Université de Toulouse

THÈSE

EN VUE DE L'OBTENTION DU

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)

Présentée et soutenue le *25/01/2024* **par :**

Thinh HOANG DINH

Techniques de Traitement des Données pour Les Trajectoires des Véhicules Routiers et Aériens

Réduction de dimensionnalité, détection d'anomalies, filtrage et compression de données avec des applications dans la perception collective V2X

JURY

Francisco Soria CHINESTA	ENSAM Paris	Rapporteur
Jérôme HÄRRI	EURECOM	Rapporteur
Florence NICOL	ENAC	Examinatrice
Sameer ALAM	NTU Singapore	Examinateur
Daniel DELAHAYE	ENAC	Directeur de Thèse
Pierre MARÉCHAL	Université Paul Sabatier	Co-directeur de Thèse

École doctorale et spécialité :
MITT : Domaine Mathématiques : Mathématiques appliquées

Unité de Recherche :
École Nationale de l'Aviation Civile - Équipe OPTIM

Directeurs de Thèse :
Daniel DELAHAYE *et* Pierre MARÉCHAL

Rapporteurs :
Francisco Soria CHINESTA *et* Jérôme HÄRRI

Université de Toulouse

Data Processing Techniques for Trajectories of Road and Air Vehicles

Dimensionality Reduction, Anomaly Detection, Filtering and Data Compression with Applications in V2X Collective Perception

Presented by
HOANG DINH Thinh

Under supervision of
Pr. Daniel DELAHAYE (ENAC)
Pr. Pierre MARÉCHAL (IMT Toulouse)

Submitted to the Doctoral School of Mathematics, Informatics, Telecommunications of Toulouse (EDMITT) in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Applied Mathematics at Paul Sabatier University.

Toulouse, France
June, 2023

Contents

Résumé

Dans le domaine des systèmes de transport intelligents (ITS), le traitement des données de trajectoire est essentiel mais difficile, car les trajectoires routières et aériennes sont souvent de grande dimension. Cette complexité peut entraver les tâches d'inférence telles que la prédiction, la compression, le filtrage et la classification. Pour répondre à cette problématique, cette thèse est structurée en deux parties qui explorent différentes méthodes d'apprentissage automatique statistique et non supervisé appliquées aux données de trajectoires. Les applications couvertes incluent la détection d'anomalies dans les données en continu, la compression en temps réel, ainsi que le lissage et le filtrage des trajectoires des véhicules routiers. Ces techniques ont été utilisées dans des contextes tels que la compression de données dans les réseaux Vehicle-to-Everything (V2X), la reconnaissance de comportements de conduite imprudents et l'extraction de trajectoires de véhicules à partir de vidéos de trafic.

La première partie de la thèse présente un modèle de signal nouvellement proposé, appelé modèle d'impulsions aléatoires (RXID), pour représenter les trajectoires routières. Des analyses comparatives des noyaux de covariance et de leurs vecteurs propres correspondants ont été effectuées pour valider le modèle par rapport à l'ensemble de données NGSIM 101. Ce modèle a été utilisé dans trois applications différentes, à savoir deux algorithmes de compression de données conçus pour atténuer la fréquence des transmissions de véhicule à véhicule (V2V) et un modèle dynamique pour le suivi et le filtrage multi-objets de véhicules routiers à partir d'un flux vidéo. .

Bien que la détection d'anomalies soit une discipline qui a atteint un certain niveau de maturité, elle n'a pas été développée de manière approfondie dans le cadre de la détection d'anomalies dans les flux de données. La suite de cette thèse est consacrée à la conception d'un algorithme de détection d'anomalies en temps réel pour de tels paramètres. Cet algorithme est basé sur deux composants clés : la réduction de la dimensionnalité et le test d'hypothèses en ligne. Une application concernant la reconnaissance d'un comportement de conduite anormal est présentée à titre d'exemple, suivie d'une comparaison avec deux méthodes courantes pour démontrer la valeur de l'approche.

La deuxième partie de cette thèse présente une technique de vision par ordinateur (CV) pour comparer le développement temporel de deux orages dans des périodes historiques distinctes, résultant en un système d'interrogation météorologique

qui permet de rechercher des jours météorologiques similaires. Il est également démontré que la réduction de la dimensionnalité s'avère bénéfique pour visualiser et comparer les schémas d'arrivée basés sur les données de trajectoire de l'aéroport de Dallas Fort Worth (DFW). La méthodologie proposée présente un potentiel prometteur non seulement pour la visualisation des données, mais également pour le développement de méthodes pouvant apprendre des données open source concernant les décisions des contrôleurs aériens. Des sujets supplémentaires, tels que la segmentation non supervisée des séries temporelles, sont présentés, ainsi qu'une extension subtile du cadre probabiliste introduit dans la première partie pour reconnaître les approches non conformes, précurseurs cruciaux des atterrissages non stabilisés.

La thèse se termine par un dernier chapitre résumant les principales contributions et offrant des perspectives sur les travaux futurs.

Mots Clés: *modélisation de trajectoire, filtrage, lissage, suivi, compression de données, réduction de dimensionnalité.*

Abstract

In the field of Intelligent Transportation Systems (ITS), trajectory data processing is essential yet challenging, as road and aerial trajectories are often high-dimensional. This complexity can hinder inference tasks such as prediction, compression, filtering, and classification. To address this issue, this dissertation is structured into two parts that explore different statistical and unsupervised machine learning methods applied to trajectory data. Applications covered include anomaly detection in streaming data, real-time compression, and the smoothing and filtering of road vehicle trajectories. These techniques have been used in contexts such as compressing data in Vehicle-to-Everything (V2X) networks, recognizing reckless driving behavior, and extracting vehicle trajectories from traffic videos.

The first part of the dissertation introduces a newly proposed signal model, denoted as the Random Impulses Model (RXID), for representing road trajectories. Comparative computation of the covariance kernels and their corresponding eigenvectors were performed to validate the model against the NGSIM 101 dataset. This model has been utilized in three different applications, namely, two data compression algorithms designed to mitigate the frequency of Vehicle-to-Vehicle (V2V) transmissions, and a dynamic model for multi-object tracking and filtering of road vehicles from a video stream.

Although anomaly detection is a discipline that has reached a certain level of maturity, it has not been thoroughly developed in the context of detecting anomalies in streaming data. The subsequent part of this dissertation is devoted to the design of a real-time anomaly detection algorithm for such settings. This algorithm is based on two key components: dimensionality reduction and online hypothesis testing. An application regarding abnormal driving behavior recognition is presented as an example, followed by a comparison against two common methods to demonstrate the value of the approach.

The second part of this dissertation presents a Computer Vision (CV) technique for comparing the temporal development of two thunderstorms in distinct historical periods, resulting in a meteorological querying system that allows searching for similar weather days. It is also shown that dimensionality reduction proves beneficial in visualizing and comparing arrival patterns based on trajectory data from Dallas Fort Worth Airport (DFW). The proposed methodology exhibits promising potential not only for data visualization but also for developing methods that can learn from open-source data regarding air traffic controller decisions.

Additional topics, such as unsupervised segmentation of time series, are presented, along with a subtle extension of the probabilistic framework introduced in the first part to recognize non-compliant approaches, crucial precursors to non-stabilized landings.

The dissertation concludes with a final chapter summarizing major contributions and providing perspectives on future works.

Keywords: *trajectory modeling, filtering, smoothing, tracking, data compression, dimensionality reduction*.

Acknowledgments

I extend my heartfelt gratitude to my advisors, Pr. Daniel Delahaye of ENAC, Pr. Pierre Maréchal of IMT Toulouse, and Vincent Martinez of NXP Toulouse, for their invaluable guidance, encouragement, and insights throughout my journey. I am particularly appreciative of the support given by Mme. Helene Weiss, Mme. Dunia Legendre, and Mme. Corinne Joffre during my early days in France.

My sincere thanks go to Pr. Peng Wei, Derek, and Chelsea at George Washington University for their guidance and stimulating discussions. I am grateful to Anh Nguyen Tuong from Dassault Systèmes Paris for sharing his expertise in C++ programming. I also greatly appreciate the insights offered by Dr. Gabriel Jarry of the DGAC on unsupervised learning. Additionally, I am thankful to Dr. Xuan Huy Pham from CNRS GIPSA Lab, whose invaluable experience has assisted me in transitioning from academia.

A special place in my heart is reserved for my parents and my fiancée Vy Tran. Their unwavering love and support have been a continuous source of strength throughout my academic journey. I also wish to express my gratitude to the many people who drove me to school, prepared my meals, and maintained a clean workspace for me, whose names I unfortunately do not know.

Lastly, I am thankful for the generous support and resources provided by NXP Toulouse, ANITI, Paul Sabatier University, ENAC, and Google Colab, which have been essential to my research. This dissertation has benefited significantly from the support of the AI Interdisciplinary Institute ANITI, funded under the French "Investing for the Future – PIA3" program, Grant agreement number ANR-19-PI3A-0004.

List of Figures

List of Tables

PART 1

Road Vehicle Trajectory: Modeling, Compression, Filtering and Smoothing

1

1

Introduction

Abstract

In this chapter, we present an overview of Intelligent Transportation Systems (ITS). Despite ITS being a vast and well-researched discipline, the emergence of new sensors and technologies continues to require advancements in signal processing, particularly with respect to vehicle trajectories. This is the motivation for our studies, which will be presented along with the goals and scope of the dissertation. The significance of the proposed work will be briefly discussed here, with a more profound exploration to follow in subsequent chapters. Additionally, we provide an overview of the thesis structure.

Contents

1.1 CONCEPTS

1.1.1 Intelligent Transportation Systems (ITS)

Transportation plays an indispensable role in modern daily life. According to data from the American Automobile Association (AAA), the average American spends 51 minutes driving each day, covering more than 220 miles per week. This equates to an astounding annual distance of nearly 11,500 miles [4]. The study also highlights an upward trend in driving time, with a rise of approximately 5% over the two years from 2014. Notably, the proportion of drivers above the age of 75 has surged, showing a 23% increase [4]. As individuals spend more time driving daily and continue to drive into their later years, the demand for a safer and more comfortable driving experience has substantially grown.

Moreover, the implications of increasing transportation activity on the environment and human health warrant attention. Road travel is accountable for nearly two-thirds of the Carbon Dioxide emissions within the transportation sector of the European Union (EU), thus making it a chief contributor to air pollution and sound disturbances in urban regions [5]. This heralds a prominent health concern for the foreseeable future. Sustained contact with polluted air and noise has been associated with a multitude of cardiovascular, neural, and auditory disorders. In response, the European Commission (EC) has put forth its vision for European transportation [6], aspiring to achieve a "green, digital transformation and become more resilient to future crises". The proposition advocates for a 90% reduction in emissions by 2050, with an extensive implementation of a "smart, safe, accessible and affordable transport system."

Intelligent Transportation Systems (ITS) are considered as a potential solution to the multifaceted challenges facing modern road transport. By examining factors that can be modified, ITS aims to create a more comfortable, efficient, and sustainable model of road transport. While the broad spectrum of considerations within ITS extends to policymaking and infrastructure planning, this dissertation focuses on a specific component: the potential application of a vast amount of data collected from an extensive array of deployed sensors. The integration and analysis of this data could lead to the development of enhanced decision-making tools and onboard automotive systems that significantly improve the driving experience.

1.1.2 Advanced Driver-Assistance Systems (ADAS)

One of the early examples of ITS in modern vehicles is the ADAS which provide safety and automated features for drivers in real-time. Examples include:

- Blind-spot monitor,

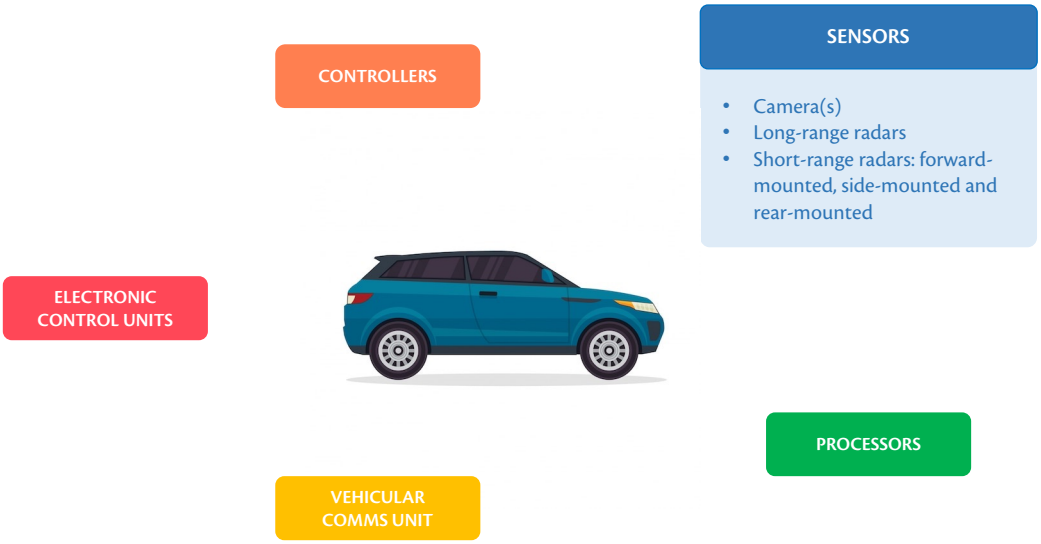


Figure 1.1: Key components of ADAS.

