Experimental Design for Human-in-the-Loop Driving Simulations

Katherine Driggs Campbell
Ruzena Bajcsy

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2015-59
http://www.eecs.berkeley.edu/Pubs/TechRpts/2015/EECS-2015-59.html

May 11, 2015

Acknowledgement

A special thanks to Victor Shia for his contributions to the work and to Guillaume Bellegarde and Robert Matthew for their contributions in setting up the simulator system and in developing the modules. The contributions and feedback from Jim Gilson regarding the safety protocols and experiment design is also greatly appreciated.

Experimental Design for Human-in-the-Loop Driving Simulations

Katherine Driggs-Campbell
Advised by Professor Ruzena Bajcsy
Department of Electrical Engineering and Computer Sciences
krdc@eecs.berkeley.edu

Abstract

In order to develop provably safe human-in-the-loop systems, accurate and precise models of human behavior must be developed. Driving and flying are examples of human-in-the-loop systems that both require models of human behavior and are safety critical. This report describes a new experimental setup for human-in-the-loop simulations. We focus on developing testbed for collecting driver data that allows us to collect realistic data, while maintaining safety and control of the environmental surroundings. A force feedback simulator with four axis motion has been setup for real-time driving experiments. The simulator will move to simulate the forces a driver feels while driving, which allows for a realistic experience for the driver. This setup allows for flexibility and control for the researcher in a realistic simulation environment. Experiments concerning driver distraction can also be carried out safely in this test bed, in addition to multi-agent experiments.

We present an application in driver modeling, in which we attempt to predict driver behavior over long time horizons for use in a semi-autonomous framework. We extend previous work that focuses on set predictions consisting of trajectories observed from the nonlinear dynamics and behaviors of the human driven car, accounting for the driver mental state, the context or situation that the vehicle is in, and the surrounding environment in both highway and intersection scenarios. By using this realistic data and flexible algorithm, a precise and accurate driver model can be developed that is tailored to an individual.

Contents

Part I

Overview of Human-in-the-Loop Simulation

1 Introduction

Recently, human-in-the-loop control has been a focus of many research projects [6, 49, 47]. These projects generally consider how automation and humans can work together or in a joint environment [6]. However, testing these algorithms and setting up experiments is often difficult due to safety concerns and lack of realistic simulations. In this report, the details of a new driving simulator at the University of California, Berkeley, are presented to address this issue, focusing on driving experiments. All code referred to in this work that is needed to run the simulator, the additional sensors, and the basic processing is available for use.[1]

When considering safety control applications in vehicles, one of the most difficult aspects is the human. Humans have a tendency to be not predictable, as their actions are often dependent on their mental state. These mental states, which could be labeled as happy, angry, sleepy, etc., are difficult to identify, especially from an engineer's perspective, and differ greatly between different people. The project that motivated this report was considering driver distraction [49]. Experimenting with distracted drivers is both difficult and dangerous due to the fact researchers cannot actively tell drivers to text while driving in a real vehicle. To ensure safety, a computer simulation environment was implemented to gather data on driver behavior [47]. However, when obtaining feedback from the subjects, it was found that a major complaint was the lack of realism in the simulations. To address this, a Force Dynamics Simulator [12] was setup for real-time use with PreScan, a standard industry simulation software [14], to create a realistic and safe test bed for driver experiments.

In this work, we presents the setup of this new testbed that is tailored to realistic simulations that mimic real driving. The experimental setup gives an overview of each component and how each contributes to the testbed. The work that has gone into designing sensors to monitor the driver will be briefly discussed as well as the data collection methodology. The paper is organized as follows: in the following section, we present a literature review of similar testbeds and applications. We then present the details of the experimental set, followed by an application that utilizes this setup to build a driver modeling methodology that can be used in a semi-autonomous framework for human-in-the-loop control.

[1] Code is available at `http://www.purl.org/simulator_code/`.

2 Literature Review

In this section, we present a brief review of work that has been completed in simulators focused on human-in-the-loop simulators and the consequent work in human-in-the-loop control.

2.1 Driving Simulators

While realistic simulators as described here are fairly standard in industry [35, 8] and in government labs [16, 20], they relatively uncommon in academia, and are generally stationary simulators. Notable work in this area and driving simulators at various universities can be found at these sites [1, 2, 4, 5, 9]. While many of these systems provide excellent visual displays and can be used in a variety of experiments, including user interface design and psychology studies, few have the motion feedback that is provided on the platform to be in active safety system design and control as presented here. As was previously described, the testbed presented here has been designed to recreate the feeling of moving in a vehicle and is equipped with monitoring devices to observe the human. The motion platform is a 4-axis motion platform simulator created by Force Dynamics [12]. In the area of human factors, a great deal of research effort has be spent of collecting data, as assessing the utility of simulated data collection [36]. Evidence of the utility of motion simulators are can be found in [38, 41].

2.2 Human-in-the-Loop Driver Modeling

In this work, we are interested in using the testbed to monitor the driver and use the driver state information in human-in-the-loop control applications. Modeling the driver has been considered by a variety of communities, ranging from the human factors to computer vision. The human factors and psychology community have focused on quantifying the amount of attention required for driving and level of distraction for different tasks during driving [22, 44]. Other communities have used a variety of sensors: eye and facial trackers [23, 40, 50], body sensors [46] or other sensors to predict driver intent [28, 31]. In [24], intersection data was collected from traffic to design driver assistance systems, but no individualized models were created.

These modeling methods fall short of easily being employed in a control framework. In [34], the authors developed a model of the human, and integrated this into a control framework. This method however did not incorporate the driver state information, which has significant influence over what the driver is likely to do [49]. In Part III of this work, we describe our proposed methodology for using this testbed to develop driver models that can be seamlessly integrated into a semi-autonomous framework.

Part II

Experimental Design of the Simulator

1 Simulator Overview

This section presents the simulator setup for use in human-in-the-loop experiments and driver data collection. There are three primary components that have been integrated to create this testbed: (1) a motion platform simulator, (2) environment and vehicle simulation software, and (3) driver monitoring devices.

1.1 The Motion Platform Simulator

The simulator is a Force Dynamics 401CR platform (Fig. 1). The platform runs using two computers: one computer to run the software, referred to as the gaming computer, and one computer to control the movement of the simulator, referred to as the control computer. This system has four axes: pitch, roll, yaw, and heave, which are used to simulate the forces a driver would experience. It is equipped with a sound system and video display, which provides full frontal view with three angled screens. For more information about the technical specifications of the simulation platform, please visit [12].

Figure 1: Force Dynamics 401CR Platform [12].

In the current setup, using rotational data, pitch and roll measurements relative to the car's frame of reference are fed back to the driver to recreate the accelerations experienced on turns and when braking or accelerating. The change in heading angle is determined

by the yaw rate, relative to the car's frame, and is calibrated to be a one to one turning experience (i.e. a 90 degree turn will result in a 90 rotation in the simulator). User feedback on the motion experience is discussed in Section 3.

1.1.1 Controlling the Simulator

To run driving simulations, PreScan is used to design and run experiments. While this software is often used for replaying collected data or for running experiments where real-time performance is unnecessary, a communication protocol between the software and the simulator was put in place to achieve real-time simulations with the simulator, currently running at a rate of 60 Hz. The input to the simulator platform is obtained though the joystick function block from Matlab's Simulink, which is then used as the input to the dynamics of the vehicle through PreScan. The position of the vehicle, in terms of pitch, yaw, and roll, is obtained from PreScan and sent to a Python script via User Datagram Protocol (UDP). This is packaged as a specific structure that is sent to the simulator control computer, again over UDP, to adjust the position of the simulator appropriately. This setup is illustrated in Figure 2. Each of these signals have gains associated them, which have been hand-tuned to re-create the feeling of driving. While majority of the gains were selected by trial and error with multiple drivers' feedback, the braking and throttle input were adjusted such that the vehicle would decelerate from sixty to zero miles per hour in about two seconds (following the idea of the four-second-rule when driving at highway speeds [3]) and would accelerate from zero to sixty miles per hour in twelve seconds.

