing the time differences from all experiments performed.

Being that these experiments are preliminary experiments for the overall see-through concept being developed, the following are important factors to take into consideration relating to the results presented and discussed in this paper:

1. **Vehicle Speed:** The vehicle speed varied depending on the experimental environment. In Experiment One, the speed varied from approximately 15 mph to 25 mph due to the combination of the V2V and V2I communication used. Higher speeds were not tested due to the testbed being on a university campus. In the event of hardware and driving environment becoming available at an off-campus site where speeds can be increased, further research into developing the algorithm for higher speeds can be explored. The vehicle speed for Experiment Two varied from approximately 5 mph to 15 mph due to this experiment using solely V2V communication. In order to obtain a continuous video to extract results, vehicle speed was lowered.

2. **Image Resolution:** The image resolution was scaled to 448x448 pixels to effectively run through the CNN used. This allowed the CNN to analyze the images at the same reso-lution each time to minimize any type of image distortion and decrease the probability of misinterpreting a feature within the image.

3. **Frame Rate:** The web camera used in this experiment captured images at 30 fps. Though this was the frame rate, the algorithm only processed one out of every ten frames taken to reduce latency while analyzing frequently enough to gather important data for a real-time interpretation.

Table 3.2 Times Displayed from Recorded Experiment Videos in Results Section

Scenario	Without See-Through	With See-Through	Time Difference (s)
Blocked Lane	7:14	7:11	3.0 Seconds
Road Debris	0:32	0:30	2.0 Seconds
Pedestrian	1:04	1:01	3.0 Seconds

Table 3.3 Best, Average, and Worst Reaction Time with See-Through

Category	Time (s)
Best Improvement in Reaction Time	1.4 Seconds
Average Improvement in Reaction Time	1.9 Seconds
Worst Improvement in Reaction Time	2.3 Seconds

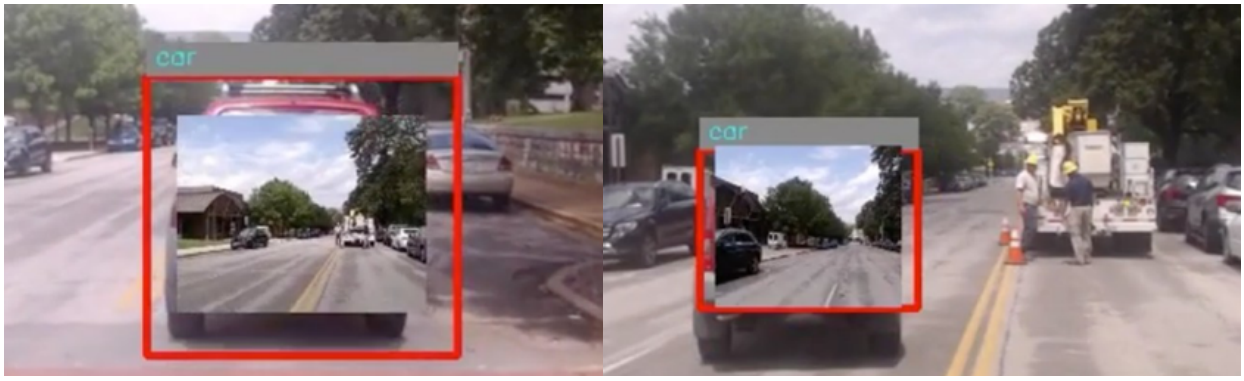
3.6 Conclusions about the See-Through Application

In this chapter we presented two experiments using a single camera on a set of vehicles, a set of access points for one experiment, and an algorithm built upon a convolutional neural network to identify vehicles and pedestrians to expand upon the concept of see-through technology. By using a CNN, the identification of an object in an image feed allowed the ability to create a dynamic output of overlapping images from one vehicle to another to create the illusion of being able to "see-through" the car in front with the driver's own eyes. Not only does this ability increase a driver's view of the roadway environment, but it also allows the driver to

have a higher sense of awareness.

As stated in the beginning of this chapter, distracted drivers made up 10 percent of the roadway crashes in 2015. This Introduction also discussed an example of how 2 seconds can make the difference between life and death: "According to their findings of the 85th percentile reaction time values, the range of driver reaction times were between 1.26 and 3.6 seconds" [31]. Depending on the way that see-through is implemented, it can be used to actively alert with a sound as well as by - at the moment of the alert - making see-through available. With a tool like this to redirect your attention back to driving, this reaction time can be improved by those 2 seconds.

From the results explained in this chapter, see-through technology allows a driver a look into what is to come on the roadway and gives them more time to think and take action. Though see-through is not the complete solution to prevent distracted drivers from avoiding accidents, the ability to give the driver at least two to three extra seconds to be aware of an obstacle in their path could be all the time they need to evaluate their surroundings and take the correct action to prevent endangering their lives or the lives of others.

