THESIS

TOWARDS ENABLING PREDICTIVE OPTIMAL ENERGY MANAGEMENT SYSTEMS FOR HYBRID ELECTRIC VEHICLES WITH REAL WORLD CONSIDERATIONS

Submitted by

Aaron Rabinowitz

Department of Mechanical Engineering

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Spring 2021

ABSTRACT

TOWARDS ENABLING PREDICTIVE OPTIMAL ENERGY MANAGEMENT SYSTEM WITH REAL WORLD CONSIDERATIONS

In the pursuit of greater vehicle fleet efficiency, Predictive Optimal Energy Management Systems (POEMS) enabled Plug-in Hybrid Electric Vehicles (PHEV) have shown promising theoretical results. In order to enable the practical development of POEMS enabled PHEV technology, if must first be determined what method and what data is needed is for providing optimal predictions. Research performed at Colorado State University and partner institutions in 2019 and 2020 pursued a novel course in considering the widest range of possible data and methods of prediction currently available including a survey of all feasible Vehicle to Infrastructure (V2I), Vehicle to Vehicle (V2V), Advance Driver Assistance Systems (ADAS), and Ego vehicle CAN data streams with classical and novel machine learning methods. Real world vehicle operation data was collected in Fort Collins Colorado, processed, and used in the development of optimal prediction methods. From the results of this research, concrete conclusions on the relative value of V2I, V2V, and ADAS information for prediction, and high fidelity predictions were obtained for 10 second horizons using specialized Artificial Neural Networks.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

With the passage of one fifth of the 21st century the world is ever more connected and interdependent thanks to increased access to information and transportation technology. While bringing a wide range of advantages, an increasingly interconnected world comes with higher energy consumption. At this time, the primary source of transportation for most people on earth is a vehicle powered by an Internal Combustion (IC) engine [4]. Observed air quality in the United States has declined over the past few years with a 15% increase in days within the unhealthy air quality range in the United States in 2018 and 2017 over the average from 2013 to 2016 due to greenhouse gas emissions. In the world at large, greenhouse gas emissions have been shown to be a significant contributor to global climate change [5] and criteria pollutants have been shown to be a contributor to lowered life expectancy in many countries [6].

Reliance on IC engine transportation is a large contributor to the total greenhouse gas emissions produced. The transportation sector is responsible for 27% of all greenhouse gas emissions produced globally and more than 50% of nitrogen oxide emissions [7]. As emissions production is a function of, amongst other things, fuel consumed, much research has been performed towards improving vehicle Fuel Economy (FE) in recent years [8]. The potential to mitigate the environmental impact of the IC engine is limited but reliance on the IC engine is likely to continue for the foreseeable future due to its convenience the high energy density of petroleum fuels, thus, much research development has also, lately, been performed on Hybrid Electric Vehicles (HEV) and Plug-in Hybrid Electric Vehicles (PHEV) which use electric powertrains in tandem with IC powertrains in order to reduce the fuel consumed per mile driven or to allow for extended Electric Vehicle (EV) operation in the latter case, only using the IC powertrain after depleting the battery charge. PHEVs have been shown to allow for high levels of improvement in fleet efficiency and emissions reduction [9], with much work still being done to determine the optimal proportion of power to be drawn from the EV and IC powertrains [10].

Although general strategies have been developed to optimize power distribution in HEVs and PHEVs, the true optimal behavior of any vehicle depends on the manner in which it is driven. Two approaches exist in applying this principle: First, Eco Driving uses driver level control (acceleration and braking requests) to guide the vehicle along a path from its current state to its destination [10], second, Optimal Energy Management Systems (OEMS) attempt to travel along a given path in the most efficient manner possible through the manipulation of powertrain parameters [9]. Research suggests that the best performance can be attained by using both in collaboration. However, while Eco

Driving requires autonomous control over the speed, acceleration, and piloting of the vehicle, EMS does not and, thus, is more readily implementable with near-term automotive technology.

Although OEMS can be implemented without any knowledge of the future motion of the vehicle, improved fuel economy can be gained using Predictive Optimal EMS (POEMS) which minimizes fuel consumption for a window of time in which a prediction of vehicle motion is available. POEMS can be used to maximize the FE of HEVs and PHEVs but it should be noted that the specific control strategy must differ between PHEVs and HEVs given the difference in the capability of the electric powertrain. PHEVs can travel long distances using only battery power, HEVs will mostly derive power, simultaneously, from both powertrains [11, 12].

With the emergence of Intelligent and Connected Vehicle (ICV) technologies, and the rapid market penetration of Advanced Driver Assistant Systems (ADAS) in the vehicle fleet, largely for the purpose of passenger convenience and safety benefits [13, 14, 15], the means of producing high-fidelity predictions of vehicle motion are, or will soon be, widely available. Modern consumer vehicles have both the means to sense and predict future vehicle states through ICV/ADAS technoloiges, and the means to actutate powertrains to achieve fuel economy optimized operation. Together these enable POEMS type hybrid vehicles. However, POEMS is far from a mature technology and significant research gaps remain.

In order to facilitate the practical introduction of predictive Optimal EMS into the automotive market, certain research gaps must be addressed. The first of these research gaps as defined in [12] is:

- The sensitivity of optimal EMS performance to actual velocity prediction is not understood. Much of the OEMS literature concentrates on predictions using drive cycle data, and idealized driving schedules. By using real-world derived vehicle velocity predictions, we can quantify how effective these techniques are under conditions closer to their real world application.

These research gaps build off of previous work from researchers at Colorado State University and other institutions. The efficacy of POEMS for improving efficiency in PHEVs was first shown in 2008 in [16] which predicted velocity via an analytical traffic model, the benefits of integrating Artificial Neural Network (ANN) prediction for POEMS were shown in 2015 in [17, 18], in 2017 and 2018 a series of studies [19, 20, 21, 22] showed the relative importance of various individual data streams for use with shallow ANN velocity prediction, in 2019 more modern machine learning techniques were introduced into the field in [23] where reinforcement learning was used to train a deep ANN to output optimal control for a power-split hybrid based on traffic data, also in 2019, [24, 25] showed that high degrees of prediction fidelity could be attained through the use of deep Long-Short-Term-Memory (LSTM) ANNs, finally, in 2020, a wide ranging analysis of various combinations of real world data streams and machine learning techniques [26, 1] showed that the highest degree of prediction fidelity could be attained through the use of LSTM ANNs with the use of Signal Phase and Timing (SPaT) and Lead Vehicle data.

This thesis summarizes research performed towards the goal of identifying the optimal combination of prediction method and prediction data-set, in order to aid in addressing the above listed research gap. Research conducted towards this goal included the categorization of available data and collection and processing methods, the collection of a real-world data-set based on said classification in Fort Collins Colorado, the processing of said data-set, the use of said processed data for LSTM ANN prediction of vehicle motion, and the used of said predictions to determine the optimal data-set in order to enable the greatest possible improvement from POEMS.

Chapter 2

Data Categorization

2.1 Data Categorization

To address the challenges associated with the research gaps identified in chapter 1, this work began by developing a categorisation of the data that would be needed to characterize the performance of PEOMS algorithms. The goals of this data characterization were to create a data-set which would serve to represent a generic ICV interacting with a generic Intelligent Transportation System (ITS). In order to determine what data to collect, it was first necessary to define a generic taxonomy for ICV data collection and processing. For clarity, the various forms of data received by the vehicle will be called Data-Streams, and the various forms of processed data will be called Data-Treatments.

2.2 Data-Streams

A taxonomy of Data-Streams is shown in Figure 2.1 with a separation between those streams which do and do not require communication with vehicles or infrastructure to produce data.

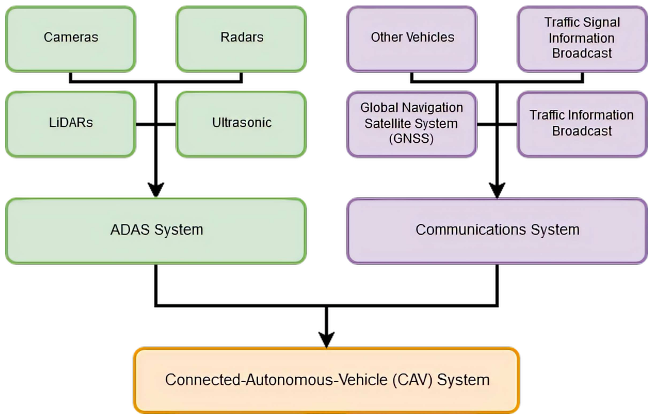
Cameras
Radars
Other Vehicles
Traffic Signal Information Broadcast
LiDARs
Ultrasonic
Global Navigation Satellite System (GNSS)
Traffic Information Broadcast
ADAS System
Communications System
Connected-Autonomous-Vehicle (CAV) System

Figure 2.1: Distinguishing between ADAS and COMM Data Streams from [1], data generation is performed on-vehicle.

Advance Driver Assistance Systems (ADAS) [27] systems are a fairly mature technology and are present on many new vehicles [28] in safety and driver assistance contexts. On-vehicle communications systems, sometimes known as vehicle to other (V2X), are an emerging technology which can presently be used for traffic system optimization [29], but require large scale implementation to be effective for vehicle level controls as opposed to ADAS systems which can be effective even if present on a limited section of the fleet. In order to avoid unintended association with specific protocols [30], communications data-streams will be referred to as COMM.

A central focus of this thesis is to evaluate the relative effectiveness of various data streams in terms of enabling enhanced velocity prediction fidelity; these data-streams are defined in more detail in [1].

2.3 Data-Treatments

While the physical implementation of CAV systems will vary greatly vehicle to vehicle and manufacturer-to-manufacturer, a generic taxonomy of data-treatments is also proposed, into which, most implementations will fit. A further subdivision into Relevance Areas RA is defined below:

1. Ego Vehicle Data (EGO): Data which concerns the state of the vehicle, such as powertrain parameters and vehicle motion.

2. Vehicle-Immediate-Surroundings (VIS): Data which concerns the position, motion, and classification of the objects in the vehicle's immediate surroundings. This data can originate from both ADAS and COMM sources and is important for safety, driver assistance, and efficiency applications.

3. Forward-Path-Information (FPI): Data concerning the condition of the transportation system such as segment speeds and traffic signal states in the planned path of the vehicle. FPI data is relevant for efficiency and planning applications.

Shown in Figure 2.2 is a more detailed subdivision of the VIS relevance area, note that certain VIS sub-areas are also relevant to FPI.

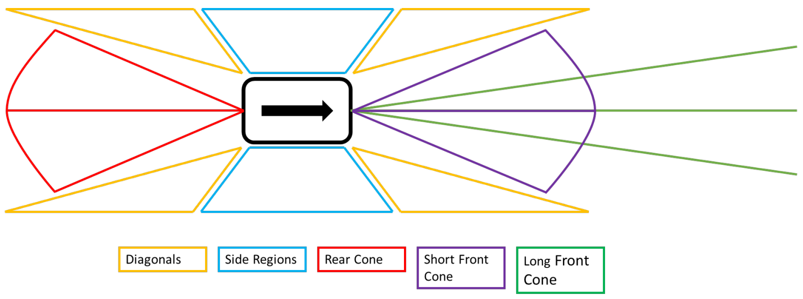

Figure 2.2: Vehicle FOV areas. Rear and front short cones, side areas, and diagonals compose VIS. Long forward cone is within FPI.

Data-treatments are related to relevance areas and data-streams below in Table 2.1:

Table 2.1: Descriptions of data treatments, their relevance areas, and data streams from which they can be acquired [1]

Relevance Area	Data Treatment	Description	Data Stream
EGO	Driver Inputs	Steering, accelerator and brake pedal traces, drive mode selection, turn signal, etc.	VEH
EGO	Vehicle Performance	Vehicle speed, engine load and speed, transmission gear, accelerations, etc.	VEH
EGO	Vehicle Position and Motion	GNSS position and motion information	COMM
VIS	Object Tracks	Relative locations and classifications of detected and tracked objects	ADAS, COMM
VIS	Lane Information	Information about vehicle position and trajectory relative to the vehicle's current lane	ADAS, COMM
VIS	Condition Information	Lighting and weather conditions in the vehicle's environment	ADAS, COMM
FPI	SPaT	Phase and timing information for traffic signals	COMM
FPI	Segment Speeds (SS)	Average vehicle speeds for segments of road	COMM
FPI	Lead Vehicle (LV)	relative position and motion of the vehicle most immediately in front of the ego vehicle	ADAS, COMM
FPI	Historical Speeds (HS)	Speeds which vehicles have historically traveled at specific locations	VEH, COMM

Chapter 3

Data Collection

3.1 Background

Data collection was performed in the Autumn of 2019 in Fort Collins Colorado using the CSU EcoCAR Mobility Challenge 2019 Chevrolet Blazer instrumented with a forward facing Mobileye 630 Camera [31] and a forward facing Bosch Mid Range Radar [32]. The following considerations dictated data collection:

1. Data collection should take place in a representative urban area where vehicles will be expected to stop frequently and traffic level will play a role in determining vehicle motion.
2. Data collection should take place in an area where all data collection methods are applicable at all times.
3. Data collection should comprise of many iterations of the same route to provide a reasonable training and testing data-set for prediction.
4. Multiple drivers should drive iterations of the route such that the effect of driver on prediction can be quantified.
5. All streams in EGO and FPI relevance area should be either directly captured or captured in proxy
6. All data collected should come from currently available and deployed technology; preferably that which might be expected to be present on a current ADAS enabled vehicle.

