REAL TIME LABELING OF DRIVER BEHAVIOR IN REAL WORLD ENVIRONMENTS

by

Christopher Seebeck
A Thesis
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Master of Science
Computer Science

Committee:

______________________	Dr. Zoran Duric, Thesis Director
______________________	Dr. Duminda Wijesekera, Committee Member
______________________	Dr. Steve Kan, Committee Member
______________________	Dr. Yi-Ching Lee, Committee Member
______________________	Dr. David Rosenblum, Chairman, Department of Computer Science
______________________	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering
Date: ______________________	Fall Semester 2020 George Mason University Fairfax, VA

Real Time Labeling of Driver Behavior in Real World Environments

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at George Mason University

By

Christopher Seebeck
Bachelor of Science
Old Dominion University, 2013

Director: Dr. Zoran Duric, Professor
Department of Computer Science

Fall Semester 2020
George Mason University
Fairfax, VA

Dedication

I dedicate this thesis to my wife Jessica, without her support this wouldn't have been possible.

Acknowledgments

I would like to thank Dr. Zoran Duric and Dr. Duminda Wijesekera whose advice and help made this research possible.

Table of Contents

Page

List of Tables

List of Figures

Abstract

REAL TIME LABELING OF DRIVER BEHAVIOR IN REAL WORLD ENVIRONMENTS

Christopher Seebeck
George Mason University, 2020
Thesis Director: Dr. Zoran Duric

Vehicle systems that use driver behavior data to determine safe and unsafe behavior need to operate in real time and in real chaotic environments. The research in developing these systems do not have publicly accessible data sets that would aid in research and development. In order to create these data sets, experiments and data collections need to be performed in an unconstrained real world environments or in a highly constrained environment. This thesis proposes a tool to collect real time data in unconstrained real world environment, called Live Driving Detection (LiDD). LiDD labels driver head rotations to determing where the driver is looking and combines the state of the car from the CANBus network to add more context to the produced data. The labeling process is designed to be simple in order to quickly label a given instance of data called a frame. LiDD is able to output labeled driver data fused with vehicle state at approximately 6 HZ. LiDD's utility was evaluated in three common real world environments: a suburban, a major highway, and a city environment. This research shows that LiDD and it's resulting data sets can be developed without requiring expensive equipment and that it's data will be useful for future research and development of Advanced Driver-Assistance Systems (ADAS).

Chapter 1: Introduction

Research in determination of driver behavior is difficult. Only a few pre-existing data sets exist with the most commonly referenced one as the State Farm Distracted Driving data set. As of this writing it is only available for use in their Kaggle competition [3]. This data set, while extensive in labeled individual images, does not provide adequate data for development and use of real time Advanced Driver-Assistance Systems (ADAS). An example of the type of image found in the data set is included in Figure 1.1. This data only captures the actual distraction, not the motions leading up to the distraction. As such, research into this area must rely on the research team to collect their own data. Further difficulties in researching driver behavior exists in the recreation of dangerous behavior. Dangerous behavior being defined as, distracted driving, drunk driving, aggressive driving, and other behavior that would be described as unsafe. This kind of research is inherently interesting

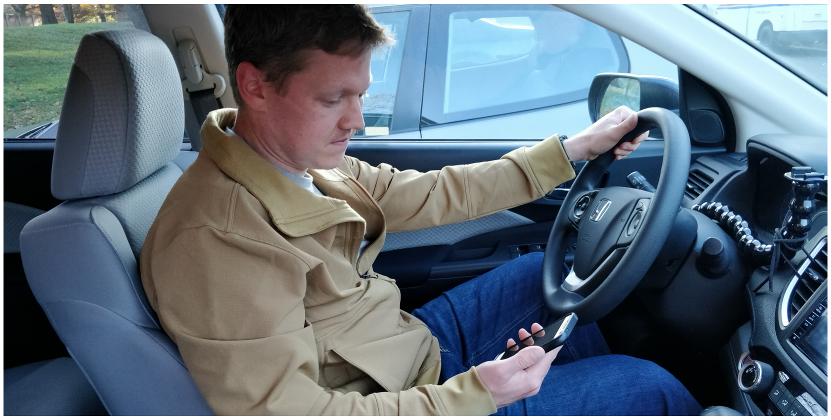

Figure 1.1: Type of image found in State Farm dataset

as: drunk driving causes almost 11,000 fatalities annually, aggressive driving causes almost 13,000 accidents annually, and texting while driving increases the likelihood of an accident 8 times [4]. Ethically this kind of research is not possible in a real world environment or must be done with severe restrictions in order to remain safe.

Simulation can provide the answer for providing a safe environment for this kind of research. How good is a simulation in capturing a similar response from the driver as a real world environment? With good data collection practices, the same set of experiments can be performed in both the real world and in simulation collecting information on the driver, the car, and the outside environment. Comparison of the data collected in both environments can be used to evaluate the effectiveness of a given simulators ability to elicit a response out of a driver that resembles a real world response. This would allow for research into dangerous driving behaviors to be performed while maintaining safety with an understanding of how the results compare to real world circumstances.

In order to understand what a driver is doing there are three categories of data context that need to be combined. The fusion of this data will help determine if a driver is being unsafe, the drivers ability, and provide input into many other new kinds of driver assistance systems. This context can be split into three categories of information: driver, car, and world.

The first category is information about the driver. This would include where the driver is looking or what the driver is doing. Information such as, is the driver looking at their phone, the road ahead, eating, would all be information contained in the first category. This is very important information as the driver is responsible for the cars actions as it moves through the world. This category has been the focus of much of the research in driver behavior [5] [6] [7] [8].

The second category is information about the car. This would include how the driver is operating the car. Vehicle speed, braking, use of the entertainment center, are all points of information that provide additional context to what the driver is doing. A driver is looking at their phone and texting, what is the car doing? Is it in motion or is the car stopped or even parked? That is a critical distinction that only using information about the driver can't answer.

The third category is information about the world, this covers a significant amount of information and contains more possible situations alone than the combination of the first two categories. These situations include stop lights, pedestrian crossings, what other vehicles are doing, deer crossing, and many more. The combination of the driver and car information categories provides context on whether a driver's actions are safe. The inclusion of information about the world provides additional context about the driver's decisions.

This thesis focuses on the importance of data collection for future research in driver behavior detection and the utility of combining the driver and car information categories. The scope of the world category is too complex to try and capture for this thesis. A tool called Live Driving Detection (LiDD), was developed for this research that combined information on the driver and information from the car in order to create simple data sets of labeled information that could be used for further research. LiDD leverages data from two sources: an open source tool called OpenFace to determine a head position and reads data off the vehicles CANBus to get vehicle state. This data is combined into a situational frame by LiDD to representing the current state of where the driver is looking and the state of the vehicle at approximately 6 Hz in real time.

Methodologies for real time driver data collection are also evaluated here. Experiments were conducted in real world environments without constraints on external factors. This was done to capture events and other unplanned circumstances that are difficult or impossible to recreate in a controlled environment. Safety is a top concern in these experiments and the difference between a controlled or simulation environment to a real world uncontrolled environment are discussed. Problems discovered with this experimentation and lessons learned are also discussed.

Chapter 2: Related Work

Labeling drivers actions is an important component of understanding what the driver is doing. Research into this area have approached this problem similarly such as running in real time, detecting eye gaze through sunglasses, and determining what the driver is doing [6] [9]. This research focuses on the creation and training of new machine learning models to perform the task of face, eye, or body position detection.

Jaeik Jo et al. focused on eye detection using OpenCV and an accelerometer placed at the back of the head to determine driver drowsiness and distraction [6]. This sensor would be considered intrusive for a driver to wear in order to collect head pose data needed for drowsy detection. Leekha et al. created a convolutional neural network to do detection of driver body pose in order to detect when the driver is distracted based on the State Farm Distracted Driver Data set [5]. Both of these approaches only use information on the driver to detect dangerous behavior. While it is a good assumption that these conditions such as the driver looking at a phone is interpreted as the driver is distracted, there is more context available to increase understanding of the situation. For example, if the driver is currently distracted but the vehicle is parked or possibly even stopped at a stop light, an ADAS system wouldn't know that the vehicle is stopped.