- Forward Collision Warning,
- Intersection Assistant,
- Lane Departure Warning System,
- Electronic Stability Control (ESC)…

The key components of an ADAS system are shown in Figure 1.1. Among these, sensors are the most essential, with the CCD camera, long-range and short-range radars (such as ultrasonic, LIDAR) being common choices. Controllers and ECUs are required if control algorithms exist to provide input to the car actuators, for example, the Anti-Lock Braking System (ABS). Processors are the components where perceptive and decision-making algorithms are run, and depending on the individual vehicle, a modem device for communication with other vehicles using a Vehicle-to-Everything (V2X) network might be equipped.

It's worth noting that ADAS are suggested for incorporation within a broader framework pertinent to the autonomy level a vehicle can accomplish. The Society of Automotive Engineers (SAE) presents six levels of autonomous driving, wherein levels 0, 1, and 2 could potentially be attained through the inclusion of ADAS in vehicles [7] alone, without any additional integration into higher levels, like a transport infrastructure. Specifically, at levels 0, 1, and 2, an automated regulator can manage the brakes and acceleration, yet the vehicle stays under the complete control of the operator. This mandates that the driver maintains full alertness and awareness. The 3rd and 4th levels of autonomous driving assign a portion of the driving task to the control system but still need the driver's interven-

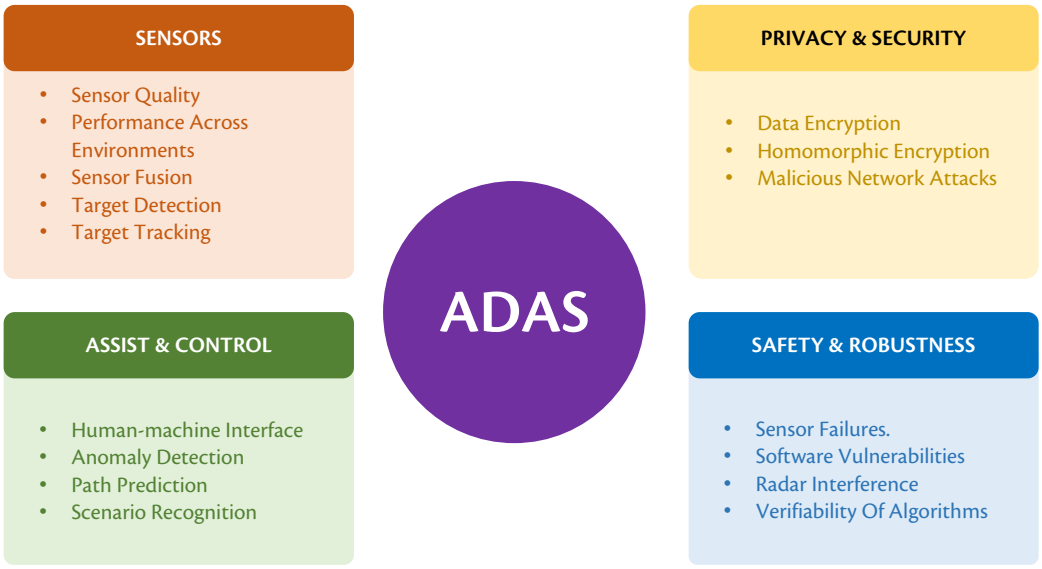


Figure 1.2: Exemplary challenges faced when developing ADAS applications.

tion when required. Level 5 signifies the pinnacle of autonomy, with no human intervention needed, even during emergencies. Although extensive research is in progress, the general agreement is that the obstacles tied to ADAS design (Figure 1.2) persist as similar to those of autonomous driving, but only supplemented by the requirement of planning and control algorithms for the generation of control inputs.

As presented in [8], most of the current quoted annual revenues for the ADAS market range from $5 billion to $8 billion, which demonstrates a substantial potential for growth when juxtaposed with audio and telematics quoted at $30 billion, and climate control at $60 billion. Despite several optimistic predictions about the adoption rate of ADAS, the utility of these systems is still generally considered marginal, while safety concerns cast a significant shadow on the adoption trends. This underwhelming performance is not generally attributed to the market's cautionary attitude, but rather because ADAS consist of a large number of separate yet interconnected challenging engineering problems, the reliable solutions for which have only recently begun to emerge.

Figure 1.2 presents a non-exhaustive list of problems that engineers usually face when developing ADAS applications, in addition to other common constraints such as cost and hardware limitations. For example, many ADAS applications require the detection of various road objects as prerequisites - such as vehicles, traffic signs, and pedestrians. State-of-the-art detection under diverse lighting conditions and appearances has only been achieved with the advent of deep neural networks [9]. As clearly depicted in Figure 1.2, barring hardware-specific issues,

the majority of challenges tend to be tied to the handling of vehicle trajectory data in some form. We posit that addressing this fundamental hurdle will induce substantial enhancements across an array of ADAS system developments, hence marking it as a crucial gap in current research that demands urgent attention. For now, we will momentarily set aside this topic, resuming with a more in-depth exploration later, in Section 1.2.

1.1.3 Vehicular Communications and V2X

While the challenge of autonomous driving is already significant, the broader aim of ITS often extends beyond the safety and comfort of individual drivers, to a more collective perspective. Essentially, coordinating multiple vehicles can unlock further potential for enhancing safety and improving the efficiency of current transport systems. This has required the creation of a cross-vehicle communication link, facilitating the transmission and receiving of data.

A key example of this concept is highway platooning, where vehicles, often trucks, travel in close proximity to reduce air drag and consequently, save on fuel. This close formation also allows the convoy to occupy a relatively constant area of the road, helping to reduce traffic congestion. The primary challenge lies in coordinating these convoys in a tightly controlled manner, requiring continuous transmission of control parameters, such as speed and direction with stringent constraint on delay to ensure safety [10].

The notion of communication between vehicles can be actualized via a Vehicle-to-Vehicle (V2V) network, capable of extension into Vehicle-to-Infrastructure (V2I), and Vehicle-to-Network (V2N), and collectively referred to as Vehicle-to-Everything (V2X). Analogous to the Internet-of-Things (IoT), V2X delivers a protocol for road entities—including vehicles, roadside sensors, traffic signals, and even data centers—to interact. This enables an expansive perception domain that surpasses the capabilities of individual participants, and the introduction of Collective Perception (CP) [11] broadens the perception scope to even those not equipped with V2X, such as pedestrians, motorcyclists, and road laborers. This can be accomplished with assistance from diverse sensors, inclusive of those installed on vehicles, road-side units, and traffic cameras, and the transfer of data via the Collective Perception Message (CPM). The evolution of the V2X concept has roots in preliminary research in Mobile Ad-Hoc Networks (MANET) and Vehicular Ad-Hoc Networks (VANET). Below are examples of already-deployed V2X services:

Example: Day One V2X Services

Examples of Day One V2X Services (as of 2019) include [12]:

1. Emergency electronic brake light
2. Emergency vehicle approaching
3. Slow or stationary vehicle(s)
4. Traffic jam ahead warning
5. Hazardous location notification
6. Road works warning
7. Weather conditions
8. In-vehicle signage
9. In-vehicle speed limits
10. Probe vehicle data
11. Shockwave damping
12. GLOSA / Time To Green (TTG)
13. Signal violation / Intersection safety
14. Traffic signal priority request for designated vehicles

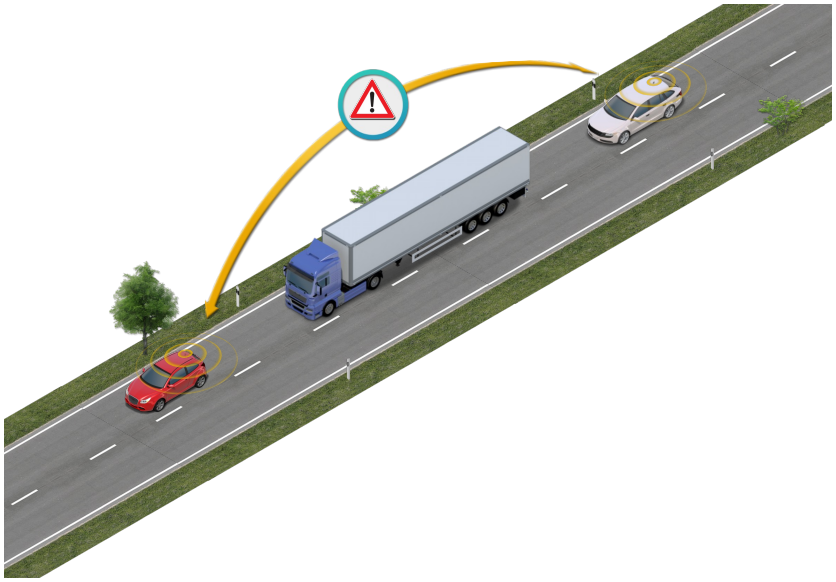


Figure 1.3: Vehicle overtaking scenario. The red vehicle can be rendered visible to the white vehicle through the continuous beaconing of its position and other information through the wireless medium with V2X [1].

Figure 1.3 depicts a scenario in which V2X technology could prove beneficial. The white vehicle intends to overtake the long truck; however, its perception is limited due to the red vehicle being concealed from its line of sight. In such a situation, if both the red and white vehicles are equipped with V2X communication systems, they would be able to detect each other through the continuous broadcasting of

their respective positions via a standardized message format, called Cooperative Awareness Message (CAM).

While V2X encompasses a wider range of road scenarios, including notification of forthcoming road hazards and vehicle platooning, the scenario represented in Figure 1.3 is one of the initial proposals and continues to be fundamental when discussing V2X's benefits. It becomes apparent once more that trajectory data forms the core of the information transmitted over the wireless medium. As such, enhancing our capability to process trajectory data more proficiently will indubitably yield positive outcomes in designing a more efficient vehicular communication system.

A more thorough introduction to V2X, as well as recent developments pertaining to data compression and congestion control will be presented in the corresponding chapter (Chapter 5).

1.2 THE NEED FOR TRAJECTORY DATA PROCESSING

1.2.1 Motivation

The preceding discussion underscores the significance of Trajectory Data Processing within numerous ITSs applications. In this section, we address the main difficulties encountered in processing this category of data and the potential impact that theoretical advancements in this field could potentially foster. These encompass the evolution of ADAS, enhancement of data fusion and object tracking performance of radar systems, in addition to the realization of a more efficient data communication scheme amongst V2X participants. This, in turn, could improve the capacity, responsiveness, and quality of service for the entire network. Furthermore, owing to its time-series characteristics, it could also make significant contributions to the broader fields of signal processing and unsupervised machine learning.

As shown above, the nature of data frequently emerging in ITS is predominantly time-series. This data type, characterized by its high-dimensional complexity, often introduces numerous processing challenges. One prominent difficulty is known as the "curse of dimensionality," which complicates machine learning tasks due to the need for larger datasets, limited availability of data visualization tools, and latent autocorrelation features. Furthermore, the absence of event localization adds an additional layer of complexity to the analysis, as very often, it requires simultaneous solving of both identification and classification tasks. Adding to these challenges, time-series data is often tainted by noise, necessitating the filtering process that requires numerous assumptions about the noise's characteristics.

As an example of time series data, Figure 1.4 shows the raw trajectories ob-

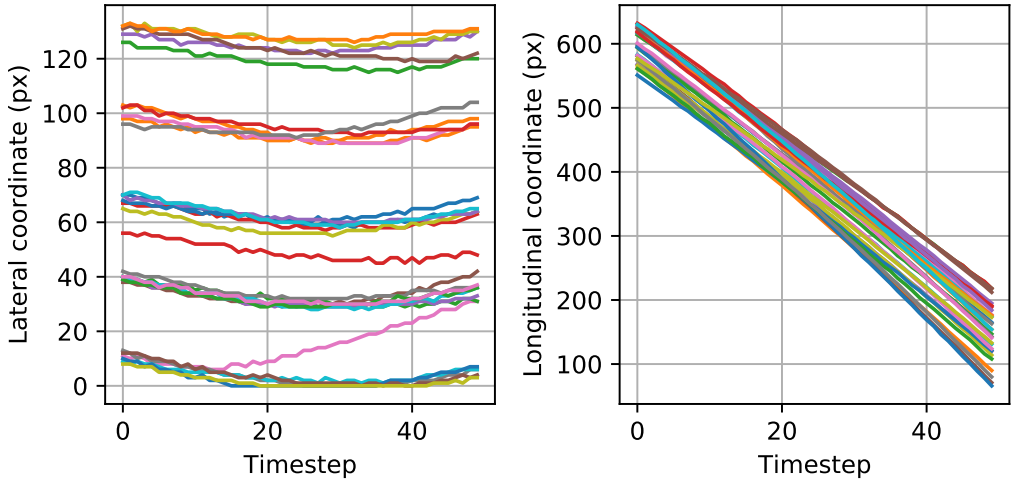


Figure 1.4: Exemplary time series data in ITS: raw trajectories of vehicles extracted from a video sample of the NGSIM-101 dataset.

tained through the tracking of vehicles from a video sample captured by a traffic camera. Each trajectory comprises two components: lateral and longitudinal, which represent lane-changing and speed characteristics, respectively. It is also obvious that this highway has five lanes and that the traffic flow is relatively stable. An ADAS that warns anomalous driving behavior will have to process this type of data. In the following, we provide a few additional examples of how trajectory data processing can assist ADAS, autonomous driving, and V2X applications.