3.2 Drive Cycle

the route illustrated in Figure 3.1 was selected as the drive cycle over which all testing would be performed for this thesis:

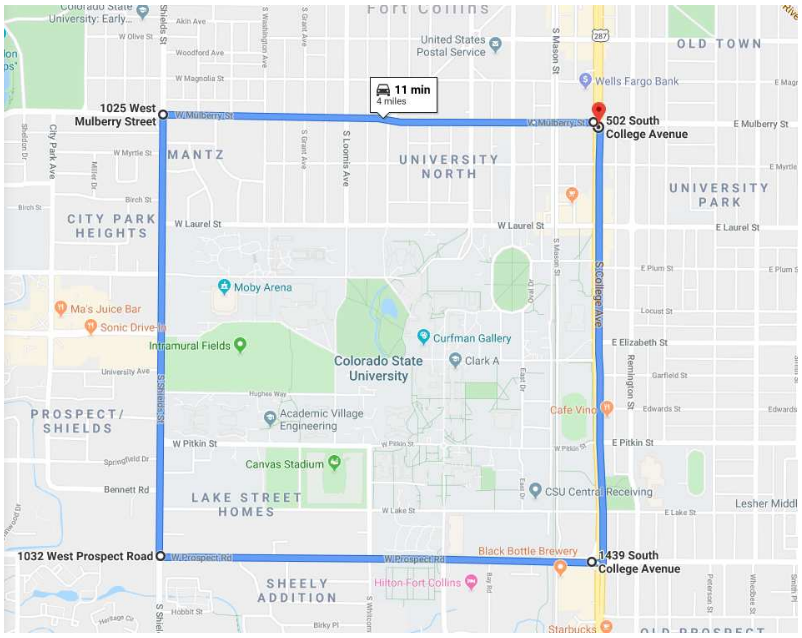

Figure 3.1: Data Collection Drive Cycle

This cycle included segments on four high volume arterial roads in Fort Collins and covered roughly 4 miles in 10-20 minutes depending on multiple factors including traffic. In contrast to previous longer routes, this shorter route was chosen as it allowed for 7-10 drive cycles to be collected in a 2 hour session and thus it was feasible to collect a large number of drive cycles with weekday driving conditions. Additionally, the selected route satisfied condition 1 by encountering 20 traffic signals, and satisfied condition 2 by encountering 4 distinct traffic monitoring zones as shown below.

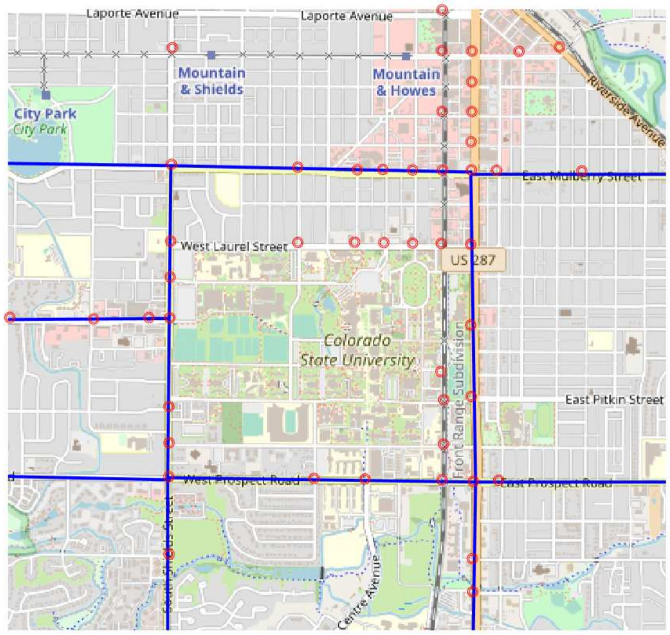

Figure 3.2: Traffic signals (red targets) and traffic monitoring segments (blue lines) encountered along data collection drive cycle

Ultimately data would, successfully, be collected on this route in three driving sessions of 5, 8, and 7 laps each with the first tow being driven by the same driver and the third being driven by a different driver. The variability of speeds and lap times is visible in figure 3.3 which compares speeds (color axis) measured at repeated locations over the 8 laps driven in session 2.

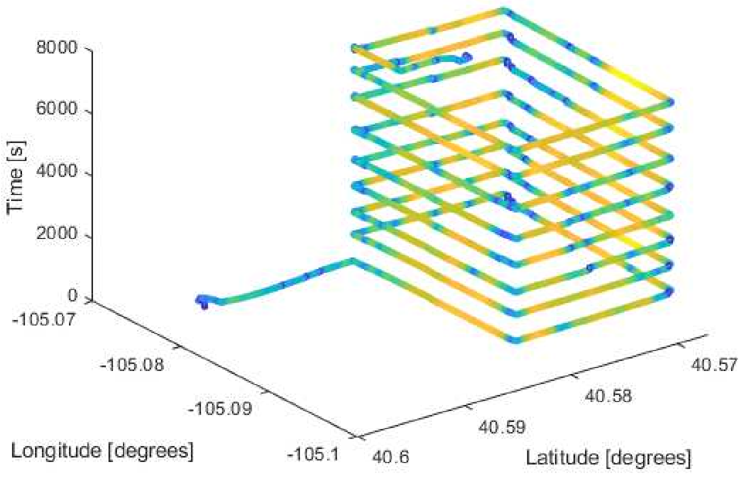

Figure 3.3: Lat/Lon Position vs Time Colored by Speed for Laps from Session 2

3.3 Collection of EGO and ADAS Data

External object data was collected using the forward mounted and facing Mobileye 630 and Bosch MRR. The most common forward facing external object sensor system for ADAS enabled vehicles is a windshield mounted camera/ grille mounted radar setup and thus this setup was intentionally replicated [28]. The practical fields of view available from these sensors overlap to produce roughly a $\pm 15^o$ cone in front of the vehicle up to 180 m as shown below.

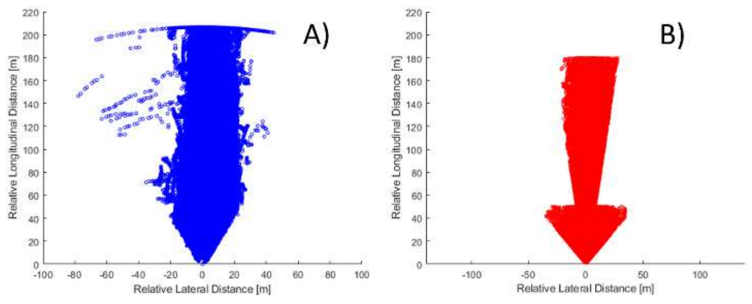

Figure 3.4: Practical Fields of View for A) Mobileye 630 Camera B) Bosch MRR

In addition to the external object monitoring system, EGO data was collected from vehicle CAN and, although Global Navigation Satellite System (GNSS) positioning data was available from the vehicle CAN network, an additional high accuracy GNSS receiver was used for reasons which will be expanded upon in chapter 4. Finally, as all of the sensors were capable of Controller Area Network (CAN) communication, all data was able to be logged centrally through the use of a Vector VN1630. The mounting of sensors is shown in Figure 3.5.

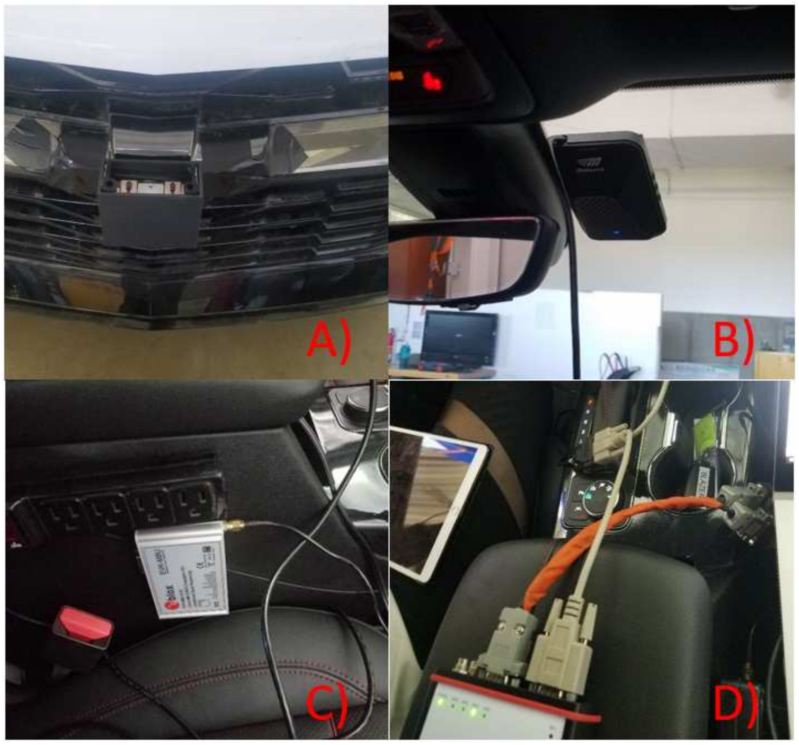

Figure 3.5: Physical locations of sensors on probe vehicle A) Radar mounted on front and center of vehicle grille B) Camera mounted on windshield slightly offset to the left C) GNSS Receiver mounted on center console, near-as-possible to vehicle center of mass D) Vector CAN interface mounted on center console

3.4 Collection of COMM Data

COMM data such as SPaT and SS data was collected from the City of Fort Collins traffic management department via TraffiCast. SPaT data is directly logged by TraffiCast at 10 Hz, all forms of traffic flow estimation, including that used by Fort Collins rely on tracking cell phones as they pass from signal to signal which creates inherent limitations. Limitations of cell phone based SS information are as follows:

1. Cell phone tracking can only track a sample of the vehicles on any given segment and is unlikely to track all vehicles.
2. ITS tracking, as opposed to that used by Google Maps or Waze for example, tracks phones as they communicate with antennae at fixed locations such as intersections, thus limiting the number of routes which can be tracked. In Fort Collins, only motion between major intersections is tracked.
3. Distances between tracking locations mean that, while SS information is relevant for traffic control, the information may be of limited relevance for any individual vehicle traveling along route.

Real time use of continuous and cloud integrated cell phone tracking data, as in Google Maps/Waze, is still not truly feasible for current ICVs and thus the immediate future of ITS/ICV interaction will heave to rely on the limited point-to-point style SS estimation as used in Fort Collins.

Chapter 4

Data Processing

In order to generate Data Treatments from Data Streams, two processes need to occur, these are:

- Sensor Fusion: The process by which different streams of information from measurements are used to track immeasurable information. For this thesis, sensor fusion was used to track external objects relative to the ego vehicle and to estimate the curvature of the vehicle's motion path.
- Localization: The process by which a vehicle's position and motion, with respect to infrastructure, is determined such that relevant ITS information can be used by the vehicle. For this thesis localization was used to assign instantaneous HS, SS, and SPaT information to the ego vehicle.

The methods used for sensor fusion and localization will be discussed in detail in this section.

4.1 Sensor Fusion

4.1.1 External Object Sensor Fusion System

The design of an External Object Sensor Fusion (EOSF) system for an ADAS system composed of radars, cameras, and LiDARs (as most currently are [33]) must take into account the following concerns:

- Variable effectiveness of sensors based on external conditions. Of the three types above mentioned, radars and LiDARs are susceptible to interference from rain or dust while cameras are susceptible to adverse lighting conditions.
- Unknown expected number of objects in immediate vicinity of vehicle, the EOSF system will have to retain functionality for conditions which involve a wide range and varying of number of actual objects to track.
- Wide variety of types of objects to track such as vehicles, pedestrians, road signs, and others.
- Regularity with which objects obstruct Line of Sight (LOS) from the ego vehicle to other objects
- Additional uncertainty due to the moving measurement platform which forces the system to account for uncertainty in the position and motion of the origin in global coordinates

- High false positive rate among external object sensors

The problem of object tracking, thus, lends itself to a Bayesian Inference type solution [34]. Uncertainty exists in both the measurement of object locations and the model used to predict their motion. Bayesian inference solutions can be used for a wide variety of tracking and prediction applications and there are many Bayesian type tracking algorithms present in the literature [35] [36]. For the motion of macroscopic objects with well understood physics, a natural algorithm of choice is the Kalman Filter (KF) [37]. The KF and its variants are currently the standard for automotive measurements including the problem of tracking external objects. The operating principle of such filters is that knowledge of an objects dynamics, even if uncertain, combined with knowledge of its location and motion, even if uncertain, allows one to track the objects location and motion with greater certainty than either that of its dynamics or its measured properties.