A data set that gets used frequently in this area is the State Farm Distracted Driver data set [3]. This data set has thousands of labeled images ready for use in a machine learning system. Figure 1.1 shows an example image from this data set. Leekha et al. used this data set to train their model and were able to build a system that can classify if the driver is distracted in a real time environment [5]. This data set only has the overt distraction action labeled. There is no data of the lead up to the action that may be critical in stopping the behavior before it happens. With a full range of data available describing an action, an ADAS system could recognize a distracted driving action happening before the attention is fully diverted away from the road.

Abouelnaga et al. created a data set similar to the State Farm data set called the American University in Cairo (AUC) Distracted Drivers Dataset [9]. Due to the restrictions on the State Farm dataset, this research set about recreating the data for use in a face and hand detector experiment to annotate their data set. From their research the data set they created appears to be similarly structured to the State Farm data set, but the annotations are applied to each image that is created from the captured video. This data set does not contain information from the vehicle or the outside world providing more context to drivers actions.

Chapter 3: Approach

The approach taken in this thesis is to create a tool to fuse where the driver is looking and information from the CANBus network in real time. The goal of LiDD is to collect data of real world drives at a rate that can be used to understand actions taken by the driver. A real world drive is defined here as one where the drives occurs without external constraints being controlled. An open source tool called OpenFace is used to determine current face pose and to provide LiDD the data to perform simple labelling to determine what the driver is looking at. A vehicles CANBus provides information on the state of the vehicle and provides context to driver actions. LiDD was developed in order to experiment with existing tools, real world collection methodologies, and real time data collections.

3.1 OpenFace

OpenFace is a tool created by Baltrušaitis et al. for determination of facial landmarks, head pose, facial action units, and eye gaze [10] [11]. The creators of OpenFace developed the tool in order to aid in research of facial actions while providing real time performance. In this thesis OpenFace is used to provide information on what the driver is looking at. As such, it's features that were used and evaluated will be described here.

3.1.1 Head Pose

From the available features of OpenFace, Head Pose estimation provides an accurate description of the drivers face in relation to the camera. OpenFace accomplishes this by determination of Facial Landmarks. An example of OpenFace running is provided in Figure 3.1. Each blue dot is a landmark that is calculated on each frame, with a frame being a set of data at a given time. Each landmark has two calculated positions, a two-dimensional position (2D) and a three-dimensional position (3D). The two-dimensional position is the landmarks position in the image itself. The 3D position is the landmarks position in space in relation to the camera. The main data points being used from OpenFace for this thesis

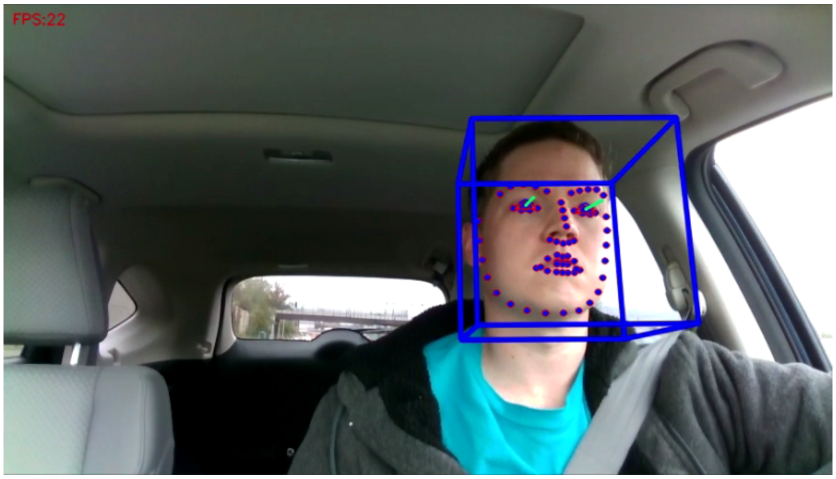

Figure 3.1: Example OpenFace Frame

are the rotations in radians with the camera acting as the origin [10] [11]. Three data variables are provided for each frame that OpenFace provides: Pitch, Yaw, and Roll. Figure 3.2 describes how these rotations are viewed in OpenFace. On the test and evaluation system, OpenFace produced frames at approximately 10 frames per second (fps). These frames were read in at 10 Hz while checking for and discarding duplicate frames. This processing rate was maintained while running the developed application for this thesis. The OpenFace source was modified in order to use images captured from the Intel RealSense API described in section 3.2.

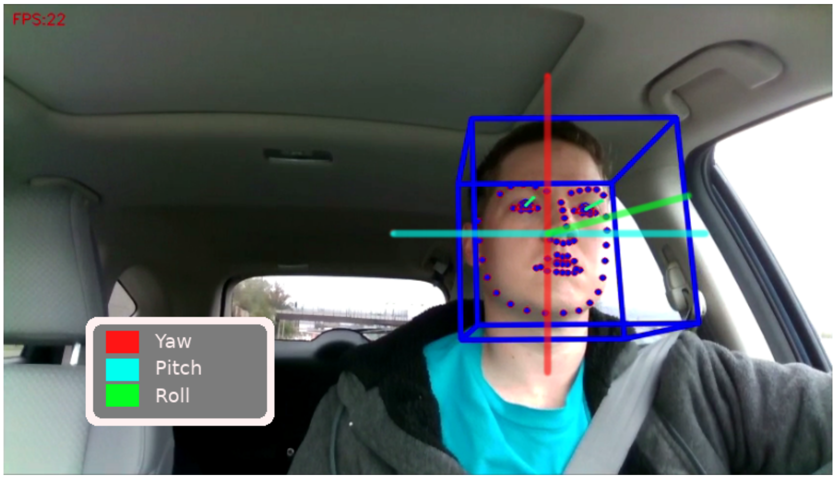

Figure 3.2: Yaw, Pitch, Roll

3.2 Intel RealSense

An Intel RealSense D435 depth camera was used to capture real time footage of a driver. Figure 3.1 shows an example frame from this camera being run through OpenFace. The camera has three cameras built in: an RGB camera, and two infra-red cameras used as a depth camera [12]. The depth camera can provide depth from the camera at a given point. The range of depth detection is between 0.105m and approximately 10m [12]. The depth camera was evaluated for use in this thesis but was not ultimately used. The RGB camera provides several options for stream selection of different fps options and resolutions. The selected configuration for this thesis is 848x480 at 30 fps.

3.3 Driver Labeling

The labeling of where the driver is looking is a critical step in understanding what the driver is doing. In a real time system the label needs to be applied quickly if decisions are going to be made on that information. If the labeling process takes n seconds after the initial frame capture, the information gained may no longer be valid for the current situation.

Table 3.1 describes the landmarks used in labeling where the driver is looking. Looking can be defined in different ways. In this thesis looking is defined as where the drivers head rotation is currently pointing, called a head pose. This thesis does not use eye gaze vectors to determine where the eyes are pointing. Drivers may use sunglasses while driving, as such the use of an RGB camera will not be sufficient to detect eye gaze. In order to determine a solution to work in all scenarios, research was prioritized on determining what the driver is looking at by head rotation alone.

Table 3.1: Landmarks and descriptions

Landmark	Description
Center	Drivers head position pointing over the steering wheel
Left Mirror	Drivers head position pointing center of left mirror
Right Mirror	Drivers head position pointing center of right mirror
Rear Mirror	Drivers head position pointing center of rear mirror
Entertainment Center	Drivers head position pointing center of entertainment center
Instrument	Drivers head position pointing center of instrument cluster

3.3.1 Calibration

In order to label where the driver is looking, a calibration step is performed to localize where the driver is in relation to the camera. This is a two-step process. First the landmarks inside the car are determined through measurement of the drivers head rotation. This measurement of the heads position, called head pose, captures the Roll, Pitch, and Yaw described in Figure 3.2 at a moment in time with the driver looking at the specified landmark. This process is repeated with the car off and not in motion. After this step, an average Yaw and Pitch are calculated for each landmark in relation to the center point in the car. This mapping is then used to configure the application for run time.