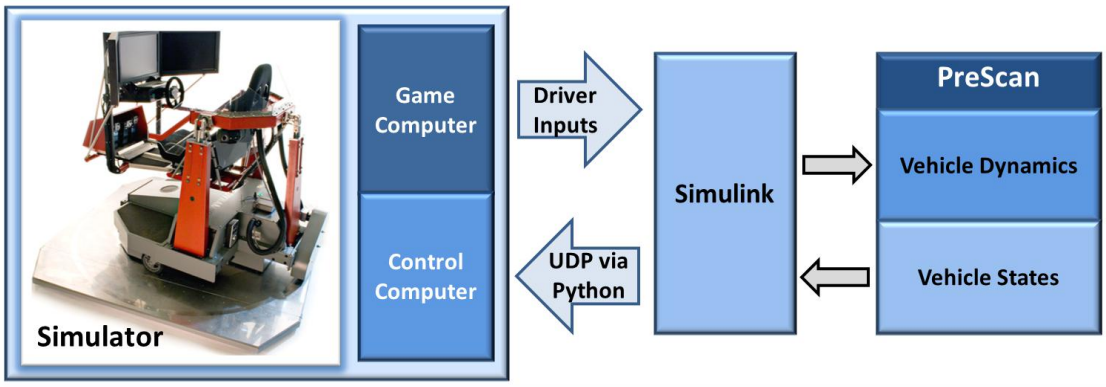


Figure 2: Block Diagram showing the communication protocol to control the simulator.

1.2 Environment and Vehicle Simulation

Once the real-time aspect was completed, the actual design of experiments was considered. To assist in design and data collection, experiment and data processing modules

have been developed. Simulink modules were created to store all input and output data from the simulator and PreScan. The components of PreScan that have been set up are briefly explained below.

Driver Inputs The input component is pulled from the simulator to be used as the input to the vehicle model as provided through PreScan. This includes the braking, throttle, and steering input.

Vehicle Information The vehicle component provides all vehicle states in addition to GPS coordinates. These states include: x, y, and z position, rotation on the x, y, and z axes, velocity, heading angle, and yaw rate. Note that these rotation values are sent to the simulator to recreate the feeling of driving.

Radar Components The radar component has three different sensors implemented, for the sides and the front of the vehicle. In PreScan, this is an idealized sensor, so the output for each sensor is the readings for all objects in the experiment world at each time step. This gives the object ID number (which is determined randomly in PreScan), the angle of detection, the elevation angle of detection, the distance to the object, the velocity of the object, and the heading angle of the object.

Object Detection Components While the radar detection modules relates the detection information with an object ID, object detection has also been integrated which classifies detected objects in a particular field of view. This is gives basic object information and bounding box data, similar to what would be produced by a vision or depth image classification algorithm.

Lane Marker Sensor This component provides all road data at the vehicle and at three predefined distances ahead of the vehicle. The data collected can provide distance to the nearest lane marker and curb on either side of the vehicle as well as the curvature of the road. Currently, this is the only data from the sensor being used, although many more measurements can be observed [14].

These components were selected to mimic the hardware that can readily be implemented on real vehicles, for future testing.

The vehicle, input, radar, and lane marker components each have an associated Simulink data collection module to collect variables that effect the driving experience. This system data is also used to provide motion feedback to the driver, by controlling the forces the driver will feel while driving. Additionally, the simulator is setup to shake if the driver collides with an obstacle in the experiment. The motion has been tuned to appropriately mimic the acceleration felt while driving. The modular setup of the components is shown in Figure 3.

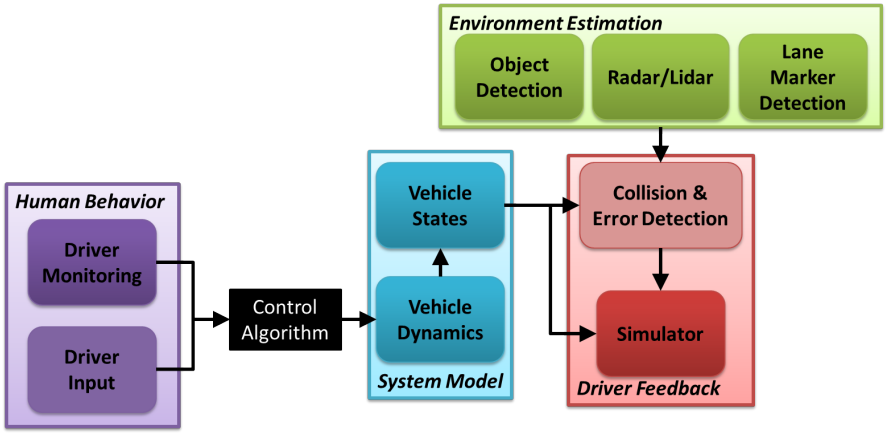


Figure 3: Schematic of the modular component setup. The black box control algorithm can be filled in to verify developed algorithms, or ignored for pure data collection.

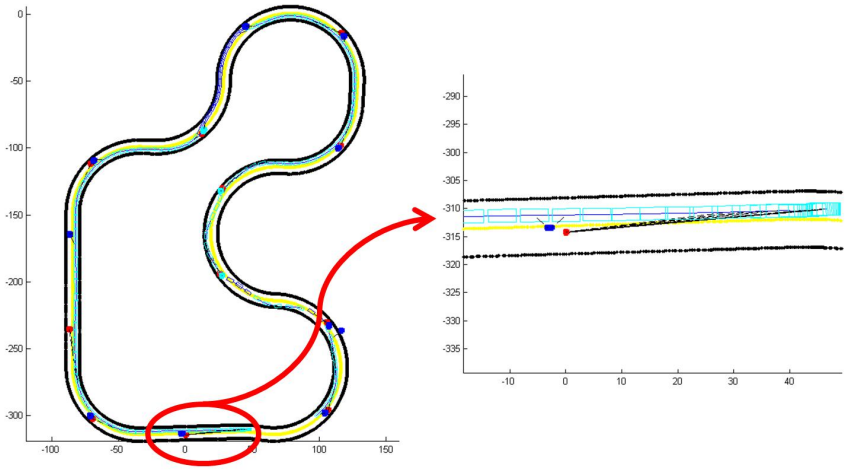


Figure 4: Matlab plot showing the experimental run. The road and lane boundaries are shown in black and yellow. The path the vehicle took is shown in a blue line and the blue or light blue box. The color of the vehicle box indicates the lane. Objects are shown in red, blue, or light blue, depending on if the front, left, or right radar detected the object, respectively. The angle and line of detection is shown from the vehicle.

1.2.1 Data Processing

Matlab scripts are available to easily unpack and do minor processing of the data. Basic functions can unpack the vehicle, input, radar, and lane marker data. The lane and vehicle data can be used to determine indicators of which lane the driver is in and offset relative to the center of the road. The radar data can be interpreted into three

components: front object, left object, and right object, giving the data for the closest obstacle. With this basic processing, the investigator can develop algorithms to assess driver behavior. The collected data can be plotted to visualize the data collected in the experiment, as shown in Figure 4.

1.3 Driver Monitoring

To gather complete data concerning driver behavior, additional components have been developed for the simulator. The following components have been added to the simulator setup:

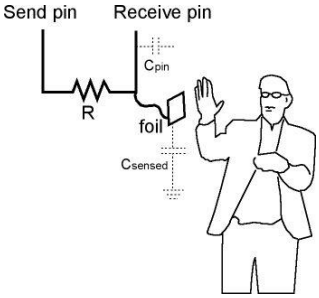
Motion Capture System: An OptiTrack motion capture system has been set up around the simulator to accurately monitor the driver pose in real time [19]. While this provides a reliable baseline for testing, the motion of the simulator often causes occlusion.

Eye Tracking: SMI Eye Tracking Glasses can be worn by the driver so his exact gaze can be monitored [7]. This is to be integrated with the motion capture system to obtain the point of attention in the proper frame of reference.

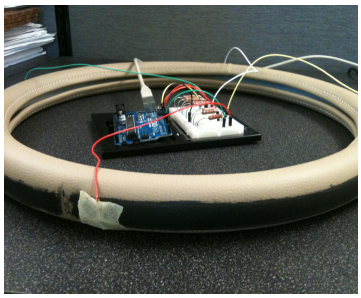
Sensor Network for Dynamic Human Modeling: In order to model the dynamic behavior of the driver and recover data points lost due to occulusion, a sensor network is being developed to attach sensors to the body of the driver. This network includes nine degree of freedom IMU sensors that communicate to a centralized location for processing. This can be incorporated with motion capture data to develop the kinematic or dynamic model of the driver.

Camera and Microsoft Kinect 2: A GoPro camera has been mounted to the simulator to monitor the driver [17]. The MS Kinect 2 system has also been mounted on the simulator to measure the driver skeleton and monitor the driver's head pose and expression [18]. This can be used for pose detection or other video processing algorithms, which would give insight to driver state as shown in [30, 47].