External Object Kalman Filter Design

The Kalman Filter is based on a motion model, F, which is an Euler sampling of the system of equations which define the transition of the object in question from its state at the current time step to the next time step, and a measurement model, H, which defines which states are being measured.

The equations of the Kalman filter are shown below and can be found in greater detail in [35], [38], [36], and [37].

Step	$KalmanFilter$
$Predict$	$\bar{X} = X + FX$
	$\bar{P} = \alpha FPF^T + Q$
$Update$	$Y = Z - H\bar{X}$
	$S = H\bar{P}H^T + R$
	$K = PH^T S^{-1}$
	$X = \bar{X} + KY$
	$P = \bar{P} - KSK^T$

In the Predict step, the object state vector, $\bar{x}$, at time step $t + dt$ is predicted using the motion model F and the object state, X, at time t while the object state covariance (uncertainty), $\bar{P}$, is predicted based on the motion model, the current state covariance P, and the process noise model Q.

In the Update step The residual Y between the measurement Z and predicted state is calculated using the measurement model H, the residual covariance which functions as a sum of uncertainty for the step S is calculated using the measurement model, predicted state uncertainty, and the measurement covariance R, the Kalman gain, which serves as a ratio of prediction uncertainty to total uncertainty, is calculated based on the prediction and residual covariance, and finally a new state and stated covariance are calculated based on the Kalman Gain.

Because the Kalman filter, in its basic form, is not capable of dealing with non-linear systems, it can be used, instead, with linearized motion and/or measurement models as an Extended Kalman Filter (EKF). The Kalman filter can also be modified to deal with non-linearity through statistical sampling using the Unscented Transform with this filter being known as the Unscented Kalman Filter (UKF) [39].

The α parameter in the state covariance matrix is the Recency-Bias Parameter serves to inflate the importance of the uncertainty from the current step and to decrease the effects of past steps allowing the filter to adapt quicker if $\alpha > 1$ or slower if $\alpha < 1$.

The decision to use a linear Newtonian model was made as the specific information required to construct a more specific motion model was not available. A simple vehicle model, as shown in section 4.1.2 requires specific information about vehicle steering geometry which was not available for EOSF vehicle targets. In the absence of a vehicle-specific model, the next best option is linear Newtonian motion, which will hold most of the time. Having selected a linear motion model, the linear KF was also selected.

In the process of designing a Kalman Filter for external object tracking first consideration was given to the state vector which would be used. In this case, as the filter must be able to track the motion of objects as different as motorcycles and road signs, it was decided to experiment with both a constant velocity (cv) and constant acceleration (ca) model. These are presented below:

$$X_{cv} = [x, \dot{x}, y, \dot{y}]^T \tag{4.1}$$

$$X_{ca} = [x, \dot{x}, \ddot{x}, y, \dot{y}, \ddot{y}]^T \tag{4.2}$$

Where x is the unit vector for motion along the longitudinal axis of the ego vehicle and y is the unit vector for motion along the lateral axis of the ego vehicle both originating at its Center of Mass (COM).

In designing the motion model for external objects consideration was given to the variety of types of objects which the vehicle might encounter. While vehicles and road signs have relatively constrained motion, pedestrians do not. Thus, a very general Newtonian motion model was adopted (equation 4.3).

$$F_{cv} = \begin{bmatrix} 1 & dt & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & dt \\ 0 & 0 & 0 & 1 \end{bmatrix} F_{ca} = \begin{bmatrix} 1 & dt & .5dt^2 & 0 & 0 & 0 \\ 0 & 1 & dt & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & dt & .5dt^2 \\ 0 & 0 & 0 & 0 & 1 & dt \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

The process noise model is meant to account for deficiencies in the motion model. For example noise can be caused by unexpected things such as road inclination and wind. Because these effects are not sustained and not unidirectional, thus they may be treated as random. The degree to which one can expect the effects of random noise to offset a prediction should be proportional to the length of the time step of the prediction. For all these reasons the process covariance is modeled as Gaussian white noise [40] modified by a tuning parameter σ.

$$Q_{cv} = \sigma \begin{bmatrix} .25dt^4 & .33dt^2 & 0 & 0 \\ .33dt^2 & dt^2 & 0 & 0 \\ 0 & 0 & .25dt^4 & .33dt^2 \\ 0 & 0 & .33dt^2 & dt^2 \end{bmatrix} Q_{ca} = \sigma \begin{bmatrix} .25dt^4 & .5dt^3 & .5dt^2 & 0 & 0 & 0 \\ .5dt^3 & dt^2 & dt & 0 & 0 & 0 \\ .5dt^2 & dt & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & .25dt^4 & .5dt^3 & .5dt^2 \\ 0 & 0 & 0 & .5dt^3 & dt^2 & dt \\ 0 & 0 & 0 & .5dt^2 & dt & 1 \end{bmatrix} \tag{4.4}$$

As the sensors employed output detections in a variety of different coordinate systems, the detections are processed into the ego vehicle 2D Cartesian coordinate system via separate logic before they are fed to the SF algorithm. The measurement vector for the system was:

$$Z = [x, y]^T \tag{4.5}$$

Thus the measurement models were:

$$H_{cv} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} H_{ca} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.6}$$

Finally, the measurement noise covariance matrices were defined for a given sensor *i* based on a *x* and *y* uncertainty as:

$$R_i = \begin{bmatrix} a_i & 0 \\ 0 & b_i \end{bmatrix} \tag{4.7}$$

where a_i and b_i represent the uncertainty for sensor i in x and y respectively. It should be noted here that the Kalman filter designed above, in either configuration is linear. Given the lack of information about the external objects being tracked there is no advantage to choosing a more specific, non-linear model. A second very important takeaway is that the entire filter is parametrized by only the following values:

$$\Theta = [\alpha, \sigma, a_1, ..., a_n, b_1, ..., b_n] \tag{4.8}$$

The length of the above parameter vector is $2n+2$ for n sensors in the system. This allows for a manageable design space when it comes to tuning the system.

Multi-Object Tracking

Thus far, object tracking has been discussed for single objects only. In reality, however, an ICV needs to track multiple objects and deal with multiple measurements. In the case of ICV object tracking, the object tracking algorithm will have to track n_t targets based on n_d number of detections, where n_t and n_d may vary by frame. n_t and n_d will vary as the number of objects in a vehicles surroundings change. Furthermore objects may come in and out of the vision line of the vehicle sensors such as when a vehicle cuts in and out of traffic or when a pedestrian walks behind a vehicle. In order to track multiple targets, detections must be assigned to tracks or used to create new tracks. Multi-object tracking is an additional problem which must be considered separately but must be optimized to be compatible with filtering. The sensor fusion logic is laid out below.

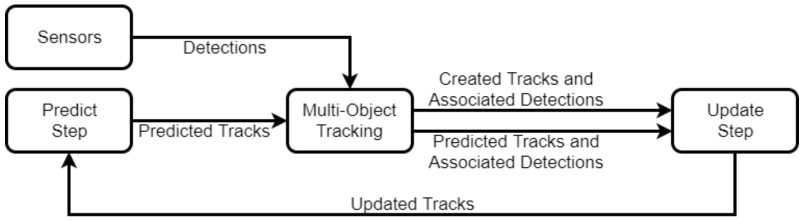

Figure 4.1: Full Logic Flow for Kalman Filter

As is shown above, another logic block has been added to the cycle called ”Multi-Object Tracking”, the function of this block is to associate detections with tracks where possible, create new tracks where needed, and remove non-valid tracks. Because sensors are noisy and can report false detections, especially in an ICV context [41], it is beneficial to have some method of determining the validity of tracks before sending them to the control system. A method of ensuring the validity of tracks is Promotion-Relegation (PR), PR systems initialize tracks as ”tentative” until they meet some criteria which allows them to be promoted to ”active” [36] and only active tracks are sent to the control system. In order to stay active a track must maintain a degree of plausibility, if this is not met the track is considered coasted until it regains the required plausibility to be active again. Usually, if a track is coasted for a sufficient number of frames it is dropped entirely. Using this framework, a system can be set up to deal with both false detections and cut-in cut-out behavior, naturally, this also introduces a series of design decisions and tuning parameters.

Data Association

The multi-object tracking block can be further subdivided, as shown below, into data association and track management.

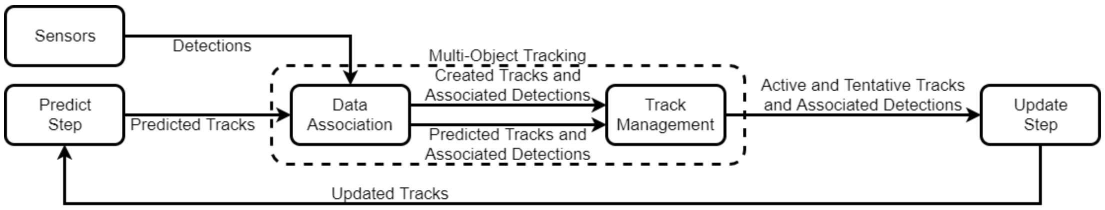
Sensors
Detections
Multi-Object Tracking
Predict Step
Predicted Tracks
Data Association
Created Tracks and Associated Detections
Predicted Tracks and Associated Detections
Track Management
Active and Tentative Tracks and Associated Detections
Update Step
Updated Tracks

Figure 4.2: Full Logic Flow for Kalman Filter

The job of the data-association logic is to find a global optimum solution for matching tracks to detections. For this task a very useful method is the Hungarian Algorithm.

Hungarian Algorithm

The Hungarian Algorithm is a method of computing a global optimum solution for matching agents to tasks based on cost [42, 43]. The method works by attempting to find the minimum cost solution for a cost matrix through the use of permutation matrices [44].

$$\min_{L,R} \sum (LCR) \tag{4.9}$$

Where L and R are the permutation matrices. As this method does not take and input or initialization parameters and may be used as a pre-developed, optimized, and hardware deployable part of the Scipy library, it is not necessary to develop custom logic for it and thus its derivation is not necessary for the context of this document [42, 44].

Mahalanobis Cost

The cost matrix for the Hungarian Algorithm must, however, be generated. For n_t tracks and n_d detections this will take the form of an n_d by n_t matrix as shown below.

$$COST(Z,X) = \begin{bmatrix} f_c(z_1,x_1) & f_c(z_1,x_2) & \dots & f_c(z_1,x_{n_t}) \\ f_c(z_2,x_1) & f_c(z_2,x_2) & & \vdots \\ \vdots & & \ddots & \\ f_c(z_{n_d},x_1) & \dots & & f_c(z_{n_d},x_{n_t}) \end{bmatrix} \tag{4.10}$$

In this case the cost $COST_{i,j}$ can be defined as the Mahalanobis distance between the detection z_i and the track x_j using the formula:

$$COST_{i,j} = f_c(z_i,x_j) = \sqrt{(z_i - x_j)^T S_j^{-1}(z_i - x_j)} \tag{4.11}$$

Track Management

There are essentially two manners in which track management can be done: via likelihood, and via age. The likelihood method involves evaluating the likelihood of each track and setting promotion and relegation levels P_p and P_r. Likelihood can be calculated using the Mahalanobis distance or likelihood ratio, as discussed earlier.

A potential issue with the likelihood method is that is sensitive to the performance of the track initialization logic. This invites the possibility of strong covaraince between the filter and track management parameters in evaluation and optimization which increases the dimensions of the design space and increases the effort needed to tune the system.

Another method of PR is the age method, where promotion is is done by age. In other words, if a tentative track is maintained (not coasted) for n_f frames then it is promoted. It would still be recommended, in this case to relegate by Mahalanobis distance as the effects of the initialization function become less pronounced over time.

Due to the lack of applicable data, it was decided to use the age method with the following parameters based on educated guesses with n_f set to 2 and a relegation distance of $d_r = 5\sigma$, so as to minimally interfere with the Kalman Filter tuning.

EOSF Kalman Filter Tuning

The EOSF Kalman Filter was tuned using designed factorial experiments and the method of steepest ascent to find optimal parameters for the filter. Relevant consequences of the particular Kalman Filter design chosen for tuning experimental design are the following:

- The limited tuning parameter vector $\Theta = [\alpha, \sigma, a_c, b_c, a_r, b_r]$ where c refers to the Mobileye and r refers to the MRR
- $Q = f(\sigma)$ so $\bar{P} = f(\alpha, \sigma) = \alpha FPF^T + Q(\sigma)$, and $R = f(a,b)$ so $P = f(\bar{P}, R) = \bar{P} - K(H\bar{P}H^T + R)K^T$, thus $P = f(\Theta)$. This suggests that significant interaction should be expected between the parameters in Θ.