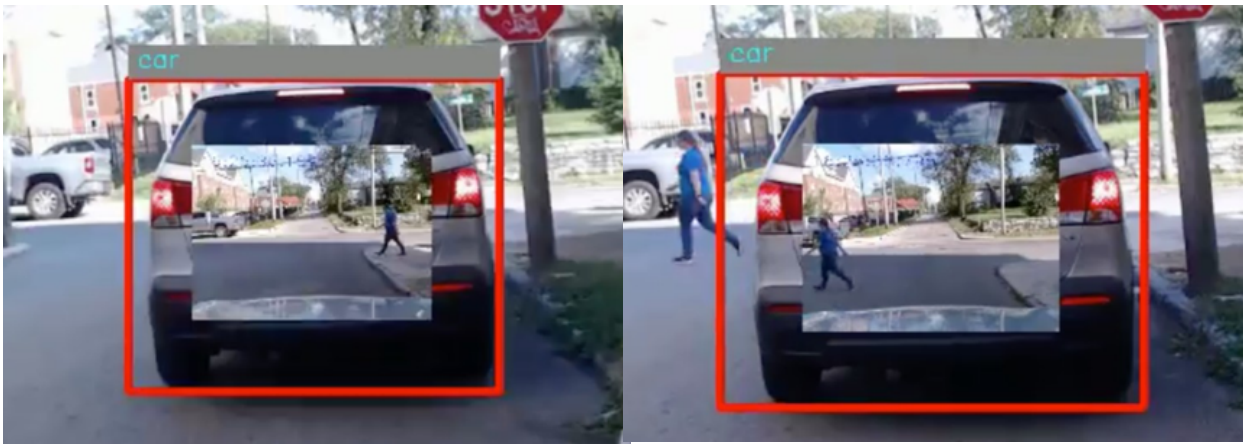


(a) (b)

Figure 3.6 Lane blocked from viewpoint of rear vehicle with see-through

(a) (b)

Figure 3.7 Road debris shown to the rear vehicle via see-through algorithm



(a) (b)

Figure 3.8 Pedestrian crossing street and viewed via the see-through algorithm

CHAPTER 4

ALL-IN-ONE MAPPING APPLICATION (AIO)

4.1 Goal of the All-in-One Urban Mapping Application (AIO)

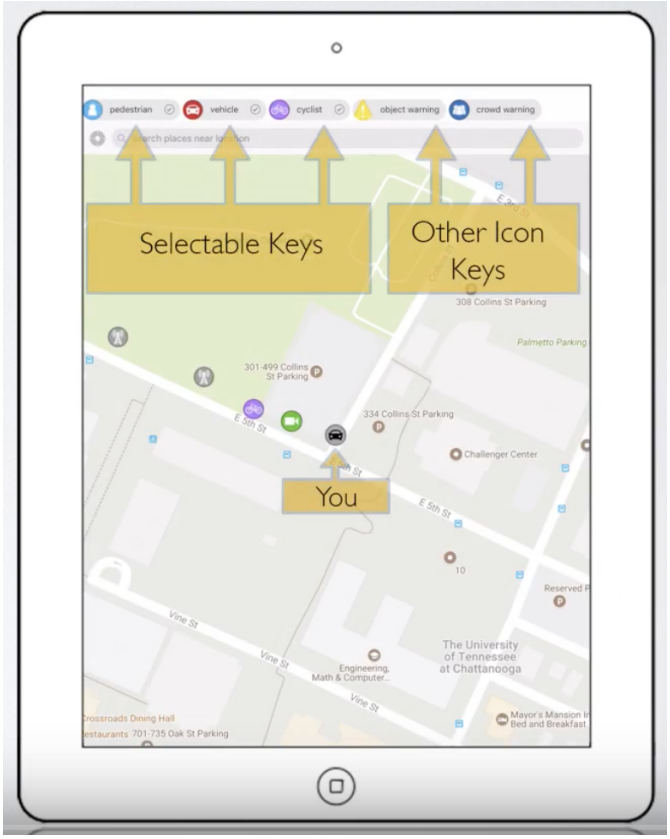


Figure 4.1 Screenshot of the AIO mobile mapping application

The goal of the All-in-One Urban Mobility Map (AIO), shown in Figure 4.1, is to be an application that allows users, pedestrians, or drivers, to see other vehicles, cyclists, and pedestrians within their immediate surroundings. Allowing the users to see what is surrounding them gives the user a better understanding of what to be aware of while traveling to

their destination. This extra awareness allows more time for the user to make decisions during their commute, with the addition to improve their safety in the environment overall.