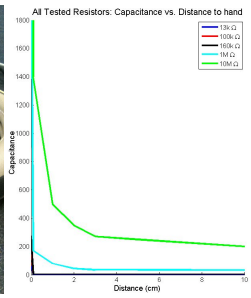
Wheel Touch Sensor: A touch sensor was built to determine if the driver's hands are on the wheel. This is done using a capacitive sensor as described in [27, 21], illustrated in Figure 5a, and is implemented using the Capacitive Sensor Library for the Arduino Uno. The steering wheel is divided into four quadrants, each acting as a sensor. Conductive paint is used in place of the conductive foil shown in the figure. Each is connected to an Arduino which continuously sends data over BlueTooth. The data contains a binary signal indicating whether each sensor is being triggered. The complete sensor is shown in Figure 5b. The sensor was calibrated with various resistor values to examine the sensitivity of the system (Fig. 5c), verifying the relationship between resistance and distance from sensor. Ultimately, a $13k\Omega$ resistor value was used in the final design, as it resulted in the most boolean-like response in the calibration process.



(a) Schematic for the capacitive touch sensor [21].



(b) Touch sensor implemented on a steering wheel cover. This shows the connections to the Arduino that relays the information over BlueTooth.



(c) Calibration plot for the touch sensor showing the capacitance value collected as a function of distance from contact to hand with various resistor values.

Figure 5: Capacitive touch sensor for the steering wheel.

Phone Application: The last additional component added to the experimental design was a phone app to simulate texting, as shown in Figure 6. This application is meant to distract the driver in the form of incoming texts. The application randomly rings in a time frame of 30-60 seconds with a question for the driver to respond to. Upon ringing, this information is sent via BlueTooth to the control server indicating that the user will soon be distracted. This app can detect when the driver picks up the phone via abnormal changes in the phone's accelerometer data and when the driver starts touching the phone.

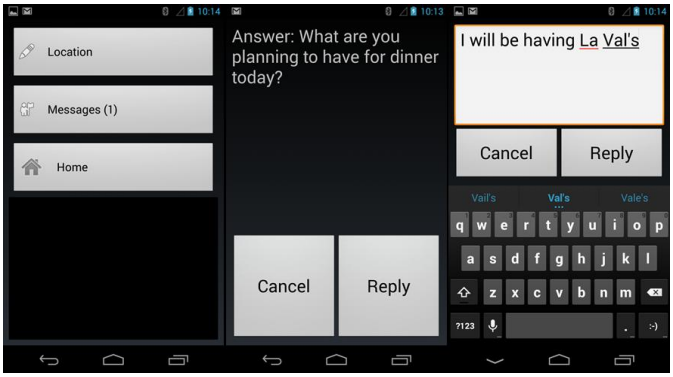


Figure 6: Android app created to simulate texting environment with randomized messages.

The touch sensor data and the phone app data are sent through the phone to the main computer. The touch sensor data is sent every 10 ms, while the phone data is event driven based on if the user is picking up the phone, touching the screen, or putting down the phone. Using this data, we can determine when the phone is being used, meaning we can determine when the driver is distracted.

Future works include integrating augmented and virtual reality systems to create a more immersive environment, and provide feedback to the human.

2 Safety Measures

The study protocol was approved by the University of California Institutional Review Board for human protection and privacy, under Protocol ID 2013-07-5459. Subjects will only be selected if they are over the age of 18, are not vulnerable to undue influence, meet height and weight requirements, are not prone to motion sickness, and are not pregnant, and will sign a form to acknowledge that they are qualified to perform the driving tasks and are aware of the experimental procedures. Assuming the subject is rational and will not intentionally attempt to injury himself, the likelihood of physical injury from use of the force dynamics simulator is minimal. For safety, the simulator is equipped with multiple emergency stop buttons and the sytstem will not start if the entry gate is not closed or if the seat belt is not on. To minimize risk, the subjects will be instructed on how to stop the simulation at any time before the experiments. Also, a waist high fence surrounds the simulator while it is active to avoid observers getting hurt. A curtain is has been installed around the simulator to avoid unnecessary distractions to the driver. To avoid potential mental distress, only animals, vehicles, or other inanimate objects will be used as obstacles. Upon request, the IRB Protocol and documentation can be provided for more detailed information.

3 User Feedback

The following user feedback was collected while running the experiments for the application described in Part III. The test subjects were asked a number of questions concerning the experiment and their experience on the simulator. When asked to rate their overall experience on the simulator, the average score was 8.2 out of 10, where 10 was a good, realistic experience and 0 was a poor, unrealistic experience. Relevant questions and the average response is provided in Table 1. The subject was asked to respond to the following statements, with *Strongly Agree, Agree, Neutral, Disagree,* or *Strongly Disagree*. The averages were calculated by assigning a score of 5 to *Strongly Agree* and a 1 to *Strongly Disagree*, with uniform steps in between.

Table 1: User feedback about the simulator experience.

Statement	**Avg. Response**
I feel comfortable in the simulator.	4.2
I feel safe in the simulator.	4.8
The setup reconstructed realistic, everyday driving distractions.	3.9
The simulator improved my driving experience.	4.3

The subjects were also asked if they felt their driving improved as more trials were conducted, which gave the average response of 4.1. This implies that performing training or practice sessions are important so the driver can become comfortable with the system and so the data is indeed realistic.

4 Discussion

This part presented the setup of our experimental platform to test human-in-the-loop driving experiments. This has been done using a force feedback simulator that allows for a realistic driving experience, while allowing complete control over the testing environment. By utilizing the PreScan software, many variables can be controlled including road configuration, weather, obstacles, vehicle dynamics, and much more. Combining these two components to form a complete set up has been completed in addition to extra features. These include driver monitoring devices (including cameras, Kinect, etc.) that can be mounted to the simulator as desired, eye tracking glasses, a touch sensor for the wheel, and a phone app to simulate distracted driving. To make experiments easy and quick to set up, modules have been created for data collection and for simple data processing that can be used by those interested in running driving experiments. In addition, safety measures have been implemented in order to guarantee the subject's safety when using the setup.

While this setup is to be used for many experiments, the primary focus will concern human interaction with semi-autonomous or autonomous vehicles. This can be used to build driver models to build smart active safety systems, as demonstrated in [47, 49] The remainder of the paper presents a project that utilizes the testbed to develop a driver model, for use in semi-autonomous vehicles [29].

Part III

Application in Driver Modeling

1 Introduction

There are number of ways to approach safety when considering safety in human-machine interaction. Some robotic systems approach safety from a mechanical point of view by creating systems that physically cannot harm the human [15]. Another approach is to develop controllers and sensor systems that can guarantee safety for a given system [32]. However, when considering systems that involve or interact with humans (called human-in-the-loop systems), deriving safety boundaries is not a simple task. Human actions and behaviors are often unpredictable and cannot easily be described by normal dynamical methods. Therefore, in order to develop provably safe human-in-the-loop systems, first a useful model of the human must be developed that can be incorporated into a safety algorithm for semiautonomous control. Here, we consider human driver behavior. This application is a key example as the driver has full control of the system, and her likely actions are highly dependent on her mental state and the context of the current situation.

It is well-known that while humans have many innate skills that make them adept at driving (e.g. flexible and adaptable to new situations, good at decision making), they are fallible and prone to error. In fact, according to a 2008 study, 93% of car crashes are due to human error. On top of that, studies show that at any given moment in America, approximately 660,000 drivers are using cell phones or another electronic device while driving, even though doing so increases the risk of getting into an accident by three times [10, 48]. This has brought rise to a great deal of research in driver modeling and autonomous vehicles to mitigate or hopefully eliminate these collisions [11]. Additionally, many semiautonomous and human-in-the-loop systems are being developed as many believe that autonomous vehicles should be introduced incrementally [33].

One example of a successful semiautonomous system that uses a reachable set methodology is the Volvo City Safety system. When driving in the city (below 35 miles per hour), the system calculates the reachable set of the vehicle for the future 500ms and predicts collisions by checking to see if a detected object is within that set [26]. As noted, this method does not work at high speeds as the reachable set of the vehicle itself becomes too large, leading to an overly invasive system. When considering high speeds, the human can no longer be considered as a disturbance in the system, as the driver has significant influence over the future trajectories of the vehicle. Ideally, the system would function at high speeds and consider the likely actions of the human by modeling the driver to create a more informative reachable set.

