AUTOREC - AN AUTOMATED DATA LOGGING SYSTEM FOR STUDYING IN-CAR EXPERIENCE

A Thesis
Presented to the Faculty of the Graduate School
of Cornell University
in Partial Fulfillment of the Requirements for the Degree of
MSc.

by
Yue Wang, Jingxuan Zhang

May 2020

ABSTRACT

We are looking forward to designing an in-car recording system with multiple data streams that can help researchers create a new driving experience for the future.

With AutoRec, a recording system embedded in the car, we can record a variety of data ranging from the car's geolocation to the physical behavior and biological data of the driver and the passenger. Through timestamp all these different streams of data, we get a comprehensive idea of all the events occurring in the vehicle. Using these various types of data, we find efficient ways to enhance peoples' overall interaction with vehicle. And this system allows all possible extensions and applied for different scenarios, including real driving and autonomous driving experience in simulation room – all these applications will be detailed illustrated in results section.

This is the final report for our Specialization Project, which is a two-semester project required for the Connective Media Master program. This project was completed by Yue Wang and Jingxuan Zhang, directed and guided by Professor Wendy Ju and other faculty members at Cornell Tech.

ACKNOWLEDGEMENTS

This research was under the guidance of Pr. Wendy Ju. We thank our colleagues from FAR Lab who provided insight and expertise that greatly assisted the research.

We thank Ilan Mandel, David Goedicke for assistance with experiment tutorials and technical support, and Sven Krome for comments that greatly improved the manuscript.

CHAPTER 1
INTRODUCTION

The main purpose of our study is to explore new ways to improve peoples' in-car experience. It is a very broad and exploratory field that is open to all kinds of possibilities. The first step of our plan is to gather as much information as we can from all the activities taking place in the vehicle. Therefore, we wish to build a system that will be able to support a wide range of future experiments by incorporating data streams of various types into the AutoRec system.

Currently, we have finished building two main kinds of data streams: one recording the data of the vehicle itself, including the GPS location of the car, the CAN-bus data, and the data from the IMU (Inertial Measurement Unit) sensor, and the other video-recording the behavior of the driver and the passengers in the car, through multiple GoPro cameras installed onto the vehicle. More data streams have also be added under different request, such as iPad display, pupil tracking and voice agent in order to record more aspects of in-car activities.

There are two research questions that we are focusing on: building a uniformed dashboard that integrate all streams of data in the AutoRec system and examining the AutoRec system in two different scenarios: one is real in-car passengers interaction called as ride-sharing interaction study, another is simulated autonomous driving with phone usage named as crosswalk cooperation driving simulator study.

CHAPTER 2
RELATED WORK

One of the most important work that lays the foundation of the problem we want to solve now is The Needfinding Machine, written by Nikolas Martelaro and Wendy Ju, published on HRI '17 Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction, on pages 355-356.

The paper mainly discusses how social robots can help researchers design new ways in which people interact with ubiquitous computing and ubiquitous robotics. In the paper, researchers act through a need finding machine to communicate with the users and observe their behavior to understand their potential needs and thus, find new ways of interaction.

A need-finding machine that is discussed in the paper is called DJ Bot, an interactive music agent. The researchers used DJ Bot to remotely control the music played in the car and communicate with the users to find out about why they are listening.

We are planning to take a similar approach to decide what kind of new interactions can we add to make driving more interesting. When working on the extending the system, major work relies on referring documentation of certain hardware, for example Rapsberry Pi, Adafruit, Go-Pro, etc.

Furthermore, we evaluate the possibility of AutoRec system on Voice Agent Application, to prove the AutoRec flexibility on enhancing future autonomous car human centred design. Key articles helping us evaluate the system needs are The effects of social interactions with in-vehicle agents on a driver's anger level, driving performance, situation awareness, and perceived workload, written by Myounghoon Jeon, Bruce N. Walker, Thomas M. Gable.

In this paper, it mainly evaluate how in-car voice agent would interact with drivers' road rage. The voice agent has been designed according to different prompts, including emotion regulation (ER), situation awareness (SA) and no agent mode. All these would display two speech style, command and directive.

As the most essential part is to induce participants' anger level and evaluate them. In their experiment, they chose frequently-used affect induction methodology by Bodenhausen, asking participant to write a description of past emotional experience, and researchers would choose on to let participant remember as clearly as possible, also filling an emotional state questionnaire. Then participant would drive in a simulator car to experience a specific scenario. As a result, the differentiation of questionnaire would be shown as how voice agent help alleviate road rage.

If applied this experiment on our system, the result would be more straightforward, and won't be affected by the deviation due to measure by questionnaire filling by participant himself, rather, we would use our system to passively measure the level of road rage based on the driving behavior, facial information and how participant reacting to voice agent.

The AutoRec system is inspired by these studies on in-car driving experience and is built to facilitate further studies that explore the field of corporative driving and automatic driving.

CHAPTER 3

[APPROACH — METHOD — IMPLEMENTATION]

3.1 Method

We will take the research through design approach in this project. Research through Design (RtD) means that we will gain new understanding of the problem that we are trying to solve by learning during the design process. The end goal of our project is to create a new in-car driving experience by exploring the new ways in which humans can interact with the car while driving.

The following is the steps we have took and accomplished in this semester:

1. Setting up the basic in-car system

Our first design about the system is that, AutoRec is an embedded system in car, which collects geolocation, CAN-bus data and video information both inside the car and surrounding. Thus, our first step is to set up the four cameras target at different information – the driver, passenger, front-view and rear view, using GoPro mounted at four different places and angles.

Also, we obtain the car information through the CAN-bus device which capture the accelerators, brakes data. In order to provide a synchronized system, we connect the data stream from all the outputs into Quadview.

2. Data Processing and Data Logger

As far as we set up the car, we have the data streams from multi-resources, the video, location, driving status, which provides us an open-ended platform to build-on. However, the data sources are complicated and not user-friendly for analyst to work on. Therefore, in this step, we aim to integrate all data stream from different sensor, different data type, and process