Example: Road Object Recognition and Tracking

Object recognition and tracking constitute pivotal components in the development of Advanced Driver Assistance Systems (ADAS) and autonomous piloting designs. These systems require the vehicle to accurately identify diverse elements in its environment, including other vehicles, pedestrians, lane markings, and traffic signs (Figure 1.5). Besides recognition, estimating the movement of these objects is critical for efficient trajectory planning. Notably, motion information can assist the recognition process in cluttered environments, and conversely, accurate recognition contributes to enhanced tracking performance. This reciprocal relationship underscores the need for integrating motion data with object appearance, a central concept in many Multi-Object Tracking (MOT) algorithms, such as SORT [13]. Consequently, processing trajectory data helps to augment the efficacy of computer vision tasks.

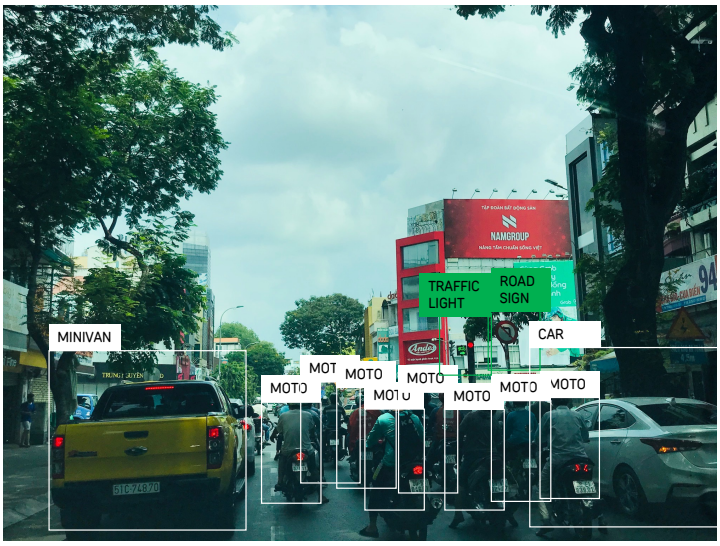


Figure 1.5: This image portrays a road scene in Ho Chi Minh City, Vietnam, with various road objects annotated. It's important to note the presence of numerous potential false sources that could complicate the recognition process, including billboards and vehicles parked on the pavement.

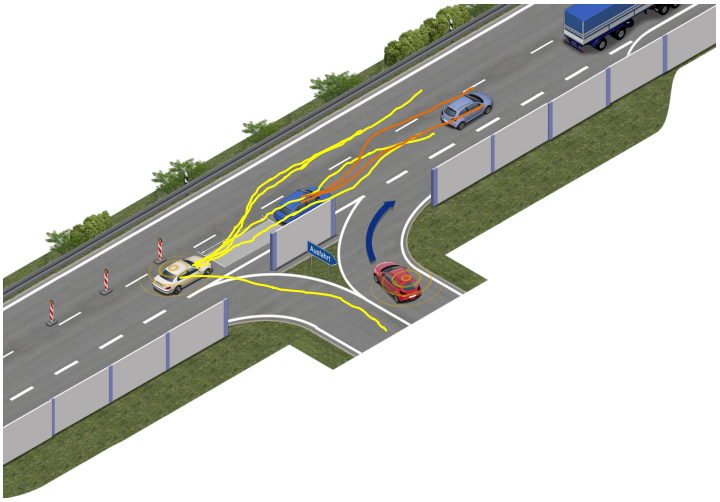


Figure 1.6: To avoid collision, the warning system usually has to anticipate the trajectories of nearby vehicles (orange curves) as well as the trajectory of itself (yellow curves). Trajectory prediction is one of the most important task in ITS.

Example: Collision Avoidance Warning

Collision Avoidance forms a critical element of autonomous driving, with trajectory prediction as its cornerstone. This component allows for the anticipation of the movements of neighboring vehicles, proffers warnings about impending collision risks, and facilitates the formation of plans to avert such threats. The literature surrounding this issue is both abundant and varied [14], reinforcing its position as an integral subset of trajectory data processing.

Example: Anomalous Driving Behaviour Detection

Applications such as the detection of wrong-way driving or reckless driving [15] require continuous observation of the vehicle's trajectory over a certain period of time. Often, it is compared to a standard collection of behaviors for classification purposes [16]. This application routinely incorporates various machine learning techniques, particularly those relevant to time series data processing, within which trajectory analysis is a case in point.

Example: V2X Collective Perception (CP) Data Compression

CP refers to a V2X service that facilitates the exchange of information regarding detected objects with other road users, via sensors mounted either on a vehicle or a road-side unit. ETSI Technical Report 103 562 v2.1.1 [3] presented simulation results demonstrating CP's substantial impact on channel resources, notably bandwidth, leading to a worsened Packet Delivery Ratio (PDR). Several mitigation strategies were proposed to address the issue, encompassing message rate regulation, congestion control mechanisms, and support for multi-channel operation. Nevertheless, as evidenced in Chapter 5, by simply considering the vehicle's trajectory, we can effectively decrease the message rate by 2.5 times, without significantly compromising information related to the trajectory, with no additional channel resources required.

Some of these examples encompass a significant portion of the content to be discussed in the ensuing chapters. This exploration is facilitated by initially analyzing the properties of road trajectory data via a novel mathematical model, followed by the model's adaptation to a diverse range of applications, extending from ADAS design to V2X data compression (Figure 1.7).

1.2.2 Model-based vs Model-free

Trajectory data processing, like all time series processing, can be divided into two main categories: model-based and model-free. The former involves deriving a model for road trajectory from the theoretical understanding of the underlying physical processes, such as the acceleration or steering mechanism of the vehicle [17–19]. Further psychological assumptions about human reactions have led to more sophisticated models, such as car-following models [20] and lane-changing models [21]. A significant advantage of trajectory models is their simplicity, which often enables the derivation of mathematical properties, thereby leading to a more comprehensive understanding of their performance and potential shortcomings. However, a key disadvantage is that the assumptions made in these models might

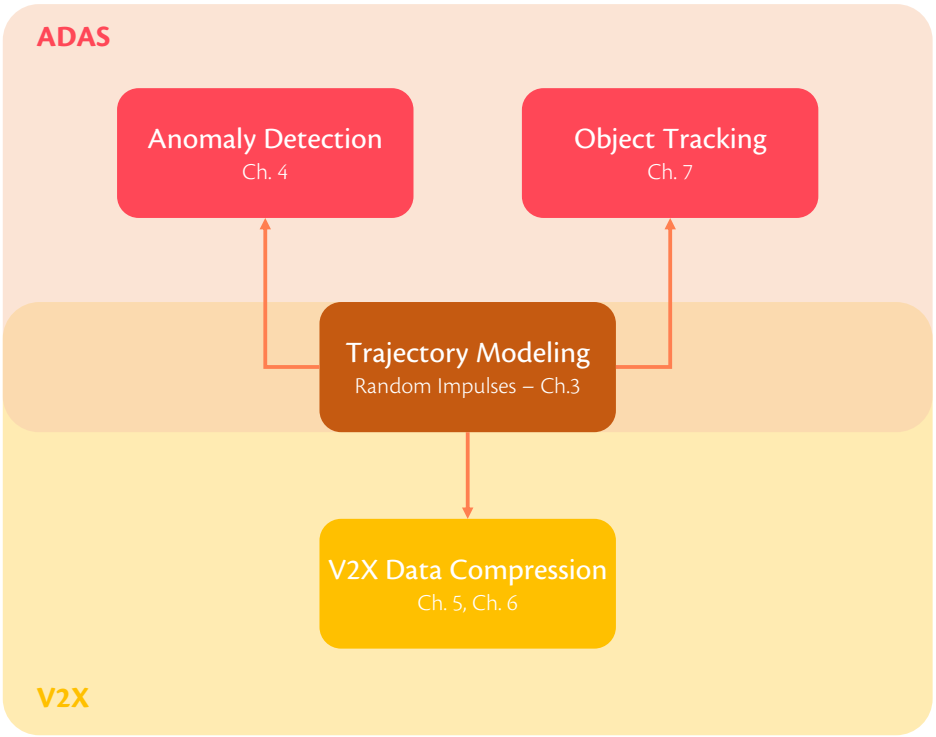


Figure 1.7: The development of a trajectory model engenders a multitude of applications in ADAS and V2X. While this dissertation primarily centers slightly more on the application of V2X Data Compression (presented in Chapter 5 and 6), the generalizability of the model holds the potential to traverse well beyond the confines of V2X and ADAS, as shown in other chapters.

lack rigor and may not correspond well to various traffic scenarios.

On the other hand, model-free trajectory models, such as those modeled with Hidden Markov Chains and deep generative models [22,23], often require a large dataset for training. Although this approach greatly simplifies the model design process and can represent complex nonlinear relationships between factors, the robustness and validity of these models are less understood. This can pose a risk for safety-critical applications, where the performance of the models cannot be guaranteed. Nevertheless, these models have achieved tremendous success, at least on benchmark datasets.

The general engineering problem extends beyond the raw predictive power of the trajectory models. Factors such as computational complexity also play a crucial role, especially when implementing algorithms on embedded platforms that require light memory footprints. Battery-powered devices impose additional constraints on energy demand, which brings to fore considerations about the cost-effectiveness of the solution. In these respects, model-based trajectory models often take the lead and are commonly found in most Intelligent Transportation System (ITS) components, such as smart traffic light controllers, V2X communication devices, and various onboard electronic components.

One could argue that the choice between model-free and model-based approaches mainly depends on the specific problem at hand, as well as any additional constraints imposed. Another question to consider when choosing a method, or even when deciding to develop a new model, is whether the problem at hand requires the use of a more complex model, such as a deep neural network, or if currently available models are enough. The answer to this question often emerges after a lengthy process of trial and error. However, in some instances, a deep theoretical analysis of the model could yield the same answer. This undoubtedly will lead to cost-saving, reduced product development time and confidence in the reliability in the performance of the model.

1.2.3 Datasets

To successfully execute model conception, analysis, and validation, a real-world dataset of vehicle trajectories is essential. Three notable datasets, known for providing high-quality trajectory data suitable for in-depth traffic flow analysis, are:

1. The well-regarded NGSIM dataset [24], provided by the U.S. Department of Transportation. This dataset provides high temporal resolution traffic data (at 10Hz) collected from southbound US 101 and Lankershim Boulevard, as well as eastbound I-80 in Emeryville, CA.
2. The Hanshin Expressway Dataset [25], which comprises five sets of one-hour traffic data collected from 2km sections of the Hanshin Expressway

Route 11, Route 4, and Route 13 in Japan. The dataset encapsulates nearly all vehicles and also includes road surface information.

3. The WUT-NGSIM dataset [26] offers high-quality data with 10Hz temporal resolution as well. Similar to NGSIM, this dataset uses an array of high-speed cameras to provide stable video streams for a sophisticated data processing framework, which employs a Kalman filter and Hungarian algorithm for tracking and estimation.

The NGSIM dataset has been widely used in research for various purposes, such as deriving lane-changing models [27], classifying driving styles [28], and predicting trajectories [29]. This dissertation also employs the NGSIM dataset to develop trajectory models, but it delves a step further to offer a more profound theoretical understanding of the physical processes observed in this dataset.

1.3 PROBLEMS, AIMS AND OBJECTIVES

The preceding discussion has shown that processing trajectory data (or timeseries) is central in ITS. In many scenarios, a suitable trajectory model must be chosen. The development of a model-free trajectory model would necessitate the availability of a trajectory dataset. It is obvious from the discussion in Section 1.2.3 that the size of these available datasets is generally small, often consisting of only a few hours and thousands of trajectories of vehicles collected exclusively in highway data scenarios. As a result, the use of a deep generative model could raise concerns about reliability and generalization capability due to the unbalanced dataset. Moreover, the majority of models used in automotive engineering still need to have their performance guaranteed. This is only possible with a deep analytical study. Given that the majority of deployment platforms are embedded systems, computational cost, hardware price, and execution time are also critical factors.