4.2 Motivation

As urban environments become more prevalent and saturated, the amount of risk a driver takes in city traffic is increasing. A summary of the National Motor Vehicle Crash Causation Sur-vey (NMVCCS) showed that driver-related attributes such as recognition errors, decision errors, etc., caused 94 percent of the NMVCCS crashes [36]. Cooperative sensing and cooperative mobil-ity have the potential to greatly reduce the number of accidents by collecting information about current road conditions and using Inter-Vehicle Communications (IVC), V2I, and Infrastructure–to-Vehicle Communications (I2V) to propagate the information throughout the fleet. To present a holistic view of current road conditions, we cannot rely strictly on a sensor suite that is installed on the vehicle. In other words, camera, radar, and light detection and ranging (LiDAR) typically cannot see beyond a hill or around a curve. Yet, data beyond the sight distance of the vehicle might be available and is being sensed by other vehicles or can be acquired from fixed roadside sensing units. In addition, image and point-cloud data are extremely processing intensive; thus, if we can augment data sensed and processed by the vehicle with data already processed elsewhere, we can improve processing efficiency and create smart reuse of the data.

Many previous explorations of this topic were focused on single data sources, such as the Global Positioning System (GPS) and Global System for Mobile Communications (GSM) coordinates [37] In recent years, the use of many heterogeneous sources at once has been adopted in place of single data sources. One such example is the dubiously named Shotgun Reading via Wi-Fi, a process inspired by DNA sequencing that outputs a directed, weighted graph of the individual devices discovered after a “burst” read of a wide area [38]. Mobile sensor networks have also been implemented, involving the use of a fusion algorithm to measure a scalar field and construct its map to discern occupants of the surveyed area [39]. Another viable

option is multi-sensor data-fusion; fusing recorded data from multiple sensors together with pre-existing knowledge [37]. This chapter explores an implementation which utilizes positioning from multiple sensors, including GPS-enabled mobile devices and wireless cameras, to anonymously track multiple subjects of interest on a unified map interface. The next section will explain the goals of the application presented and methods used, which will cover gaining the information to map the absolute GPS coordinate via mobile devices, the relative GPS coordinate via object detection and infrastructure communication, and steps taken to achieve rerouting algorithms based on obstacles hindering the driver on their typical route.

4.3 Methods

Two forms of detection and identification were used during this project. The first used a single camera to capture and send an image to an off-site server, analyze the image for any useful information, and send that information to a database to be used for adding the relative location of an object to a map updated in real-time. The second used a mobile device, such as an iPad or iPhone, to send an absolute location of an application user to the database and map. Both forms of identification and localization simultaneously appear on the mapping system after being unified in the database. Each of these components will be further explained in this section, beginning with the process of tracking objects using computer vision.

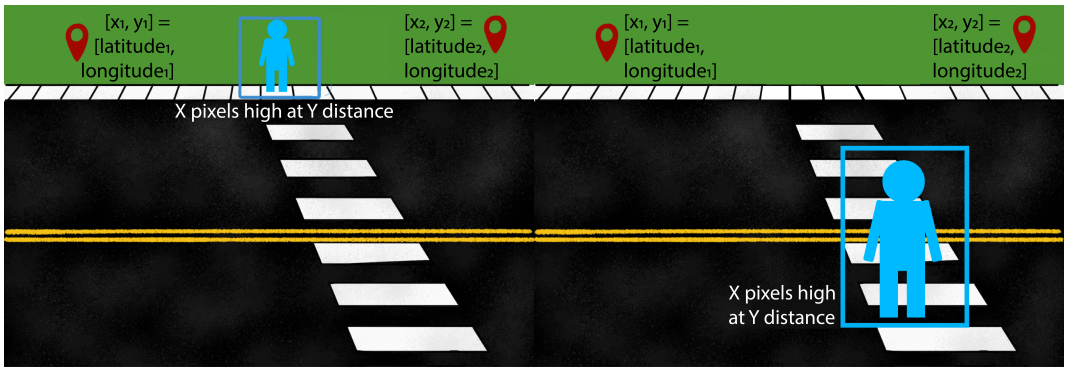
4.3.1 Tracking Objects via Computer Vision

Should absolute location be unavailable via the mobile application, there is an alternative method for localizing objects. This algorithm does not rely on the use of GPS services directly; instead it utilizes a trilateration algorithm which requires three known points accompanied by three known distances based on designated reference points. The following steps are taken to obtain the needed reference points.

1. **Object Detection:** For object detection, YOLO version 2, which can detect 80 different class types, was used [7]. These classes include people, bicycles, cars, motorbikes, airplanes,

and 75 other common objects. Using YOLO, multiple common roadway objects can be detected and tracked, further increasing driver awareness.

2. **Camera Calibration Process:** Obtaining three known points to perform the trilateration algorithm was done through a calibration procedure. During the calibration procedure, the latitude and longitude are used together to determine the geolocation of the camera. Once the geolocation of the camera has been determined, the camera location and the geolocation points from objects in the camera's field of view are used to determine a specific area to the pixel values of the reference points. The pixel values can be approximate due to other portions of the algorithm post-calibration. During the calibration process, a person, ideally of average height, must stand a known distance away in the frame.