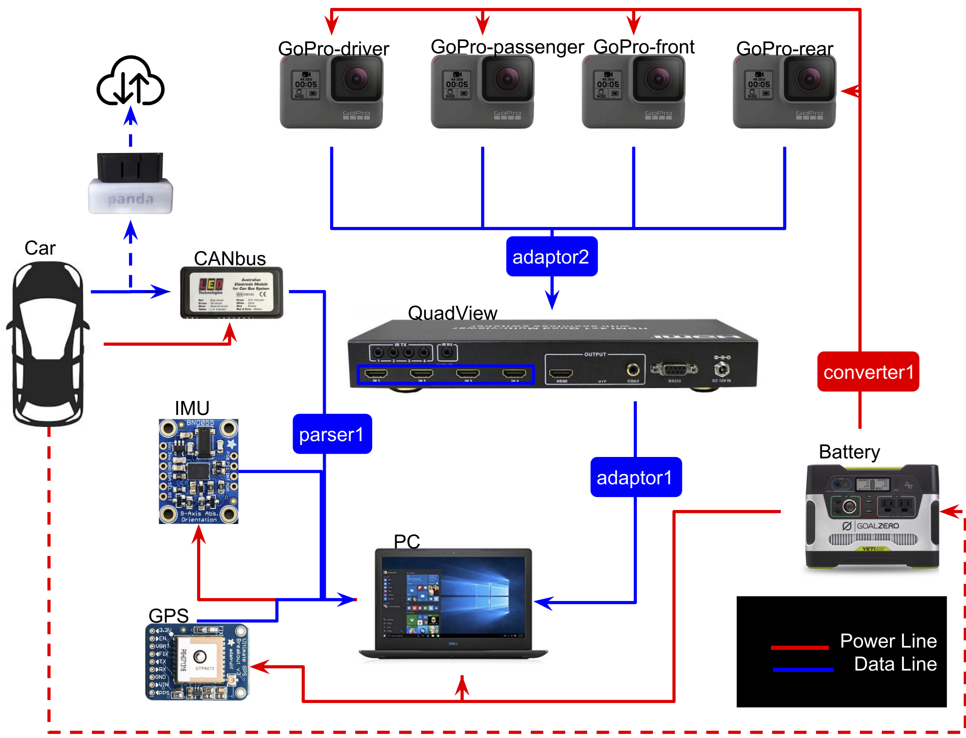


Figure 3.1: Implementation Overview for AutoRec.

them through our data-logger scripts. (Git Repo: `https://github.com/FAR-Lab/toyota-data-logger`)

Detailed implementation and result would be illustrate in next section.

3. Extension of the System and Dry-run experiment

First and second round testing is driving on Roosevelt Island. And after obtaining the data, we evaluate how complicated this experiment procedure was, as right now, we use a shell scripts to start and boost other python scripts to read data from sensors, and make plan for next semester to adding time-stamps to all data stream and visualization. After going through extensive literature review and research on different needs for in-car experiment, we decided to extend the

AutoRec system with eye tracking feature, so as participant's attention can be better captured. In order to solid this feature, we also considered to integrate with face-recognition based on the video data from GoPro camera.

3.2 Implementation

3.2.1 Part 1: Hardware Embedded System

The final version of AutoRec system is illustrated in this figure (See Figure 3.1).

Camera

The interior cameras are GoPro Hero 4 cameras set to the Medium viewing angle, 1080p 30 fps. These cameras are connected via HDMI to a 4 input video stitcher (video quad). The video quad outputs a 2x2 1080p 30fps video stream which is then input to the data recording laptop using a Inogeni HDMI to USB 3 video capture card. The USB 3 cable for the capture card is then plugged into a USB 3 port on the computer. Note that it should be the only device on the USB 3 port (it should not be going through a USB 3 hub) in order to have enough bandwidth.[3]

After several dry run, we also re-consider the angles of camera and re-orient them. Since as the research to observe participants, not only their face expressions but also their body gestures.

CANbus

CANbus is an automation fieldbus commonly used in the automotive industry as the main network bus to allow communications between the many on-board ECUs on modern vehicles.

The Linux kernel has native CAN bus support at network layer, with a lot of drivers for both embedded and USB CAN bus controllers, so it is fairly easy to add a CAN bus interface to any Linux laptop and have a playground with it.

We use a USB2CAN device to collect data from the ODB port in the front driver side of the car. The cable connects from the car's OBD to a DB-9 connector into the main body of the USB2CAN. It then connects to the USB hub in the computer box.

The actual usage model of CAN-bus (See Figure 3.2): each packet has single address of either 11 or 29 bits, and a maximum payload of 8 bytes. The idea behind this is that address are used to identify the type of frame (i.e. what is the meaning of the payload), and all nodes on the bus can send and receive any type of frame. As the bus can have a lot of traffic on it, but most nodes only care about some specific frame IDs and does not have the computational resources to handle the full traffic, each CAN-bus controller implements some sort of "acceptance filter", which is usually a bitmask to decide which range of frame addresses should be received by the node, while all the others are discarded.[4]

Another more straight-forward approach is using PandaCan[2] to connect the CAN-bus portal with mobile or computer through Wifi. And the platform itself provides cloud storage of driving experience, and interface to parse the CANbus data.3.3

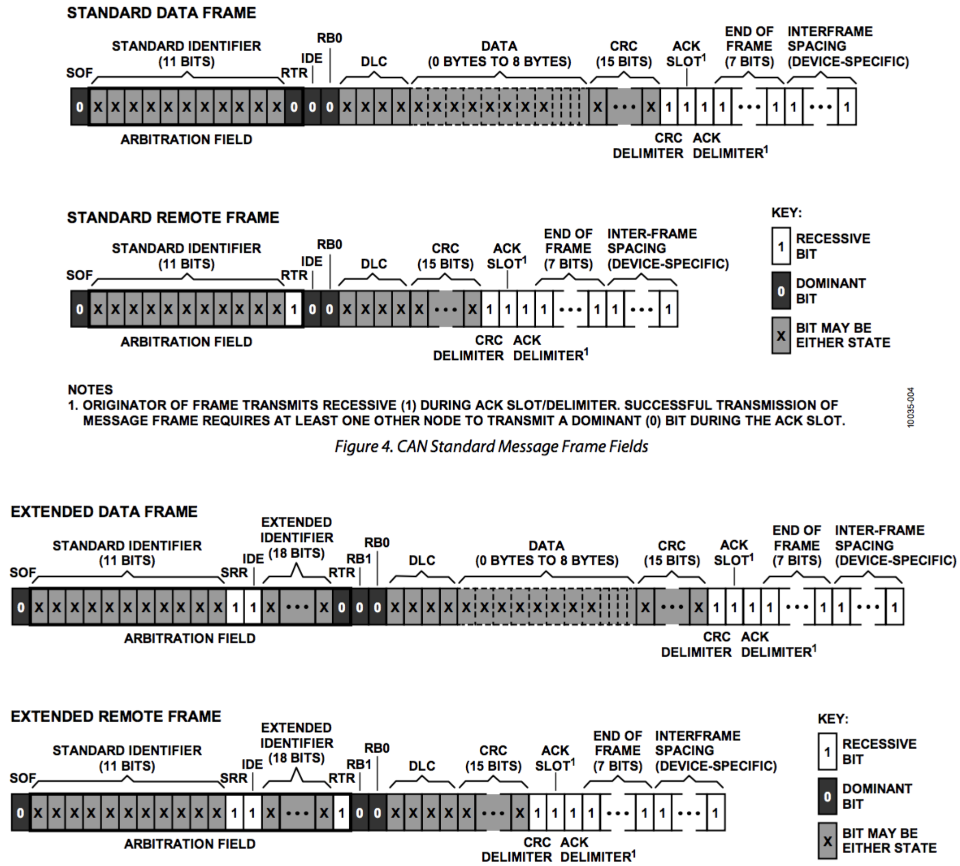


Figure 3.2: CAN Extended Message Frame Fields [4]