We propose the use of a driver model that incorporates knowledge of an individual driver's likely set of actions to create a set of likely states the driver will visit, to minimize the amount of intervention by an active safety system, as first described in [49]. We utilize the experimental setup previously described that allows us to collect realistic driver data that can be used to create a model of the human mental state, in a safe manner. We hypothesize that by incorporating a mental model of the human and by developing a context aware system, we can better predict driver behavior and thus create a better active safety system. This system can be tailored to a specific driver, and if highly precise, it can be less invasive, while maintaining accuracy.

The work here expands upon the data-driven methodology presented in [47] by creating a more precise and accurate model of human driving behavior. Our contributions include (1) using a human-in-the-loop testbed to collect realistic driving and mental state data; (2) enhancing the existing driver modeling algorithm with improved results; and (3) extending the model to consider intersection scenarios in addition to highway driving. We relax some of the assumptions from previous work on the underlying model of the vehicle, by including all inputs (steering, throttle, and braking) into the model. This creates more variability in the predictions, as we consider nonlinear behaviors of the human inputs.

The following work is structured as follows. In Section 2, we will present our methods and its advantages over other methods. The setup and execution of the experiments will be described in Section 3. Section 4 will present our metrics and results for analyzing our system when considering highway driving. Section 5 will present an extensions of this algorithm to intersection behaviors. While this work focuses on the development of the human model, the methods for applying the model are discussed in Section 6. Finally, we will conclude with a discussion of the results and of the future works.

2 Methodology

The following section will introduce the formulation and implementation of the proposed model. In this work, we are interested in the driver modeling that will influence the control. Refer to [47] for details concerning the larger control framework which this driver model can be applied and the implementation process.

2.1 Modeling Human Behavior

In this subsection, we provide a mathematical formulation of the driver model. We denote the past observed information with $\mathcal{O}$ and the information at the current time with $\mathcal{I}$.

Existence of a Driver State Function Suppose in the current information set, we are able to obtain driver monitoring data such that:

$$\theta : \mathcal{O} \times \mathcal{I} \to Q \tag{1}$$

where Q is our set of discrete mental states that the driver could be in. In this work, we assume this set of mental states to be attentive, partially attentive, and distracted. This is similar to work in psychology and discrete event systems, which identifies tasks to have no, low, or high mental workload or cognitive distraction, and adjusts based on discrete mental modes [39, 42].

Existence of Distinct Driver Modes Given that driver behavior heavily depends on context and mental state, we assume that there exist distinct driver modes that depend on this information. As was previously mentioned, we are interested in long time horizon trajectory predictions that will encapsulate the uncertainties and the bounds of the potential future states of the vehicle. This is described as follows:

$$\begin{aligned} \underset{\Delta \subset \mathbb{R}^n}{\text{argmin}} \quad & |\Delta| \\ \text{subject to} \quad & P_{\theta(\mathcal{O},\mathcal{I})}\left[(X(k) - x_0) \subset \Delta | \mathcal{O}, \mathcal{I}\right] \geq \alpha \\ & \forall k \in \{0, \ldots, N\} \end{aligned} \tag{2}$$

where X is the random variable representing the future vehicle trajectory, $x_0 \in \mathbb{R}^n$ is the initial position, N is the time horizon, P_θ is the probability distribution on the trajectories given the driver state and the information sets, and Δ the minimum area set that contains the future trajectory of the vehicle with at least probability $\alpha \in [0, 1]$. This can be interpreted as the α-likely reachable set of the vehicle given the past and current information we have observed from the driver. The output of this optimization program at time k return a set that has a probability distribution that will be referred to as $\Delta_k(\mathcal{O}, \mathcal{I}, \alpha)$.

2.2 Model Implementation

In this subsection, we describe how we implement the driver model described in Section 2.1. Rather than making assumptions on human behavior and fit P_θ to a known probability distribution, we estimate an empirical distribution from the observed data. Here, suppose that for a given context and environment observations, there are distinct modes of behavior. While we assume that the level of distraction and context are known (i.e. we have full knowledge of θ and know when the vehicle is on a highway or approaching an intersection), identifying a mode associated with the environment constraints is more difficult. These constraints are related to the road boundaries and curvature and the surrounding obstacles. We uncover the modes of the environment from sensor data using the k-means algorithm on observed scenario data [37]. This environment representation also includes the level of traffic and relative states via sensing the surrounding vehicles.

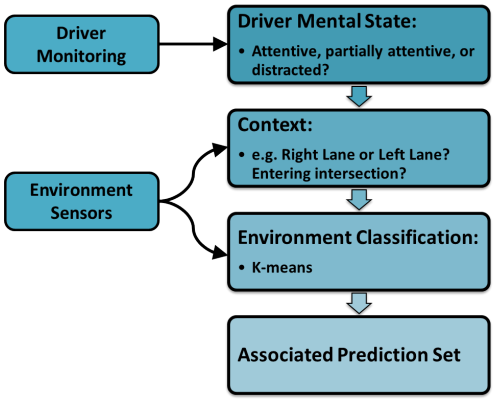


Figure 7: A flow diagram of the parsing used to identify the driver modes.

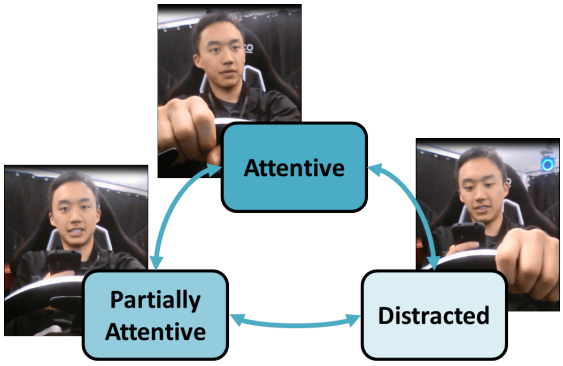


Figure 8: Diagram showing the transitions between mental states when distractions or actions occur.

The sensors chosen to collect environment data mimic current commercially available sensors such as front and side radar and lane detection modules [13]. Using the front and side radar, we detect the number of obstacles in the vicinity of the ego vehicle and observable states (i.e. relative position and velocity), relative to the ego vehicle. The lane sensors allow us to extract future road bounds.

To create a context aware system, we implement a hierarchical algorithm with the levels as shown in Figure 7. The first two levels determine driver mode using the driver mental state (described in Eq. 1 and shown in Fig. 8) and context (e.g. approaching an intersection, position on the road, etc.), which provide insight to the future trajectories of the vehicle. For instance, drivers behave differently during city driving than highway driving and drivers demonstrate different behaviors when in the right or left lane of a two lane road. By parsing the data based off of these detectable contexts, we hypothesize that the system will be able to more accurately identify driver modes. This is a significant improvement to the model proposed in [47] which uses k-means on a time-series vector of driver state, context and environment.

To summarize the algorithm, driver data is parsed using a decision tree-like method. The first level utilizes the driver state and the second utilizes the position of the car on the road. The final level is dictated by which cluster the observed surroundings belong to. By separating the training data in this manner, we then build predictive sets using the future observed trajectories at each instance of this mode, effectively linking the dataset to the current point in time. From these sets, we can calculate an empirical probability distribution for future trajectories and inputs. In essence, this model answers the question: given the context and the driver's mental state, what has the driver done in the past?

3 Experiments

In this section, we describe the experiments, using the previously described simulator setup. A test subject in the simulator and visualization seen by the driver is shown in Fig. 9 and Fig. 10, respectively.

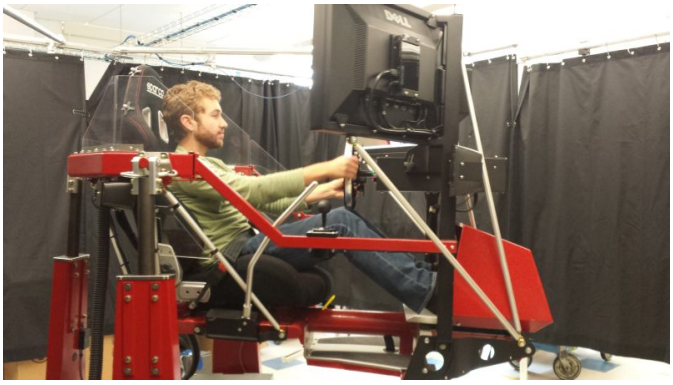
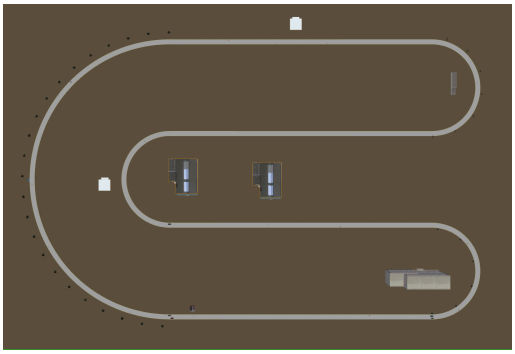


Figure 9: Picture of test subject driving the simulator.