These considerations have driven demand for the availability of a model-based model that is still capable of learning and adapting, but to a point that still leaves room for analysis, deriving performance guarantee bounds, and wide adaptation in various ADAS and autonomous driving applications. It must also be computationally lightweight enough to be implemented on time-critical systems.

Most model-based counterparts in literature, however, suffer from different weaknesses, which we shall detail as follows.

1. Physical models are especially suitable for controlling engine speed and actuators, like ABS. They effectively capture physical effects, such as friction. However, these models are often deemed too detailed for analysis on a larger scale, such as on a road segment.

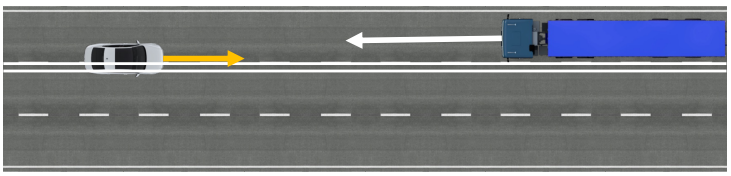


Figure 1.8: Illustration of a wrong-way driving scenario that can threaten safety. Anomaly detection in this case could be very useful.

2. Car-following and lane-changing models are frequently too complex for comprehensive analysis since they also simulate interactions between vehicles, such as the distance to the leading car and their relative velocities. Finding solutions often involves solving an intertwined system of partial differential equations, a task that is far from trivial.
3. Current kinematic models, such as those assuming constant velocity or constant yaw rate, typically have a very short prediction horizon. Despite this, they perform exceptionally well in terms of computational cost and simplicity, making them ideal for implementation on embedded platforms where deep analytical analysis is possible. However, the limited prediction horizon can be attributed to their failure to account for various factors such as the geometrical shape of the road and the effects of nearby vehicles.

Regarding this aspect, this dissertation aims to address this gap by introducing two models for the lateral and longitudinal components of a road vehicle trajectory, termed **Random Impulses Models**. Their validity is demonstrated by verifying against real-world trajectories, for which we use the NGSIM dataset. Through the computation of covariance kernels, it can be further established whether *information* has been fully extracted or not. This enhances confidence in the validity of the models and also bolsters the assurance of using the results derived in this dissertation for various applications.

While the rationale and analysis of the models are important, the application aspects should not be underestimated. This dissertation also explores various machine learning, filtering, and target tracking problems that could benefit from the newly introduced trajectory models. In particular, we address the following issues in depth:

- **Dimensionality reduction of trajectory data.** By reducing dimensionality, the representation (or embedding) becomes more amenable to visualization and other machine learning tasks, such as trajectory prediction and clustering.
- **Data compression.** With the possibility of representing roughly the same amount of information about the trajectory with a vector of fewer compo-

nents, we can compress trajectory data. While it is straightforward to perform offline batch compression of trajectory data, performing online compression is more challenging since the vehicle's full trajectory has not yet been completed. This problem has significant implications for vehicular networks, such as V2X. Compressing data could allow for serving more road participants, reducing the package collision rate, and improving the transmission range.

- **Spherical Codec** is the first data compression algorithm for CAMs, which relies on predicting future trajectory and transmitting information about a trajectory's representation rather than the trajectory itself in the time domain. This approach has led to a 2.5 times improvement in the channel busy ratio for the same number of road participants.
- **Linear Extrapolation over the Mean-Subtracted Trajectory (LEMONS)** is another simple algorithm which can be proven to be the optimal transmission algorithm that works directly with the trajectory in the time domain. It leverages the introduced models for optimal performance guarantees and performs comparably to the Spherical Codec. However, it surpasses the Spherical Codec in computational cost due to the absence of a representation estimator.
- **Anomaly Detection** is another important application. Figure 1.8 illustrates an exemplary scenario where a car oversteps the road centerline and drives in the wrong direction relative to a truck. Detecting such scenarios could be lifesaving. Traditional approaches may require access to a knowledge base about lane divisions, direction assignment, and so on. However, with an unsupervised learning approach, there is no such need, and as a result, detection algorithms can be adapted more quickly to various road scenarios, even in countries with different driving rules. The key challenge is that while this method is effective for offline classification (i.e., classification when the whole trajectory is available), online classification (i.e., classification when only part of the trajectory is available) is much more difficult. The solution typically involves using a sliding window and repeated decomposition of trajectories [30, 31], but these methods neglect the autocorrelation feature between data points lying across different sliding windows. To help address these problems, we present three anomaly detection algorithms, two of which serve the same purpose, while the third one serves a slightly different purpose, as follows:
- (1) **The Likelihood Method for Anomaly Detection** is similar to the computation of the Bayes factor for a presumed distribution of the representation of trajectory data, which acts as the prior distribution. In particular, we

introduce a vanilla case and a noise-corrupted case, which lead to two real-time hypothesis testing algorithms that reject the null hypothesis that the trajectory is nominal.

- (2) **The Bayesian Method** performs real-time hypothesis testing on the representation domain, but this approach leads to difficulty in the computation of a generalized Chi-square distribution. We propose an approximation method to use the Gaussian distribution instead, when the number of observations is sufficiently large.
- (3) **An anomaly localization algorithm** based on nearest neighbors, which exploits the non-correlation between Functional Principal Component Analysis (FPCA) components to avoid resampling the training set every time a new observation is received. This algorithm is capable of localizing the anomaly in time, and we propose its application in detecting anomalous *aircraft* landing trajectories.
- Another application is related to **filtering and smoothing of trajectory data.** The Random Impulses Models provide justifications and the correct implementation of the process model, rather than employing arbitrary random walk process models like those used in [13].
- **Estimation and learning of model parameters**. Here, the solution emerges not directly, but through an examination of an associated problem that grapples with the intertwined challenges of segmenting the time series and ascertaining the number of modes. In addressing this, we utilize a Dirichlet Prior to identify the number of change points, and subsequently detect these change points within a hypothesis testing framework.

1.4 SIGNIFICANCE

The previous section underscores the potential transformative impact the development of such a model could have on various facets of automotive applications. We highlight several avenues through which our work might bolster the growth of this field:

1. Our Random Impulse models are both simple and lightweight, making them ideal for use on embedded platforms where energy and computational constraints are rigorous.
2. These models provide valuable physical interpretation and intuition. For example, the mean-trajectory can be interpreted as representing the regulatory effect of drivers keeping a safe distance from each other.
3. Despite their simplicity, these models boast adaptability, particularly through adjustments of the mean-trajectory.

4. The theoretical analysis we present clarifies the conditions that enable convergence of the covariance kernel. This understanding should instill greater confidence in applying these models to a variety of potential road scenarios.

Our Random Impulses models are poised to bring about advancements within the fields of ITS, automotive technology, and vehicular networking. Notably, the applications addressed in this dissertation showcase the promising potentials of our work:

1. Two introduced codecs enable a considerable reduction in V2X channel utilization during Cooperative Awareness Message (CAM) broadcasting. This efficiency enhancement improves the network's overall performance and capacity, particularly critical during instances of heavy congestion.
2. We offer two important anomaly detection algorithms—likelihood-based and Bayesian-based—for the immediate detection of irregular trajectories. These methods, applicable to vehicle paths, prove invaluable in identifying instances of wrong-way, under-speed, over-speed, and reckless driving. As such, they hold the potential to become crucial components of various ADAS applications, fostering safer roads.
3. In the second part, we introduce a related anomaly localization algorithm. This tool can detect unsteady aircraft approaches in real-time, providing a critical line of defense against potential approaches that bear significant safety risk.
4. Additionally, we present a changepoint detection framework designed to extract notable patterns from time series data in an unsupervised manner. This algorithm can play an important role in deriving training sets for various machine learning algorithms that involve time series.
5. The topic of dimensionality reduction is briefly revisited as we incorporate Multidimensional Scaling for the recognition of traffic patterns affected by thunderstorms within the Terminal Maneuvering Area (TMA). This method facilitates quick visualization and comparison of traffic patterns, thereby accelerating situational understanding. It also helps various planning algorithms by considering similar weather factors.

1.5 DISSERTATION STRUCTURE

The subsequent sections of this dissertation are arranged as follows. Chapter 2 provides an in-depth background on ADAS and vehicular networking concepts, coupled with a thorough literature review. A detailed discussion on the strengths and weaknesses of previous researches is included, offering a clearer perspective on the research questions presented in this chapter.

The first part of this dissertation, entitled "Road Vehicle Trajectory: Modeling, Compression, Filtering and Smoothing," encompasses Chapters 3, 4, 5, 6 and 7. Chapter 3 is devoted to the exploration of Random Impulses Models, presenting both the design rationale and the validation process against the NGSIM-101 dataset. The approximate linear forms of these models give rise to the KL transform—a linear transformation that efficiently represents a vehicle trajectory with minimal components. The applications of these models are explored in the subsequent chapters. Chapter 4 introduces a probabilistic framework designed for real-time anomaly detection in vehicle trajectory data. Chapter 5 and 6 delve into data compression techniques in the context of vehicular networking, while Chapter 7 lays out a framework for filtering and smoothing trajectories derived from primary data sources such as aerial videos.

The second part of the dissertation features three chapters on other themes relevant to machine learning on trajectory data. Specifically, Chapter 8 presents a framework for unsupervised segmentation of time series, based on frequently observed patterns. This is crucial for identifying parameters for the Random Impulses Models introduced in the first part. Chapter 9 uses Multidimensional Scaling (MDS) to depict the air traffic situation in various thunderstorm scenarios. In line with this theme, Chapter 10 introduces a framework for localizing anomalies, specifically designed to detect unsteady aircraft approaches based on energy characteristic curves.

The dissertation concludes with Chapter 11, where key results are recapitulated and potential future research directions are explored. Figure 1.9 illustrates the dependencies between the chapters, serving as a useful roadmap for readers.

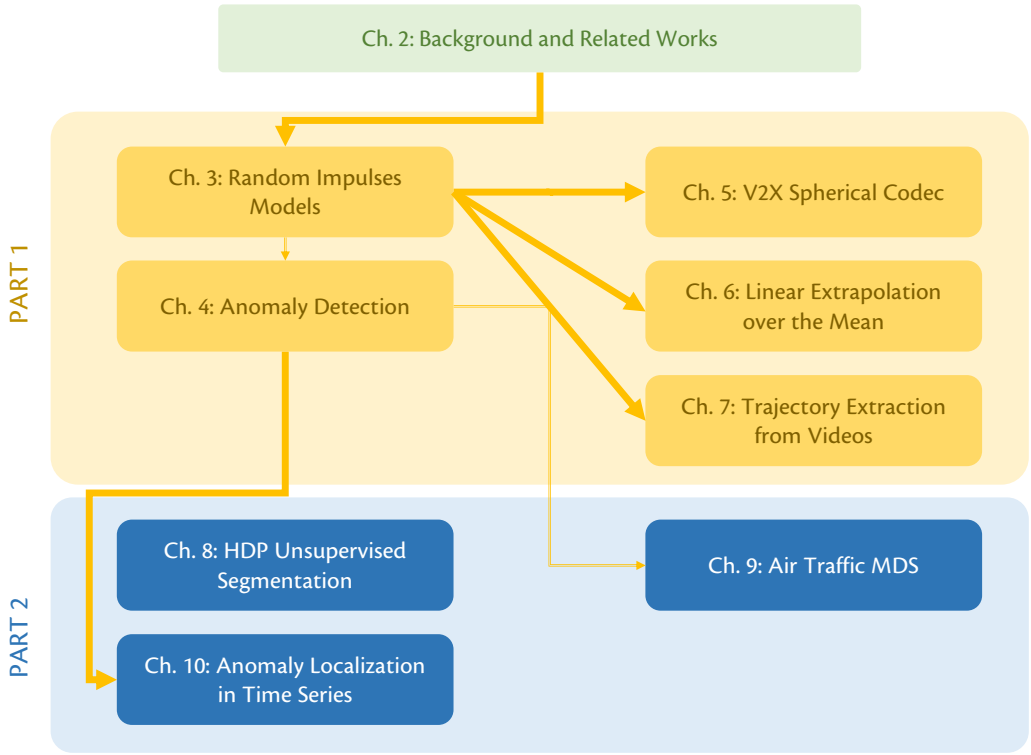


Figure 1.9: The dependency graph of the chapters. The thick arrows indicate content dependency, indicating that the following chapter reused many results from the preceding chapter. The thin arrows indicate related chapters, indicating that similar ideas may exist in the two chapters, but it is not necessary to read one chapter before another.