The ultimate function of a Kalman filter is to maintain its tracks with a high degree of certainty, in order to do this, the initialization of the model must allow the filter to quickly acquire a reasonable mean for a track, understand its uncertainty, and quickly adjust should this change. To this end, Mahalnobis distance was implemented as a metric of statistical distance between a track and its measurements. The equation for the Mahalanobis distance in Kalman notation is:

$$d = \sqrt{Y^T S^{-1} Y} \tag{4.12}$$

Where the distance is in units of standard deviations. For convenience this can be transformed into a pseudo-likelihood using the Likelihood Ratio of the residual and a residual of zero length.

$$L = P(Y,S)/P(\hat{0},S) \tag{4.13}$$

Where the likelihoods are calculated using the multivariate normal distribution [45].

$$P(X) = f(Y,S) = \frac{1}{\sqrt{(2\pi)^n \det(S)}} \exp\left[-\frac{1}{2} Y^T S^{-1} Y\right] \tag{4.14}$$

This likelihood ratio was the metric of performance used for tuning.

6 Parameter Tuning

Initially, due to the expected presence of high leverage interaction terms between all factors an experiment was designed to account for all parameters in Θ. Because the purpose of the study was to find an optimal set of tuning parameters and the experiment can be run repeatedly, it was decided to run a factorial experiment around a starting educated guess and, if significant terms were present $p(> t) \leq .05$, then to use the method of steepest ascent to find a stationary point.

The initial experiment was initially conceived as a 3^6 full factorial with two blocks, for the ca and cv models. Because this would have required 1458 runs to complete, at about 10 minutes per run, this was not thought to be feasible. Instead a 2 block 3^{6-3} fractional factorial was selected, which would require 54 runs to complete. The assignment of parameters was as follows:

Table 4.1: Assignment of Parameters in $2 * 3^{6-3}$ Fractional Factorial

A	B	C	AB	AC	BC	D
α	a$_c$	a$_r$	σ	b$_c$	b$_r$	model

Where A through BC are the fractional factorial and D is the model blocking. In determining which parameters to assign to which letters the main concern was aliasing. In any case, three parameter effect will be aliased with three interactions; the above assignment was chosen such that these aliases would be:

- $\alpha : a_c \longrightarrow \sigma$
- $\alpha : a_r \longrightarrow a_c$
- $a_c : a_r \longrightarrow b_r$

It was determined that these aliases represented the least likely set of high leverage interaction terms possible. The 3^{6-3} fractional factorial was orthogonal as all rows, dotted with all other rows, resulted in zero, and all rows were unique from all other rows. As shall be shown, the blocking for model was eventually dropped and the factorial became:

Table 4.2: Assignment of Parameters in 3^{6-3} Fractional Factorial

A	B	C	AB	AC	BC
α	a_c	a_r	σ	b_c	b_r

Once the entire factorial run was complete the results Y were regressed onto an inputs vector X, which was composed of the parameters and multiples and combinations of the parameters, using statsmodels [46] Ordinary Least Squares (OLS) regression to create a linear equation $Y = ||\beta * X||$, where β are the model coefficients. in order to determine if a model was sufficient to use for generating the next factorial, regression analysis was performed. The coefficient of determination (R^2) which represents the portion of the result variable deviation form mean which can be explained by the model, and the Prediction Sum of Squares (*PRESS*) which represents the error with which the model can be expected to predict the result for a data point not in its training set. For the model to be acceptable, $R^2 \geq 0.8$ and $PRESS \leq 0.2$; if these criteria were not met the model had to be altered until they were before the next factorial could be generated.

Having performed regression and returned a suitable model, a new factorial can be generated by using the model coefficients and a fixed step size. In this case the new center-point for an input parameter x_i can be calculated as follows:

$$\bar{x_i} = x_i + \frac{\beta_i}{\beta_{max}} step \tag{4.15}$$

With the new center-points generated the factorial can be run again and the analysis performed again. This process can be repeated until a stationary point is reached at which no more significant terms remain.

Iteration 1

The starting point for the parameters Θ was:

Table 4.3: Tuning Parameter Levels for Evaluation

Level	α	σ	a_1	b_1	a_2	b_2
-1	1.5	.01	.001	.001	.001	.001
0	185	.1	.01	.01	.01	.01
1	2.1	1	.1	.1	.1	.1

The values for the sensor uncertainties were based on information from confidential documents provided to the team which led to ball-parking the uncertainties at 10^{-2} meters. The values for α and σ were educated guesses spread widely to cover a large design space. for regression the levels for σ and the sensor parameters were expressed as their exponent.

Table 4.4: Tuning Parameter Levels for First Regression

Level	α	σ	a_1	b_1	a_2	b_2
-1	1.5	-2	-3	-3	-3	-3
0	1.8	-1	-2	-2	-2	-2
1	2.1	0	-1	-1	-1	-1

The experiment was conducted at these levels and the result were distributed as follows:

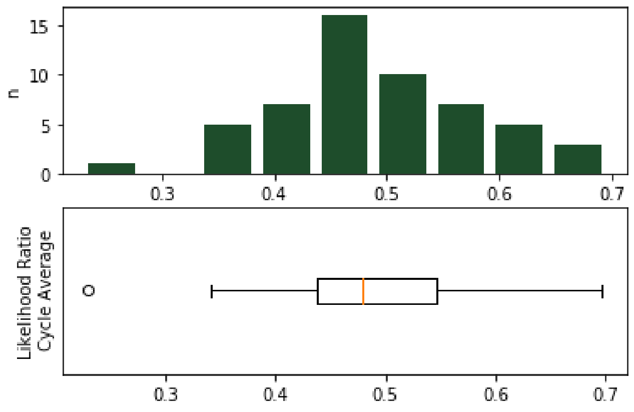

Figure 4.3: Distribution of Likelihood Ratio for First Iteration

This uni-modal distribution indicated that a decent linear fit should have been possible. The following models were tried and their R^2 and Predicted Error Sum of Squares (*PRESS*) are shown:

Table 4.5: Models Tried for First Regression

Model	R^2	$PRESS$
$L \approx C(model) + \alpha + \sigma + a_c + b_c + a_r + b_r$	.689	
$L \approx C(model) + \alpha + \sigma + a_c + b_c + a_r + b_r + \alpha^2 + \sigma^2$	.7030	.1322
$L \approx C(model) + \alpha * \sigma + \alpha^2 + \sigma^2 + a_c * b_c * a_r * b_r$	.7965	.0905
$L \approx C(model) + \alpha * \alpha^2 + \sigma + \sigma^2 + a_c * b_c * a_r * b_r$	.7966	.0905

Where $C()$ means that the variable is categorical and the $*$ symbol means to consider the interaction term as well. With the last model at just less than $R^2 = 0.8$, the residuals and fit were visualized in order to assess the reasons for the lack of fit.

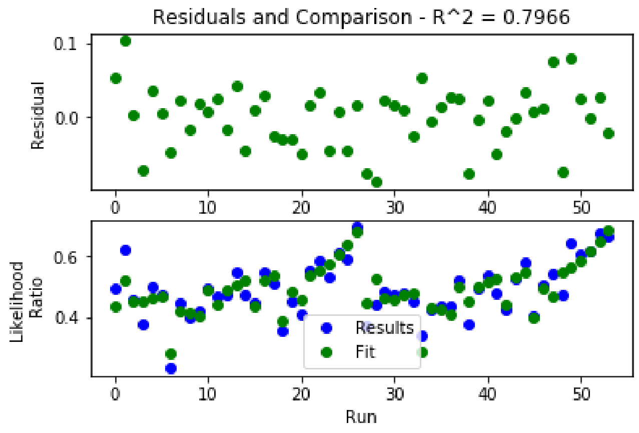

Figure 4.4: Visualization of Residuals and Fit for First Regresion

While the residuals appeared to be uniformly random, figure 4.4 illustrates that the trend of results was different between the ca and cv models meaning that each would have to be optimized independently. For this reason the decision was made to focus on optimizing the cv model as it is computationally cheaper and its performance appeared to be roughly equal to the ca model. After dropping the ca model (and reducing the run-time of the experiment by half), the process of generating the next factorial was started.

Based on the last model the significant parameters were:

Table 4.6: Significant Terms from First Regression

Parameter	β
α	0.128253
$a_c : a_r$	$-.116933$
$a_r : b_r$	$-.130162$

Thus α was the variable used for steepest ascent, and the next iteration could be run.

Iteration 2

The second iteration was run and fitted with the following results:

Table 4.7: Models Tried for Second Regression

Model	R^2	*PRESS*
$L \approx \alpha * \sigma + \alpha^2 + \sigma^2 + a_c * b_c * a_r * b_r$	.8868	.0162

There was only one significant term for the second regression and that was:

Table 4.8: Significant Terms from Second Regression

Parameter	β
b_r	$-.449644$

Using the b_r term the factorial for the third iteration was generated.

Iteration 3

The third iteration was run and fitted with the following results:

Table 4.9: Models Tried for Third Regression

Model	R^2	*PRESS*
$L \approx \alpha * \sigma + \alpha^2 + \sigma^2 + a_c * b_c * a_r * b_r$	.9314	.0096

From the third regression no significant terms were discovered, thus implying that the parameter vector was near a local maximum. The final parameter vector was:

Table 4.10: Final Tuning Parameters

α	σ	a_c	b_c	a_r	b_r
1.855	0.1124	0.0091	0.0116	0.0116	0.0094

2 Parameter Tuning

For the sake of simplicity and the ability to visualize results, the parameter vector was reduced to $\Theta = [\alpha, \sigma]$ for fine tuning.

Iteration 1

Initially a 3^2 full factorial was run on the two parameters around the final point from the 6 parameter analysis. This was then fitted as follows:

Table 4.11: Models Tried for First 2 Parameter Regression

Model	R^2	*PRESS*
$L \approx \alpha * \sigma + \alpha^2 + \sigma^2$	.9167	.0004

With no significant terms produced as a result of the regression, the assumption of being near a local extreme point was confirmed. In order to determine the direction of travel for fine tuning the experiment was switched to a 3^2 center circumscribed design [47].

Iteration 2

After regressing the results of this experiment onto the same model, with satisfactory diagnostics, a quadratic equation was constructed from the coefficients and a stationary point was calculated to be located at $(1.8057, 10^{-0.1861})$, with a likelihood ratio of 0.5915. This point is shown relative to the search points below:

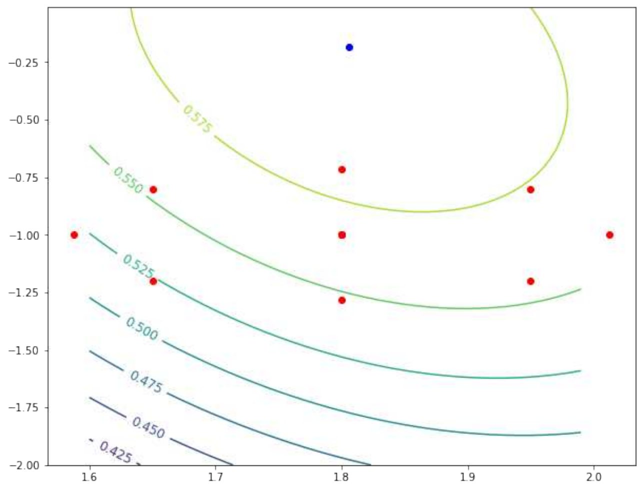

Figure 4.5: Visualization of Second Iteration Search points (Red) and Stationary point (Blue) along Likelihood Ratio Contours

From the figure it was apparent that the difference between the current center-point and the stationary point was much greater in σ than in α. For the third iteration the center-point was moved to the stationary point.