(a) Pedestrian detection located distant from the camera

(b) Pedestrian detection located close to the camera

Figure 4.2 Graphical representation of the camera calibration process

At this point in the calibration process, YOLO provides an approximate pixel-height of the detected person where "x" is width and "y" is height. The process then prompts the user to manually provide the distance between the camera and the person to get the location of the camera compared to the person and the two initial points and the object. The x and y of the detected object will change if the object moves closer or away from the camera due to the concept of visual depth. Therefore, if the number of pixels in the detected object's bounding box increases, we can assume the object is moving closer

increases, we can assume the object is moving closer to the camera, and if the pixel numbers decrease, we can assume the object is away from the camera based on the bounding box pixel area from the original calibration (see Figure 4.2).

3. **Obtaining Distance from Three Known Points:** There are three distances which we require: that from each of the two reference points and that from the camera. The distance from the reference points can be approximately calculated using a pixel scalar. This scalar is calculated using a formula which provides the distance in meters between two geo-locations, which can be used to determine the physical distance between the two reference points. Once complete, the physical distance can be used with the pixel distance between the reference points to create a ratio which can approximately convert pixels to meters within the image. Next, the distance in pixels between the bottom-center point of a detected object and each of the known geolocations are calculated. By choosing the bottom-center point, the Y-value of the detected object will not change as a result of size in reference to the camera; it will only change as a result of actual movement.

   The distance from the camera can be calculated by using the bounding box distance and height from the calibration process. Combining these two along with the current height of the bounding box gets us a distance:

$$\frac{DistanceAtCalibration}{CurrentPixelHeight/PixelHeightAtCalibration} \tag{4.1}$$

   The trilateration process must be adapted from typical implementations, as all measurements taken via computer vision are approximate. The algorithm starts by creating a ratio from the actual distance and pixel distance between the reference points. This ratio is then applied to the pixel distance between the subject and both reference points. This gives two approximate radii for the circles needed to be formed. The third circle is formed from the point where the camera is mounted. An approximate distance from the camera can be calculated using data from the calibration process. During this process, the pixel-height of a person standing at a

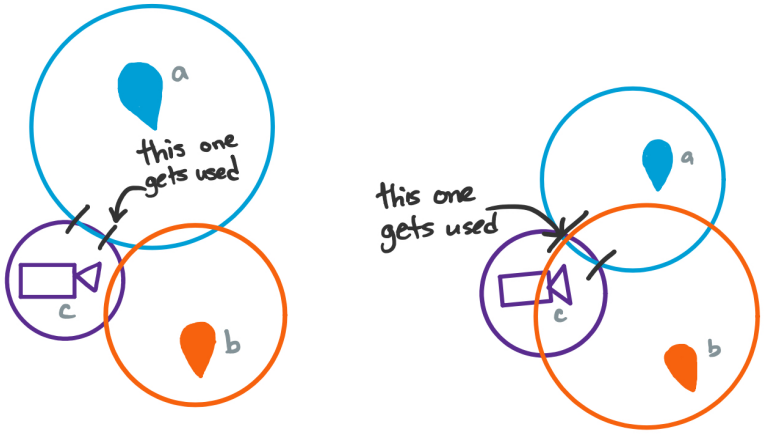


Figure 4.3 Graphical representation of object mapped using trilateration

known distance is recorded, along with the known distance. With this, the new pixel-height can be used to approximate a new distance. The intersections of the three circles can be used to determine approximately where the subject is located in real space.

Using the distance and geolocation, two circles are formed: one at the camera and one at either reference point; the radii of each circle being their respective distance. Next, the two intersections between these circles are found and the intersection closest to the unused reference point is chosen, unless it is smaller; in this case, the further point is chosen (see Figure 4.3). Through this method, it is possible to take the pixel coordinates of any object in an image and convert them to a geo-location that is not relative to the scene.

4.3.2 Mapping Objects

Figure 4.4 Custom icons created for mobile mapping application

Once the objects have been identified and the location has been determined, the mobile application or computer vision algorithm will automatically upload the relative information, such as the identity, latitude, and longitude of the object, to Google's Firebase database. Using the relative information from the devices and existing locations in the database, the database places a custom icon corresponding to the identification of the object onto the Google Maps API used for this project; this process is repeated every half second to update the map. Each of the icons shown in Figure 4.4 represent a different object based on the user's preference on the mobile device and the identity the computer vision algorithm gave the object upon detection. These details were successfully implemented into a mobile-based map as well as a web-based map to show each of the icons moving in real-time along with the user or object detected.