2

Related Works about Trajectory Data Modeling and Functional Principal Component Analysis

Abstract

This chapter provides a review of diverse trajectory data models, ranging from rudimentary physical models to their intricate, interaction-driven counterparts. Despite the abundance of available trajectory models, there remains an unmet need for an novel approach. This emergent model should merge the elegance of statistical models with the requisite accuracy to cater to a broad spectrum of applications. These findings pave the way for the forthcoming introduction of the Random Impulses Models in Chapter 3. Additionally, this chapter introduces foundational concepts related to Functional Principal Component Analysis (FPCA), also alternatively known as the Karhunen-Loève Transform. Discussions pertaining to specific applications, including anomaly detection and V2X, are reserved for their respective subsequent chapters.

Contents

2.1 TRAJECTORY SIGNAL MODELS

The trajectory prediction problem is central to trajectory signal processing, and itself alone is already an extensive subject. A comprehensive review was given in [32]. Several attempts at trajectory formalization could be divided into physical, manoeuvre-based, and interaction-aware models [32]. In physical models, the differential equations governing the dynamics of a vehicle are derived, and inputs to this model are usually modelled either as Gaussian white noise or by some other simple equations. While the dynamics model is usually nonlinear, [18] showed that linearization up to the second degree was sufficient to accurately characterize the vehicle's motion. In [19], the authors proposed that GPS or inertial navigation data could be used to complement the observation model. The same idea also inspired the work of [33], in which the driver inputs were considered to be reactions from surrounding objects' dynamics, modelled as a random variable following exponential distributions. Then, Monte-Carlo sampling enabled the discovery of future scenarios. A comprehensive review of the road vehicle dynamics model can be found in [34–36]. While physical models can well-describe the motion of vehicles down to a minimal temporal solution, they seem to be more beneficial for automatic control system designs rather than in traffic research, mainly because they are too complicated, and the predicted trajectory is only valid for a short period of time. They do not generally consider the driver's intention, road traffic scenario, or other external factors.

The mentioned shortcomings can be remedied by switching to another class of methods that trade resolution for lower computational complexity. The simplest ones assumed constant velocity, acceleration, or yaw rate [37–39], thus it is unsurprising to see that the prediction was still only valid for a short time. A prominent extension of the above methods was to treat inputs as random variables rather than deterministic ones. Different weights could then be applied depending on the modelled constraints, such as road topology or aberrant acceleration values [40,41]. On the other hand, manoeuvre-based models incorporate a model for the driver's input on top of another model describing the vehicle's motion, such as polynomials and Gaussian processes. The latter approach usually results in a more usable form than a set of nonlinear differential equations. Parameters learned from a sample dataset or the dataset itself can be directly used for the task. This is made possible by the introduction of a distance function, among which Euclidean [42], Hausdorff [43], and Dynamic Time Warp [44] are among the most popular. Gaussian processes (GP) were shown to be surprisingly good at approximating vehicle trajec-

tories [45, 46]. In this approach, each trajectory is considered to be a path sample from a Gaussian process characterized by an unknown covariance function. The discrete implementation requires solving for N^2 entries of the covariance matrix for length-N trajectories. As a consequence, the computational cost of training could be prohibitive. In [47], the velocity field of the entire area was modelled as a Gaussian random field, which is the generalization of the Gaussian process to two dimensions. The motion patterns were then classified in an unsupervised manner by imposing a Dirichlet Process as the prior distribution. Another important class of methods couples the motion model with a driver's intent estimator, usually realized by some machine learning algorithm, such as Support Vector Machine [48], or Hidden Markov Model (HMM) [49]. In [50], the driver's intent was coded into a "goal curve", and uncertainties are learned into a set of "prefix points" following Gaussian distribution. The solution curve was then derived by fusing the two, thus adjusting the goal curve to learned realistic constraints. In [51], a lane departure and prediction algorithm was introduced that uses SVM to classify the driver's intent - which is modelled with an HMM and a Bayesian filter to generate the trajectory.

With the advent of deep learning methods for trajectories - a particular instance of time series - many neural network designs have been extensively studied. Recurrent Neural Networks belong to the earliest group of design that emerged in the field of trajectory prediction. In [52], a Long-Short Term Memory (LSTM) network was used to predict the driver's intention, followed by a context-reasoning process to yield a continuous prediction. In [53], an autoencoder structure was proposed that predicted future trajectories with a beam-search algorithm, while in [54], the prior map generated by a Generative Adversarial Network (GAN) was combined with a convolutional autoencoder network for modelling stochastic multimodality.

Traffic simulators typically use interaction-based models to generate artificial traffic demands employed in traffic research. These models assumed that the driver's intent depends on the traffic situation around, and is therefore subjected to the interactions between vehicles. General Motors's traffic model is probably the simplest and most well-known model assuming constant sensitivity and acceleration of the following vehicle [55]. Improvements in the driver's reaction to situation change were made, which resulted in the Gazis-Herman-Rothery (GHR) model [55]. Observations about the intention to maintain a safe distance and avoid collision provided an empirical formula for the velocity of the ego-vehicle following the leading vehicle shown in [56, 57].

Emphasis on the interaction between vehicles as the main driving force behind vehicle dynamics has led to the widespread belief that Convolutional Neural Networks can be helpful, with most approaches employing a top-down view of the traffic map. In [58], a hybrid LSTM-CNN was used to model road agents'

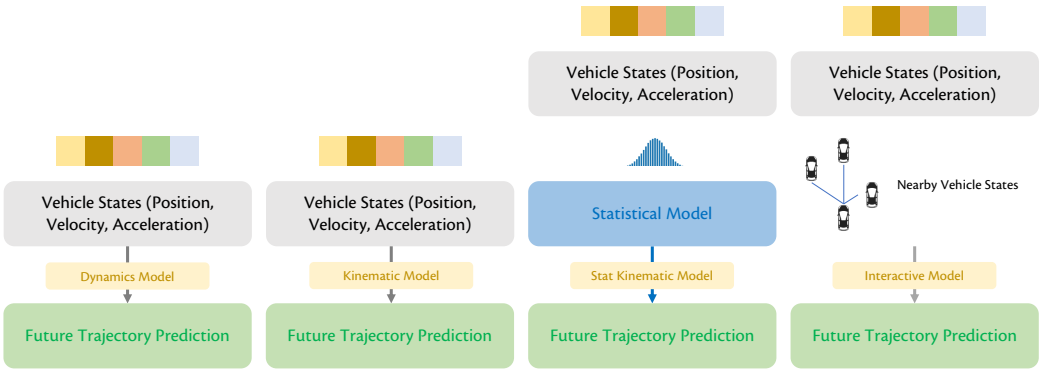


Figure 2.1: Taxonomy of vehicle trajectory models. From left-to-right: physical model (or dynamic model), kinematic model, statistical kinematic model and interactive model.

dynamics, shapes, and behaviours and their interactions to predict the future trajectory. The advantage was the capability to predict multiple agent behaviours in one sweep. RNN's primary disadvantage was the fading gradient problem, which was remedied with the introduction of the attention mechanism [59]. In [60], an attention mechanism involving global and partial attention to surrounding vehicles was used to model their relative importance to each driver. This is achieved via multi-head attention pooling, which serves as a junction between individual vehicles' LSTM encoder and decoder. Other essential ideas involved Graph Neural Networks [61], edge-enhance graph convolutional neural network [62], and deep reinforcement learning [63, 64].

For a more comprehensive review, we refer readers to [32, 65] and [14].

Figure 2.1 highlights the key distinctions among the models previously presented in the literature. For batch trajectory signal processing tasks, such as trajectory prediction algorithms executed on a modern System-on-Chip, deep learning methodologies or interactive models might be excessively demanding. Furthermore, the advantages of these methods, when averaged over numerous vehicles, may not be readily apparent. In other words, interactive models could offer clear benefits when focusing solely on the vehicle of interest. However, when considering a scale of 1,000 or 10,000 vehicles, the benefits might become less discernible as statistical kinematic models accurately capture performance expectations. It should be noted that some vehicles might perform better than others. This variability in performance is a crucial reason for this dissertation's focus on statistical kinematic models. They maintain the simplicity of alternative models while, when processing data on a large scale, provide performance that converges with that of interactive models.

2.2 THE ROLE OF TRAJECTORY MODELS IN APPLICATIONS

The preceding discussion in Chapter 1 has unveiled that the model-free approach typically demands a considerable dataset and often culminates in an enigmatic black-box. This complexity impinges upon the safety and robustness assessment due to the absence of rigorous performance guarantees analysis. Additionally, model-free methods generally require a significant amount of computational power, rendering them unsuitable for deployment on cost-effective, low-energy embedded platforms that are ubiquitous in vehicles.

In the following, we will endeavor to highlight the role that trajectory modeling plays in various ADAS and V2X applications. Furthermore, we will examine how the creation of a novel trajectory model could instigate transformative changes across an array of applications. Many of these points will be made clear in subsequent chapters.

1. Operating within an identical estimation framework (e.g., Kalman filter), prevalent models such as Constant Velocity - Constant Yaw Rate [56, 57] deliver a time frame for which the prediction remains valid (the horizon) that is markedly shorter than that yielded by the Random Impulses Models. This enhancement in performance is solely derived from the utilization of a better model, with no other variables being altered (refer to Chapter 3).
2. Within the context of anomaly detection, the presence of a trajectory model demonstrates that certain foundational steps, such as learning of representations from a training set, can be entirely circumvented. This process not only decreases the volume of data necessary for training and calibration of the framework but also promotes rapid market adoption in regions with divergent driving regulations, such as those where driving is done on the left.
3. In fields like radar object tracking, antenna beamforming, and traffic supervision, the provision of signal models can augment the efficacy of tracking algorithms, mitigating the rate of misassociation and tracking loss. Furthermore, the model also provides a sound basis for the design of the underlying algorithm, eliminating the need for making unwarranted assumptions such as the state vector following a random walk.

2.3 FUNCTIONAL PRINCIPAL COMPONENT ANALYSIS (FPCA)

2.3.1 *Background*

In signal processing, unitary transforms are widely used. Informally, a unitary transformation T is an isomorphism between Hilbert spaces $\mathcal{H}_1$ and $\mathcal{H}_2$ that pre-

serves the inner product:

$$T : \mathcal{H}_1 \rightarrow \mathcal{H}_2 \tag{2.1}$$

$$\langle x, y \rangle_{\mathcal{H}_1} = \langle Tx, Ty \rangle_{\mathcal{H}_2}, \forall x, y \in \mathcal{H}_1 \tag{2.2}$$

Very often, we will consider the particular case where $\mathcal{H}_1$ is the space of real square-integrable functions under the Lebesgue measure $L^2(\mathbb{R}, dx)$ and $\mathcal{H}_2$ is the sequence space l^2. Fourier Transform falls under such category. It is also worth mentioning the equivalent discrete version where both $\mathcal{H}_1$ and $\mathcal{H}_2$ are Euclidean vector spaces $\mathbb{R}^n$. In this case, T is expressed as a $\mathbb{R}^{n \times n}$ orthogonal matrix.

With a slight abuse of notation, for a given probability space $(\Omega, \mathcal{A}, P)$ and a measurable space $(S, \mathcal{B})$, the signal is modeled as a stochastic process:

$$\{X_t \triangleq X(t, \omega); t \in T, \omega \in \Omega\} \tag{2.3}$$

where t usually denotes time. In other words, the signal can be considered as a collection of S^T random variables. We further assume that S is an interval over $\mathbb{R}$ (i.e., X_t has finite support), the process's mean is zero, is square integrable and the statistics are known up to the second order. Let us define the covariance function:

$$R(s, t) = \mathbb{E}\left[X_t X_s\right]; t, s \in S \tag{2.4}$$

$R(s, t)$ defines a linear operator $A : L^2(S) \rightarrow L^2(S)$:

$$x \in L^2(S), (Ax)(s) = \int_S R(s, t) x(t) dt \tag{2.5}$$

It follows from Mercer's Theorem that the covariance function admits the following decomposition:

$$R(s, t) = \sum_{n=0}^{\infty} \lambda_n e_n(s) e_n(t) \tag{2.6}$$

where the functionals $e_n(\cdot)$ are eigenfunctions of the operator A corresponding to the eigenvalues λ_n, sorted from largest to smallest. The Karhunen-Loève Theorem says there exists a decomposition of X_t [66]:

$$X_t = \sum_{n=0}^{\infty} \sqrt{\lambda_n} V_n e_n(t) \tag{2.7}$$

$$V_n = \frac{1}{\sqrt{\lambda_n}} \int_S X_t e_n(t) dt \tag{2.8}$$

Equation (2.7) is the KLT of X_t. It could be shown that $\mathbb{E}[V_n] = 0$ and $Var[V_n] = 1$. Proof of (2.7) converging uniformly in S can be found in [67]. Generalization of the above discussion to the discrete case should be straightforward. Let $x[k] \triangleq x[k, \omega], k \in 1, 2, \ldots, N, \omega \in \Omega$ be a discrete stochastic process. The covariance function $R(s, t)$ is then replaced by the covariance matrix $C \in \mathbb{R}^{N \times N}$ where:

$$[C]_{ij} = \mathbb{E}\left[x[i]x[j]\right] \tag{2.9}$$

then, the KLT basis are eigenvectors of $[C]_{ij}$.