The second calibration step is done at run time when the driver calibrates the head pose of the center landmark. This calibration process runs automatically at application startup. The software captures the current head pose to be the calibration target to begin the calibration process. On each new frame, the new pose is compared to the pose from the calibration target frame. Each axis of rotation of the new frame is compared with its axis from the previous frame. If the difference in rotations are greater than .5 radians the calibration is considered failed, the current frame is used as the new calibration target, and the process restarts.

3.3.2 Labeling

The labeling process was designed to work in a real time environment. This is accomplished by simplifying the labeling process in order to label a given head rotation on average under 1 ms. Using the vehicle landmark mapping determined in step one of the calibration procedure, the current head pose is compared to the head pose at the mapped landmark. Specifically the yaw and pitch rotations are compared. If the difference in yaw and pitch rotations are less than 0.1 radians then the frame is labeled as the driver looking at the mapped landmark. If either yaw or pitch is within that difference, then it is labeled as a possible that the driver is looking at that landmark. The results of this calculation is then collected into a Driver frame.

This calculation is done using the Python3 multiprocessing. When a new frame is processed, a new process is created in order to perform the position calculation on an available CPU core. This reduces resource contention as processes are only created when a new frame needs to be processed.

3.4 CANBus

The CANBus (Controller Area Network Bus) has been a standard in consumer vehicles since 1996 and has been mandatory in vehicles since 2008 [1]. The CANBus provides a method of packet transmission between vehicle subsystems. For example, a CAN packet may contain information from the steering wheel such as the current steering angle or could contain information about individual wheel speeds. All systems on the CAN network will see all packets that are on the network.

The CANBus runs on two networks called CAN high and CAN low [1]. These two networks split vehicle systems depending on the critical nature of its information. The CAN high line contains systems such as braking and steering angle. The CAN low line contains systems such as wipers or climate control. These CAN lines are accessible through the OBD-II or diagnostic link connector. This port is generally found below the steering wheel in most vehicles, with Figure 3.3 showing the most common locations [1]. Figure 4.6 showing the location of the ODB-II port on the experiment vehicle with a CANBus cable hooked in.

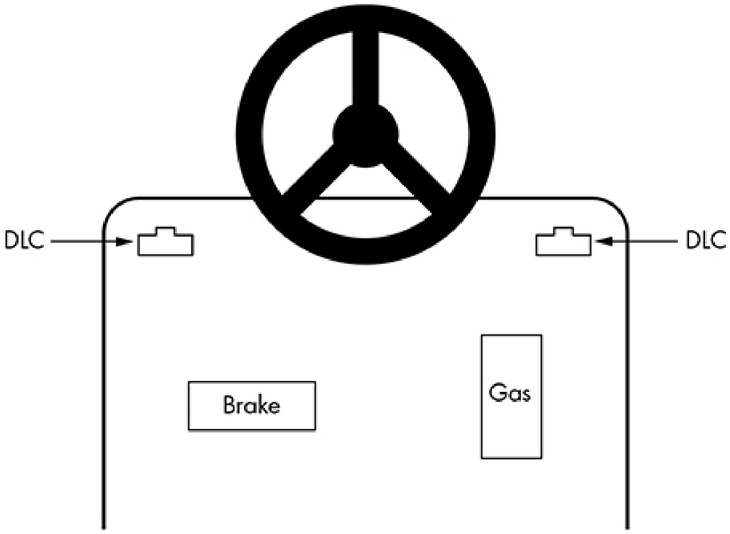

Figure 3.3: Typical ODB-II port location [1]

3.4.1 Connecting to the CANBus

There are manufactures who make tools to connect into the OBD-II port and access the CAN network. A similar search for ”OBD2” on Amazon returns over 1,000 items. Many of these are diagnostic scanners, but also includes devices that could be used to read data off of the port. There are other manufactures that provide interfaces designed to stream data off of the CANBus to another device. For this thesis two CANBus communication equipment was utilized. First CANUSB from CANUSBShop, a device that acts as a controller on the CANBus and is able to send and receive CAN packets. This device has an attached usb connection allowing for data to be streamed to a laptop [13]. The second to be used was a CAN to OBD2 cable from gridconnect [14]. The CANUSB device does not have an OBD2 port, it ends in a 9-pin D-Sub socket. This cable correctly maps the OBD-II connections to the connections on the CANUSB device. To ensure connectivity, this required looking at pin outs from each manufacturer to verify that both devices could work together. In determining which OBD-II to D-Sub cable several vendors were checked for potential use, but many did not have a matching pin out.

SocketCAN is an open source tool that has been available in the Linux kernel since 2006 [1]. Using this network interface, CAN applications can be written to stream data from a CAN device like the CANUSB mentioned above. There are a multiple tools that can be used to interact with the CANBus on the command line. For example candump will read CAN packets and display them as they occur. Similarly there are Python libraries for interacting with SocketCAN. The two libraries used in this thesis are python-can and cantools [15] [16]. Python-can provides abstractions to hardware devices and utilities for sending and reading CAN packets [15]. Cantools provides tools for encoding and decoding of CAN messages as well as other utilities [16]. Figures 3.2 and 3.3 are two example CAN frames captured from the research vehicle using the tools discussed here.

With the use of the ODB-II port, relatively inexpensive equipment, and open source tools, data on the CANBus can be easily accessed. There are challenges in working with

Table 3.2: Example Gear Selection CAN Frame

Tag	Data
GEAR_SHIFTER	P
GEAR2	169
GEAR	169
ZEROS_BOH	0
BOH	1
COUNTER	2
CHECKSUM	14

Table 3.3: Example Steering CAN Frame

Tag	Data
STEER_ANGLE	3.90
STEER_ANGLE_RATE	-3
COUNTER	0
CHECKSUM	7

CANBus as each vehicle manufacturer uses unique CAN packets that can vary from different makes and even models of vehicle [1]. This encoding is usually kept secret by the manufacturer in order to maintain network security, but reverse engineering processes can be performed to reverse engineer packets. This is a time consuming process unless previous research or published mappings are able to be applied to a chosen vehicle. The Car Hackers Handbook provides instructions for this reverse engineering process [1].

3.4.2 CAN Frame

A CAN frame is a set of labeled data representing the current state of the vehicle at a given moment in time. The labeling of this data is straight forward because all of the work needed to determine what a particular CAN packet means is handled ahead of time when integrating with a particular vehicle. The actual mechanism in LiDD, is a simple Python dictionary update of the received CAN packet.

The Tags in Table 3.4, is the CAN information that was used from the research vehicle, a 2016 Honda CR-V. Each of these Tags were chosen to represent vehicle state from the available CAN data from the research vehicle. There is other information available such as: engine rpm, lock status, wiper speed, and steer angle rate. This information could be useful in understanding more of what the driver is currently doing. The Tags in table 3.4 were selected for representing what the vehicle is doing. More information is available from the CANBus, such as information from the Entertainment Center and Air Conditioning controls. Information from these systems would prove beneficial for algorithms in determining if the driver is distracted. This information was not accessed as part of this thesis.

Table 3.4: CANBus Data Tags

Tag	Description
STEERING_ANGLE	Angle of the steering wheel 0 being no left/right turn
BRAKE_PRESSED	Represents state of the brake pedal pressed/not pressed
GAS_PRESSED	Represents state of the gas pedal pressed/not pressed
WHEELS_MOVING	Represents state of wheels if the vehicle is moving/not moving
RIGHT_BLINKER	State of the blinker stalk right blinker on/off
LEFT_BLINKER	State of the blinker stalk left blinker on/off
GEAR_SHIFTER	State of the gear selector [Park,Reverse,Drive,Low,Sport]
CAR_SPEED	Current vehicle speed in mph (did not work)
XMISSION_SPEED	Current vehicle transmission speed
XMISSION_SPEED2	Current vehicle transmission speed (possible second sensor)
MAIN_ON	State of the cruise control system main on/off

3.5 Data Fusion

At each iteration, LiDD combines both the current CAN frame and the Driver frame into one frame for the given moment in time. On the development and evaluation machine this process happens at approximately 6 Hz. This fusion of data provides situational awareness into both the driver and vehicle states at a specific time. These frames are called Situational frames, as both the Landmark and Vehicle frames provide context helping explain what is happening.