Iteration 3

Repeating the steps from iteration 2, the new stationary point was calculated at $(1.8241, 10^{-0.1833})$, with a likelihood ratio of 0.6216. This point is shown relative to the search points in figure 4.6:

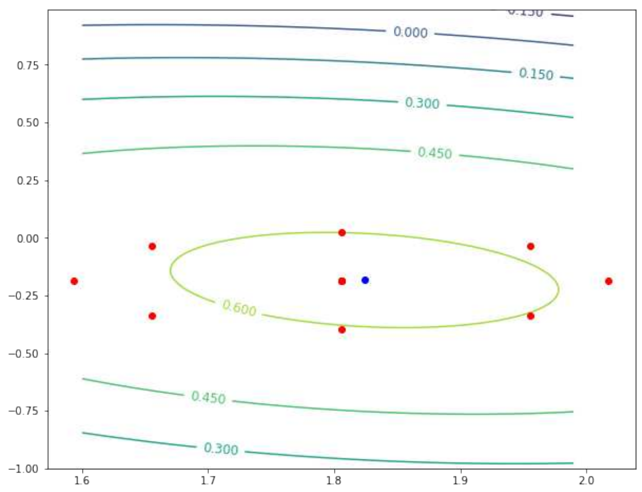

Figure 4.6: Visualization of Third Iteration Search points (Red) and Stationary point (Blue) along Likelihood Ratio Contours

Iteration 4

Once more repeating the steps from the previous iteration, the new stationary point was calculated at $(1.8241, 10^{-0.1833})$, with a likelihood ratio of 0.6216. This point is shown relative to the search points below:

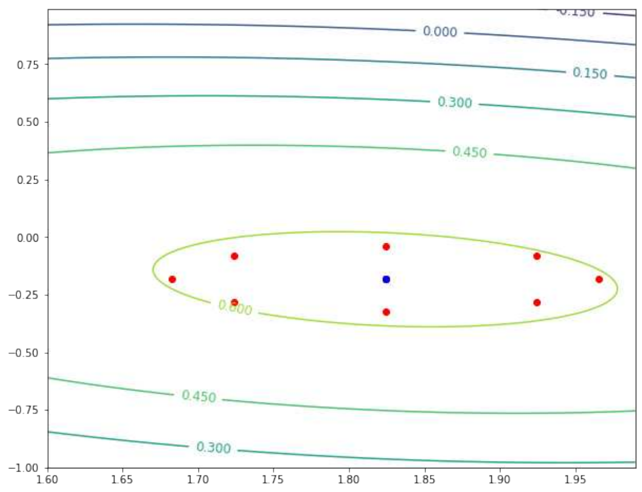

Figure 4.7: Visualization of Fourth Iteration Search points (Red) and Stationary point (Blue) along Likelihood Ratio Contours

Having clearly converged at a maximum, the search was suspended after the fourth iteration.

Conclusions

The final parameter vector chosen was:

Table 4.12: Final Tuning Parameters

α	σ	a_c	b_c	a_r	b_r
1.8241	0.6556	0.0091	0.0116	0.0116	0.0094

Having arrived at the parameters trough, first, a six parameter search, then a 2 parameter search, the experimenter was left with an uneasy feeling based on the results for all parameters other than α. If anything was proven by this tuning it was that the effect of α is so large in the selected region that it dwarfs all other effects; perhaps a second analysis should be carried out with α fixed at its confidently determined optimal value. Another takeaway, which was not confirmed by the above analysis but could be by subsequent analysis, is that the other parameters do not hold much pull on the results in the range selected, This conclusion is counter-intuitive as the likely ranges for these parameters were included in the range of the initial experiment. One possible explanation for this relates to the construction of the filter, because the generic motion model was used, it provides a lot of uncertainty and leads to a high degree of mis-prediction, thus mitigating the effect of measurement uncertainty. Because of the low maximum likelihood ratio, the above explanation is plausible. As a result of the investigation documented in section 4.1.1, a high performing External Object filter was arrived at.

4.1.2 Lead Vehicle Classification

External Object Sensor Fusion, on its own, does not allow for the processing of ADAS data into any of the Data-Treatments mentioned in Chapter 2. In order to produce LV information it is also necessary to estimate the curvature of the Ego vehicle's instantaneous trajectory and to put the world into terms of that trajectory.

Defining the vehicle trajectory requires projecting both the speed of the vehicle and the curvature of the vehicle path out to a point. Path curvature $\kappa = 1/R$ is the inverse of the path radius. Without the use of predictive machine learning or Artificial Intelligence, only constant velocity or acceleration and constant curvature trajectories may be predicted. On a small time scale, however, linearization leads to acceptable results. Only the curvature is needed for lead vehicle identification. In order to estimate the curvature of the trajectory one must derive the vehicle single-track model [38], this is a simplified vehicle dynamics model which treats the front and rear axes of the vehicle as single tires which operate around a Center of Mass (COM). The single track model, often referred to as the "bicycle model", is frequently used in high level vehicle controls as it is computationally simple [48]. Combined with assumptions of constant cornering stiffness and steering rack ratio and neglecting the effects of load transfer, the single track vehicle dynamics model can be reduced to a few equations as follows.

First the variables for the geometry of the vehicle motion are shown below:

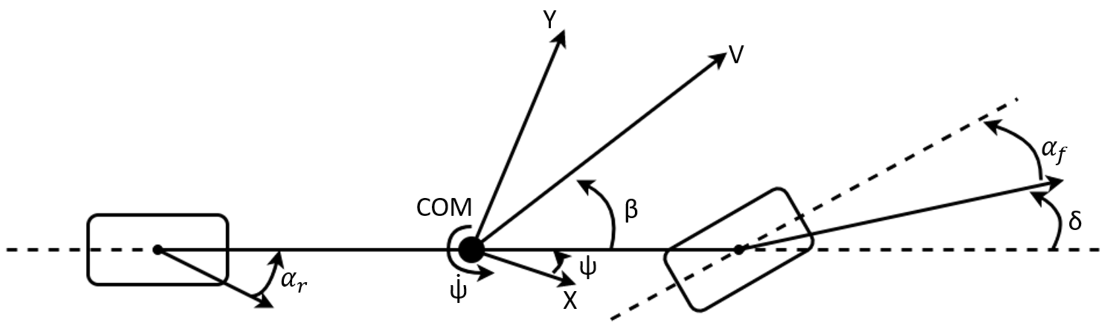

Figure 4.8: Single Track Model Motion Geometry. X and Y define the 2D coordinate system, V is the vehicle velocity vector, δ is the steered angle, α_r and α_f are the rear and front tire slip angles (angle between heading and motion) and β is the vehicle sideslip angle, ψ and $\dot\psi$ are the yaw (heading) angle and yaw rate. Vehicle dynamic coordinate systems have the positive Z axis pointing down so thus positive angles are clockwise

For a single track model the states X, inputs U, and measurements Z are:

$$\begin{aligned} X &= [\dot\psi, \beta]^T \\ U &= [\delta, V]^T \\ Z &= [\dot\psi, A_y]^T \end{aligned} \tag{4.16}$$

Where A_y is the lateral acceleration of the vehicle. Sideslip is a state but as it cannot be measured we must use the relationship between yaw rate and lateral acceleration to calculate it. The equations of motion are given as:

$$\dot X = \begin{bmatrix} -\frac{C_{\alpha,f} l_f^2 \cos\delta + C_{\alpha,r} l_r^2}{I_{zz} V} & \frac{-C_{\alpha,f} l_f \cos\delta + C_{\alpha,r} l_r}{I_{zz}} \\ -1 - \frac{C_{\alpha,f} l_f \cos\delta - C_{\alpha,r} l_r}{mV^2} & -\frac{C_{\alpha,f}\cos\delta + C_{\alpha,r}}{mV} \end{bmatrix} X + \begin{bmatrix} \frac{C_{\alpha,f} l_f \tan\delta}{I_{zz}} \\ \frac{C_{\alpha,f}\sin\delta}{mV} \end{bmatrix} \tag{4.17}$$

$$Z = \begin{bmatrix} \frac{-C_{\alpha,f} l_f \cos\delta + C_{\alpha,r} l_r}{mV} & -\frac{C_{\alpha,f}\cos\delta + C_{\alpha,r}}{m} \end{bmatrix} X + \begin{bmatrix} \frac{C_{\alpha,f}\sin\delta}{m} \end{bmatrix} \tag{4.18}$$

With the vehicle parameters:

$$\Theta = [C_{\alpha,f}, C_{\alpha,r}, l_f, l_r, m, I_{zz}] \tag{4.19}$$

$C_{\alpha,f}$ and $C_{\alpha,r}$ are the cornering stiffnesses of the axles, l_f and l_r are the distances form the axles to the COM, m is the vehicle mass and I_{zz} is the yaw inertia.

In this form the equations are non-linear due to the presence of trigonometric functions. Linearizing the equations using the small angle assumption produces:

$$\dot{X} = \begin{bmatrix} 1 - dt\frac{C_{\alpha,f}l_f^2 + C_{\alpha,r}l_r^2}{I_{zz}V} & dt\frac{-C_{\alpha,f}l_f + C_{\alpha,r}l_r}{I_{zz}} \\ -dt - dt\frac{C_{\alpha,f}l_f - C_{\alpha,r}l_r}{mV^2} & 1 - dt\frac{C_{\alpha,f} + C_{\alpha,r}}{mV} \end{bmatrix} X + \begin{bmatrix} \frac{C_{\alpha,f}l_f}{I_{zz}} \\ \frac{C_{\alpha,f}}{mV} \end{bmatrix} \delta \tag{4.20}$$

$$Z = \begin{bmatrix} 1 & 0 \\ \frac{-C_{\alpha,f}l_f + C_{\alpha,r}l_r}{mV} & -\frac{C_{\alpha,f} + C_{\alpha,r}}{m} \end{bmatrix} X + \begin{bmatrix} \frac{C_{\alpha,f}}{m} \end{bmatrix} \delta \tag{4.21}$$

Where dt is the time step. In the linearized form an EKF can be used where only the UKF can be used for the non-linear form. Given the states of the vehicle, the radius of the path can be calculated as:

$$R(\dot{\psi}, V) = \begin{cases} \frac{V}{\dot{\psi}}, & \dot{\psi} \neq 0 \\ \infty, & \text{otherwise} \end{cases} \tag{4.22}$$

Curvilinear Coordinates

With the radius of curvature of the path calculated one must now put track locations into coordinates which are relative to the trajectory. Curvilinear coordinates [49] are a general representation of coordinates of which Cartesian and polar coordinates are examples. Where in the Cartesian system the base unit vectors $[e_1, e_2, ..., e_n]$ lie along orthogonal, straight axes and are the same length, curvilinear systems are defined by base curves $[\theta_1, \theta_2, ..., \theta_n]$ which are not necessarily orthogonal, straight, or the same length. An example of a curvilinear coordinate system is cylindrical coordinates where $[\theta_1, \theta_2, \theta_3] = [r, \theta, z]$.

The vehicle path coordinates must be able to be transformed from 2D Cartesian coordinates to the SL curvilinear coordinates and back. First we define the path coordinates as the SL coordinate system as shown in [33] chapter 6.

$$\Theta = [s, l] \tag{4.23}$$

where s is distance along the path and l is radial deviation from the path. The relationship between SL and Cartesian coordinates is shown below:

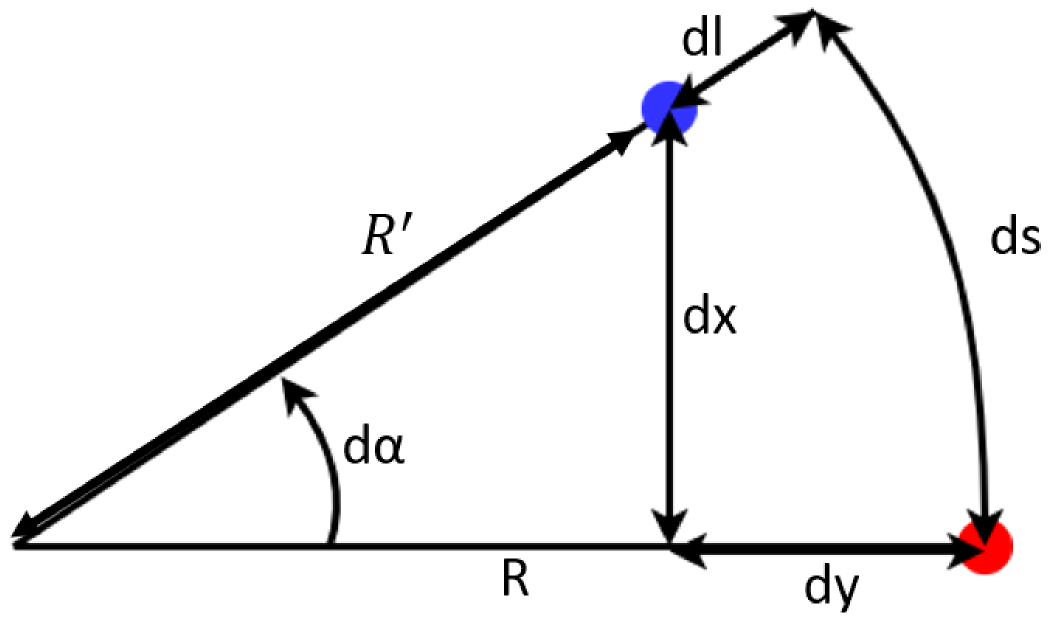

Figure 4.9: SL and 2D Cartesian Coordinates. The distance between two points marked red and blue is (ds, dl) in SL and (dx, dy) in 2D Cartesian

The following observations are made related to the SL system:

$$\begin{aligned} ds &= R d\alpha \\ dl &= R - R^{'} \end{aligned} \tag{4.24}$$

The following observations are made related to the Cartesian system:

$$\begin{aligned} dx &= R^{'} \sin\alpha + d\alpha - R \sin d\alpha \\ dy &= R^{'} \cos\alpha + d\alpha - R \cos d\alpha \end{aligned} \tag{4.25}$$

Since the start point is the ego vehicle (red rod in figure 4.9), $\alpha = 0$, thus:

$$\begin{aligned} dx &= R^{'} \sin d\alpha = (R - dl) \sin \frac{ds}{R} \\ dy &= R^{'} \cos d\alpha - R = (R - dl) \cos \frac{ds}{R} - R \end{aligned} \tag{4.26}$$

With dx and dy in terms of ds and dl, The SL coordinates can now be solved for. The final transformations can be written as:

$$E = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} = F_E(\Theta, R) = \begin{bmatrix} (R - l) \sin \frac{s}{R} \\ (R - l) \cos \frac{s}{R} - R \end{bmatrix} \tag{4.27}$$

and

$$\Theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} s \\ l \end{bmatrix} = F_\Theta(E) = \begin{bmatrix} \arctan\left(\frac{x}{y+R}\right) R \\ R - \frac{y+R}{\cos\left(\arctan\left(\frac{dx}{dy+R}\right)\right)} \end{bmatrix} \tag{4.28}$$

Cost Function

Thus, having the radius of curvature calculated and having all tracks in SL coordinates, The tracks can be assigned a cost based on their location relative to the vehicle path. This cost function should heavily penalize vehicles for being outside of a reasonable radial band around the trajectory, the width of this band will be represented as w_b. The cost function should also penalize radial deviation in the band and select the closest vehicle along the path. for this reason the following cost function is proposed:

$$COST(s_i, l_i) = \begin{cases} s_i^a + l_i^b, & |l_i| \leq w_b \\ \infty, & \text{otherwise} \end{cases} \tag{4.29}$$

Where $b >> a$. As there can only ever be one lead vehicle, there is only one possible permutation and thus the lead vehicle is the active track with the lowest cost.