A key feature that needed to be addressed was the ability to delete items from the Firebase database. This was necessary to avoid having objects displayed on the map that have already left the camera's field of view or if a user has closed the application. The solution was simple for the mobile-based communication; if the device stopped sending information for a certain period of time, the database will delete their entry and delete their icon from the map. Regarding deleting the entry from the database in relation to the camera, if the object moves out of the camera's field of view, the database entry related to that object is deleted. In doing this, the stability of tracking of the object and retaining the unique identification information of the object had to be continuously taken into account.

For the mobile front-end map application, choosing Google Maps had several benefits in addition to its multifaceted API. Aside from its availability on several platforms, this choice was advantageous for its ability to utilize custom icons for its markers. The latitude and longitude

stored in the database were used to determine the location of these markers, and their location was updated in a regular interval to correspond to the user's movement. This polling rate from the server on the front-end corresponds with the update rate of the locations from the computer vision algorithm. A main feature that must be addressed when talking about Google Maps is that the map can be viewed via "street view" and "satellite view". Street View allows the user to see a minimal graphical representation of the environment around them while satellite view gives the user more detailed information comparable to what we would see as humans through an aerial view. Though satellite view gives more information, we felt that because of the level of detail displayed it would distract the user more that the street view; the level of detail given on satellite view was not necessary for the stage of the experiments presented in this thesis.

In addition to the mobile front-end, a web interface can be made for testing and performance benchmarking. The Google Maps and Google Firebase APIs both offer JavaScript implementations, which aid in the creation of a rudimentary map which can track the same subjects the mobile front-end can see. This JavaScript-based front-end uses the same polling rate from the database as the mobile front-end and the computer vision algorithm.

4.3.3 Tracking Object via GPS-Enabled Devices

Cellular GPS

The second form of object identification was done by utilizing the user's mobile device. The All-in-One (AIO) mobile mapping application was downloaded onto a user's iOS device. The AIO mapping application was built using the Ionic mobile development platform [40] and deployed and tested on iOS devices. The application used the native Google Maps SDK to visualize information on the surrounding environment and send and retrieve object locations in real-time using Google's Firebase database. The application utilized a Cordova plugin to provide

Firebase database. The application utilized a Cordova plugin to provide current GPS information collected from a combination of network signals such as the IP address, Wi-Fi, Bluetooth MAC addresses, RFID, and GSM/CDMA cell IDs [41]. The user's device geo-coordinates were updated every 250 milliseconds while the user had the application open and active.

Mobile Application Anonymity and Mobile Type Classification

The mapping application required iOS privacy control permissions to be granted by the user before it could access its location. The application did not have the ability to monitor a user's location while inactive on the mobile device. Privacy and anonymity concerns were prioritized, and no self-identifying ID, service provider, or name information was associated to any of the participating device's geo-coordinates stored in the database. Random IDs were generated by the application and used to identify entries on the Firebase server. A user could choose to identify their type of transportation as pedestrian, vehicle, or cyclist. A user who chose not to identify their transportation type was treated as a pedestrian. The description of the user was associated with the device's geo-coordinates and sent back to the cloud database. The transportation's type was then used to determine the type of icon that would be used to visualize the coordinates on the mobile map.

4.4 Results

4.4.1 Object Tracking

Combing the components discussed in the methods section provides an integrated solution for congregating data gathered. During the testing and data gathering process, it was discovered that the AIO application icons created a distraction caused by the continuous updating of the icons' locations. This distraction was in the form of "flickering' caused by removing and replacing all of the icons on the map during each iteration. To minimize this distraction, existing markers were moved and shifted in a way that corresponds with the

subject's actual movement. This helped prevent the icons from"flickering" and decreased the possibility of distraction from updating the relative positions of the objects.