2.3.2 Optimality of Karhunen-Loève Transform

The KLT attracts attention mostly due to two following important properties. Firstly, notice that C is symmetric positive-definite, hence C admits the spectral decomposition:

$$C = F\Lambda F^T \tag{2.10}$$

where Λ is a diagonal matrix. Then the equivalent KLT in the discrete domain is given by:

$$y = Fx \tag{2.11}$$

thus, the covariance of the transform coefficients is:

$$Cov[y] = \mathbb{E}\left[(y - \mathbb{E}[y])(y - \mathbb{E}[y])^\intercal\right] = FCF^\intercal \tag{2.12}$$

combined with (2.10), we have:

$$Cov[y] = \Lambda \tag{2.13}$$

In other words, all the coefficients in the KL domain are independent, thus any manipulation of one component should not induce any change in the others. Another property is that KLT is the optimal transform in many performance criteria. [68] showed that KLT is the optimal unitary transform for performance measures of type:

$$\zeta(FCF^H) = \sum_{i=1}^{N} G(u_i^T C u_i^*) \tag{2.14}$$

where $G(\cdot)$ is continuous, increasing and convex and u_i are eigenvectors of C, i.e., columns of F. It was also shown in [68] that distortion rate, basis restriction errors are performance measure of this kind.

2.4 RESEARCH GAP IN THE LITERATURE

As a continuation of the discussion in Chapter 1, it is clear that despite the plethora of existing trajectory models, there remains an imperative need to develop a novel model. This new model should strike a balance between computational efficiency and the ability to maintain predictive power over an extended time period. For example, while the differential equations inherent in Physical Models may be suitable for brake control under varying weather conditions, they fall short in predicting the trajectory of nearby vehicles. Similarly, manoeuvre-based models, which incorporate additional state variables indicating the driver's intent, pose considerable estimation challenges. Gaussian Processes models and deep neural networks, despite their potential, demand extensive computational resources, rendering them unsuitable for designing a data compression algorithm to be executed on a low-power V2X modem SoC. Consequently, this dissertation seeks to bridge these gaps among different models and introduces a new model with broad applicability across numerous ADAS and V2X applications, the details of which will be expounded upon in the forthcoming chapters.

While the literature review presented in this chapter is informative, it certainly falls short of capturing the myriad background concepts and state-of-the-art developments related to specific ADAS applications, such as anomalous driving behavior detection or V2X data compression. Nonetheless, given the weak interdependence among the chapters, it is our belief that these contents are better suited for introduction within their corresponding chapters.

3

Exploring the Random Impulses Trajectory Model for Dimensionality Reduction and Applications

Abstract

Intelligent Transportation Systems (ITS) hold great potential for improving the safety, efficiency, and comfort of road users. These applications heavily rely on processing trajectory data, a significant component of which is the road vehicle trajectory model. This chapter presents a kinematic statistical model for large-scale ITS that is more suitable for batch data processing, especially in systems with limited hardware capabilities such as vehicular communication modems and embedded systems.

Our proposed model approximates slipping velocity and acceleration as random impulses originating from stationary distributions. After validation, we present two applications of this model.

Our derived Functional Principal Component Analysis (FPCA) basis allows for effective dimensionality reduction, a key step for various machine learning tasks such as trajectory prediction and anomaly detection.

Results on the NGSIM 101 dataset demonstrate that highway trajectories can be significantly reduced to vectors with very small number of dimensions for the first application. For the second application, the combination of filtering and smoothing yields a smoothed estimate of velocity with a confidence interval up to two times smaller than that achieved by filtering alone.

We believe that the proposed models are suitable for implementing stringent-

constrained data processing at the edges.

Contents

3.1 INTRODUCTION

This chapter presents the "Random Impulses" trajectory model, tailored to meet the specific requirements of a kinematic statistical model in batch data processing contexts. At its core, this model perceives individual vehicle trajectories as variations of a "mean trajectory". This mean trajectory represents the safety distance-keeping by drivers, coupled with a slow-varying random component that captures the influence of various factors considered to be Markovian. This approach simplifies the interactive effects produced by surrounding vehicles, while preserving the simplicity of the more basic kinematic models that assume constant velocity or acceleration [37, 69, 70]. The model's simplicity and computational efficiency make it particularly attractive for deployment in large-scale ITS, especially those operating under hardware constraints.

The considerable value of this model becomes more apparent in the subsequent sections of this chapter, where we demonstrate its applications through various

A version of this chapter, titled "Exploring the Random Impulses Vehicle Trajectory Model for Dimensionality Reduction and Motion Extraction from Aerial Videos", has been submitted for review to the *IEEE Transactions on Intelligent Transportation Systems* in 2023. The authors of this submission are Thinh Hoang, Vincent Martinez, Pierre Maréchal, and Daniel Delahaye.

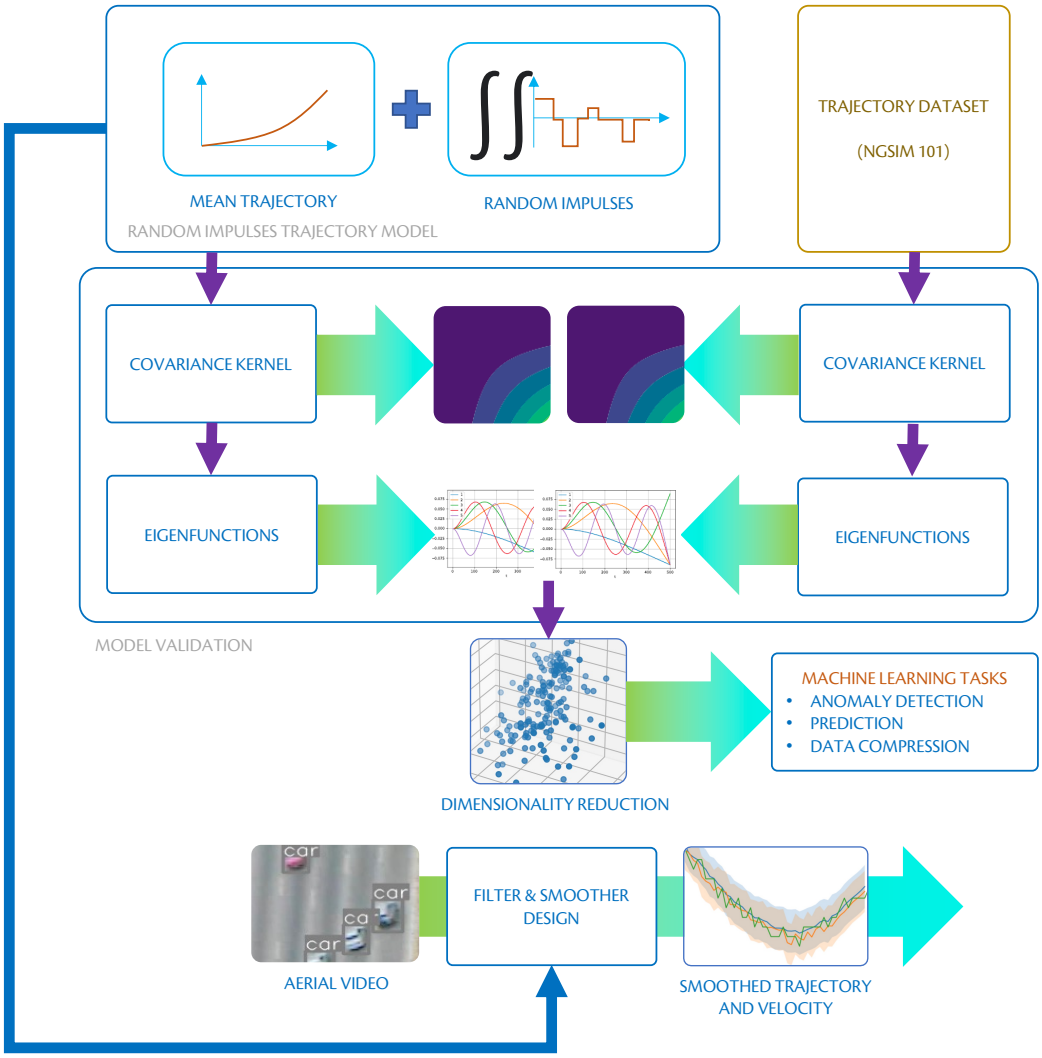


Figure 3.1: Overview of the chapter: we introduce novel signal models for road vehicle trajectories and validate these models through the computation of covariance kernels. The eigenfunctions of these kernels can be used for dimensionality reduction, a process that plays a critical role in several machine learning tasks. A filtering and smoothing framework based on these models will be left for a later chapter.

application frameworks related to the Karhunen-Loève Transform (KLT) - also known as FPCA - and an approximated linear state-space model. The KLT has strong ties to Principal Component Analysis (PCA), a technique that has found widespread use in the machine learning community for data dimension reduction. The intricate nature of the lateral impulses distribution invites the use of a Gaussian Mixture implementation for the filter and smoother. Achieving these applications with car-following or constant velocity models would have posed substantial challenges, which likely contributes to their limited implementation in edge processing facilities. This stark contrast underscores the innovative contribution of the "Random Impulses" model to the field.

Fig. 3.1 provides the overview of the chapter. It unfolds as follows: Section 3.2 introduces the "Random Impulses" Model and Section 3.3 justifies its validity through the analytical derivation of the covariance kernel. Section 3.4 expounds on the model's eigenfunctions for trajectory dimensionality reduction, a crucial component for anomaly detection in Advanced Driver Assistance Systems (ADAS). Section 3.5 that follows will provide various discussions around the assumptions and the results, and the chapter concludes with Section 3.6.

3.2 RANDOM IMPULSES MODEL

In this section, we formulate the Random Impulses Models for both the longitudinal and lateral components of the vehicle trajectory. The former represents the "fast-slow" characteristics of speeding, while the latter characterizes lane-changing behaviors. For reasons that will become apparent, we will refer to these models as the Random Acceleration Impulses Model (RAIM) and the Random Yawing Impulses Model (RYIM), respectively (Fig. 3.2). Consider the trajectory of vehicle i, denoted as $W_t^{(i)}, Y_t^{(i)}$. Here, $W_t^{(i)}$ specifies the vehicle's relative position with respect to the left roadside, and $Y_t^{(i)}$ indicates the longitudinal distance the vehicle has traveled into the segment (Fig. 3.3a). We conceptualize each vehicle's trajectory as an independent realization of two corresponding random processes, Z_t and X_t. These processes will be defined in Equations (3.1) and (3.6).

3.2.1 Random Acceleration Impulses Model (RAIM)

Consider a road segment with several lanes, all directing vehicles in the same direction. While vehicles might temporarily stop within the segment due to, for example, a traffic light, we simplify our model by assuming all vehicles enter and exit the segment at the same location (Figure 3.3b).

To avoid a collision, the motion of any vehicle entering a road segment is influenced by nearby cars. Similarly, the movement of these proximate vehicles also

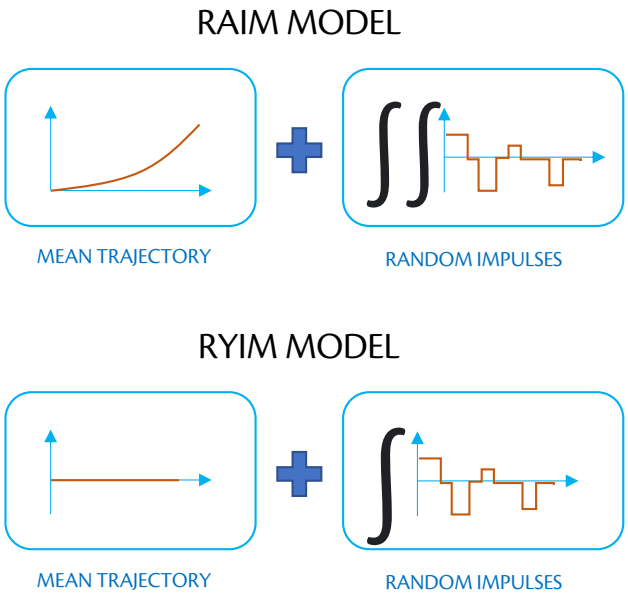


Figure 3.2: An overview of the RAIM and RYIM Models for longitudinal and lateral components of the trajectory time series. The "mean" component represents the self-regulatory effect of distance-maintaining behavior for safety assurance. The *Random Impulses* model the effects of various stochastic factors.