Figure 10: PreScan simulation environment from the driver's perspective.

Fourteen subjects (two female and twelve male) between the ages of 21 and 36 were recruited to test this prediction algorithm for highway and intersection scenarios. The years of driving experience ranged from two to sixteen years. The intersection model is presented in Section 5. For the highway scenario, subjects were asked to drive on four courses each lasting twenty minutes to generate three training sets and one test set. These courses were generated using PreScan to run on the simulator, and consisted of

(a) Training Course 1



(b) Training Course 2

(c) Training Course 3

(d) Test Course

Figure 11: Training and test courses used to build the data sets.

two lane roads with turns of various curvatures, with different levels of traffic that moved independently of the ego vehicle with no opposing traffic. On these courses, the drivers faced a number of obstacles, some that were static (e.g. trailers in the road, cardboard boxes, etc.) and others that were moving (e.g. balls rolling in the road, other vehicles, etc.). The driver was asked to drive as they would normally at about 50 mph. The final test course consisted of obstacles and road patterns that had not been experienced in the training set, to verify the flexibility of the model. The top views of the courses are illustrated in Fig. 11.

To simulate distraction, the driver was given an android phone with a custom application to randomly ping the driver to respond to a text message 30-60s after the driver responded to the previous text. The application also recorded phone acceleration and touch to determine when the driver state in real time (see Part II, Section 1.3). A few example questions are:
(1) What did you have for lunch today?
(2) What is your major?
(3) Where are the Olympics this year?

To determine the driver state, we assume that the driver is attentive when there is no distraction, partially distracted after the phone rings and she considers answering, and fully distracted when she is physically typing on the phone.

4 Evaluation

In this section, we describe the metrics used to evaluate the model and results on the above experiments.

4.1 Model Metrics

Before presenting the results, the evaluation techniques will be briefly described. Since we are considering a set prediction for the driver model, there is a trade-off between the precision and the accuracy. To clarify, if we were only interested in a model with high accuracy, then the reachable set of the car would suffice. However, this does not provide us any useful information into what the driver is likely to do. Therefore, we define accuracy as the number of samples that fall within the boundaries of the set (Eq. 3) and precision as the area of the set relative to the size of the reachable set (Eq. 4). More information about these metrics can be found in [47]. These metrics are formally defined as follows:

$$A = \frac{1}{M} \sum_{i=1}^{M} \prod_{k=0}^{N} \mathbb{1} \left\{ (x_i(k) - x_i(0)) \in \Delta_k(\mathcal{O}, \mathcal{I}, \alpha) \right\} \tag{3}$$

where M is the number of observed trajectories, $x_{i,N}(k)$ is the state of vehicle of the i^{th} trajectory at time k, and $\mathbb{1}$ is the indicator function.

$$P = \max \left\{ 1 - \frac{1}{M} \sum_{i=1}^{M} \frac{\left| \cup_{k=0}^{N} \Delta_k(\mathcal{O}, \mathcal{I}, \alpha) \right|}{\left| \cup_{k=0}^{N} \mathcal{R}(k, \mathcal{I}) \right|}, 0 \right\} \tag{4}$$

where $\mathcal{R}(k, \mathcal{I})$ is the reachable set beginning at $x_i(0)$, for a constant velocity. For a given mode, the median velocity of the associated set of observed trajectories is used. Since we consider the reachable set for a constant velocity (meaning there is no throttle input), it is possible that the predicted set is larger than the reachable set used for comparison. The precision is set to zero if this occurs, the prediction is deemed less useful than the standard reachable set.

4.2 Highway Experiments

We evaluate this metric at various time horizons $T = \{0.5, 1.0, 1.2, 1.5, 2.0\}$ seconds. For comparison, we compare these results to the reachable set (denoted RS) of the vehicle, which is always accurate, but lacks precision. The accuracy and precision metrics versus the number of clusters are shown in Tables 2 and 3, respectively. These results are the averaged metrics for each of the individualized models.

Table 2: Driver Model Accuracy Results, where k is the number of clusters used and RS denotes the reachable set.

T	**0.5 s**	**1.0 s**	**1.2 s**	**1.5 s**	**2.0 s**
RS	1.000	1.000	1.000	1.000	1.000
$k = 1$	0.998	0.998	0.998	0.998	0.967
$k = 5$	0.993	0.992	0.992	0.991	0.894
$k = 10$	0.985	0.983	0.982	0.981	0.818
$k = 15$	0.961	0.959	0.958	0.956	0.757
$k = 20$	0.955	0.952	0.950	0.948	0.721
$k = 25$	0.925	0.922	0.920	0.916	0.676
$k = 30$	0.911	0.908	0.906	0.902	0.642

Table 3: Driver Model Precision Results, where k is the number of clusters used and RS denotes the reachable set.

T	**0.5 s**	**1.0 s**	**1.2 s**	**1.5 s**	**2 s**
RS	0.000	0.000	0.000	0.000	0.000
$k = 1$	0.000	0.000	0.000	0.000	0.000
$k = 5$	0.000	0.000	0.087	0.244	0.337
$k = 10$	0.000	0.182	0.336	0.457	0.528
$k = 15$	0.000	0.313	0.447	0.551	0.613
$k = 20$	0.000	0.400	0.521	0.614	0.671
$k = 25$	0.000	0.458	0.570	0.657	0.709
$k = 30$	0.000	0.488	0.596	0.679	0.729

From these results, a few key observations can be made. As expected, the accuracy decreases over time. This can partially be explained by the uncertainty of the environment, which can drastically change over the course of two seconds. Therefore, this model is best used at a time horizon of 1 to 1.5s. Also note that the accuracy decreases as a function of the number of clusters. Here, we note the trade off between the precision, which improves with an increase in clusters. This intuitively makes sense, as we finely separate the data, we expect smaller set predictions, sacrificing the accuracy of the model. When the environment is ignored, meaning only a single cluster is used, precision is completely lost. This implies that the environment must be taken into consideration for useful prediction.

It can be seen that for a time horizon of 0.5s, the driver model has zero precision. This implies that when considering a short time horizon, the directly using the reachable set is justified. However, when predicting over long time horizons, the model is becomes more useful than the alternative. The trade-off between accuracy and precision is visualized in Fig. 12, which shows both metrics versus the number of environment clusters used.

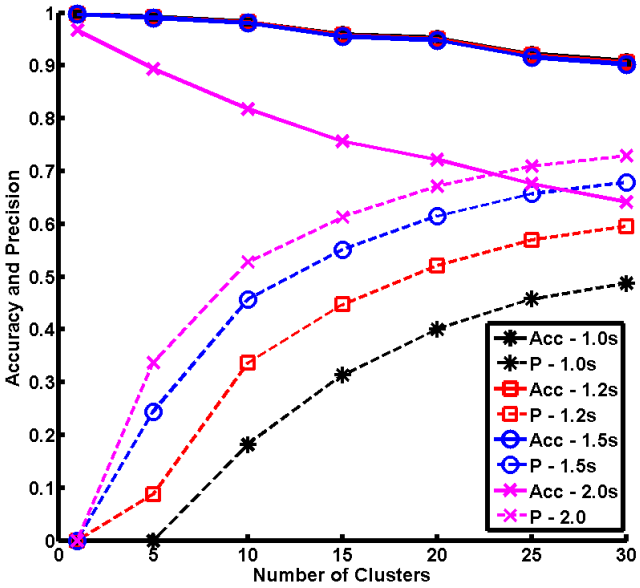
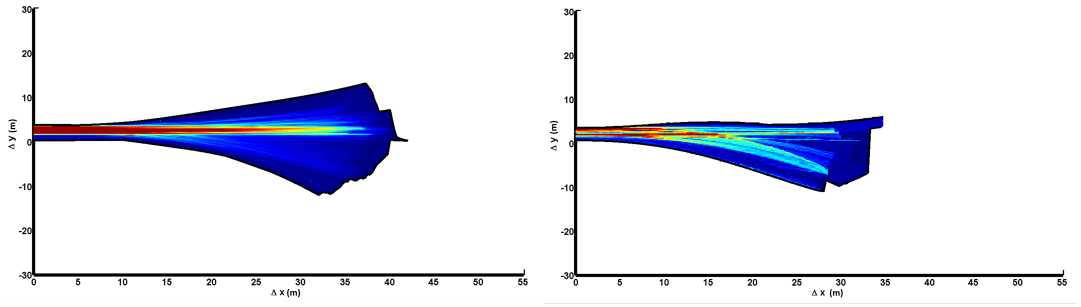


Figure 12: Plot illustrating the trade-off between accuracy and precision as the number of clusters increase, when $T = 1.0$, 1.2, 1.5, and 2.0s.