Figure 4.5 Infrastructure camera maps information in real-time on application

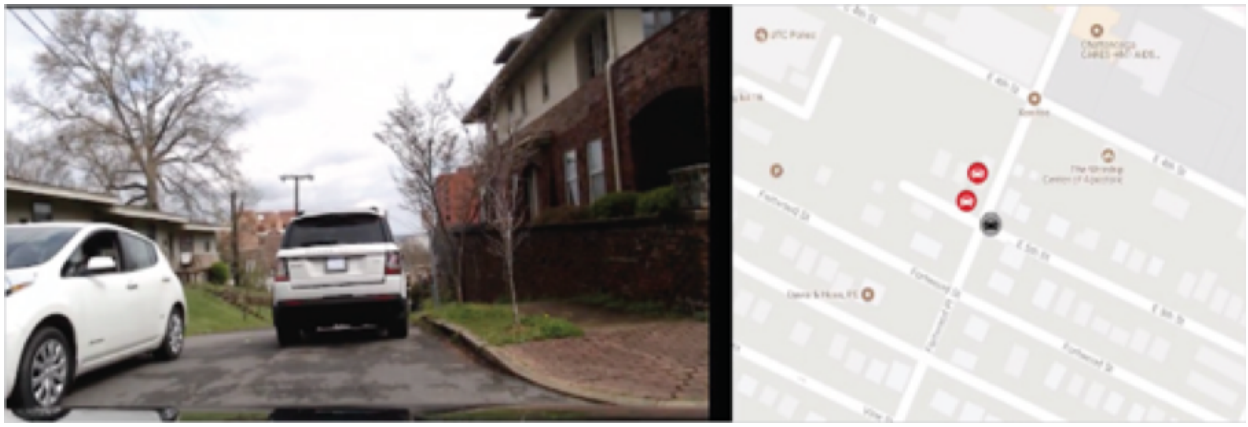


Figure 4.6 A vehicle waits as another passes to allow driver to pass safely

Figure 4.5 and Figure 4.6 show the driver's actual field of view (left) relative to the AIO application communication range (right). The application also gives the driver a wider view of objects to come ahead while driving and can help them prepare decision-making tasks earlier compared to what they are able to see only from their field of view. For example, in Figure 4.6, the driver is able to see a stalled vehicle on the side of the road, but the hill in front of them prevents the ability to see the vehicle approaching them from the other

lane. If the driver wereto pass the stalled vehicle, an accident could occur, but using the AIO mapping application, the driver is able to see a vehicle approaching on the map and avoid an accident by waiting for the driver to pass and rechecking the map to make sure the lane is clear before advancing around the stalled vehicle.

As the web-based front-end is written in JavaScript, it is trivial to keep its features in line with the mobile application. As such, a web front-end has been launched for testing purposes with no issue. This interface does not offer tracking of the current user (mostly due the lower accuracy of IP-based geolocation), but it does allow the user to pan through the Google Maps window and view all tracked subjects.

4.5 Conclusions on AIO

The All-in-One mobile mapping application has the potential for short-term and long-term success if well-utilized and further developed. This application has the ability to give drivers, passengers, and pedestrians a real-time view of their surrounding environment relating to mobility. With the use of individual app users and installed infrastructure cameras anonymously identifying the location of individuals, users can be more aware as they travel throughout their day and increase the chances of making a well-planned decision to help make urban streets and roadways safer for our communities.

CHAPTER 5

CONCLUSION

5.1 Final Thoughts

During this thesis, two applications were presented using the concept of V2V Communication and V2I Communication to help improve roadway safety, whether they be cyclists, drivers, passengers, or pedestrians, during their commute. With the utilization of fiber-connected V2X Communication in combination with object detection and t he ubiquitous connection the world now has with GPS-enabled devices, we have the capability to create technology that can benefit lives worldwide. What has been presented here is only one piece of the equation, but with the cooperation and creativity of others, new applications can be created to help further the goal of creating a safer roadway for our community and the community of others.

A common question that has been asked throughout presenting these applications at public events is how do these applications lessen driver distraction compared to causing more distraction due to the visuals the applications produce. This is a great question and one I have thought about myself from time to time. These applications are in the beginning s tages in a setting where experiments are being performed by non-autonomous vehicles. Therefore, in order to help validate and share our results, a visual component is necessary. I dealistically, the data gathered f rom these applications, such as the location of a pedestrian, will be used to assist the decision-making in autonomous vehicles upon approaching the locations stored. A nother thought is that once these applications are f unctioning well within an autonomous vehicle, the passengers may find an interest in visually viewing applications such as these out of the interest of seeing the correlation of the vehicle's action and what they see on the screen or they can

view it for pure entertainment during their commute.

5.2 Future Research

Future research for these applications can be divided into short-term research goals and long-term research goals.

5.2.1 Short-Term Research Goals

1. Immediate next steps to be taken will involve combining the two applications that were presented to create a single safety application. The two applications will be combined in a way that see-through will still be performed, but it will not be visualized for the driver; at least not in the way depicted in Chapter 3. Instead, the data gathered from the see-through application will work similar to the infrastructure camera. It will identify the object, estimate its location based on the vehicle's location, and place an icon on the mapping application. This will create less distraction without impeding the integrity of the mobile application.

2. The second feature to be added to the applications will be the transition from using Google Firebase to web sockets to allow for scalability. The web sockets reduce latency and elimi-nates the data limit we currently have while using Google Firebase.

3. Another feature to be added to the AIO application could be the detection of objects on the road that could hinder the ability for a smooth commute. By expanding our algorithm to detect objects on the road that are not classified as a cyclist, pedestrian, or vehicle, we can label it as a generic object and place it on the map. If the object has not been detected after a set length of time, we can assume the object has been moved and the icon can be removed until another object, or the same object, is detected and will be placed back on the map.