Trajectory Estimation Kalman Filter Tuning

In order to enable accurate lead vehicle identification, a designed experiment was performed on tuning parameters for filters for both the linear and non-linear single-track models developed earlier. The parameter vector for these models is shown below:

$$\Theta_m = [C_{\alpha,f}, C_{\alpha,r}, l_f, l_r, m, I_{zz}] \tag{4.30}$$

Where, thanks to sponsor provided confidential information, only the cornering stiffnesses are unknown. The tuning parameters for the filters are shown below:

$$\Theta_f = [\alpha, \sigma, a, b] \tag{4.31}$$

Where a is the uncertainty for yaw rate measurement and b is the uncertainty for lateral acceleration measurement. That means that the combined parameter vector for the experiment was:

$$\Theta_f = [\alpha, \sigma, a, b, C_{\alpha,f}, C_{\alpha,r}] \tag{4.32}$$

The method used for tuning was the same as that described in 4.1.1 and, for the sake of brevity, will not be recounted. The final parameter vector arrived at was.

Table 4.13: Tuning Parameter Levels for Regression

α	σ	a_1	b_1	a_2	b_2
1.4584	0.9552	0.0045	0.0045	31584	30422

A comparison of the yaw rate tracking and measurements, as well as the lead vehicle identification it allows for are shown below.

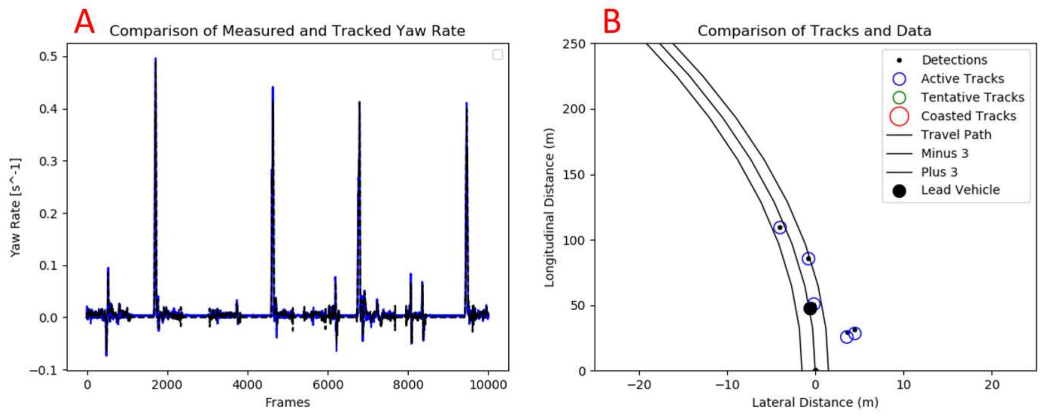

Figure 4.10: A) A comparison of measured and tracked yaw rate, B) Lead vehicle is selected from a series of tracks based on SL distance cost

4.2 Localization

In order to produce HS, SS, and SPaT information, or really to make any use of ITS Data-Streams it is necessary to accurately place a vehicle on a map and to determine where that vehicle is going. With the knowledge of what road and lane a vehicle currently occupies and where it is heading SS can be generated by associating the vehicle with a monitored speed segment, SPaT can be generated by associating a vehicle with a traffic signal, and HS can be generated by comparing recorded speeds at the location the vehicle occupies.

Localization is a quite difficult problem and the difficulty scales with the required precision for a certain application. Often, localization relies on HD maps [33, 50] which are not available for this case. For the purposes of determining HS, SS, and SPaT however, lower definition maps can be used with the aid of precision GNSS position information and vehicle path information calculated as described in 4.1.2. For this thesis Open Street Map [51] data was used.

The process of data association for SPaT and SS was as follows:

- Pull road points from map within a 50 meter radius

- Using ego GNSS lateral/longitudinal position and estimated path curvature, first convert all points into ego Cartesian coordinates and then into ego SL coordinates.
- Select the closest point (in ego SL coordinates) in front of the ego vehicle as the next ego vehicle location.
- Using current and next location of the ego vehicle determine the current road and lane of the ego vehicle and assign corresponding SPaT and SS data. Current location determines the road the vehicle is on where next location determines the direction of travel.

For HS data, in the absence of information from other vehicles, the speeds of the ego vehicle at a given point along the data collection route for all laps save the current lap, was used as a proxy.

Chapter 5

Velocity Prediction

Predictive Optimal Energy Management Systems (POEMS), as defined in Chapter 1 are smart systems which use knowledge of future vehicle motion in order to command optimal powertrain behavior. As opposed to Eco Driving systems which guide the vehicle along a minimal energy path, POEMS does not influence the motion of the vehicle, rather finding an optimal powertrain management solution for the given path to be traveled. The obvious contingency is that POEMS requires knowledge of future motion to operate, thus necessitating accurate prediction of future vehicle motion.

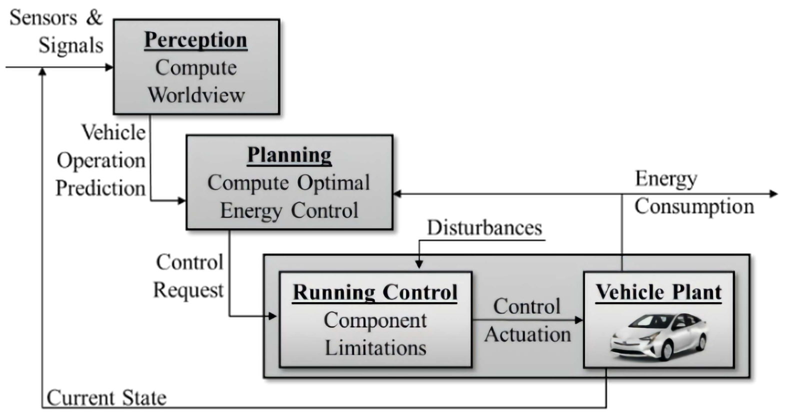

Figure 5.1: The system-level viewpoint of predictive optimal energy management, enabling POEMS requires optimization of signals, perception, and prediction

The requirement for high fidelity velocity prediction comes from the nature of the optimal energy problem. Generally the optimal energy problem is solved recursively from a set point in the future. Constraints to the energy optimization problems can be obtained using the prediction of future driving conditions. Hence, the fidelity of the predicted vehicle velocity directly impacts the effectiveness of the controls in terms of both efficiency and safety. Due to the dependence of POEMS on prediction horizon and fidelity there is a pressing need to develop accurate and robust approaches for predicting vehicle speed and understanding which input signals and prediction strategies enable the most accurate prediction.

Artificial Neural Networks (ANN) are a natural choice for such a task as they are capable of modeling and extracting unseen characteristics and properties which allow for reliable and simple solutions to sophisticated modeling problems [52]. With the use of ANNs, high fidelity predictions of the future outputs of systems which enable the use of optimal control principles are achievable. Many researchers have recognized the potential of ICV technology-derived data streams for use in ANN predictive models whose outputs can be used to derive an FE optimal control strategy [53, 54, 18, 17, 55, 56, 57, 58, 25].

5.1 General ANN Structure

ANNs are a machine learning technique which is composed of artificial neurons and attempts to mimic the structure of the brain. ANNs are an increasingly common and well understood solution to a wide variety of data analysis problems not only in hard sciences and engineering, but also in the soft sciences and arts due to their ability to accurately predict outputs from complex systems which would otherwise have to be modeled from constituents [59]. The most basic form of an ANN is a shallow, feed-forward network consisting of a layer of input neurons, a hidden layer of fully connected neurons, and a fully connected layer of output neurons. The neurons themselves contain a set of weights for their inputs and an activation (gating) function, often a sigmoid or hyperbolic tangent, which translates the weighted sum of the inputs into an output value. The weights and activation function parameters are tuned through back-propagation using training data.

Recurrent ANNs are able to display ”memory” where the state of the hidden layers at time-step t becomes an input to the hidden layers at time-step $t+1$, thus allowing the network to continue to learn after training. Deep ANNs include multiple hidden layers which allow the networks to understand more complicated behavior without over-fitting at the cost of increased training time.

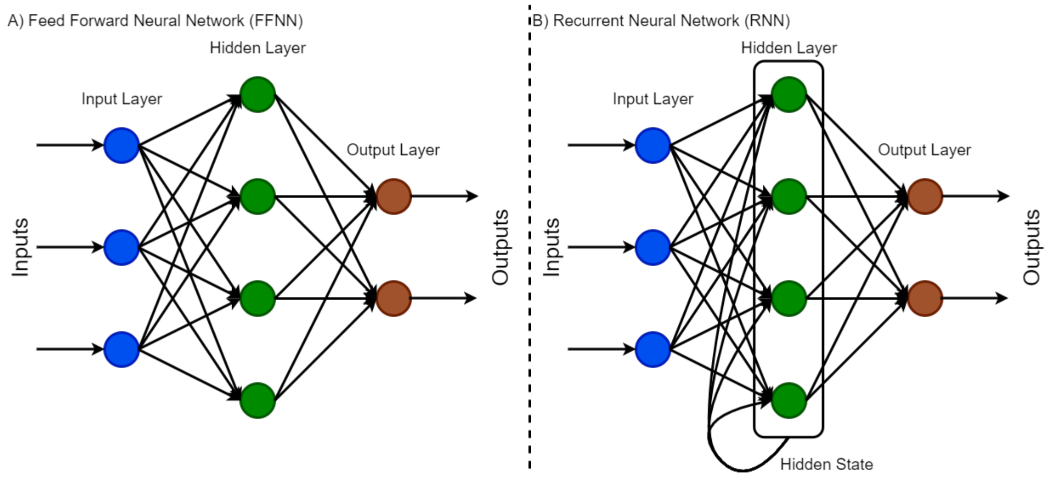

Figure 5.2: Schematic of FFNN and RNN [2]

The Long Short-Term Memory (LSTM) nework is a special kind of recurrent neural network (RNN) designed by

Hochreiter and Schmidhuber [60] which uses at least one hidden layer of specialized LSTM neurons. LSTM neurons differ from traditional recurrent neurons in that it is capable of recurrent behavior within each neuron.

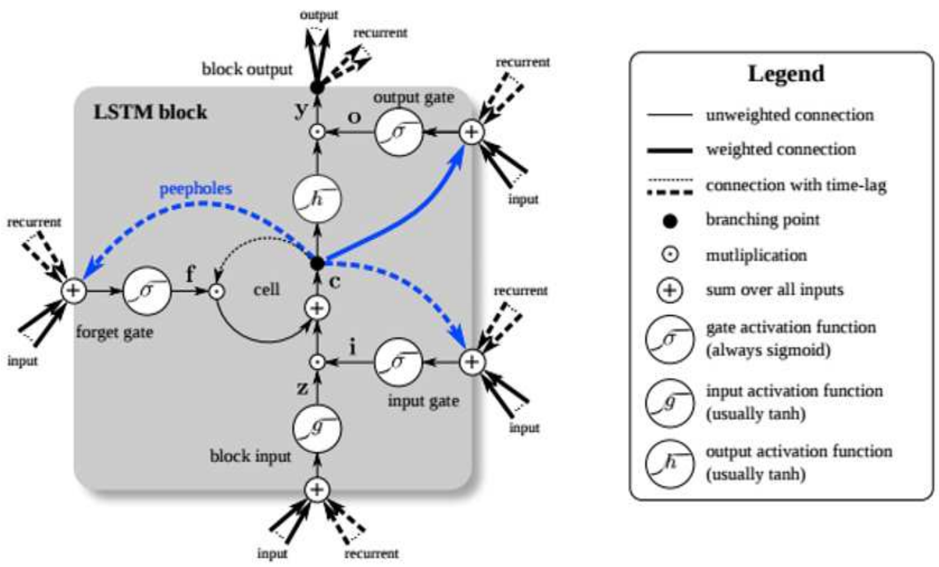

Figure 5.3: Schematic of LSTM Neuron from [3]

In comparison to a conventional recurrent neuron which uses a single gate to translate weighted inputs into an output, the LSTM neuron contains a series of gates which determine what input information is remembered for future steps, what stored information is forgotten, and the ratio of new and remembered information used to compute the output. Thus, where the outputs of a conventional RNN are only affected by the hidden state from the previous time step, an LSTM network has the additional capability to remember information for multiple time steps.[2, 61, 62]. LSTM networks prove especially useful in applications wherein the relationship between inputs and outputs are delayed [3].