Power

A GoalZero Yeti 150 provides power to the video equipment, computer, and 4G router. This acts both as power distribution and as an uninterrupted power supply. 12V power is used for the video equipment and 4G router. A 3-port 12V hub distributed power from the GoalZero to the various devices.[1] The 120V AC port on the GoalZero is used for powering the laptop. Note that only a 60W laptop power supply should be used. When running the study, ensure that the 12V power cable running from the read of the Prius is plugged into the GoalZero. This keeps the GoalZero charged rather than it running purely on

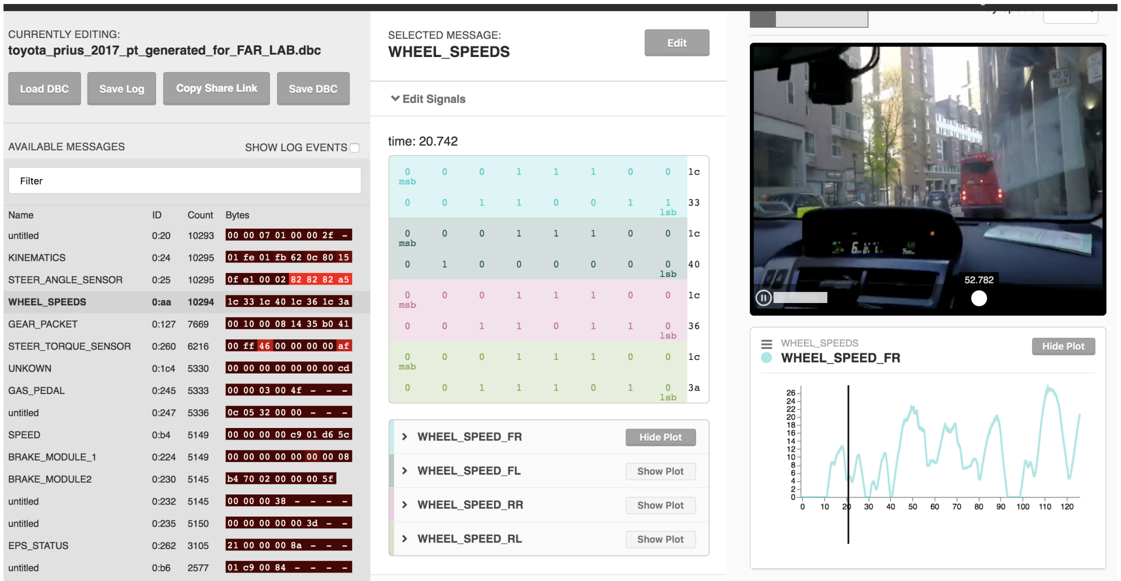


Figure 3.3: Parsing CANbus data through PandaCAN: on the left part is the raw data; on the middle part is the format of selected data; on the right-top is the video from mobile camera; on the right-down is the visualization of selected data (here is wheel speed)

battery power.

Eye Tracking

In order to fulfill the goal of auto recording participant's behavior, we add this extra dimension of system to capture participant's eye movement and tracking pupil's staring direction to measure how would contextual environment would interfere participant's attention. Through this way, the result would be more straight forward than investigating the video information from participant's angle. The rationality behind eye tracking is using three micro-camera: Two Eye Cameras:The eye camera arm slides in and out of the headset frame, which can slide the eye camera arm along the track. World Camera: The world camera aimed to focusing the view from participant, and the output from all cameras will be projected to calculate the real eye movement based on the distance from eye camera to pupil.

PART TWO: Data Logging System

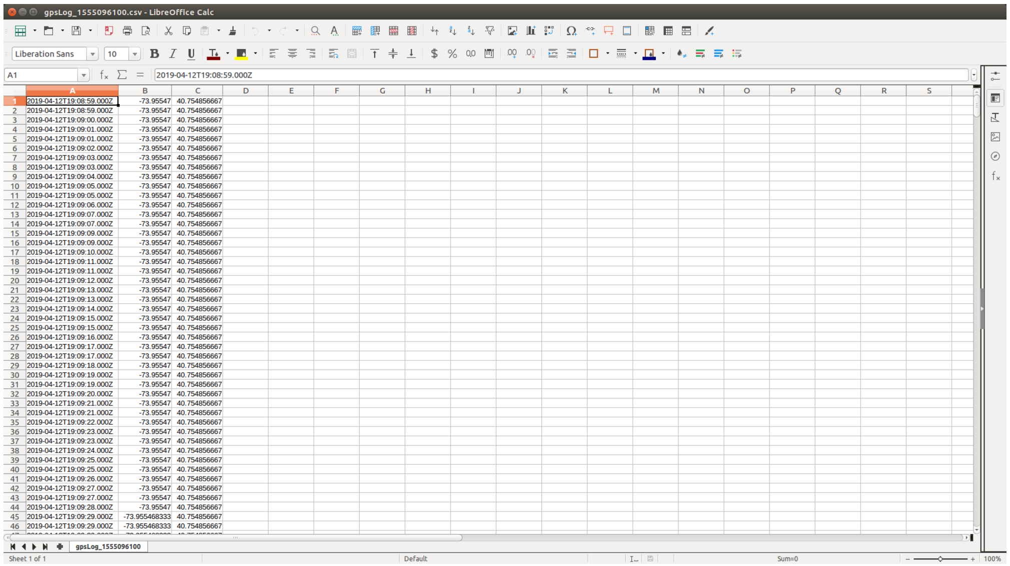
The data logging system records the data collected through the hardware embedded system as described in the previous paragraphs. This data logging system is based on an existing logging system build by Nikolas Martelaro from Accenture Technology Labs, which originally runs on the OS X operating system and records data of a Toyota vehicle. Because in our project, we will be migrating the setup to a Linux environment, changes need to be made in order to migrate the original logging system to our new platform.

The data logging system mainly consists of four main parts: recording geolocation data, recording CAN-bus data, recording IMU data, recording video data, and eye tracking data.

Currently, the Python code for GPS, IMU, and CAN-bus data recording can be started by one Shell script: start_all.sh.

GPS Logger

GPS data is logged with the script *gpsLogger_linux.py*, which takes information collected through a GPS sensor that runs on an Adafruit board and is located on the roof of the vehicle. The Python script will save the real-time longitude and latitude of the car in CSV format (See Figure 3.4).