An example of a situational frame is provided in Table 3.5. This Situational frame is an example of the driver making a right lane change. The driver is pressing the gas pedal, looking at the right mirror, and the right blinker is on. Subsequent frames would show the steering angle changing as the driver performed the lane change.

Table 3.5: Example Situational Frame: Right lane change

Landmark	State	Vehicle	State
Center	0.0	Steering Angle	-6.4
Left Mirror	0.0	BRAKE_PRESSED	0
Right Mirror	1.0	GAS_PRESSED	1
Rear Mirror	0.0	WHEELS_MOVING	1
Entertainment Center	0.0	RIGHT_BLINKER	1
Instrument	0.0	LEFT_BLINKER	0
Yaw	0.26	GEAR_SHIFTER	D
Pitch	-0.232	CAR_SPEED	0.0
		XMISSION_SPEED	54.12
		XMISSION_SPEED2	53.9
		MAIN_ON	0

Chapter 4: Experimentation

The testing of LiDD described in chapter 3 focused on capturing data in real world environments. Each experiment focused on capturing driver data on real roads in a variety of situations. Three environments were used to capture driving data: a suburban area, a major highway, and a city. In each environment a set of events were tracked for each individual drive. Each drive was repeated three times in daytime conditions, twice without sunglasses and once with sunglasses.

4.1 Suburban Environment

The chosen route for the suburban environment is described in Figure 4.1. This route contains a stop sign and two stop lights. Along the way there are pedestrian crosswalks, houses, a school, and on the day of the experiment little to no traffic. The start point is located at point A. Point B had a stop sign only for cross traffic, which could impact the start of the drive and on the return trip there was no stop sign. Point C is the first stop light, where the right turn allows right turn on red. Point D is the second stop light and featured a u-turn where the route retro-traversed. At the beginning of the experiment the car was started from a parked state and the vehicle was then parked before the experiment was stopped. On each individual run the same route was taken, provided it was possible, and the car started and ended in the same parking space.

The following events were tracked during the drives in this environment. If the driver had to stop at the first stop light at location C. If the driver had to stop at the second stop light at location D. If the driver had to stop to turn left on the return trip at location D. If the driver had to slow down for pedestrians.

This drive was repeated once at night. The segment between point A and point C is poorly

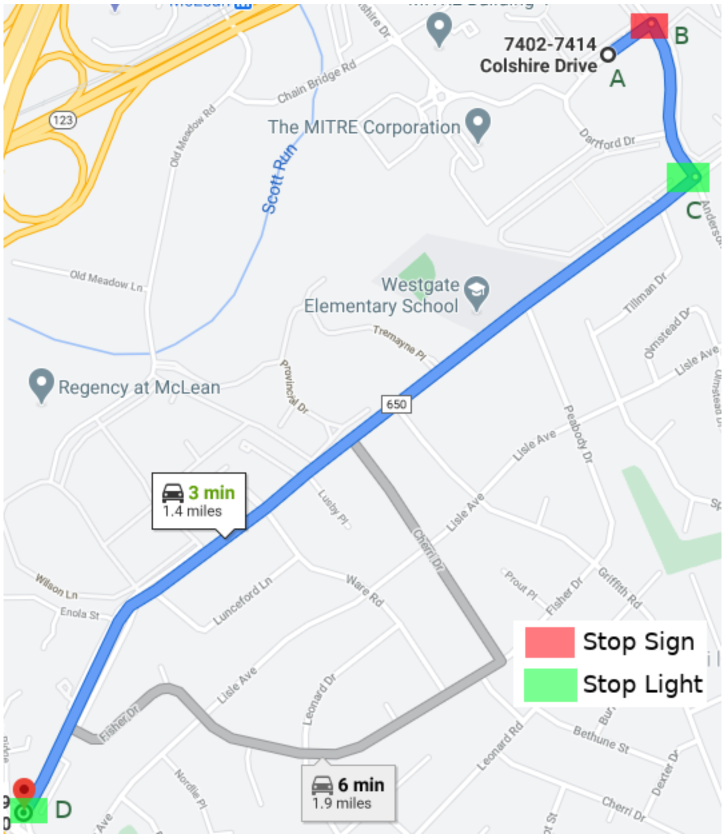

Figure 4.1: Suburban Experiment Route

lit with few if any street lights. The segment from point C to point D has a series of street lights providing additional light sources. Besides taking place at night the drive featured the same turns as the first three drives. Light conditions inside the vehicle were kept at the ambient light from the instrument cluster and entertainment center. The calibration step was performed with the overhead light on and then that was turned off before starting the drive. This was done because the start point did not have enough ambient light for the camera to pick up the face.

4.2 Highway Environment

The route chosen for the highway environment is described in Figure 4.2. This environment starts at a point along the way in the Suburban environment, but the majority of the route is over a major four lane highway. The turn around point, location D, is an exit ramp where the driver does not merge off and continues around to the highways entrance ramp location E. There are two stop lights near the beginning of the route at location B. The driver enters the highway at location C. The route was repeated on each individual experiment and the

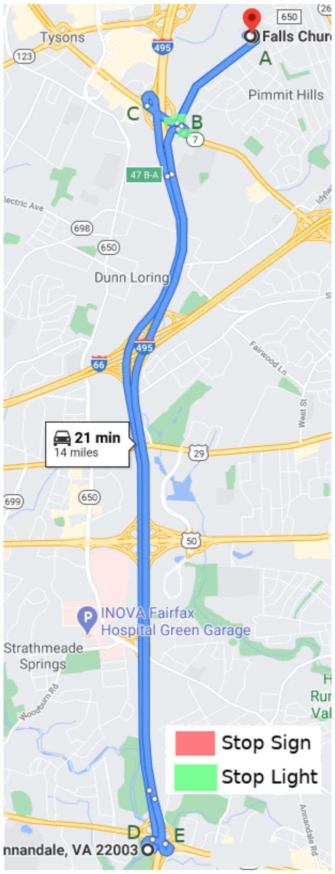

Figure 4.2: Highway Experiment Route

experiment was repeated three times. Each run began and ended in a small apartment parking lot. The route had less track-able events than the Suburban environment with only two traffic lights and significantly less pedestrian traffic. The environment is still complex as traffic conditions can vary from location to location.

The following events were tracked for this series of experiments. If the driver stopped at the first stop light. If the driver had to slow down on the highway due to traffic at an exit for another major highway. If the driver had to merge with other cars at locations E and D. If the driver was able to make the three lane change needed to turn left at location B on the return trip. The last event was added after the first experiment.

4.3 City Environment

The route chosen for the city environment is described in Figure 4.3. This environment is significantly more complicated to describe than the suburban environment. The suburban environment was able to be described in a few sentences, where in this route of 2.1 miles contained 15 interactions with stop lights. This route also contains a section from location L to O, with small one way roads and stop signs. This is in contrast to the route from location A to K which several stop lights and several places for pedestrians to cross.

The route was the same on each experiment and the experiment was repeated three times. Each time starting from a small parking lot at location A and looping through each location going through location B twice. The first time to start the route heading towards location C. The second time turning left to head back to location A.

The following events were tracked during the drives in this environment. If the driver had to stop at any of the stop lights or stop signs. If a lane change occurred because a car ahead was stopped to turn left. If the driver came to a stop for pedestrians.