Final ANN Design

The nature of the velocity prediction problem includes a series of specific constraints. The reaction of the ego vehicle driver to inputs is necessarily slower than the rate at which new data is acquired. inputs such as SPaT and LV can change rapidly, and driver response to inputs is not repeatable. Bbased on previous work [61, 63], research was focused on LSTM RNNs and CNNs. Finally, a deep network containing both LSTM and convolutional layers was also considered. All ANNs were implemented in Python using Keras libraries [64].

A broad study of various machine learning methods conducted in [26] using the data collected for this thesis revealed the optimal structure for ICV vehicle velocity prediction to be a deep LSTM ANN of 64, 32, and 12 LSTM layers, 10% dropout after the first LSTM layer, and a fully connected layer at the end.

5.2 Effects of Data Categories

Having determined the optimal construction of an ANN to predict vehicle velocity, the question naturally arises as to which Data Treatments are most important, and thus worth focusing on for future development. In order to determine which groups would perform best, a $3 * 2^4$ full factorial experiment was run on combinations of EGO data with all combinations of HS, LV, SPaT, and SS for 10, 20, and 30 second prediction horizons. For this experiment a best fit, in terms of Mean Absolute Error (MAE) was found for each Data Treatment group allowing for small changes to the network and fitting to convergence; raw results of this experiment are shown in Figure 5.4.

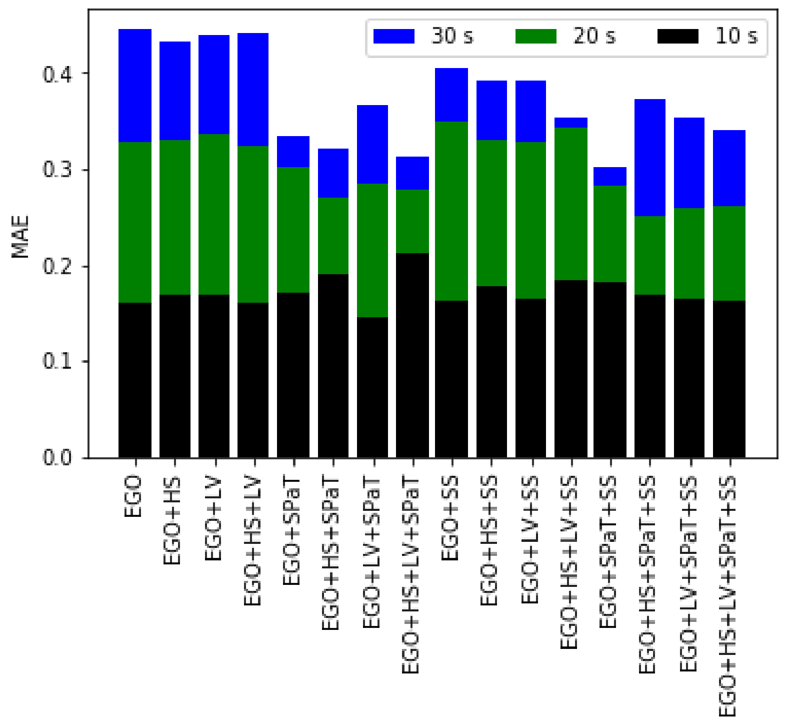

Figure 5.4: Performance of groups of signals with best network

A few takeaways of note from the raw results are the following:

- Prediction fidelity decays with extended prediction horizon, this is to be expected but it holds system level implications for POEMS. The Planning algorithm will be more effective with a longer scale prediction but the quality of that prediction will decrease.
- the distinction between the best data groups in the 10 second horizon level was minimal

A regression analysis on the Data Treatments revealed the following:

Table 5.1: Data group regression results

Factor	β	$stderr$	t	$P > \|t\|$	2.5th	97.5th
Intercept	0.1077	0.015	7.201	0.000	0.077	0.138
horizon	0.0102	0.001	18.494	0.000	0.009	0.011
HS	-0.0016	0.009	-0.180	0.858	-0.020	0.017
LV	-0.0021	0.009	-0.237	0.814	-0.020	0.016
SPaT	-0.0429	0.009	-4.757	0.000	-0.061	-0.025
SS	-0.0098	0.009	-1.090	0.282	-0.028	0.008

Results show significance in only one of the individual Data Streams, SPaT, which had a negative coefficient of $-.0429$ meaning that the presence of SPaT data was shown to be important in reducing error. Significant results were not seen in any of the other individual products. Additionally, no cross products were shown to be significant. The group that performed well enough on all levels of prediction horizon to be considered the best group, from visual analysis, was EGO+LV+SPaT, with this group performing best at the 10 second prediction horizon level as is visible in Figure 5.4. A conclusion from this evidence is that traffic signal information is more important for velocity prediction than other Data Streams.

Questions may exist as to whether this matters from an overall POEMS standpoint. As figure 5.5 below, shows, the difference between the best group at a 10 second horizon and the group which only included EGO data is not very noticeable to the naked eye.

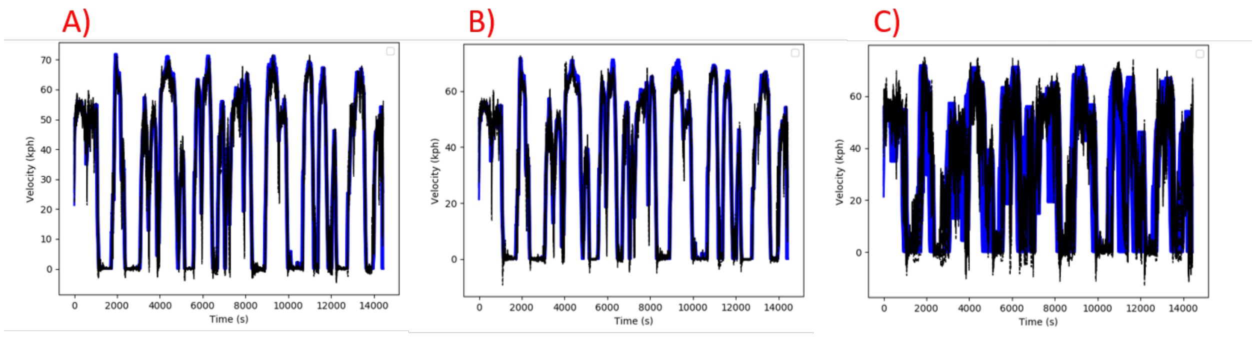

Figure 5.5: Comparison of recorded and predicted velocity traces for A) Best group at 10 seconds B) EGO only group at 10 seconds C) Best group at 30 seconds

Other takeaways from Figure 5.5 are that performance of the network was not dramatically reduced after being tested on a different driver from the one it was trained on (MAE rose from .1447 to .1884) and that the difference between 10 and 30 second prediction horizons was qualitatively different. The most harmful thing for POEMS planning algorithms are mis-predictions which are significantly wrong in the magnitude or direction of acceleration [22] with which the 30 second horizon prediction is replete, meaning that the 30 second prediction may be far less usable than the 10 second prediction; more research is needed on this topic.

5.3 Prediction Algorithm Portability

A question that must also be answered is to what degree is the prediction algorithm portable. In order to evaluate this property of the prediction algorithm, data was collected with two different drivers as explained in Chapter 3, then the best ANN was evaluated for portability between the data-sets. As LSTM ANNs have recurrent behavior, the network would adjust to the new driver given enough time, thus the evaluation of portability was conducted over only three validation drive-cycles (12 miles).

The first experiment was to compare the performance of the network trained on driver 1 and tested on driver 1, trained on driver 1 and tested on driver 2, and trained on driver 2 and tested on driver 1. The results of this experiment are shown in Figure 5.6 and table 5.2.

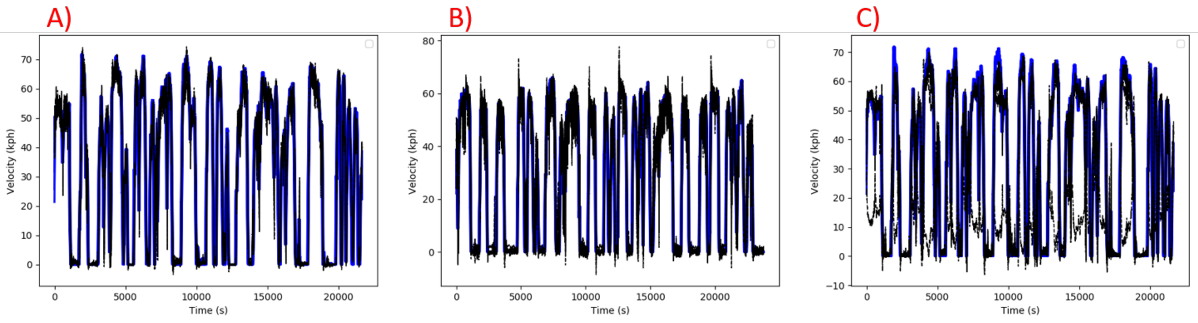

Figure 5.6: Initial portability assessment A) trained on driver 1 and tested on driver 1 B) trained on driver 1 and tested on driver 2 C) trained on driver 2 and tested on driver 1

Table 5.2: Initial portability assessment MAE

Training Set	Testing Set	MAE
Driver 1	Driver 1	.1482
Driver 1	Driver 2	.1820
Driver 2	Driver 1	.2952

The results showed that the algorithm cannot be considered portable from one driver to the next immediately. However, in practice the network would be trained on a much greater data-set with data from many drivers before being deployed. The algorithm was also trained using combined data from both drivers and then tested on each driver's data with results shown in figure 5.7 and Table 5.3.

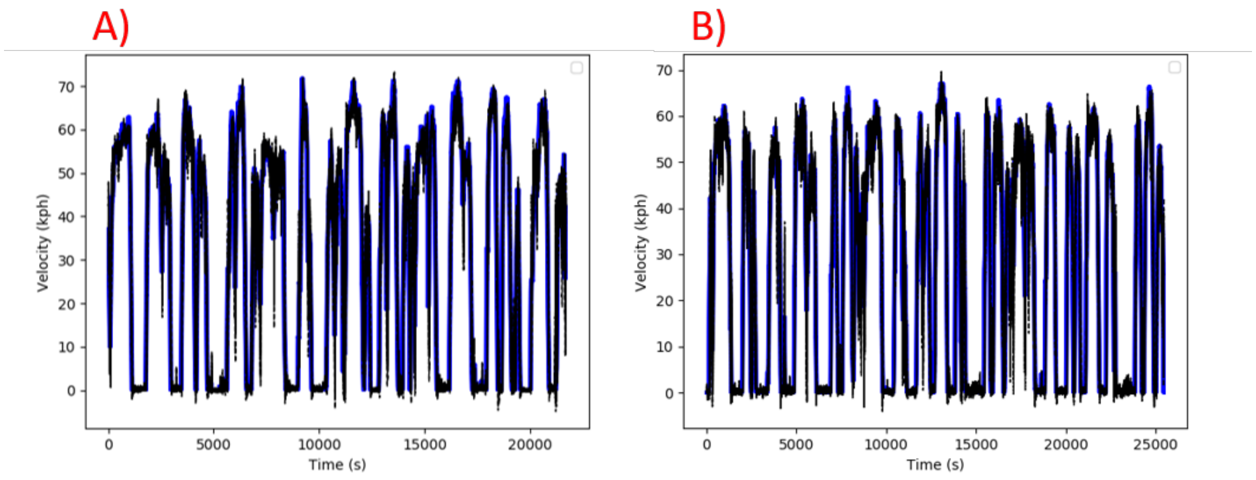

Figure 5.7: Combined portability assessment A) trained on both drivers and tested on driver 1 B) trained on both drivers and tested on driver 2

Table 5.3: Combined portability assessment MAE

Training Set	Testing Set	MAE
Both	Driver 1	.1537
Both	Driver 2	.1537

The results show that, when trained on a combined data-set, the network was able to predict each driver equally well. Due to the nature of LSTM ANNs, the network would eventually become more driver specific, but a combined data-set would be an adequate starting point.