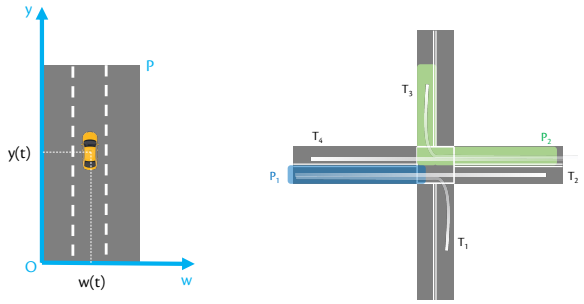


Figure 3.3: (a) Definition of the P-segment coordinate system. W_t, Y_t are relative coordinates, and absolute coordinates require road topology be taken into account. (b) An example of two *road segments* P_1 - the blue rectangle, and P_2 - the green L-shape road segment. Here, both trajectories T_1 and T_2 are considered valid for the *road segment* P_1, since vehicles enter and leave at the same places. However, for *road segment* P_2 only trajectory T_3 is considered valid because T_4 leaves P_2 prematurely.

Figure 3.4: All vehicles in the segment (blue) are considered to be following a virtual vehicle (green). If the car-following is exact, the separation (distance) between each vehicle is maintained, which is also the optimal scenario.

impacts the motion of other neighboring cars. This interaction results in a complex set of interlocking equations describing vehicle relationships without an easy solution. However, for simplicity, let's consider all vehicles in a given road segment as following a "virtual vehicle" in a "leisurely" manner. By "leisurely," we mean that the process of following is not exact. Nevertheless, any vehicle whose motion deviates substantially from the trajectory of the virtual vehicle runs an increased risk of colliding with other cars, as illustrated in Fig. 3.4. Formally, let $\{Y_t^{(i)}\}_{i=1}^{N_v}$ be the set of N_v longitudinal trajectories of vehicles traveling on the road segment P, $Y_t^{(i)} \in \mathbb{R}^{\mathbb{R}_+}$. Let $\bar{Y}_t$ be the trajectory of the virtual vehicle. Statistically, we consider $\{Y_t^{(i)}\}_{i=1}^{N_v}$ to be independent sample paths of a stochastic process:

$$\{X_t \triangleq X(t, \omega)\}, \tag{3.1}$$

where the sample function is defined as:

$$X(\cdot, \omega) : \mathbb{R}_+ \to \mathbb{R},$$

over the probability space $(\Omega, \mathcal{A}, P)$ indexed by $t \in \mathbb{R}_+$. The RAIM model assumes:

$$\mathbb{E}[X_t] = \bar{Y}_t, \tag{3.2}$$

and the "residual deviation" of the trajectory is generated from doubly integrating path-wisely the random acceleration impulses model A_t:

$$X_t = \mathbb{E}[X_t] + \int_{s=0}^{t} \int_{\tau=0}^{s} A_\tau \mathrm{d}\tau \mathrm{d}s. \tag{3.3}$$

We now describe the process A_t. For each $k \in \{0, 1\}$, let $\{I_{k,i}\}_i, i \in \mathbb{N}^*$ be a sequence of independent random variables (with respect to i), called renewal intervals associated with the state k, given by the distribution functions:

$$F_k(u) = P(I_{k,i} \leq u), \quad k \in \{0, 1\}, \quad i \in \mathbb{N}^*.$$

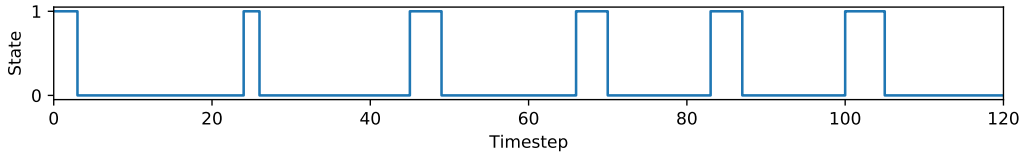


Figure 3.5: One sample path of $\mathcal{L}_t$ with renewal intervals $I_{0,k}$ given by $\Gamma(\alpha = 5, \beta = 0.5)$ and $I_{1,k}$ given by $\Gamma(\alpha = 30, \beta = 0.5)$. $\mathcal{L}_t = 1$ marks the period where there is acceleration activity, and $\mathcal{L}_t = 0$ otherwise. There are 11 regeneration points in this exemplary sample path located at $t \in \{3, 22, 24, 46, 48, 72, 75, 85, 88, 100, 105\}$.

Equipped with two sequences of renewal intervals corresponding to state $k = 0$ and $k = 1$, we define a sequence of regeneration points corresponding to moments where there is a jump from state 0 to state 1 and vice-versa:

$$W_I = \{\omega_i\}, \quad i \in \mathbb{N}^*,$$

such that:

$$\omega_i = \begin{cases} \omega_{i-1} + I_{1,\frac{i+1}{2}} & \text{if } i \text{ is odd,} \\ \omega_{i-1} + I_{0,\frac{i}{2}} & \text{if } i \text{ is even.} \end{cases}$$

This allows us to formally define a Markov renewal process $\mathcal{L}_t$, which is the *acceleration activity process* as follows:

$$\mathcal{L}_t = \sum_{i=1}^{+\infty} \left(\mathbf{1}_{\omega_i \leq t \leq \omega_{i+1}} \times \delta_{\{i \equiv 0[2]\}} \right), \tag{3.4}$$

where $i \equiv 0[2]$ indicates an even value of i.

Informally, the acceleration activity process is composed of alternating periods of idleness and acceleration. The lengths of these periods are determined by the renewal intervals of $I_{0,\cdot}$ and $I_{1,\cdot}$ respectively (Fig. 3.5). The RAIM represents this physical reality, suggesting that drivers following a virtual vehicle but intermittently apply and release the gas or brake pedals over random periods of time, at random times. This process results in the vehicle's motion deviating from that of the virtual vehicle.

Building upon (3.4), the RAIM model further assumes that acceleration impulses are*identically independent* samples from a *stationary* distribution function $F_a(u)$, resulting in the acceleration impulses model A_t:

$$A_t = \sum_{i=1}^{+\infty} \left(M_i \times \mathbf{1}_{\omega_i \leq t \leq \omega_{i+1}} \times \delta_{\{i \equiv 0[2]\}} \right), \quad M_i \sim F_a(u) \tag{3.5}$$

A path sample is shown in Fig. 3.6. It is also worth mentioning that the RAIM

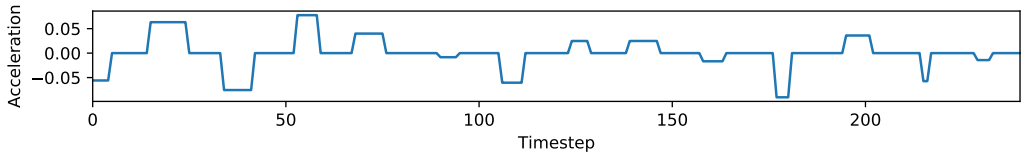


Figure 3.6: One sample path of A_t with renewal intervals $I_{0,k}$ given by $\Gamma(\alpha = 5, \beta = 0.5)$, $I_{1,k}$ given by $\Gamma(\alpha = 30, \beta = 0.5)$ and $X_k \sim \mathcal{N}(0, 0.05^2)$.

model is a particular case of the more general Piecewise Markov Process and the Markov Renewal Process. Other interesting properties, including expended and remaining times in a segment, can be further found in [71].

3.2.2 Random Yawing Impulses Model (RYIM)

We now shift our focus onto the lateral component of the trajectory. As before, we assume that the set of lateral trajectories $\{W_t^{(i)}\}_{i=1}^{N_v}$ are independent path samples of a stochastic process Z_t. The RYIM model assumes:

$$Z_t = \mathbb{E}[Z_t] + \int_{s=0}^{s=t} B_s \mathrm{d}s. \tag{3.6}$$

Introduce the coordinate system as depicted in Fig. 3.7(a), where Oy is aligned with the centre marking. With a small yawing angle θ, it follows that $v_x \approx v\theta$ and $v_y \approx v$. The dynamics of θ can be attributed to the driver's steering wheel input, and this gives a physical interpretation to the RYIM model. As a vehicle navigates a road, small adjustments to the steering wheel by the driver keep the vehicle in its lane, leading to θ oscillating around zero. Occasionally, a significant shift in θ might be observed when the driver decides to change lanes.

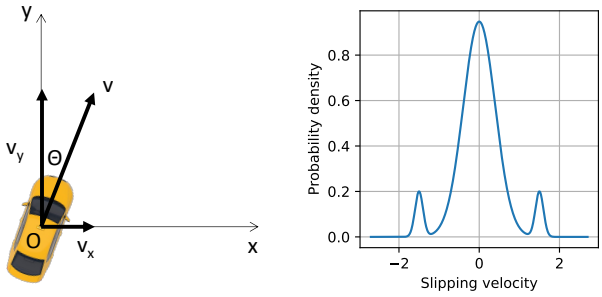
In comparison with the RAIM model, there are still 2 states $\{0, 1\}$ which correspond to lane-keeping and lane-changing respectively. The only difference is that Z_t only has one integration of the random impulses process rather than two, as shown in (3.3).

As above, the "slipping activity process" $\mathcal{S}_t$ is defined as:

$$\mathcal{S}_t = \sum_{i=1}^{+\infty} \left(\mathbf{1}_{\omega_i \le t \le \omega_{i+1}} \times \delta_{\{i \equiv 0[2]\}} \right). \tag{3.7}$$

The slipping impulse model is then defined as

$$B_t = \sum_{i=1}^{+\infty} R_i \times \mathbf{1}_{\omega_i \le t \le \omega_{i+1}} \times \delta_{\{i \equiv 0[2]\}} \quad \text{in which} \quad R_i \sim F_{lc}(u). \tag{3.8}$$



(a) (b)

Figure 3.7: (a) Illustration of the yawing angle θ, the lateral velocity v_x and the longitudinal velocity v_y. (b) Probability density of the Slipping Velocity Distribution with three modes indicating changing to the left lane, lane keeping and changing to the right lane.

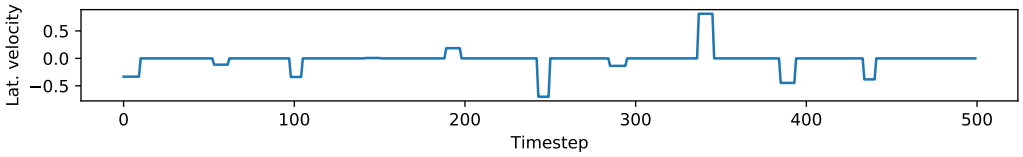


Figure 3.8: The slipping impulses generated by the process B_t defined in (3.8) with renewal intervals $I_{0,k}$ given by $\Gamma(\alpha = 30, \beta = 0.25)$, $I_{1,k}$ given by $\Gamma(\alpha = 150, \beta = 0.25)$ and $R_i \sim \mathcal{N}(0, 0.5^2)$.

The distribution F_{lc} dictates the behavior of drivers in terms of lane-keeping and lane-changing, while the distributions $I_{\cdot,k}$ manage the frequency and reaction rates of drivers. The sample paths of the slipping activity process, denoted as $\mathcal{S}_t$, are shown in Fig. 3.5, and of B_t are presented in Fig. 3.8, with its corresponding F_{lc} showcased in Fig. 3.7(b).

3.3 VALIDATION OF THE MODEL THROUGH COMPUTATION OF THE COVARIANCE KERNEL AND ITS EIGENFUNCTIONS

The RAIM and RYIM models operate on the assumption that the residual components of the trajectory time series, once the mean trajectories have been subtracted, originate from a Markov process. However, the validity of such assumptions still needs verification. To address this, we analytically derive the covariance kernel of the processes X_t, Z_t and the corresponding eigenfunctions. The stationary process assumption could be deemed valid if the convergence of the covariance function can be established via the law of large numbers.

3.3.1 RYIM Model's Covariance Kernel and Eigenfunctions

We study the time integration of B_t in Eq. (3.8). Analysis for the acceleration process (3.5) should be similar.

$$B_t = \sum_{i=1}^{+\infty} R_i \times \mathbf{1}_{\omega_i \le t \le \omega_{i+1}} \times \delta_{\{i \equiv 0[2]\}}. \tag{3.9}$$

Proposition 3.1. The covariance kernel for the process Z_t defined in (3.6) converges to the covariance kernel for the Wiener process if the regeneration time $I_0 + I_1$ goes to zero.