	A	B	C
1	2019-04-12T19:08:59.000Z	-73.95547	40.754856667
2	2019-04-12T19:08:59.000Z	-73.95547	40.754856667
3	2019-04-12T19:09:00.000Z	-73.95547	40.754856667
4	2019-04-12T19:09:01.000Z	-73.95547	40.754856667
5	2019-04-12T19:09:01.000Z	-73.95547	40.754856667
6	2019-04-12T19:09:02.000Z	-73.95547	40.754856667
7	2019-04-12T19:09:03.000Z	-73.95547	40.754856667
8	2019-04-12T19:09:03.000Z	-73.95547	40.754856667
9	2019-04-12T19:09:04.000Z	-73.95547	40.754856667
10	2019-04-12T19:09:05.000Z	-73.95547	40.754856667
11	2019-04-12T19:09:05.000Z	-73.95547	40.754856667
12	2019-04-12T19:09:06.000Z	-73.95547	40.754856667
13	2019-04-12T19:09:07.000Z	-73.95547	40.754856667
14	2019-04-12T19:09:07.000Z	-73.95547	40.754856667
15	2019-04-12T19:09:09.000Z	-73.95547	40.754856667
16	2019-04-12T19:09:09.000Z	-73.95547	40.754856667
17	2019-04-12T19:09:10.000Z	-73.95547	40.754856667
18	2019-04-12T19:09:11.000Z	-73.95547	40.754856667
19	2019-04-12T19:09:11.000Z	-73.95547	40.754856667
20	2019-04-12T19:09:12.000Z	-73.95547	40.754856667
21	2019-04-12T19:09:13.000Z	-73.95547	40.754856667
22	2019-04-12T19:09:13.000Z	-73.95547	40.754856667
23	2019-04-12T19:09:14.000Z	-73.95547	40.754856667
24	2019-04-12T19:09:15.000Z	-73.95547	40.754856667
25	2019-04-12T19:09:15.000Z	-73.95547	40.754856667
26	2019-04-12T19:09:16.000Z	-73.95547	40.754856667
27	2019-04-12T19:09:17.000Z	-73.95547	40.754856667
28	2019-04-12T19:09:17.000Z	-73.95547	40.754856667
29	2019-04-12T19:09:18.000Z	-73.95547	40.754856667
30	2019-04-12T19:09:19.000Z	-73.95547	40.754856667
31	2019-04-12T19:09:19.000Z	-73.95547	40.754856667
32	2019-04-12T19:09:20.000Z	-73.95547	40.754856667
33	2019-04-12T19:09:21.000Z	-73.95547	40.754856667
34	2019-04-12T19:09:21.000Z	-73.95547	40.754856667
35	2019-04-12T19:09:22.000Z	-73.95547	40.754856667
36	2019-04-12T19:09:23.000Z	-73.95547	40.754856667
37	2019-04-12T19:09:23.000Z	-73.95547	40.754856667
38	2019-04-12T19:09:24.000Z	-73.95547	40.754856667
39	2019-04-12T19:09:25.000Z	-73.95547	40.754856667
40	2019-04-12T19:09:25.000Z	-73.95547	40.754856667
41	2019-04-12T19:09:26.000Z	-73.95547	40.754856667
42	2019-04-12T19:09:27.000Z	-73.95547	40.754856667
43	2019-04-12T19:09:27.000Z	-73.95547	40.754856667
44	2019-04-12T19:09:28.000Z	-73.95547	40.754856667
45	2019-04-12T19:09:29.000Z	-73.955468333	40.754856667
46	2019-04-12T19:09:29.000Z	-73.955468333	40.754856667

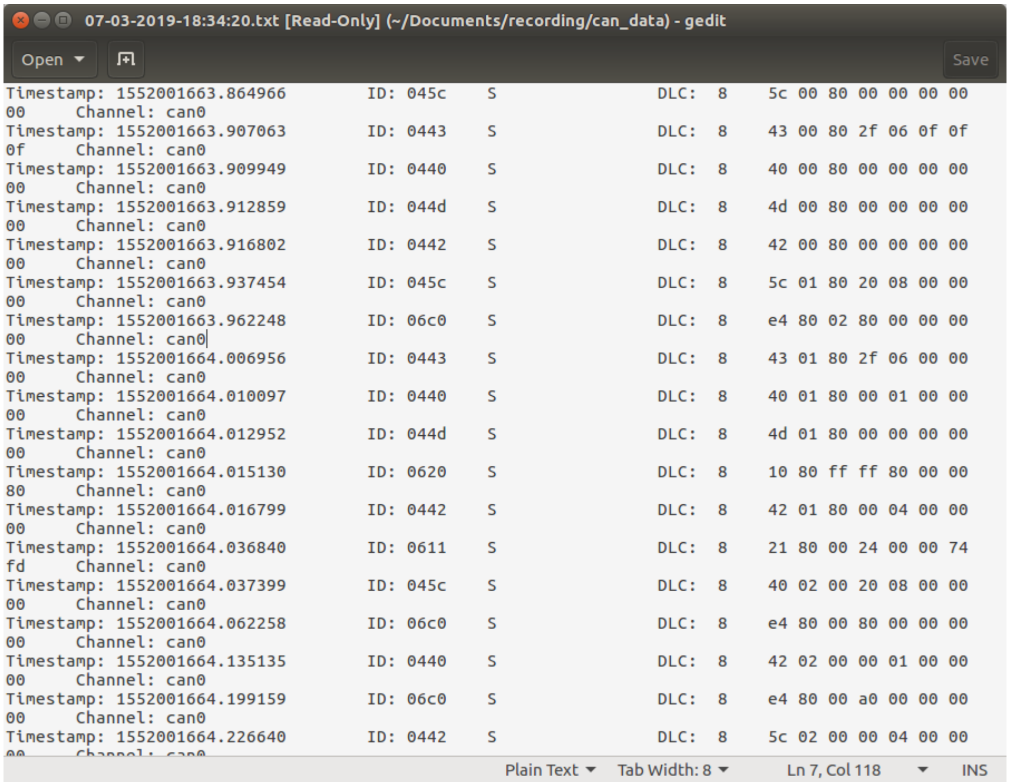
Figure 3.4: Data recorded from GPS logger.

IMU Logger

IMU (Inertial Measurement Unit) data collection is implemented in *imu_logger.py*. The script will collect IMU data from the sensor and store the results in CSV format.

CAN-bus Data Logger

CAN-bus data collection is implemented in *toyotaCan.py*, which also stores the data as a CSV file. The Python script can also be called by running the Shell script *start_can.sh* (See Figure 3.5).




```
Timestamp: 1552001663.864966        ID: 045c    S               DLC:  8    5c 00 80 00 00 00 00
00      Channel: can0
Timestamp: 1552001663.907063        ID: 0443    S               DLC:  8    43 00 80 2f 06 0f 0f
0f      Channel: can0
Timestamp: 1552001663.909949        ID: 0440    S               DLC:  8    40 00 80 00 00 00 00
00      Channel: can0
Timestamp: 1552001663.912859        ID: 044d    S               DLC:  8    4d 00 80 00 00 00 00
00      Channel: can0
Timestamp: 1552001663.916802        ID: 0442    S               DLC:  8    42 00 80 00 00 00 00
00      Channel: can0
Timestamp: 1552001663.937454        ID: 045c    S               DLC:  8    5c 01 80 20 08 00 00
00      Channel: can0
Timestamp: 1552001663.962248        ID: 06c0    S               DLC:  8    e4 80 02 80 00 00 00
00      Channel: can0
Timestamp: 1552001664.006956        ID: 0443    S               DLC:  8    43 01 80 2f 06 00 00
00      Channel: can0
Timestamp: 1552001664.010097        ID: 0440    S               DLC:  8    40 01 80 00 01 00 00
00      Channel: can0
Timestamp: 1552001664.012952        ID: 044d    S               DLC:  8    4d 01 80 00 00 00 00
00      Channel: can0
Timestamp: 1552001664.015130        ID: 0620    S               DLC:  8    10 80 ff ff 80 00 00
80      Channel: can0
Timestamp: 1552001664.016799        ID: 0442    S               DLC:  8    42 01 80 00 04 00 00
00      Channel: can0
Timestamp: 1552001664.036840        ID: 0611    S               DLC:  8    21 80 00 24 00 00 74
fd      Channel: can0
Timestamp: 1552001664.037399        ID: 045c    S               DLC:  8    40 02 00 20 08 00 00
00      Channel: can0
Timestamp: 1552001664.062258        ID: 06c0    S               DLC:  8    e4 80 00 80 00 00 00
00      Channel: can0
Timestamp: 1552001664.135135        ID: 0440    S               DLC:  8    42 02 00 00 01 00 00
00      Channel: can0
Timestamp: 1552001664.199159        ID: 06c0    S               DLC:  8    e4 80 00 a0 00 00 00
00      Channel: can0
Timestamp: 1552001664.226640        ID: 0442    S               DLC:  8    5c 02 00 00 04 00 00
```