Chapter 6

Conclusions

In order to enable the implementation of Predictive Optimal EMS, this thesis has sought to define which data streams are necessary to enable high fidelity,long horizon predictions. This work extends research previously performed at Colorado State University in this domain. In this thesis a classification of available types of data for ICVs is presented, and a data-set was gathered representing data that would be available to a generic ICV. A broad study of predictive methods was conducted arriving at an optimal solution in the form of a deep LSTM ANN, an experiment was run on various combinations of data to determine which was the most important and what was the optimal data-set, and finally the portability of said predictive network between drivers was tested.

The scope of this study was limited to only the data collection and prediction steps, and in order to provide a complete model of the efficacy of POEMS algorithms, more research on optimization, path planning, and actuation would be required. Despite that caveat, a few takeaways from this study can be noted expliticly:

1. The power of the LSTM ANN to predict vehicle velocity robustly and with low error.

2. More data available for use with the ANN algorithm is not necessarily better, in fact, certain ITS data combinations performed worse than vehicle CAN bus data alone.

3. By far, the greatest impact on vehicle velocity trace in an urban environment is from traffic signals. As urban driving is dictated by speed limits and frequent signals it makes sense that the SPaT information was the most important. HS, SS, and LV data merely describe the speeds of vehicles in the network whereas SPaT dictates those speeds. In the future, automotive manufacturers and municipalities should focus on the transmission of SPaT data or some proxy over other sources of data.

4. Although drivers 1 and 2 were deliberately selected because they have very different driving styles, an ANN algorithm trained on a combined data-set from both was able to predict velocity from either at an equal and acceptable level.

Having developed an optimal process of data capture, processing, and velocity prediction, more research may now be performed into the optimal use of said predictions in energy management, bringing the mobility industry one step closer to the practical implementation of PEOMS.

References

[1] A. I. Rabinowitz, T. Gaikwad, S. White, T. Bradley, and Z. Asher, “Infrastructure data streams for automotive machine learning algorithms research,” tech. rep., SAE Technical Paper, 2020.

[2] C. Olah, “Understanding lstm networks,” 2015.

[3] C. Nicholson, “A beginner’s guide to lstms and recurrent neural networks.” `https://pathmind.com/wiki/lstm`.

[4] B. M. Al-Alawi and T. H. Bradley, “Review of hybrid, plug-in hybrid, and electric vehicle market modeling studies,” *Renewable and Sustainable Energy Reviews*, vol. 21, no. C, pp. 190–203, 2013.

[5] I. E. Agency and International Energy Agency, “CO2 emissions from fuel combustion 2016,” 2016.

[6] World Health Organization, *World Health Statistics 2016: Monitoring Health for the SDGs Sustainable Development Goals*. World Health Organization, June 2016.

[7] S. Smog, “Other air pollution from transportation,” *US Environmental Protection Agency*, 2015.

[8] A. E. Atabani, I. A. Badruddin, S. Mekhilef, and A. S. Silitonga, “A review on global fuel economy standards, labels and technologies in the transportation sector,” *Renewable Sustainable Energy Rev.*, vol. 15, pp. 4586–4610, Dec. 2011.

[9] C. M. Martinez, X. Hu, D. Cao, E. Velenis, B. Gao, and M. Wellers, “Energy management in plug-in hybrid electric vehicles: Recent progress and a connected vehicles perspective,” *IEEE Trans. Veh. Technol.*, vol. 66, pp. 4534–4549, June 2017.

[10] S. Uebel, N. Murgovski, C. Tempelhahn, and B. Bäker, “Optimal energy management and velocity control of hybrid electric vehicles,” *IEEE Trans. Veh. Technol.*, vol. 67, pp. 327–337, Jan. 2018.

[11] R. Rajamani, “Vehicle dynamics and control: Springer science & business media,” 2011.

[12] Z. D. Asher, A. A. Patil, V. T. Wifvat, A. A. Frank, S. Samuelsen, and T. H. Bradley, “Identification and review of the research gaps preventing a realization of optimal energy management strategies in vehicles,” *SAE International Journal of Alternative Powertrains*, vol. 8, no. 2, 2019.

[13] Y. Luo, T. Chen, and K. Li, “Multi-objective decoupling algorithm for active distance control of intelligent hybrid electric vehicle,” *Mech. Syst. Signal Process.*, vol. 64-65, pp. 29–45, Dec. 2015.

[14] Y. Luo, T. Chen, S. Zhang, and K. Li, “Intelligent hybrid electric vehicle ACC with coordinated control of tracking ability, fuel economy, and ride comfort,” *IEEE Trans. Intell. Transp. Syst.*, vol. 16, pp. 2303–2308, Aug. 2015.

[15] Q. Wang and B. Ayalew, “A probabilistic framework for tracking the formation and evolution of Multi-Vehicle groups in public traffic in the presence of observation uncertainties,” *IEEE Trans. Intell. Transp. Syst.*, vol. 19, pp. 560–571, Feb. 2018.

[16] Q. Gong, Y. Li, and Z.-R. Peng, “Trip-based optimal power management of plug-in hybrid electric vehicles,” *IEEE Transactions on Vehicular Technology*, vol. 57, no. 6, p. 3393–3401, 2008.

[17] C. Sun, X. Hu, S. J. Moura, and F. Sun, “Velocity predictors for predictive energy management in hybrid electric vehicles,” *IEEE Trans. Control Syst. Technol.*, vol. 23, pp. 1197–1204, May 2015.

[18] C. Sun, S. J. Moura, X. Hu, J. K. Hedrick, and F. Sun, “Dynamic traffic feedback data enabled energy management in plug-in hybrid electric vehicles,” *IEEE Trans. Control Syst. Technol.*, vol. 23, pp. 1075–1086, May 2015.

[19] Z. D. Asher, J. A. Tunnell, D. A. Baker, R. J. Fitzgerald, and others, “Enabling prediction for optimal fuel economy vehicle control,” 2018.

[20] D. Baker, Z. Asher, and T. Bradley, “Investigation of vehicle speed prediction from neural network fit of real world driving data for improved engine on/off control of the ecocar3 hybrid camaro,” *SAE Technical Paper Series*, 2017.

[21] D. Baker, Z. D. Asher, and T. Bradley, “V2v communication based real-world velocity predictions for improved hev fuel economy,” *SAE Technical Paper Series*, 2018.

[22] J. A. Tunnell, Z. D. Asher, S. Pasricha, and T. H. Bradley, “Towards improving vehicle fuel economy with adas,” *SAE Technical Paper Series*, 2018.

[23] X. Qi, Y. Luo, G. Wu, K. Boriboonsomsin, and M. Barth, “Deep reinforcement learning enabled self-learning control for energy efficient driving,” *Transportation Research Part C: Emerging Technologies*, vol. 99, p. 67–81, 2019.

[24] K. Liu, Z. Asher, X. Gong, M. Huang, and I. Kolmanovsky, “Vehicle velocity prediction and energy management strategy part 1: Deterministic and stochastic vehicle velocity prediction using machine learning,” 2019.

[25] T. D. Gaikwad, Z. D. Asher, K. Liu, M. Huang, and I. Kolmanovsky, "Vehicle velocity prediction and energy management strategy part 2: Integration of machine learning vehicle velocity prediction with optimal energy management to improve fuel economy," tech. rep., SAE Technical Paper, 2019.

[26] T. Gaikwad, A. Rabinowitz, F. Motallebiaraghi, T. Bradley, Z. Asher, L. Hanson, and A. Fong, "Vehicle velocity prediction using artificial neural network and effect of real world signals on prediction window," tech. rep., SAE Technical Paper, 2020.

[27] "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," 2018.

[28] Y. K. Ryosuke Okuda and K. Terashima, "A survey of technical trend of adas and autonomous driving," 2014.

[29] E. Z. Paolo Bellavista, Luca Foschini, "V2x protocols for low-penetration-rate and cooperative traffic estimations," 2014.

[30] J. M. P. Gaurang Naik, Biplav Choudhury, "Ieee 802.11bd 5g nr v2x: Evolution of radioaccess technologies for v2x communications," 2019.

[31] "Mobileye 8 connect - collision avoidance system."

[32] "Mid-range radar sensor (mrr rear)."

[33] S. Liu, L. Li, J. Tang, S. Wu, and J. Gaudiot, *Creating Autonomous Vehicle Systems*. 2017.

[34] *Bayesian Statistics*. `https://en.wikipedia.org/wiki/Bayesianstatistics`.

[35] R. Labbe, *Kalman and Bayesian Filters in Python*, 2019. `https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python`.

[36] B.-S. Y. and F. T.E., "Tracking and data association," *Naval Research Logistics Quarterly*, 1955.

[37] R. J. Meinhold and N. D. Singpurwalla, "Understanding the kalman filter," *The American Statistician*, vol. 37, no. 2, p. 123–127, 1983.

[38] C. Lundquist. PhD thesis, Liu-Tryck, 0AD.

[39] E. A. Wan and R. Van Der Merwe, "The unscented kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pp. 153–158, 2000.

[40] *Colors of Noise*. `https://en.wikipedia.org/wiki/Colorsofnoise`.

[41] A. Ziebinski, R. Cupek, H. Erdogan, and S. Waechter, "A survey of adas technologies for the future perspective of sensor fusion," *Computational Collective Intelligence Lecture Notes in Computer Science*, p. 135–146, 2016.

[42] H. W. Kuhn, "The hungarian method for the assignment problem," *Mathematics in science and engineering*, 1955.

[43] *scipy.optimize.linear sum assignment*, 2016. `https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.optimize.linear_sum_assignment.html`.

[44] *Steps of the Hungarian Algorithm*. `http://www.hungarianalgorithm.com/hungarianalgorithm.php`.

[45] *scipy.stats.multivariate normal*, 2014. `https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.stats.multivariate_normal.html`.

[46] S. Seabold and J. Perktold, "statsmodels: Econometric and statistical modeling with python," in *9th Python in Science Conference*, 2010.

[47] "fractional factorial design specifications and design resolution." `https://www.itl.nist.gov/div898/handbook/pri/section3/pri3344.htm`, journal=5.3.3.4.4. Fractional factorial design specifications and design resolution.

[48] L. Menhour, D. Lechner, and A. Charara, "Steering control based on a two-level driver model: experimental validation and robustness tests," in *2009 IEEE Control Applications, (CCA) Intelligent Control, (ISIC)*, pp. 125–130, 2009.

[49] "9. curvilinear coordinates," *Mechanics*, p. 213–250, Jun 2018.

[50] `https://www.here.com/`.

[51] *Main Page*. `https://wiki.openstreetmap.org/wiki/Main_Page`.

[52] H. B. Demuth, M. H. Beale, O. De Jess, and M. T. Hagan, *Neural Network Design*. USA: Martin Hagan, 2nd ed., 2014.

[53] F. A. Bender, M. Kaszynski, and O. Sawodny, "Drive cycle prediction and energy management optimization for hybrid hydraulic vehicles," *IEEE Trans. Veh. Technol.*, vol. 62, pp. 3581–3592, Oct. 2013.

[54] T. Donateo, D. Pacella, and D. Laforgia, "Development of an energy management strategy for plug-in series hybrid electric vehicle based on the prediction of the future driving cycles by ICT technologies and optimized maps," 2011.

[55] Q. Gong, Y. Li, and Z. Peng, "Power management of plug-in hybrid electric vehicles using neural network based trip modeling," in *2009 American Control Conference*, pp. 4601–4606, June 2009.

[56] A. Rezaei and J. B. Burl, "Prediction of vehicle velocity for model predictive control," *IFAC-PapersOnLine*, vol. 48, pp. 257–262, Jan. 2015.

[57] Q. Gong, Y. Li, and Z. Peng, "Trip-Based optimal power management of plug-in hybrid electric vehicles," *IEEE Trans. Veh. Technol.*, vol. 57, pp. 3393–3401, Nov. 2008.

[58] M. A. Mohd Zulkefli, J. Zheng, Z. Sun, and H. X. Liu, "Hybrid powertrain optimization with trajectory prediction based on inter-vehicle-communication and vehicle-infrastructure-integration," *Transp. Res. Part C: Emerg. Technol.*, vol. 45, pp. 41–63, Aug. 2014.

[59] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, 2018.

[60] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, pp. 1735–1780, Nov. 1997.

[61] C.-J. Huang and P.-H. Kuo, "A deep CNN-LSTM model for particulate matter (PM2.5) forecasting in smart cities," *Sensors*, vol. 18, July 2018.

[62] O. B. Sezer, "LSTM_RNN_Tutorials_with_Demo."

[63] S. Bai, J. Zico Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," Mar. 2018.

[64] Keras, "Keras documentation: Keras api reference." `https://keras.io/api/`.