Once these sets are created, the empirical probability distributions can be derived to show the likelihood of the trajectory sets over time. Example sets and their distributions are visualized in Figure 13.

5 Extension to Intersections

In this section, we describe an extension to the model on city driving with intersections. At intersections, we assume knowledge of the driver intention to turn or drive straight, similar to if the driver where to obey navigation commands from a Global Position System navigation device.



(a) Example set for attentive driver on a straight road.

(b) Example set for distracted driver on a slightly curved road.

Figure 13: Visualization of prediction sets for an individual driver ($k = 20$), where the probability distribution is plotted over Δx and Δy, representing the longitudinal and latitudinal change in position in meters. The dark red regions represent areas of high probability and darker blue regions represent low probability.

5.1 Model Extension Formulation

The model proposed in Section 2.2 easily extends to intersection with some slight modifications to the context detection level. In the context detection level of the decision tree, we detect if the driver is driving on a straightaway or approaching an intersection, which is defined as being within a distance r of the intersection. In the model presented here, we set $r = 60m$. This is then conditioned on the status of the intersection (e.g. stop sign, traffic light), which would influence the behavior. We include the status of the traffic light for the previous two seconds in the feature set. The model considers the following four scenarios: (1) approaching a stop sign with the intent of continuing straight; (2) approaching a stop sign and turning right; (3) approaching a traffic light; and (4) city driving.

Since the time scale at which events occur in city driving, this model is formulated to test for a 5 second prediction, tested in 1 second increments. Generally, accurate and precise predictions over such long time horizons are extremely difficult if not impossible due to the potential changes in the environment. However, this long time horizon allows us to analyze the entire execution of a maneuver that might occur at an intersection.

5.2 Experimental Setup

The course for this extension is shown in Fig. 14, where the training and testing data was completed in the same road configuration, with different traffic flows, obstacles, and environmental distractions. The driver was asked to navigate the loop of intersections by driving as they would normally at about 25 mph while stopping at stop signs and at traffic lights, as one would in city driving for half an hour. It was noted that the drivers

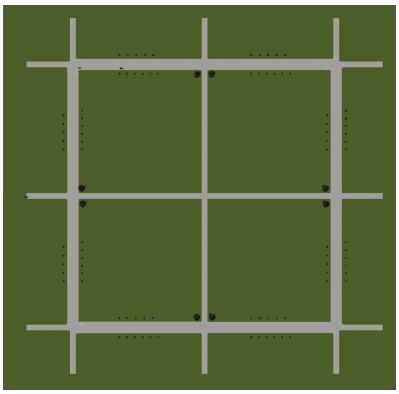
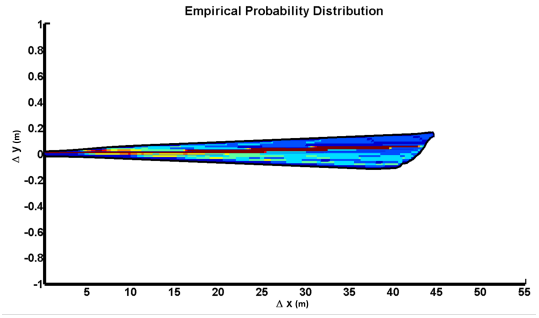


Figure 14: Course used to build the intersection model.

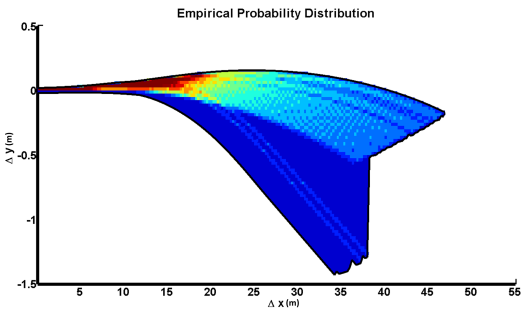
generally respond to the texts while stopped, so the driver state was excluded in this section due to lack of sufficiently rich and useful data.

5.3 Results

We present the evaluation of this model using the metrics in Section 4.1. The accuracy and precision results are also shown in Table 4 and 5, respectively. A visualization of the predictions sets for intersections are shown in Figure 15.



(a) Example set when approaching a stop sign.



(b) Example set when turning.

Figure 15: Prediction set for intersection model, where $r = 50m$ and $k = 5$. The probability distribution is plotted over Δx and Δy, representing the longitudinal and latitudinal change in position in meters. Dark red areas represent high probability and dark blue represents low probability.

From these metrics, a similar relationship between precision and accuracy can be observed as before. The accuracy for this model is quite high. This was expected from the data, as it was observed driver behaviors are more consistent and spontaneous behavior is less likely to occur (e.g. a driver will change lanes with very low probability when approaching an intersection, in our dataset).

Table 4: Intersection Model Accuracy Results, where k is the number of clusters used and RS stands for the reachable set.

T	**1 s**	**2 s**	**3 s**	**4 s**	**5 s**
RS	1.000	1.000	1.000	1.000	1.000
$k = 1$	0.999	0.998	0.997	0.997	0.967
$k = 5$	0.993	0.990	0.986	0.984	0.912
$k = 10$	0.986	0.981	0.976	0.973	0.874
$k = 15$	0.979	0.972	0.966	0.961	0.835
$k = 20$	0.974	0.965	0.959	0.952	0.797
$k = 25$	0.969	0.958	0.951	0.944	0.772

Table 5: Intersection Model Precision Results, where k is the number of clusters used and RS stands for the reachable set.

T	**1 s**	**2 s**	**3 s**	**4 s**	**5 s**
RS	0.000	0.000	0.000	0.000	0.000
$k = 1$	0.868	0.894	0.844	0.704	0.00
$k = 5$	0.865	0.858	0.855	0.833	0.00
$k = 10$	0.898	0.899	0.898	0.883	0.529
$k = 15$	0.933	0.930	0.927	0.912	0.382
$k = 20$	0.948	0.948	0.944	0.927	0.470
$k = 25$	0.966	0.967	0.9607	0.941	0.531

This formulation also allows us to closely examine the control aspect of the human when approaching the intersection. Using the driver's inputs to the vehicle, we can build an empirical probability distribution similar the trajectory sets previously presented to consider driver behaviors in terms of control. This can be used to understand how human's behave and control the vehicle in the different driver modes, but can also be used to derive bounds on the human input in different scenarios. An example of this is illustrated in Fig. 16. Using the empirical distribution, we can assess what the driver's likely actions will be in a reliable and flexible manner. Because we are directly observing the human behavior and creating the boundaries based of these observations, the predicted sets are not necessary smooth, however realistic and accurate.

6 Applying the Model

As was discussed in Section 1, accurate and precise models of human behavior are crucial for human-in-the-loop systems for developing provably safe control mechanisms or giving feedback to the driver. This model is able to identify the likely set of actions, which can be thought of a highly probably reachable set. This set formulation also allows us

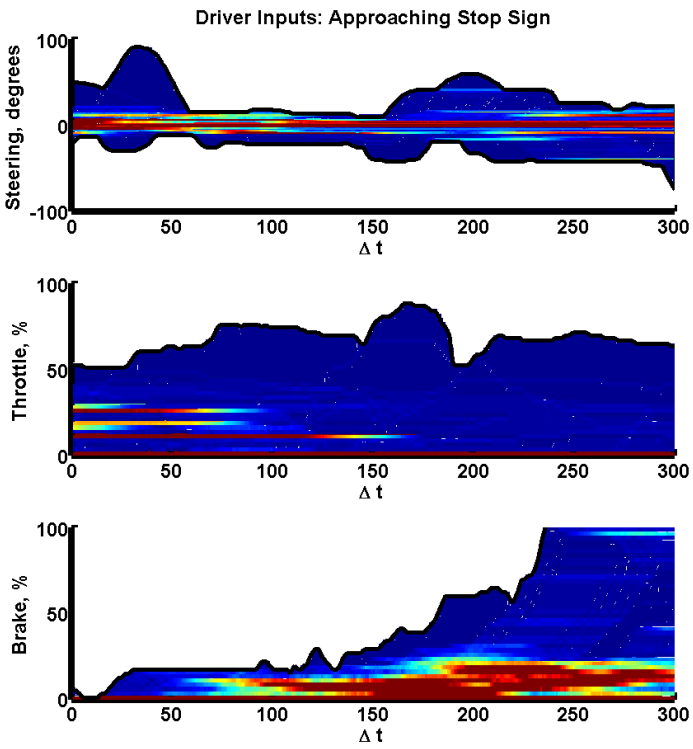


Figure 16: An example set of driver's inputs when approaching a stop sign, associated with the trajectory set in Fig. 15a. The dark blue regions are associated with low probability, while red regions represent high probability.

to examine the varying behaviors of people depending on the context in a quantitative manner. Using this empirical model, we can quantify the likelihood of "good" driving behavior, as was shown in [45]. This is valuable as the driver would be able to receive useful feedback on their regular driving behaviors and can be used to develop a provably correct controller.