Figure 3.5: Data recorded from CAN-bus data logger.

Video Input

Video streams from the four GoPro cameras are integrated by the QuadView. To record the video on the QuadView, we used a software on the Linux system called Open Broadcaster Software(OBS) to display and record the video. Currently, video recording need to be started manually. Although this is still not too troublesome, we might also consider writing a new script in the future to automate this process.

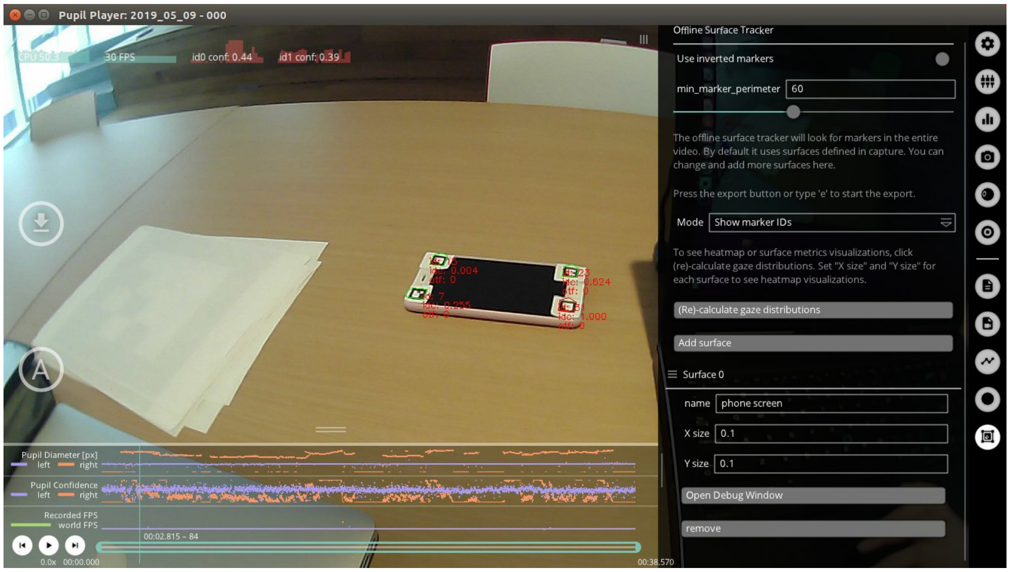


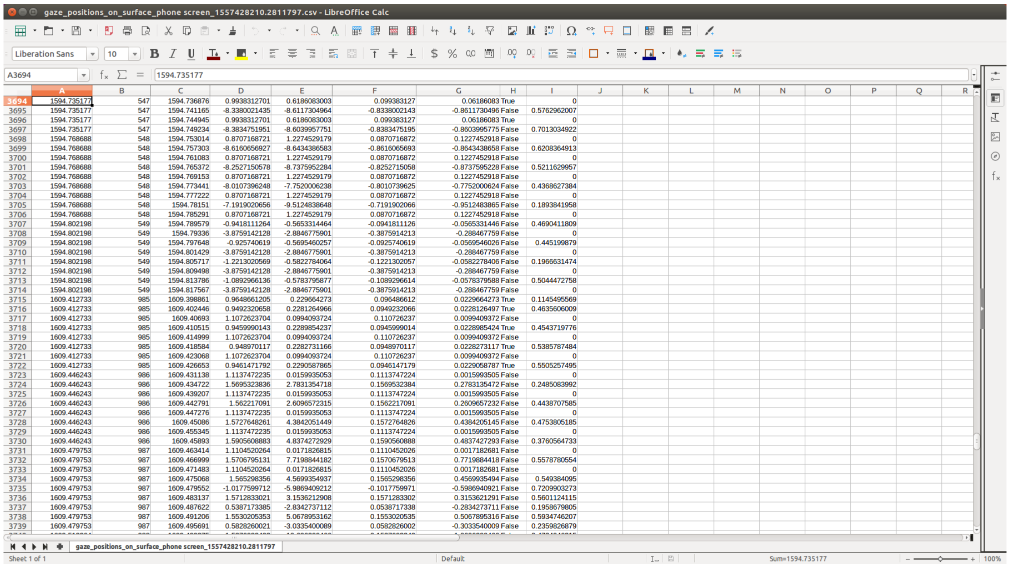
Figure 3.6: Pupil Player replaying surface tracking video.

Eye Tracking

In order to track the participants' eye movement inside the car, we are using devices and software provided by pupil labs (Homepage: `https://pupil-labs.com/`), which include a headset with three cameras (a world camera and two eye cameras as mentioned above) and Pupil, an open-source eye tracking platform. We will mainly be using two components of the Pupil platform: Pupil Capture, to record videos and data from the three cameras, and Pupil Player, to visualize previously recorded data.

Basically, after the user puts on the Pupil headset and start recording with Pupil Capture, the timestamp and location information of the pupils will be recorded along with MP4 files from the cameras.