4. A final short-term goal is to include a type of automatic detection to the process. To save energy and processing power, we can find a way to only activate the algorithms and report locations if there is activity within an area where a version of the application is running. If

there is no activity in a designated area, there is no reason to continue using processing power to run this hardware and software. A possible method for this is to apply a geofence around the devices connected to the application. If someone enters that geofence, the equipment and algorithms will be activated and begin analyzing the area, and any geofenced area that is overlapping the activated area will be activated as well to create a seamless transition for connectivity as well as giving the user a broader look at their surroundings. If the algorithms have not detected anything within a certain time frame, we can assume there is no activity and the equipment can hibernate until another instance triggers it. This type of functionality would be ideal in situations where there is little to no activity, such as college campuses during academic breaks.

5.2.2 Long-Term Research Goals

1. If the applications presented were ever introduced into actual autonomous vehicles, a visual representation would not be needed. Therefore, the database storing the locations gathered by the application would be the primary focus. If the vehicle were to have access to the database storing the objects' locations, the visual part of the application would not be necessary for the vehicle anymore and keeping that visual within the vehicle for passengers to see may cause unnecessary anxiety. The visual application would of course still be used by pedestrians, cyclists, and the necessary hubs utilizing the application for safety purposes, such as monitoring the vehicle's action or potentially for crime investigation.

2. Another long-term goal would be to implement automatic rerouting based on objects or construction detected via computer vision object detection reports. Rerouting can be added in a short amount of time, but the reason I place it under long-term goals is because we want to create a more effective rerouting algorithm that will also take into consideration how many vehicles have already been rerouted in each direction instead of rerouting all vehicles in a single direction. This will involve a much deeper understanding of different

functionalities behind cooperative mobility on a large scale.

3. A key aspect that needs to be researched is how to seamlessly transfer the video stream from V2I to V2V in the event of no infrastructure access points connected to fiber are available. In addition to the transfer, the challenge of controlling the latency during the transition will need to be taken into account.

4. Lastly, in the event of a user needing to be notified, we want to integrate an auditory indicator, and potentially a visual representation with see-through if necessary, to make sure the driver is aware of their surroundings in the case of a person crossing the street as they approach or a vehicle driving through a red light.

REFERENCES

[1] N. S. Council. (2018) 2017 estimates show vehicle fatalities topped 40,000 for second straight year.

[2] U. N. H. T. S. Administration. Motor vehicle traffic crashes as a leading cause of death in the united states, 2015. https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812499.

[3] N. H. T. S. Administration. Driver assistance technologies. https://www.nhtsa.gov/equipment/driver-assistance-technologies, last accessed on 05/28/2018.

[4] Mobileye. Autonomous driving. https://www.mobileye.com/future-of-mobility/history-autonomous-driving/, last accessed on 05/28/2018.

[5] D. G. R. Bradski and A. Kaehler, *Learning Opencv, 1st Edition*, 1st ed. O'Reilly Media, Inc., 2008.

[6] M. Bertozzi, L. Castangia, S. Cattani, A. Prioletti, and P. Versari, "360Â° detection and tracking algorithm of both pedestrian and vehicle using fisheye images," in *2015 IEEE Intel-ligent Vehicles Symposium (IV)*. IEEE, pp. 132–137.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016.

[8] L. Wang, Y. Lu, H. Wang, Y. Zheng, H. Ye, and X. Xue, "Evolving boxes for fast vehicle detection," in *2017 IEEE International Conference on Multimedia and Expo (ICME)*.

[9] C. Papageorgiou and T. Poggio, "Trainable pedestrian detection," in *1999 International Con-ference on Image Processing*, vol. 4. IEEE, pp. 35–39.

[10] H. Cho, P. E. Rybski, and W. Zhang, "Vision-based bicyclist detection and tracking for intelligent vehicles," in *2010 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 454–461.

[11] MathWorks". Object detection. https://www.mathworks.com/discovery/object-detection. html, last accessed on 04/28/2018.

[12] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005*, vol. 1. IEEE, pp. 886–893.

[13] Opencv: Face detection using haar cascades. https://docs.opencv.org/3.4.1/d7/d8b/tutorial_py_face_detection.html, last accessed on 06/01/2018.

[14] Opencv: Canny edge detection. https://docs.opencv.org/3.4.1/da/d22/tutorial_py_canny.html, last accessed on 06/01/2018.

[15] B. U. Töreyin, Y. Dedeoğlu, U. Güdükbay, and A. E. Cetin, "Computer vision based method for real-time fire and flame detection," *Pattern Recognition Letters*, vol. 27, no. 1, pp. 49–58, 2006.

[16] S. Banerjee, J. Sweet, C. Sweet, and M. Lieberman, "Visual recognition of paper analytical device images for detection of falsified pharmaceuticals," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pp. 1–9.