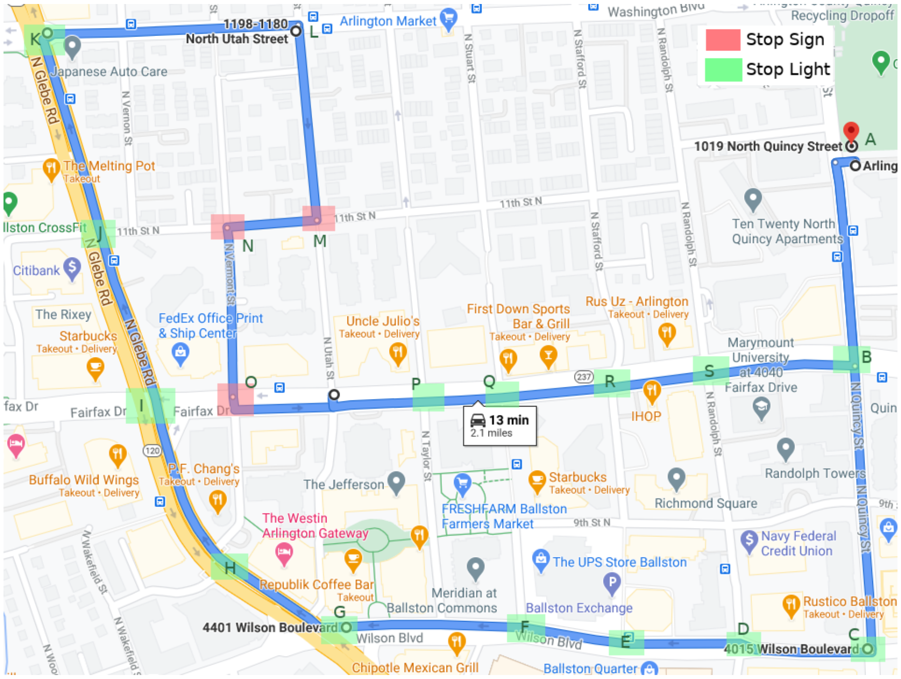

Figure 4.3: City Experiment Route

4.4 Setup

4.4.1 Research Vehicle

The research vehicle used was a 2016 Honda CR-V. The vehicle is equipped with a right blind spot camera. This equipment displays a camera feed on the entertainment center from under the right mirror of the driver's right blind spot. This allows a driver to see if the right blind spot is clear without having to turn their head to look. This differs from other cars that could be used in these experiments, since the driver could check the right mirror or the blind spot camera to check if a right lane change is clear.

The decoding of CANBus data of this vehicle was derived from publicly available database files from comma.ai GitHub repository [17]. This repository did not contain the specific file needed to decode a 2016 Honda CR-V, but did contain files for decoding other Honda vehicles. Through trial and error, a set of decoding rules were discovered that allowed decoding of the messages used in the research vehicle.

4.4.2 Computer Environment

The computer used for the development and experimentation was a Acer Nitro 5 AN515-54. Ths laptop is equipped with a Intel Core i5-9300H quad core processor, 8 GB of RAM, and a NVIDIA GeForce GTX 1650. The NVIDIA card was not used as part of the application with OpenFace and all other processes running on the CPU. The laptop had Ubuntu 18.04 LTS installed. The application was developed for use with Python 3.8.2 with other standard Python packages. OpenFace was compiled with g++ 6.5.0.

4.4.3 Equipment setup

Using an Intel RealSense camera, various mounting options for a stable camera position were tried. The first attempt had the camera mounted directly in front of the instrument cluster like in Figure 4.4. This position provided a good view of the driver as the driver moves the different points of interest in the car. In practice this position will lead to blockage of the camera during turns. Other mounting options considered during the first selection were: attached to the rear view mirror, above the steering wheel, and from the sun visor. These options were not used due to mounting hardware, field of view, and that OpenFace would need a modification to use an inverted image.

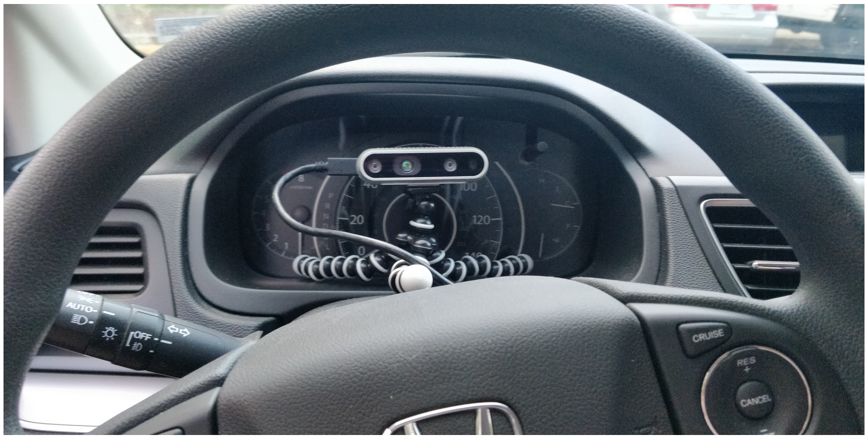

Figure 4.4: First Camera Setup

New camera positions with different mounting techniques were tried. GoPro style mounting equipment is cheap and readily available for camera mounting. A goose neck attachment provided a stable platform to mount the camera above and to the right of the steering wheel. This mounting point is shown in Figure 4.5 and was used for each of the experiments. Another mount point that was considered was using a windshield mount near the center of the windshield to keep the drivers half clear. This option was not used due to lose of face detection when the driver turns away from the camera.

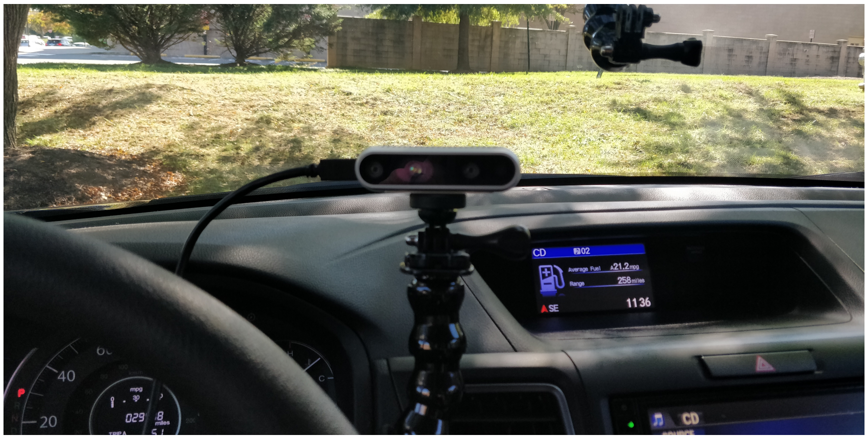

Figure 4.5: Camera Setup

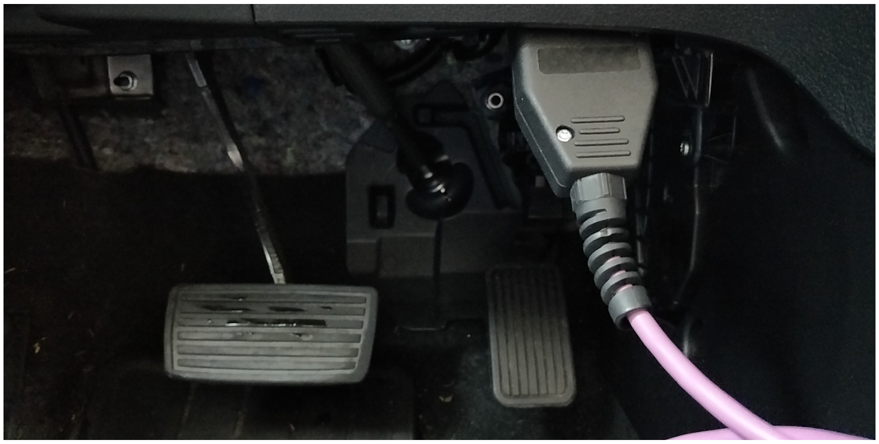

Figure 4.6: ODB-II port location on experiment vehicle

The CANBus equipment used is described in Section 3.4 consisting of a ODB-II to DB9 cable [14] to a CANUSB [13] device plugged into the experiment laptop. The connection to the vehicle with the ODB-II cable is shown in Figure 4.6. The cable was secured to the side of the driver foot well in order to prevent entanglement while driving.