The main reason why we would want to use eye tracking in our system is that we would like to know if the participants will look at certain places,



	A	B	C	D	E	F	G	H	I
3694	1594.735177	547	1594.736876	0.9938312701	0.6186083003	0.099383127	0.06186083	True	0
3695	1594.735177	547	1594.741165	-8.3380021435	-8.6117304964	-0.8338002143	-0.8611730496	False	0.5762962007
3696	1594.735177	547	1594.744945	0.9938312701	0.6186083003	0.099383127	0.06186083	True	0
3697	1594.735177	547	1594.749234	-8.3834751951	-8.6039957751	-0.8383475195	-0.8603995775	False	0.7013034922
3698	1594.768688	548	1594.753014	0.8707168721	1.2274529179	0.0870716872	0.1227452918	False	0
3699	1594.768688	548	1594.757303	-8.6160656927	-8.6434386583	-0.8616065693	-0.8643438658	False	0.6208364913
3700	1594.768688	548	1594.761083	0.8707168721	1.2274529179	0.0870716872	0.1227452918	False	0
3701	1594.768688	548	1594.765372	-8.2527150578	-8.7375952284	-0.8252715058	-0.8737595228	False	0.5211629957
3702	1594.768688	548	1594.769153	0.8707168721	1.2274529179	0.0870716872	0.1227452918	False	0
3703	1594.768688	548	1594.773441	-8.0107396248	-7.7520006238	-0.8010739625	-0.7752000624	False	0.4368627384
3704	1594.768688	548	1594.777222	0.8707168721	1.2274529179	0.0870716872	0.1227452918	False	0
3705	1594.768688	548	1594.78151	-7.1919020656	-9.5124838648	-0.7191902066	-0.9512483865	False	0.1893841958
3706	1594.768688	548	1594.785291	0.8707168721	1.2274529179	0.0870716872	0.1227452918	False	0
3707	1594.802198	549	1594.789579	-0.9418111264	-0.5653314464	-0.0941811126	-0.0565331446	False	0.4690411809
3708	1594.802198	549	1594.79336	-3.8759142128	-2.8846775901	-0.3875914213	-0.288467759	False	0
3709	1594.802198	549	1594.797648	-0.925740619	-0.5695460257	-0.0925740619	-0.0569546026	False	0.445199879
3710	1594.802198	549	1594.801429	-3.8759142128	-2.8846775901	-0.3875914213	-0.288467759	False	0
3711	1594.802198	549	1594.805717	-1.2213020569	-0.5822784064	-0.1221302057	-0.0582278406	False	0.1966631474
3712	1594.802198	549	1594.809498	-3.8759142128	-2.8846775901	-0.3875914213	-0.288467759	False	0
3713	1594.802198	549	1594.813786	-1.0892966136	-0.5783795877	-0.1089296614	-0.0578379588	False	0.5044472758
3714	1594.802198	549	1594.817567	-3.8759142128	-2.8846775901	-0.3875914213	-0.288467759	False	0
3715	1609.412733	985	1609.398861	0.9648661205	0.229664273	0.096486612	0.0229664273	True	0.1145495569
3716	1609.412733	985	1609.402446	0.9492320658	0.2281264966	0.0949232066	0.0228126497	True	0.4635606009
3717	1609.412733	985	1609.40693	1.1072623704	0.0994093724	0.110726237	0.0099409372	False	0
3718	1609.412733	985	1609.410515	0.9459990143	0.2289854237	0.0945999014	0.0228985424	True	0.4543719776
3719	1609.412733	985	1609.414999	1.1072623704	0.0994093724	0.110726237	0.0099409372	False	0
3720	1609.412733	985	1609.418584	0.948970117	0.2282731166	0.0948970117	0.0228273117	True	0.5385787484
3721	1609.412733	985	1609.423068	1.1072623704	0.0994093724	0.110726237	0.0099409372	False	0
3722	1609.412733	985	1609.426653	0.9461471792	0.2290587865	0.0946147179	0.0229058787	True	0.5505257495
3723	1609.446243	986	1609.431138	1.1137472235	0.0159935053	0.1113747224	0.0015993505	False	0
3724	1609.446243	986	1609.434722	1.5695323836	2.7831354718	0.1569532384	0.2783135472	False	0.2485083992
3725	1609.446243	986	1609.439207	1.1137472235	0.0159935053	0.1113747224	0.0015993505	False	0
3726	1609.446243	986	1609.442791	1.562217091	2.6096572315	0.1562217091	0.2609657232	False	0.4438707585
3727	1609.446243	986	1609.447276	1.1137472235	0.0159935053	0.1113747224	0.0015993505	False	0
3728	1609.446243	986	1609.45086	1.5727648261	4.3842051449	0.1572764826	0.4384205145	False	0.4753805185
3729	1609.446243	986	1609.455345	1.1137472235	0.0159935053	0.1113747224	0.0015993505	False	0
3730	1609.446243	986	1609.45893	1.5905608883	4.8374272929	0.1590560888	0.4837427293	False	0.3760564733
3731	1609.479753	987	1609.463414	1.1104520264	0.0171826815	0.1110452026	0.0017182681	False	0
3732	1609.479753	987	1609.466999	1.5706795131	7.7198844182	0.1570679513	0.7719884418	False	0.5578780554
3733	1609.479753	987	1609.471483	1.1104520264	0.0171826815	0.1110452026	0.0017182681	False	0
3734	1609.479753	987	1609.475068	1.565298356	4.5699354937	0.1565298356	0.4569935494	False	0.549384095
3735	1609.479753	987	1609.479552	-1.0177599712	-5.9869409212	-0.1017759971	-0.5986940921	False	0.7209903273
3736	1609.479753	987	1609.483137	1.5712833021	3.1536212908	0.1571283302	0.3153621291	False	0.5601124115
3737	1609.479753	987	1609.487622	0.5387173385	-2.8342737112	0.0538717338	-0.2834273711	False	0.1958679805
3738	1609.479753	987	1609.491206	1.5530205353	5.0678953162	0.1553020535	0.5067895316	False	0.5934746207
3739	1609.479753	987	1609.495691	0.5828260021	-3.0335400089	0.0582826002	-0.3033540009	False	0.2359826879

Figure 3.7: Gaze positions recorded with Pupil Capture.

such as the rear-view mirror and their phones, while driving and if so, when and for how long. Pupil platform allows us to achieve that is through surface tracking. In order to track surfaces, we first need to define a surface by adding markers (See Figure 3.8) to the borders of an object and register the surface in Pupil Capture. After a surface is registered and surface tracking enabled in Pupil Capture, we will then be able to know if the participant's gaze is landing on the defined surface or not when we replay the recorded video in Pupil Player or simply by checking the CSV file exported by Pupil Player (See Figure **??**).

However, up to now, the eye-tracking part of the system is not yet complete. We have set up the Pupil software and tested the devices. But we are still working on adjusting surface tracking (See Figure 3.9 and 3.7).