In addition, this model can be incorporated in a semiautonomous framework. There are two main control frameworks in which this model will easily integrate: switched and augmented control. Consider the following vehicle dynamics:

$$x(k+1) = f(x(k), u(k)), \quad \forall k \in \mathbb{N} \tag{5}$$

where $x(k) \in \mathbb{R}^n$ is the state at time k, $u(k) \in U$ is the input to the vehicle at time k where $U \subset \mathbb{R}^m$ is a compact, connected set containing the origin, and the initial state of the car, $x(0)$, is given.

Suppose that given $\mathcal{I}$, the unsafe regions of the environment can be estimated, denoted as $\mathcal{C}$. It is assumed that for a given fixed time horizon, $N \in \mathbb{N}$, and a given cost function,

there exists an optimal control algorithm that is able to keep the vehicle outside the unsafe set. This assumption can be satisfied by model predictive control (MPC) [25].

Ideally, the optimal semiautonomous system would be minimally invasive. Using this model, determining when the system should intervene can be calculated using the following probabilistic intervention function, denoted G:

$$G(\alpha, \tau, \mathcal{O}, \mathcal{I}) = \begin{cases} 1 & \text{if } \exists k \text{ s.t. } P\left[\Delta_k(\mathcal{O}, \mathcal{I}, \alpha) \cap \mathcal{C}_k(\mathcal{I})\right] \geq \tau \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where $k \in \{0, \ldots, N\}$ represents the time step in the time horizon N, $\Delta_k(\alpha, \mathcal{O}, \mathcal{I})$ refers to the probability distribution on the set of α probable trajectories at time k as defined in Eq. 2, $\mathcal{C}_k(\mathcal{I})$ is the unsafe set given the current information $\mathcal{I}$, and $\tau \in [0, 1]$ is a predefined threat threshold. This means that the vehicle intervention function is 1 if the probability of the α probable trajectory set intersecting with an obstacle at any time k is greater than the threshold τ, indicating that the driver is unsafe.

This framework allows for the uncertainty of modeling and prediction to be incorporated in the threat assessment of the driver in a particular situation. By using the intervention functions, a decision can be made by the semiautonomous system as to whether or not control should be applied. Assuming the prediction is accurate and has good precision, the system will intervene only when necessary leading to fewer interventions than a simpler method using the reachable set of the vehicle.

Switched Control

The most obvious method of semiautonomous intervention is switched control. By this, it is meant that if the system detects danger for the driver, complete control will be taken from the driver, operating under the assumption that the system can outperform the driver. In the framework presented here, the controller would take over whenever the intervention function was set to 1.

There are a number of issues that arise from this method. The most prominent issue is dealing with handing control back to the human. This is an interesting engineering question, but also has some implications for human factors and psychology, concerning how a driver will react to the intervention and determining when it safe to hand control back to the human.

Augmented Control

Instead of completely relieving the human of his duties as driver, augmented control adds the minimum amount of control to the driver's input to keep the driver safe.

The augmented control is always on, removing the need to switch between autonomous and human control. The simulation described in the experimental setup also tested an augmenting control strategy using MPC. The controller algorithm runs in real-time to minimize a quadratic cost function as well as the following minimization problem:

$$\begin{array}{ll} \text{minimize} & \delta u^2 \\ \text{subject to} & G(\alpha, \tau, \mathcal{O}, \mathcal{I}) \leq 0 \\ & x(k+1) = f(x(k), u_{dm}(k) + \delta u(k)), \\ & \qquad \forall k = \{0, \ldots, N\} \end{array} \tag{7}$$

where all variables are as before and u_{dm} is given by the driver model. This minimization problem adds the minimal input needed to keep the driver safe. This method has been implemented in a real-time framework and has shown, promising and successful results [47].

7 Conclusion

The contributions of this application in driver modeling extend previous work by developing a realistic testbed for data collection, and increasing the utility and accuracy of this driver modeling method. In addition, we relax some of the assumptions on the underlying model of the vehicle, by including all inputs (steering, throttle, and braking) into the model. This creates more variability in the generated prediction sets, as we consider nonlinear behaviors of the human. Regardless, this implementation of the model exhibits comparable precision and significantly improved accuracy. The accuracy of the previous model ranged from 79.2% to 92.0% at 1.2 second time horizon.

By developing this testbed and this extended algorithm, we are able to collect realistic driving data and accurately predict driver behavior. This experimental setup is unique in that it allows us to collect data for and test human-in-the-loop systems, while maintaining safety measures and control of the environmental surroundings. This aids in creating a robust system as we can push the data collection to the search out corner cases or infrequent events that often arise in driving scenarios. By creating a flexible, context aware system, the identification is limited to regions that it has seen before yet is flexible enough to handle variances in scenarios. As was shown in [47], this formulation can be used in a semi-autonomous framework that is able to robustly respond to uncertain human behaviors.

By using these realistic data and flexible algorithm, a precise and accurate driver model can be developed that is tailored to an individual and usable in semi-autonomous frameworks and in driver behavior analysis. Future works include adding more contexts, like night-time driving, poor weather conditions, icy roads, levels of traffic, etc.; examining different distractions and the resulting variation in behaviors; and testing various control methods while the human is driving to verify that the system is minimally invasive and maintains appropriate safety margins. In particular, implementing and identifying

parameters for the probabilistic control framework will be explored to verify feasibility and reliability. We will also consider use in a real vehicle, through new, more realistic experiments and by examining the relationship between driving behaviors in a simulator and in an actual vehicle with respect to this model, as has been studied here [43].

Acknowledgements

A special thanks to Victor Shia for his contributions to the work and to Guillaume Bellegarde and Robert Matthew for their contributions in setting up the simulator system and in developing the modules. The contributions and feedback from Jim Gilson regarding the safety protocols and experiment design is also greatly appreciated.

Funding for these studies and the simulator was granted by the National Science Foundation Award ECCS-1239323 and by the Office of Naval Research MURI Award ONR-N000141310341 and DURIP Award N000141310679. The study protocol was approved by the University of California Institutional Review Board for human protection and privacy, under Protocol ID 2013-07-5459. Each subject was first informed on the experimental procedure and written informed consent was obtained.

References

[1] Driving Simulation Laboratory at The Ohio State University. http://drivesim.osu.edu/facility/. Accessed: 2014-1-20.

[2] Driving Simulator Lab, Department of Psychology at Clemson. http://www.clemson.edu/psych/research/labs/driving-simulator-lab/. Accessed: 2014-1-20.

[3] Follow the 4-Second Rule for Safety Spacing. http://dmv.vermont.gov/sites/dmv/files/pdf/DMV-Enforcement-SM-4_Second_Rule.pdf/. Accessed: 2014-1-20.

[4] Laboratory for Intelligent and Safe Automobiles, Intelligent Vehilce novel Experimental Test Beds. http://cvrr.ucsd.edu/LISA/TestbedThrust.htm/. Accessed: 2014-1-20.

[5] Language & Cognition Lab at the University of California, San Diego. Driven to distraction. http://www.cogsci.ucsd.edu/spotlight/9/. Accessed: 2014-1-20.

[6] Secretary of Defense for Acquisition Technology. DoD Modeling and Simulation Glossary. http://www.dtic.mil/whs/directives/corres/pdf/500059m.pdf/. Accessed: 2014-1-20.

[7] SMI Eye Tracking Glasses. http://www.eyetracking-glasses.com/. Accessed: 2014-1-20.