[17] T. Brosnan and D.-W. Sun, "Improving quality inspection of food products by computer vision—-a review," *Journal of food engineering*, vol. 61, no. 1, pp. 3–16, 2004.

[18] X. Fontaine, R. Achanta, and S. Süsstrunk, "Face recognition in real-world images," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 1482–1486.

[19] A. Kulkarni and S. Shinde, "Monitoring driver distraction in real time using computer vision system," *International Journal of Computer Science and Engineering*, 2017.

[20] E. Kocabey, M. Camurcu, F. Ofli, Y. Aytar, J. Marin, A. Torralba, and I. Weber, "Face-to-bmi: Using computer vision to infer body mass index on social media," 2017, https://arxiv.org/pdf/1703.03156.pdf.

[21] C. Jiang and B. Zhang, "Weakly-supervised vehicle detection and classification by convolutional neural network." IEEE, 2016, pp. 570–575.

[22] Karpathy. Cs231n convolutional neural networks for visual recognition. http://cs231n.github.io/convolutional-networks/, last accessed on 10/03/2017.

[23] Ujjwalkarn. An intuitive explanation of convolutional neural networks. https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/, last accessed on 10/12/2017.

[24] J. Fu. Vehicle-detection. https://github.com/JunshengFu/vehicle-detection, last accessed on 03/12/2017.

[25] P. Gomes, C. Olaverri-Monreal, and M. Ferreira, "Making vehicles transparent through v2v video streaming," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 930–938, 2012.

[26] F. Rameau, H. Ha, K. Joo, J. Choi, K. Park, and I. S. Kweon, "A real-time augmented reality system to see-through cars," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 11, pp. 2395–2404, 2016.

[27] N. H. T. S. Administration, "Distracted driving 2015," *National Highway Traffic Safety Administration*, 2017, https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/documents/812 381 distracteddriving2015.pdf.

[28] C. for Disease Control and Prevention. Motor vehicle safety. https://www.cdc.gov/motorvehiclesafety/distracted driving/index.html, last accessed on 10/13/2017.

[29] A. T. What are the 100. https://www.itcanwait.com/, last accessed on 10/13/2017.

[30] T. A. C. National Highway Traffic Safety Administration. Stop texts stop wrecks. http://stoptextsstopwrecks.org/, last accessed on 10/13/2017.

[31] W. H. Thomas Triggs, "Reaction time of drivers to road stimuli," *Monash University Accident Research Centre*, 1982.

[32] Y.-F. Wang, "Computer vision analysis for vehicular safety applications," in *International Telemetering Conference Proceedings*. International Foundation for Telemetering, 2015.

[33] H. Zhu, K.-V. Yuen, L. Mihaylova, and H. Leung, "Overview of environment perception for intelligent vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 10, pp. 2584–2601, 2017.

[34] J. Janai, F. G¨uney, A. Behl, and A. Geiger, "Computer vision for autonomous ve-hicles: Problems, datasets and state-of-the-art," 2017, http://adsabs.harvard.edu/abs/ 2017arXiv170405519J, last accessed on 10/15/2017.

[35] H. H. C. S. Rebekah Thompson, Jose Stovall, "See-through technology using v2x communi-cation," *ACM Southeastern Conference Proceedings*, 2017.

[36] N. H. T. S. Administration, "Critical reasons for crashes investigated in the national motor vehicle crash causation survey," 2015, https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812115, last accessed on 05/30/2018.

[37] M. Veloso, C. Bentos, and F. C. Pereira, "Multi-sensor data fusion on intelligent transport systems," *MIT Portugal Program, Transportation Systems Focus Area, ITS-CM-09-02*, 2009.

[38] M. Zhou, A. K-s. Wong, Z. Tian, V. Y. Zhang, X. Yu, and X. Luo, "Adaptive mobility mapping for people tracking using unlabelled wi-fi shotgun reads," *IEEE Communications Letters*, vol. 17, no. 1, pp. 87–90, 2013.

[39] H. M La and W Sheng, "Distributed sensor fusion for scalar field mapping using mobile sensor networks," *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 766–778, 2013.

[40] Ionic. All about ionic. https://ionicframework.com/docs/v1/guide/preface.html, last accessed on 03/24/2017.

[41] Jcesarmobile. Mirror of apache cordova plugin geolocation. https://github.com/apache/cordova-plugin-geolocation, last accessed 04/05/2018.

VITA

After graduating from Gordon Lee Memorial High School in Chickamauga, Georgia in 2010, Rebekah Thompson attended Georgia Southern University in Statesboro, Georgia. In December 2014, she was awarded a Bachelor of Science with Honors in Graphic Communications Management and a minor in Marketing. After working as a project manager and graphic designer in Statesboro, Georgia, she decided to return home to pursue a master's degree in August 2015. Rebekah graduated with a Master of Science in Computer Science from the University of Tennessee at Chattanooga in August 2018.