CHAPTER 4

RESULTS

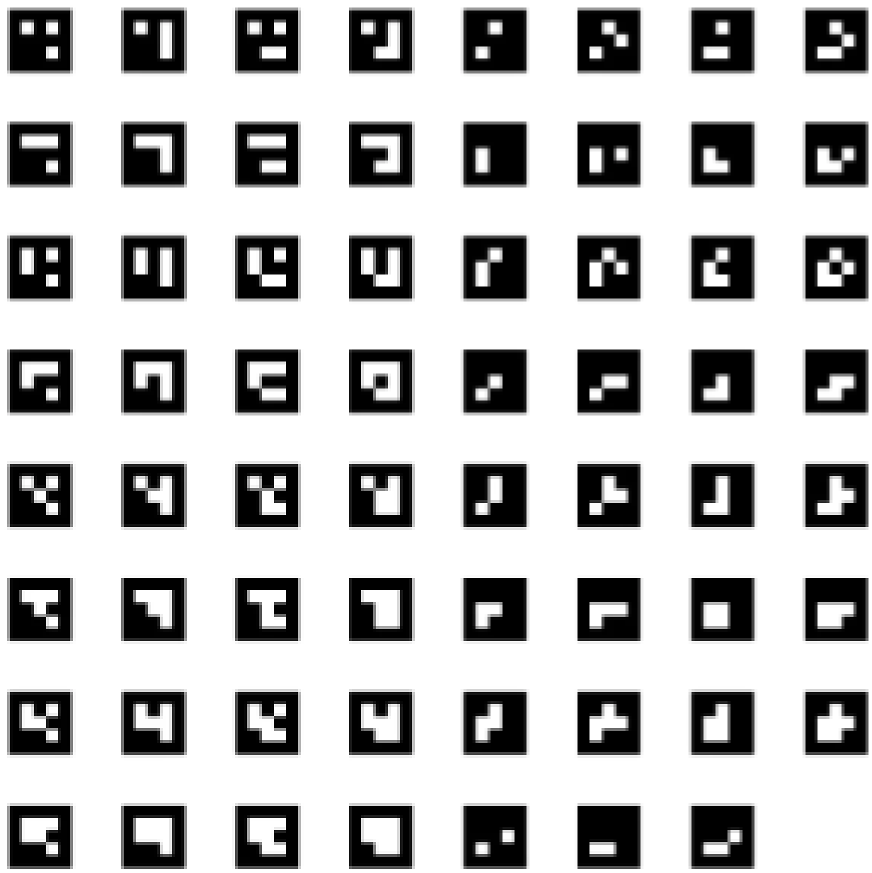


Figure 3.8: Markers used to define surfaces. Picture credit to pupil labs.

After our implementation of AutoRec system, it is essential to put it into real context. Therefore, we work with two different groups of study and test the potential and feasibility of AutoRec system. Even the two scenarios is both related to driving and human-machine interaction, their objectives and environments are entirely different which provide us broader definition and applicable scene. First is real vehicle experience but participant would only be sitting in back row. Second takes place in simulation room where paritipant would be a

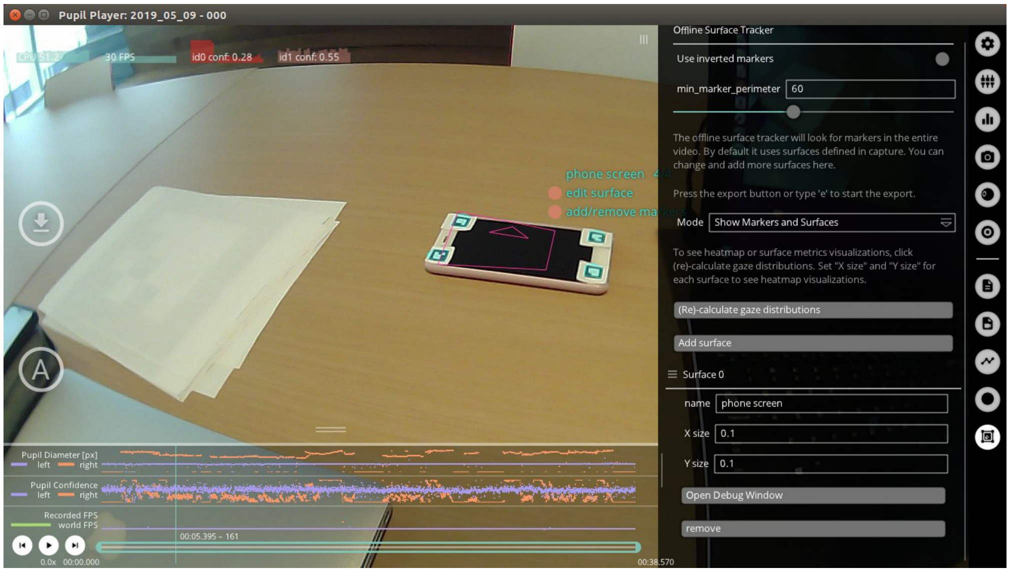


Figure 3.9: Pupil Player replaying surface tracking video.

driver in an autonomous car.

4.1 Roosevelt Island Ride-sharing Study

4.1.1 Objective

First application of AutoRec system is used to study awkward silence for ride sharing experience. The area we want to explore is how to alleviate or break in-car awkward silence through passive or active interaction with car. To narrow down the research scope, we set the scenario as ride-sharing which is popular nowadays. And we only involved in the experiment setup, pivoting and dry run stages, therefore, after recruiting participants, it might undergo iterations on hardware or software updates.

4.1.2 Experiment Setup

Based on AutoRec system, ride sharing study added several extentions: iPad display connected to Quadview, voice command, music player, and allowing remote control of experiment conductor. Hardware setup should follow the steps in implementation section, and depending on different research need, adjust aspect ratio and point of view.

4.1.3 Participant

As this study is real-world driving experience, it involves four participant: one driver, who is not the research subject and does not involve capturing and have no intended interaction with anyone inside vehicle; one actor sitting in back row, who has been told the experiment process, acting as a trigger of awkward silence; one real participant sitting in back row, where there is no restriction and no disclosure of experiment process. The last one is experiment conductor sitting in front passenger seat, and controlling voice command, music choosing survey and music player.

4.1.4 Process

In Roosevelt Island ride sharing study, participant would experience a 30-min in vehicle experience with a stranger (actor) for one round on Roosevelt Island. Whole experiment would take approximately 1 hour, and in the end, participant would be paid for 15 dollar for their volunteering. The ride sharing study mainly focused on the three dimensional interactions: participant, actor

Informational	Recreational
Description of surroundings	Random animations
Traffic status	Surprising sounds
Weather forecast	Vote for music

and vehicle, which could be concluded in two categories: First, before the study, actor would be told trying not to talk to participant as few as he/she could. And a task is given to actor to count how many blinks the participant has if the red light inside the car is lit up. This could trigger awkward staring and silence between the passengers.

Then, study would start after driving out of the parking lot. Interventions would be addressed by experiment conductor through using the in-car voice agent, from description of Roosevelt Island and Cornell Tech, then the traffic status and metro information when approaching tram station, last one would be surprising sounds such as car crash sound, and something like "Wow it's really awkward in here".

In the halfway of the drive, experiment conductor will send music voting link to both of the passengers, and the results will be directly reflected on music player. Throughout the driving, participant will be accessible to an iPad hanging in front of he/she, and it would randomly playing recreational and informational information, from weather forecast and animations.

All these passive and active interventions are trying to use in-vehicle interaction to encourage in car conversations out from awkward silence.

Next step is to integrate biometric sensor to detect positive and negative responses and the intensity towards the response, since based on our literature review it is hard to distinguish between different categories of emotions. Also another evaluation would based on the "talking" data collected from microphone and it can also differentiate which participant is talking.

4.2 Crosswalk Cooperation Driving Simulator Study

4.2.1 Objective

Another application of the AutoRec system is used to study people's relationship with autonomous driving systems.

4.2.2 Experiment Setup

The main hardware components of the experiment consist of a vehicle, projector screens, one GoPro camera, a smartphone running on Android, and an eye-tracker.