4.4.4 Startup Proceedure

The startup procedure for these tests was as follows. First the vehicle was turned on and the camera attached to the top of the GoPro mount as in Figure 4.5. Next the ODB-II cable was plugged in with the CANUSB module attached as in Figure 4.6. Both USB cables for the camera and CANUSB device were then plugged into the experiment laptop. The Intel RealSense Viewer, an application for viewing the camera data, was used to verify connection to the camera. Next the network interface to the CANUSB device was setup and verified as receiving CAN frames. After the Suburban experiments were completed an additional step of setting up and recording the laptops screen was added here. The application was then started and calibration took place. Finally once calibration was finished, a visual inspection of the produced logs was performed to verify the application was running as expected and the experiment began.

Chapter 5: Results and Discussion

5.1 Face Detection

The experiment setup with OpenFace, is able to provide head rotations in most scenarios. Figure 5.1 shows a typical head pose captured with OpenFace. These situations and others where the majority of the face is in view, OpenFace performs very well. Figure 5.1 shows a head pose captured with OpenFace while the driver is wearing sunglasses. In these situations the face landmarks have shifted. Without eye data, the eye landmarks are determined to be at the bottom of the sunglasses with bad eye gaze data. The eyebrow landmarks are determined to be at the top of the sunglasses.

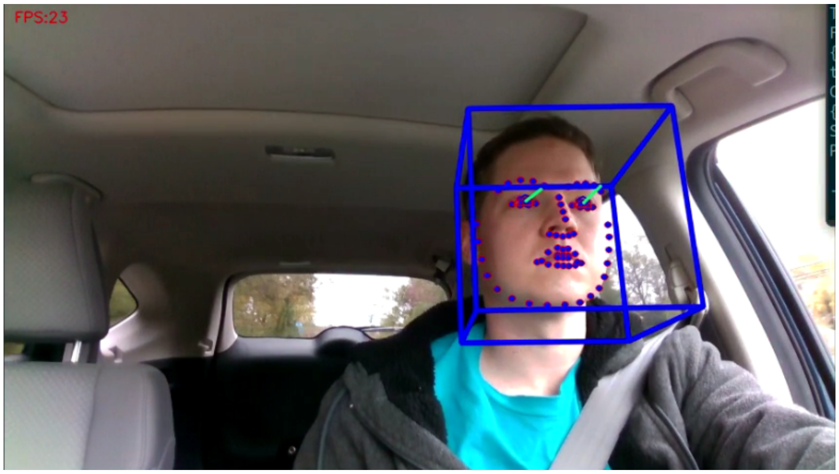

Figure 5.1: Example of the typical head pose from OpenFace

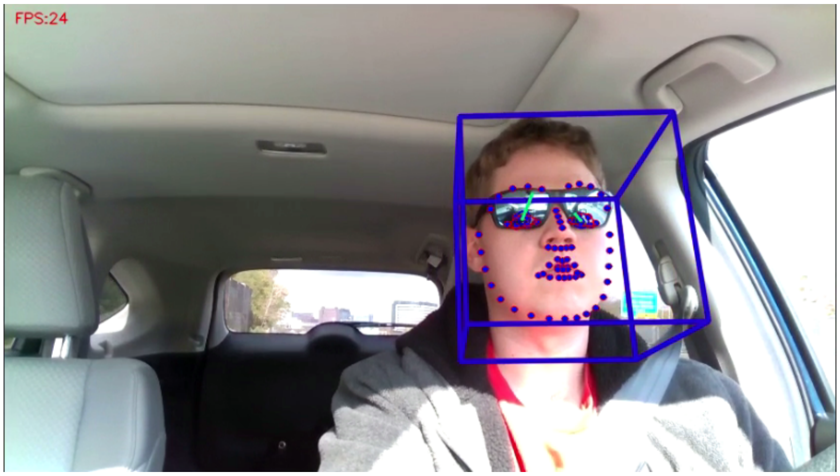

Figure 5.2: OpenFace head pose with sunglasses

Figure 5.1 is an example where the position of the camera does not allow OpenFace to form a head pose. In this Figure, the driver is checking the left blind spot. The drivers head is rotated further than the camera is able to form a head pose. This only happens in some scenarios, generally when checking the blind spots. In these case the data that is being reported in the Situational Frame is the last head pose that was formed.

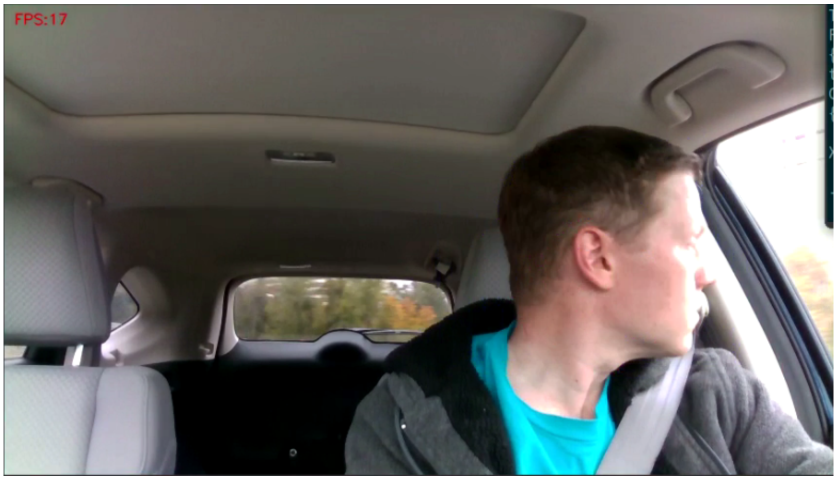

Figure 5.3: OpenFace unable to provide a head pose

5.2 Driving Collection Analysis

At 6 Hz, this style of data collection produces significant amounts of data. In the suburban data collect, the average number of frames collected was 2581. With 2575, 2612, and 2558 being the number of frames collected from drives one through three respectively. The highway set of data collections produced 5415, 5489, and 5714 frames for each drive. The city set of data collections produced 4221, 3909, and 4080 frames.

Table 5.1: Unclear Head Pose example 1

Landmark	State
Center	0.5
LeftMirror	0.0
RightMirror	0.0
RearMirror	0.0
Entertainment	0.0
Instrument	0.5
Yaw	-0.069
Pitch	-0.231

A limitation of this style of data collection is the determination of correctness of the labeled face pose data. At 6 Hz even short drives can produce lots of data with fused head pose and car state. The car state can be taken as ground truth as the vehicle is reporting what it is doing. Determining correctness of the labeled face pose data is difficult. For example during the drives two of the following situations were observed in Tables 5.1 and 5.2. In Table 5.1, the face pose was labeled as between the Center and Instrument landmarks. Similarly in Table 5.2, the face pose is between the Center, Instrument, and Rear Mirror landmarks. In both cases is it difficult to determine what specifically the driver is looking at. In both cases the driver was most likely looking at a point between each landmark.

Table 5.2: Unclear Head Pose example 2

Landmark	State
Center	0.5
LeftMirror	0.0
RightMirror	0.0
RearMirror	0.5
Entertainment	0.0
Instrument	0.5
Yaw	-0.04
Pitch	-0.142

Another case where this style of face pose labeling needs improvement were situations where more cases were reported as possible. In Table 5.3, the software correctly labeled the driver as looking at the Right Mirror landmark while all others except the Rear Mirror reported as possible. Due to the pitch being within the margin of the pitch used to identify the other points, the software labelled the other landmarks as possible.

Table 5.3: Case with bad output

Landmark	State
Center	0.5
LeftMirror	0.5
RightMirror	1.0
RearMirror	0.0
Entertainment	0.5
Instrument	0.5
Yaw	-0.834
Pitch	-0.082

These examples highlight the importance of finding software solutions that will correctly label the face pose. LiDD was developed for this thesis to demonstrate the utility that even a less computationally difficult solution could bring usable results. By combining an open source software solution with a straightforward labeling solution a simple face pose labeling could be produced that worked in many cases.