[8] Toyota Research. Pursuit for Vehicle Safety: Driving Simulator (Safety Technology Innovation from a Driver's point of view). http://www.toyota-global.com/innovation/safety_technology/safety_measurements/driving_simulator.html. Accessed: 2014-1-20.

[9] Volkswagon Automotive Innovation Lab at Stanford University. http://www.standford.edu/group/vail/. Accessed: 2014-1-20.

[10] Traffic safety facts, research note. driver electronic device use in 2011. http://www-nrd.nhtsa.dot.gov/Pubs/811719.pdf, 2011. Accessed: 2014-09-30.

[11] GM studying operator behavior in self-driving vehicles: Staying aware considered key to autonomous vehicle operation. http://media.gm.com/media/us/en/gm/news.detail.html/content/Pages/news/us/en/2012/Jun/0620_humanfactors.html, 2012. Accessed: 2014-09-30.

[12] Force Dynamics. http://www.force-dynamics.com, 2014. Accessed: 2014-09-30.

[13] Mobileye. http://www.mobileye.com, 2014. Accessed: 2014-09-30.

[14] PreScan Simulation Software. https://www.tassinternational.com/prescan, 2014. Accessed: 2014-09-30.

[15] Rethink robotics: Safety and compliance. http://www.rethinkrobotics.com/safety-compliance/, 2014. Accessed: 2014-09-27.

[16] Desdemona: The next generation in motion simulation. http://wwww.desdemona.eu/desdemona.html, 2015. Accessed: 2015-05-01.

[17] GoPro. http://pt.gopro.com, 2015. Accessed: 2015-05-01.

[18] Microsoft Kinect for Windows. https://www.microsoft.com/en-us/kinectforwindows/develop/, 2015. Accessed: 2015-05-01.

[19] OptiTrack: Motion Capture System. http://www.optitrack.com, 2015. Accessed: 2015-05-01.

[20] Simulation Laboratories: NASA Ames Research Center. http://simlabs.arc.nasa.gov/, 2015. Accessed: 2015-05-01.

[21] P. Badger. Capacitive Sensing Library. http://playground.arduino.cc//Main/CapacitiveSensor?from=Main.CapSense/. Accessed: 2014-1-20.

[22] D. Basacik and A. Stevens. Scoping Study of Driver Distraction. *Transport Research Laboratory. Road Safety Research Report*, 95, Feb 2008.

[23] L. M. Bergasa, J. Nuevo, M. A. Sotelo, R. Barea, and M. E. Lopez. Real-Time System for Monitoring Driver Vigilance. *IEEE Transactions on Intelligent Transportation Systems*, 7(1):63–77, Mar. 2006.

[24] H. Berndt, S. Wender, and K. Dietmayer. Driver braking behavior during intersection approaches and implications for warning strategies for driver assistant systems. In *IEEE Intelligent Vehicles Symposium*, pages 245–251, June 2007.

[25] F. Borrelli, M. Baotic, A. Bemporad, and M. Morari. Efficient on-line computation of constrained optimal control. In *Proceedings of the 40th IEEE Conference on Decision and Control*, volume 2, pages 1187–1192 vol.2, 2001.

[26] E. Coelingh, L. Jakobsson, H. Lind, and M. Lindman. Collision Warning With Auto Brake - A Real-Life Safety Perspective, Apr. 2007.

[27] B. Conductive. Making a Capacitive Proximity Sensor with Bare Paint. http://www.bareconductive.com/capacitance-sensor/. Accessed: 2014-1-20.

[28] A. Doshi, B. T. Morris, and M. M. Trivedi. On-Road Prediction of Driver's Intent with Multimodal Sensory Cues. *IEEE Pervasive Computing*, 10(3):22 – 34, Sept. 2011.

[29] K. Driggs-Campbell, V. Shia, and R. Bajcsy. Improved driver modeling for human-in-the-loop control. In *2015 IEEE International Conference on Robotics and Automation*, May 2015.

[30] K. Driggs-Campbell, V. Shia, R. Vasudevan, F. Borrelli, and R. Bajcsy. Probabilistic driver models for semiautonomous vehicles. *Digital Signal Processing for In-Vehicle Systems*, October 2013.

[31] L. Fletcher, N. Apostoloff, L. Petersson, and A. Zelinsky. Vision In and Out of Vehicles. *IEEE Intelligent Systems*, pages 12 – 17, June 2003.

[32] J. H. Gillula and C. Tomlin. Guaranteed safe online learning via reachability: tracking a ground target using a quadrotor. In *2012 IEEE International Conference on Robotics and Automation*, pages 2723–2730, May 2012.

[33] N. Gordon-Bloomfield. Nissan changes expectations, timeline for autonomous drive technology. `https://transportevolved.com/2014/07/17/nissan-changes-expectations-timeline-autonomous-drive-technology/`, 2014. Accessed: 2014-09-30.

[34] A. Gray, Y. Gao, T. Lin, J. Hedrick, and F. Borrelli. Stochastic predictive control for semi-autonomous vehicles with an uncertain driver model. In *Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on*, pages 2329–2334, Oct 2013.

[35] J. Greenberg and T. Park. The ford driving simulator. *SAE Technical Paper 940176*, 1994.

[36] P. A. Hancock, D. A. Vincenzi, J. A. Wise, and M. Mouloua. Human factors in simulation and training. 2008.

[37] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society, Series C (Applied Statistics)*, 18(1):100–108, 1979.

[38] K. Hulme, E. Kasprzak, K. English, D. Moore-Russo, and K. Lewis. Experiential learning in vehicle dynamics education via motion simulation and interactive gaming. *International Journal of Computer Games Technology*, 2009.

[39] B. Hyun, C. Park, W. Wang, and A. Girard. Discrete event modeling of heterogeneous human operator team in classification task. In *American Control Conference*, pages 2384–2389, June 2010.

[40] M. Jabon, J. Bailenson, E. Pontikakis, L. Takayama, and C. Nass. Facial Expression Analysis for Predicting Unsafe Driving Behavior. *IEEE Pervasive Computing*, 10(4):84 – 95, Apr. 2011.

[41] R. S. Jacobs and S. N. Roscoe. Simulator cockpit motion and the transfer of initial flight training. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 19(2):218–226, 1975.

[42] L. Jin, Q. Niu, H. Hou, H. Xian, Y. Wang, and D. Shi. Driver cognitive distraction detection using driving performance measures. *Discrete Dynamics in Nature and Society*, 2012.

[43] N. A. Kaptein, J. Theeuwes, and R. V. D. Horst. Driving simulator validity: Some considerations. *Transportation Research Record: Journal of the Transportation Research Board*, 1550:30–36, January 1996.

[44] M. Regan, J. D. Lee, and K. L. Young. *Driver Distraction: Theory, Effects, and Mitigation*. CRC, 2008.

[45] D. Sadigh, K. Driggs Campbell, A. A. A. Puggelli, W. Li, V. Shia, R. Bajcsy, A. L. Sangiovanni-Vincentelli, S. S. Sastry, and S. A. Seshia. Data-driven probabilistic modeling and verification of human driver behavior. Technical Report UCB/EECS-2013-197, EECS Department, University of California, Berkeley, Dec 2013.

[46] A. Sathyanarayana, S. Nageswaren, H. Ghasemzadeh, R. Jafari, and J. H. L. Hansen. Body Sensor Networks for Driver Distraction Identification. In *IEEE International Conference on Vehicular Electronics and Safety*, pages 120 – 125. IEEE, Sept. 2008.

[47] V. Shia, Y. Gao, R. Vasudevan, K. Campbell, T. Lin, F. Borrelli, and R. Bajcsy. Semiautonomous vehicular control using driver modeling. *IEEE Transactions on Intelligent Transportation Systems*, PP(99):1–14, 2014.

[48] B. W. Smith. Human error as a cause of vehicle crashes. `http://cyberlaw.stanford.edu/blog/2013/12/human-error-cause-vehicle-crashes`, 2013. Accessed: 2014-09-30.

[49] R. Vasudevan, V. Shia, Y. Gao, R. Cervera-Navarro, R. Bajcsy, and F. Borrelli. Safe semi-autonomous control with enhanced driver modeling. In *American Control Conference*, pages 2896–2903, 2012.

[50] E. Wahlstrom, O. Masoud, and N. Papanikolopoulos. Vision-Based Methods for Driver Monitoring. In *Intelligent Transportation Systems*, volume 2, pages 903 – 908. IEEE, Oct. 2003.