The software components include video recording software (Game Capture HD), Unity, and complementing calibration software that comes with the eye-tracker. The video recording will capture the experiment process from four viewpoints: the participant inside the vehicle, the driving route Unity scene that is shown on the projector screens, the view from the eye-tracker, and the smartphone's realtime screen.

Figure 4.1: Crosswalk Driving Simulation Setup.

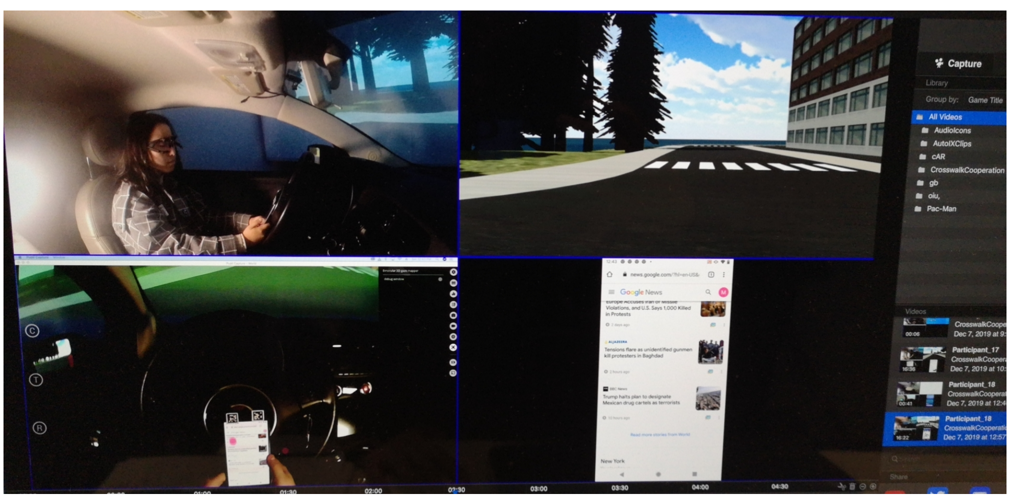


Figure 4.2: Video recording for Crosswalk Driving Simulation.

4.2.3 Participant

One preferred requirement of participants is that they should be able to see clearly without wearing glasses. Otherwise, the eye-tracker might not fit due to its structure.

4.2.4 Process

In the Crosswalk Cooperation Driving study, the participant will be sitting in a driving simulator that is located in front of projection screens. During the experiment, a scene made with Unity will display a driving route around Roosevelt Island to simulate the process of an automated driving experience. While the simulator is running, the participant will be sitting in the driver's seat and facilitate with the car to decide when is an appropriate time to drive through the crosswalk while there are pedestrians on the street.

Before the experiment begins, the participant will need to fill out a "Pre" questionnaire that focuses on the participant's attitude towards autonomous driving, such as if they find automation useful and whether they had previous experience with automation technology.

Then, after the eye-tracker is calibrated and the driving simulating scene starts, the participants will browse the smartphone like they do in their everyday lives. Whenever the car reaches a crosswalk with pedestrians nearby, the simulator will stop and a notification will pop up on the smartphone to prompt the participant to focus on the road and click "Proceed" when they decide that it is safe enough to drive through.

After three loops of driving around the island, the simulation will come to an end and the participants will have to fill out a ”Post” questionnaire which evaluates their level of trust in the autonomous driving system.

CHAPTER 5
DISCUSSION

5.1 Data Oriented

At very beginning, AutoRec system is merely a system for collecting all data streams passively and only related to researchers. However, based on the two case studies, with more dimensions added to the system, AutoRec enables experiment conductor interact with participant under the scene and situated in real time data to make decisions.

Furthermore, AutoRec system lower the barrier of integrating more data sensors. As the current system already provides template code for dealing with electronic sensors, video capturing, microphone, embeded CANbus, external eye tracking, and etc, for further study, if more features added, only replicated work is needed instead of overthrowing all previous work. The current categories of data AutoRec covering ranges from visual, audio to biometric.

5.2 Limitations

The biggest limitation of AutoRec system is its reliability, which in the sense that experiment conductor may encounter out of battery at any node of the system and have to debug and solve. As AutoRec is a system with combination of both hardware and software, the hardware parts becomes the biggest obstacle through out our implementation and research. Wire connection and sensor setup need follow the documentation, however based on the study objective, extensions would be added to the system, therefore more considerations and fixing problems need to be take care.

Due to the limit of time and several weeks waiting for IRB trainig and permission related work, we haven't gone through the procedure of analyzing the outcome and data of AutoRec system. However, based on our sample data collected from dry runs, software part of AutoRec system is not that flexible as it is hard to integrate with further Api such as facial expression.

5.3 Future work

With funding and time to spare what is the ideal future work that you would want to address. It is important to point out the most fruitful next direction.

From the previous researches that we explored and the current experiments that we have been helping running, we can see that the AutoRec system can be useful in addressing a variety of problems in the field of user driving experience. As we mentioned before, the AutoRec system aims at collecting and recording data streams that can help researchers study questions of interest. Therefore, there are still many types of potentially helpful data that we could try and incorporate into the system. For example, biometric data is a type of data that we have yet to explore thoroughly. It would probably be useful to measure and record the passengers' heart rate or body temperature during the ride-sharing study so that we can more accurately quantify the level of awkwardness that the passengers are experiencing.

Other proposed improvements of the system are more focused on hardware performance. While we were running experiments, we noticed that powering the GoPro cameras was a rather frustrating task. Because the battery life of GoPros is somewhat short, sometimes it ran out of power during the drives even when it is being charged by a portable battery inside the car. It would greatly improve the usability of the system if we can find more efficient ways to power the cameras or find substitutes that have longer endurance.

To sum up, AutoRec is a highly extensible system that has a lot of flexibility. Researchers have the freedom to cutomize the system to fit their needs based on the question at hand and we look forward to discovering more creative applications of the system.

CHAPTER 6
CONCLUSION

AutoRec is an in-car recording system with multiple data streams that can help researchers create a new driving experience for the future.

Throughout this two semester, we provide solid work on how we redesign the AutoRec system based on previous work, how to construct and implement and then based on two studies related to vehicle driving, proved the flexibility and efficiency in terms of using AutoRec system to facilitate researchers conducting experiment from scratch to data collecting. We believe that the system still has a lot of room to be improved and extended. Considering that the system is extremely flexible, it should provide a example of how to effectively extract and gather data for researchers with similar interests.

BIBLIOGRAPHY

[1] Goal zero yeti 1250 generator.

[2] Pandacan, manila, Jun 2017.

[3] Nikolas Martelaro and Wendy Ju. The needfinding machine. *Internet of Things Social Internet of Things*, page 51–84, 2018.

[4] Dr. Conal Watterson. Controller area network (can) implementation guide. www.analog.com, May 2015. Retrieved April 23, 2019 from `https://www.analog.com/media/en/technical-documentation/application-notes/AN-1123.pdf`.