There are examples in the data in which what action the driver is taking is clearer. Table 5.4 describes a section of the drive near the end of the City route at location O. In this frame the driver is at a stop, left turn signal on, with the heads yaw rotated past the landmark of the left mirror. Preceding this Situational frame, is a series of frames where the vehicle state is not changing but the drivers head is sweeping from left to right waiting for traffic to clear to make a left turn.

Table 5.4: Driver stopped turning left City Drive 3

Landmark	State	Vehicle	State
Center	0.0	Steering Angle	45.90
Left Mirror	0.0	BRAKE_PRESSED	1
Right Mirror	0.0	GAS_PRESSED	0
Rear Mirror	0.0	WHEELS_MOVING	0
Entertainment Center	0.0	RIGHT_BLINKER	0
Instrument	0.0	LEFT_BLINKER	1
Yaw	-0.621	GEAR_SHIFTER	D
Pitch	-0.284	CAR_SPEED	0.0
		XMISSION_SPEED	0.0
		XMISSION_SPEED2	0.0
		MAIN_ON	0

Another example is described in 3.5. In this frame the driver is looking at the right mirror while driving at highway speeds checking if the right lane is clear. This frame was at location D of the Highway route during a merge onto the exit lane. Similarly to the example of Table 5.4, the frames preceding the frame in Table 3.5 show the drivers head rotation right, the turn signal activating, and the driver changing speed to merge.

5.3 Determining Correct Driver Actions

Using external factors to determine if the driver is acting accordingly will be difficult. The following example helps explain these difficulties. Consider Figure 5.3, this situation happened during the Highway experiments. While the driver was on the entrance ramp for the highway, the drivers head was turned to face the blue square area looking ahead of where the car is currently going. The driver also checked the green square area briefly to check for traffic that the driver would be interacting with in the merge lane. This is a check that isn't required, but rather informs the driver of what conditions to expect in the merge lane. In situations where the driver looked at the green area, the driver had been watching the blue area and had determined it was safe to glance at the green area while still moving. An algorithm that is only watching if the driver is looking ahead would miss this situation.

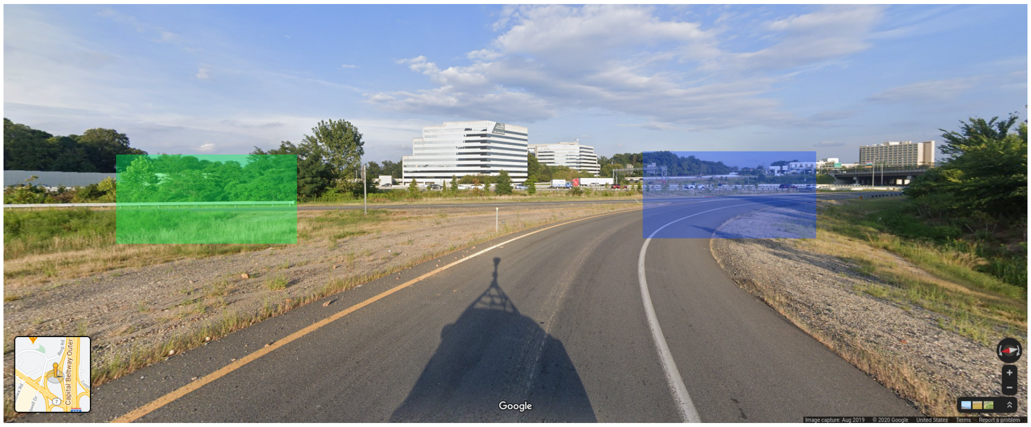

Figure 5.4: View at location C used in Highway drives (image not from experiment) [2]

This thesis focused on experiments in which the driver was acting correctly. The manual action of labeling the fused data as correct or incorrect in the situation was outside the scope of this thesis and would have been difficult with the set up as is. Including additional information would make this task easier. For example collecting the image from that frame with a forward facing image of the state of the world in front of the car would aid in the labeling of this data.

5.4 Repeat-ability

Given that these experiments are done in real world conditions, repeating a drive exactly as it happened is next to impossible. There are too many external factors in play in order to do a full recreation of a given experiment. For example consider the events in the suburban experiment, Table 5.5. The first drive was only a approximately 2.8 miles, and had three different outcomes of the tracked events. These events do not include other useful information such as interactions with other cars, actual state of the stop light when the driver interacted with the light.

Table 5.5: Suburban Collection Events

Tracked Events	Drive 1	Drive 2	Drive 3	Drive 4 Night
Stop at first stop light	X		X	
Stop at second stop light	X			
Stop for left turn w/o stop light		X		
Slowed for pedestrian crossing			X	X

The suburban environment showed that even focusing on a few factors it becomes very difficult to repeat a given experiment. In the highway environment, there were fewer stop lights but more variability from traffic conditions. For example on the return trip at location B Figure 4.2, the space from the exit ramp to the left turn light made it difficult depending on traffic to cross three lanes and make the turn. Out of three attempts only once did traffic conditions allow for the left turn to be made.

Of the three highway experiments, two were done back to back with similar traffic conditions. These traffic conditions are difficult to describe and track traffic interactions. Foe example during the first half of the drive, there is a major exit to another highway. In the first experiment there was a slow down at this location in which the driver had to slow down and look for an opening in the next lane to continue making progress. In the second experiment the slow down was not present and the driver was able to continue to make progress at highway speeds. These events and others are captured in Table 5.6. Tracking lanes changes became difficult and were not consistent from one experiment to the next.

Table 5.6: Highway Collection Events

Tracked Events	Drive 1	Drive 2	Drive 3
Stop at first stop light	X	X	X
Slow down at major exit	X		
Merge with cars at turn around E	X		X
Merge with cars at turn around D			
Made three lane change to left turn		X	

Table 5.7 references the tracked events in the city experiments. This environment is very complicated but has more easily track-able events than the highway environment. There were instances of having to wait to turn for pedestrians to clear the intersection. Other events were left out of tracking included a police car sitting at an intersection that had to be driven around, and lane changes made during each drive. The parking lot that was chosen as the start point had significant traffic and by the time the route was finished generally was in a different state then it was at the beginning. As such one of the three drives started and ended in a different parking spot.

Table 5.7: City Collection Events

Tracked Events	Drive 1	Drive 2	Drive 3
Stop at stop light B			X
Stop at stop light C		X	X
Stop at stop light D		X	
Stop at stop light E			
Stop at stop light F	X	X	X
Stop at stop light G	X	X	X
Stopped for pedestrians at G	X		
Stop at stop light H	X	X	
Stop at stop light I	X	X	
Stop at stop light J			X
Stop at stop light K			X
Stop at stop sign M	X	X	X
Stop at stop sign N	X	X	X
Stop at stop sign O	X	X	X
Stop at stop light P	X		
Stop at stop light Q		X	
Stop at stop light R			X
Stop at stop light S			
Stop at stop light B return			X
Stopped for pedestrians at B return	X		
Park in same location as start		X	X

5.5 Safety

Safety was also a very import factor in these data collections. During the first data collection, some unsafe behavior was attempted. This included the driver looking at a cell phone while stopped, looking away from center over the steering wheel while driving straight for an extra second or two, only after checking that the road ahead was clear. It was difficult to achieve some of these because they could be very dangerous particularly in a high speed highway environment, and as such were not attempted in the highway or city data collect. During each drive, the driver attempted to maintain safe driving practices such as turn signal use, blind spot checking, and no phone use.

5.6 Light Conditions

Light conditions will vary depending on time of day, weather, and whether the vehicle is inside a building. In the Suburban data collect the calibration step was not possible at night without the interior lights on because of the lack of external light. Further during that drive, there were periods of time where there was not enough light to capture where the face was, reference Figure 5.6. The face was only illuminated when there was good street lighting or an oncoming car with it's headlights on, reference Figure 5.6.

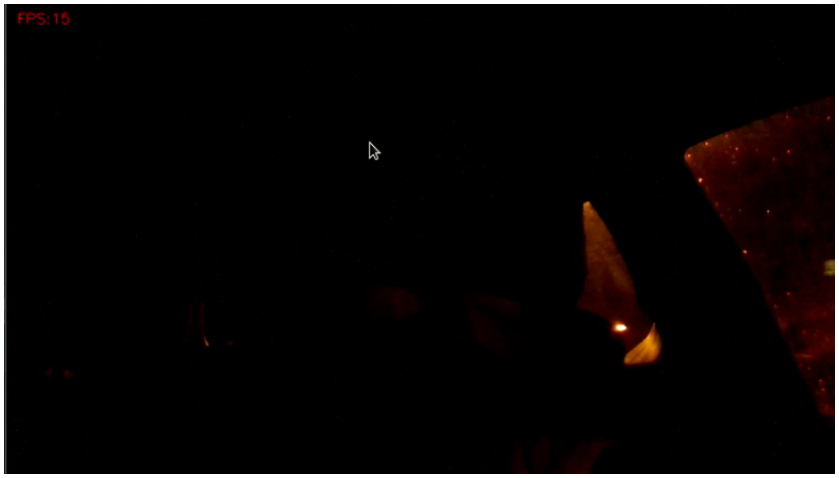

Figure 5.5: Illumination from area without direct street light

Camera choice would come into play in these scenarios. A higher end camera that can adjust to lower levels of light could provide better performance. A 4 watt infra red illuminator was evaluated to provide non invasive lighting for the camera. This did not provide enough infrared light to sustain a face pose as the driver looked at the different points in the car. If the light was held close, it did provide enough infra red light to allow OpenFace to provide a face pose. A setup with near IR illumination from multiple sources at different angles could provide enough light to keep the face illuminated to maintain a face pose.

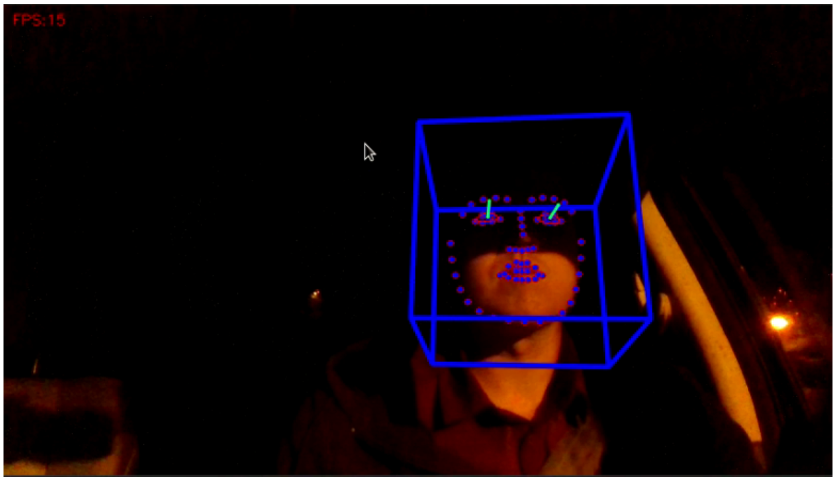

Figure 5.6: Illumination from street light

Chapter 6: Future Work

This thesis provides a simple method for combining information about the driver and the vehicle, but does not propose a solution for obtaining information about the current state of the world outside the vehicle. Information about the world outside the vehicle provides critical context for understanding driver actions. More research in the collection and understanding of this external data would aid in research and development of systems that provide driver situational awareness. The inclusion of this data will aid in better data sets representing driver behavior, acting as ground truth for decisions the driver makes.

LiDD would benefit from more research and development in determining additional information that would improve the created data sets. For example, research into the driver's body pose would need an additional camera pointing at the drivers side like in the State Farm data set [3]. Additionally, capturing an image of the driver and of the road ahead at each frame would provide extra context of what the driver was doing at that given frame.

More experimentation with LiDD would be beneficial. These experiments should include other drivers of varying characteristics. This would allow for determination of improvements that could be made to driver data collection tools and methodologies. At the time of this writing, the virus Covid-19 was still active and for safety concerns, other drivers were not considered.

Research into methodologies of recreating unsafe driver behavior in a safe and ethical manner also needs to be undertaken. A data set of good driving behavior while beneficial, is not as impactful without data showing bad driving behavior. The goal of much of the research in this area is focused on keeping the driver safe. If collected data sets do not contain data from bad driving behaviors, then research and development of solutions to detect and prevent bad driving behaviors cannot happen.

Research into camera options for mounting and improved collection capabilities would be beneficial. This research was limited to one camera and commercially available mounting options. Other cameras could provide better solutions in the areas of positioning, field of view, low light performance, and mounting.

Chapter 7: Conclusion

LiDD provided significant data from real world drives without controlled external variables. A difficulty found in this process is being able to parse this data to understand what is happening. Without additional context of where the vehicle is or some other kind of labeling of conditions external to the vehicle, the data can be hard to understand. The collected data and experiments performed have shown that there is utility here that should continue to be explored. The head pose labeling, while simple and not as accurate as other research done in this area, still provided a sufficient labeling of data to allow for analysis of produced data to take place. With the publication of this research comes the hope for the creation of more data sets enabling more research into this area for a safer road tomorrow.

Bibliography

Bibliography

[1] D. B. Craig Smith, *THE CAR HACKER'S HANDBOOK A Guide for the Penetration Tester.* no starch press, 2016.

[2] Google. Google maps. [Online]. Available: https://www.google.com/maps/@38.9127309,-77.2191173,3a,75y,99.43h,88.48t/data=!3m6!1e1!3m4!1s20t6jnkZmKVMCVbIBlza5g!2e0!7i16384!8i8192

[3] eerimoq. State farm distracted driver detection. [Online]. Available: https://www.kaggle.com/c/state-farm-distracted-driver-detection

[4] M. B. Sims. State farm distracted driver detection. [Online]. Available: https://www.autoinsurance.org/driving-hazards/

[5] M. Leekha, M. Goswami, R. R. Shah, Y. Yin, and R. Zimmermann, "Are you paying attention? detecting distracted driving in real-time," in *2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM)*, 2019, pp. 171–180.

[6] J. Jo, "Vision-based method for detecting driver drowsiness and distraction in driver monitoring system," *Optical engineering*, vol. 50, no. 12, pp. 127 202–127 202, 2011.

[7] S. Shan, M. F. Faisal, S. Rezaul Haque, and P. Saha, "Iot and computer vision based driver safety monitoring system with risk prediction," in *2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2)*, 2018, pp. 1–4.

[8] C. C. Ukwuoma and C. Bo, "Deep learning review on drivers drowsiness detection," in *2019 4th Technology Innovation Management and Engineering Science International Conference (TIMES-iCON)*, 2019, pp. 1–5.

[9] Y. Abouelnaga, H. M. Eraqi, and M. N. Moustafa, "Real-time distracted driver posture classification," 2017.

[10] P. R. Tadas Baltrušaitis and L.-P. Morency, "Openface: An open source facial behavior analysis toolkit," 2016.

[11] Y. C. L. Tadas Baltrušaitis, Amir Zadeh and L.-P. Morency, "Openface 2.0: Facial behavior analysis toolkit," 2018.

[12] Intel. Intel realsense depth camera d435. [Online]. Available: https://www.intelrealsense.com/depth-camera-d435/

[13] C. Shop. (2020) Canusb. [Online]. Available: https://canusbshop.com/index.php?route=product/productproduct$_id = 51$

[14] gridconnect. (2020) Can to obd2 cable. [Online]. Available: https://www.gridconnect.com/products/can-to-obd2-cable

[15] hardbyte. python-can. [Online]. Available: https://github.com/hardbyte/python-can/tree/master

[16] eerimoq. python-can. [Online]. Available: https://github.com/eerimoq/cantools

[17] comma.ai. (2020) opendbc. [Online]. Available: https://github.com/commaai/opendbc

Curriculum Vitae

Christopher Seebeck received his Bachelor of Science in Computer Science from Old Dominion University in 2013. He went on to receive his Master of Science in Computer Science at George Mason University in 2020. He has continued to work as a Software Engineer in the field of Autonomous Vehicles.