Proof. The RYIM model assumes:

$$\begin{aligned} Z_t = &\quad \mathbb{E}[Z_t] + \int_0^t B_s \mathrm{d}s \\ = &\begin{cases} \mathbb{E}[Z_t] + \sum\limits_{1 \le k \le \frac{i}{2}} M_k I_{1,k} \quad \text{if } \omega_i \le t \le \omega_{i+1} \quad \text{and} \\ i \text{ is even;} \\ \mathbb{E}[Z_t] + \sum\limits_{1 \le k \le \frac{i-1}{2}} M_k I_{1,k} + M_{\frac{i+1}{2}}(t - \omega_i) \quad \text{if} \\ \omega_i \le t \le \omega_{i+1} \text{ and } i \text{ is odd.} \end{cases} \end{aligned}$$

We define $\overline{Z_t}, \underline{Z_t}$ as the almost sure upper and lower bounds of Z_t respectively:

$$\underline{Z_t} \le Z_t \le \overline{Z_t} \quad \text{almost surely.}$$

where

$$\underline{Z_t} = \mathbb{E}[Z_t] + \sum_{1 \le k \le \lfloor \frac{i}{2} \rfloor} M_k I_{1,k} \quad \text{if} \quad \omega_i \le t \le \omega_{i+1},$$

and

$$\overline{Z_t} = \mathbb{E}[Z_t] + \sum_{1 \le k \le \lfloor \frac{i+1}{2} \rfloor} M_k I_{1,k} \quad \text{if} \quad \omega_i \le t \le \omega_{i+1}.$$

We define $F_\pi(u)$ as the product distribution of $F_{lc}(u)$ and $F_k(u)$. Obviously $F_\pi(u)$ is stationary if the component distributions are stationary. Let $Q_k = M_k I_{1,k}$ then $Q_k \sim F_\pi(u)$. The lower bound process could be rewritten into a more suggestive form:

$$\underline{Z_t} = \mathbb{E}[Z_t] + \sum_{1 \le k \le \zeta_t} Q_k, \tag{3.10}$$

where ζ_t is the the number of regenerations before t of the process $I_0 + I_1 \triangleq \{I_{1,k} + I_{0,k}\}_k$. In other words, the process bounds are essentially renewal-reward processes, whose reward function is Q_k, and the holding times $I_{1,k} + I_{0,k}$ indicate the time between two consecutive renewals into state 1. Let $\rho_k \triangleq \omega_{2k+1}$, then $\{\rho_1, \rho_2, \dots\}$ are the times when $I_0 + I_1$ regenerates. From the Key Renewal Theorem

$$\mathbb{E}[\zeta_{\tau+h}] - \mathbb{E}[\zeta_\tau] \to \frac{h}{\mu} \text{ as } \tau \to +\infty, \tag{3.11}$$

where

$$\mu = \mathbb{E}[I_0] + \mathbb{E}[I_1].$$

Equation (3.11) implicitly assumes that the process has been running *long enough*. We also assume that the longer the run time is, the more renewals into state 1 there will be, with probability one:

$$\zeta_t < \zeta_s \Rightarrow t < s \quad \text{a.s.} \tag{3.12}$$

Let $t = \tau + t', s = \tau + s'$, we can calculate the covariance function of $\underline{Z_t}$:

$$\begin{aligned}
\underline{\mathrm{Cov}}(t,s) &= \mathbb{E}[(\underline{Z_t} - \mathbb{E}[\underline{Z_t}])(\underline{Z_s} - \mathbb{E}[\underline{Z_s}])] \\
&= \mathbb{E}[\mathbb{E}[(\underline{Z_t} - \mathbb{E}[\underline{Z_t}])(\underline{Z_s} - \mathbb{E}[\underline{Z_s}])]|\zeta_t, \zeta_s] \\
&= \mathbb{E}\left[\sum_{1\le i\le\zeta_t}\sum_{1\le j\le\zeta_s} \mathbb{E}[(Q_i - \mathbb{E}[Q_i])(Q_j - \mathbb{E}[Q_j])]\right] \\
&= \mathbb{E}\left[\min(\zeta_t, \zeta_s)\,\mathrm{Var}[Q_i]\right] \\
&= \mathbb{E}\left[\min(\zeta_{\tau+t'}, \zeta_{\tau+s'})\,\mathrm{Var}[Q_i]\right] \\
&\to \frac{1}{\mu}\min(\rho_{\zeta_t}, \rho_{\zeta_s})\,\mathrm{Var}[Q_i]
\end{aligned} \tag{3.13}$$

where we have used the law of total probability to go from the second line to the third line, independence of Q_i to go from the third line to the fourth, and (3.11) to go to the final line if $t < s$ implies $\zeta_t < \zeta_s$ (3.12).

By definition, $\rho_{\zeta_{t'}} = t'$, $\underline{\mathrm{Cov}}(t,s) = (1/\mu)\min(t,s)\,\mathrm{Var}[Q_i]$. Similarly for $\overline{Z_t}$:

$$\begin{aligned}
&\overline{\mathrm{Cov}}\,(t,s) = \mathbb{E}[(\overline{Z_t} - \mathbb{E}[\overline{Z_t}])(\overline{Z_s} - \mathbb{E}[\overline{Z_s}] \\
=\ &\mathbb{E}[\mathbb{E}[(\overline{Z_t} - \mathbb{E}[\overline{Z_t}])(\overline{Z_s} - \mathbb{E}[\overline{Z_s}])]|\zeta_t, \zeta_s] \\
=\ &\mathbb{E}\left[\sum_{1\le i\le\zeta_t+1}\sum_{1\le j\le\zeta_s+1} \mathbb{E}[(Q_i - \mathbb{E}[Q_i])(Q_j - \mathbb{E}[Q_j])]\right] \\
=\ &\mathbb{E}\left[\min(\zeta_t + 1, \zeta_s + 1)\,\mathrm{Var}[Q_i]\right] \\
=\ &\mathbb{E}\left[\min(\zeta_{\tau+t'} + 1, \zeta_{\tau+s'} + 1)\,\mathrm{Var}[Q_i]\right] \\
\to\ &\frac{1}{\mu}\min(\rho_{\zeta_t+\mu}, \rho_{\zeta_s+\mu})\,\mathrm{Var}[Q_i]
\end{aligned} \tag{3.14}$$

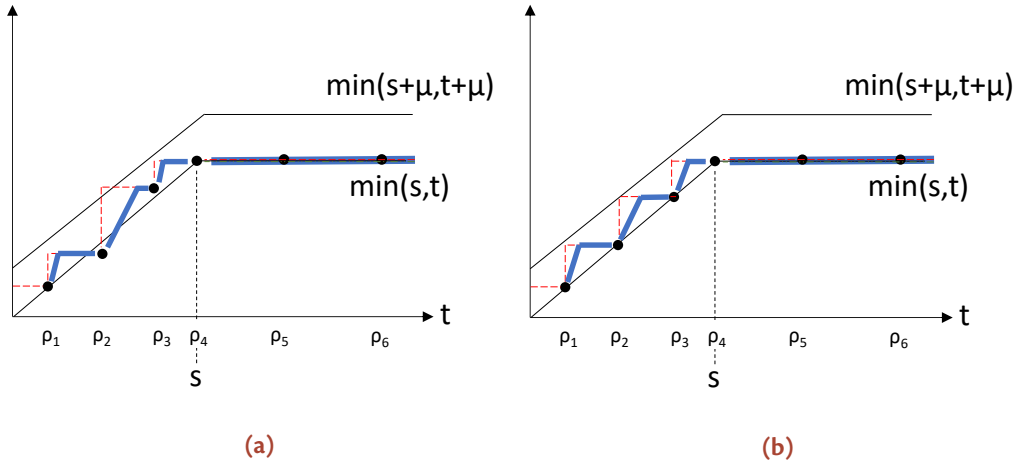


Figure 3.9: (a) depicts one instance of $\mu \min(\zeta_t, \zeta_s) \operatorname{Var}[Q_i]|\zeta_t, \zeta_s$, with s fixed, as the blue graph. The black dots are regeneration points. The expectation over all ζ_t, ζ_s, when $\tau \to +\infty$ gives the covariance function shown on (b). The lower bound of the process coincides with $\min(s, t)$, while the upper bound coincides with $\min(s + \mu, t + \mu)$. Note the relative position of the regeneration points with respect to the two bounds $\min(s, t)$ and $\min(s + \mu, t + \mu)$.

and we also have $\overline{\operatorname{Cov}}\,(t, s) = (1/\mu) \min(t + \mu, s + \mu) \operatorname{Var}[Q_i]$. Figure 3.9 shows the covariance functions' upper and lower bounds. Obviously, if $I_0 + I_1$ regenerates fast enough, $\mu \to 0$ and we can approximate:

$$\operatorname{Cov}[(t, s)] \approx \frac{1}{\mu} \min(t, s) \operatorname{Var}[Q_i]. \tag{3.15}$$

□

Because the covariance kernel's eigenfunctions are also the KL basis [72], we will use the terms interchangeably.

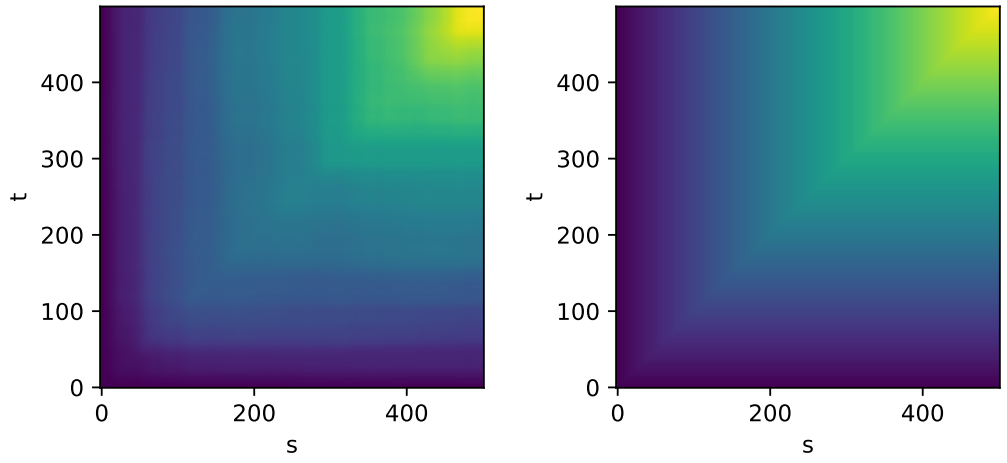
Corollary 3.2. The KL decomposition for Z_t is also the same as the Wiener process and is given by:

$$Z_t = \mathbb{E}[Z_t] + \sum_{k=1}^{+\infty} \beta_k \sqrt{2} \sin\left(\left(k - \frac{1}{2}\right) \pi t\right). \tag{3.16}$$

where β_k are normally distributed Gaussian random variables.

Proof. See [73]. □

The validity of the model is confirmed by Fig. 3.10. The similar structure of the covariance kernel attests to the assumptions proposed in Proposition 3.1, which implies that the process may be considered Markovian with impulses being independent and identically distributed samples from a stationary distribution.



(a) Observed **(b)** Prediction

Figure 3.10: (a) Computed covariance kernel from the NGSIM dataset (b) Predicted covariance kernel from the RYIM model.

The eigenfunctions shown in Fig. 3.12, exhibiting a harmonic shape, also lend weight to this finding. However, the wobbly patterns indicate that the conditions for convergence are not ideal. This is also illustrated in Fig. 3.11, which displays covariance kernels associated with larger idle times between regenerations. If the regeneration times are large, the kernels will appear more "pixelated".

3.3.2 *RAIM Model's Covariance Kernel and Eigenfunctions*

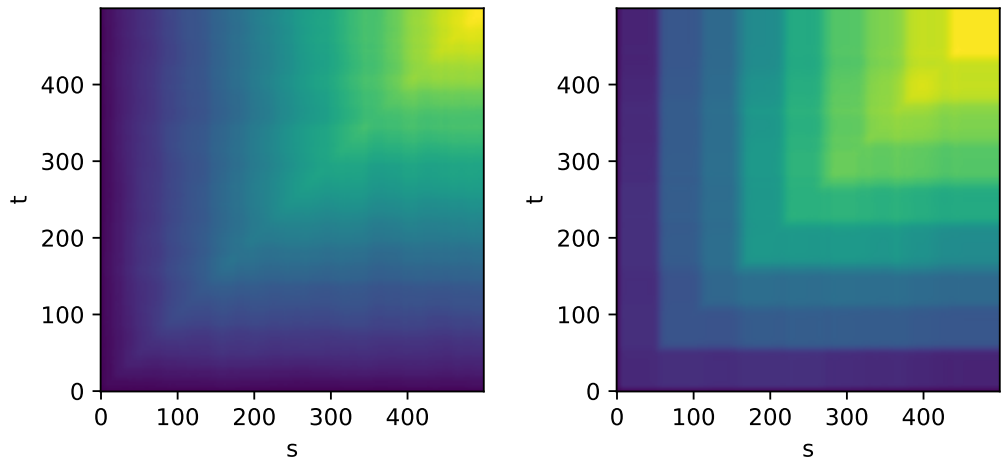
The derivation of the covariance kernel and eigenfunctions of RAIM is more involved.

Proposition 3.3. Under the assumptions made in Proposition 3.1, the covariance kernel eigenfunctions are given by:

$$\phi(t) = \sum_{k=1}^{+\infty} \frac{c_k}{\pi^2(i-1/2)^2} e_k(t).$$

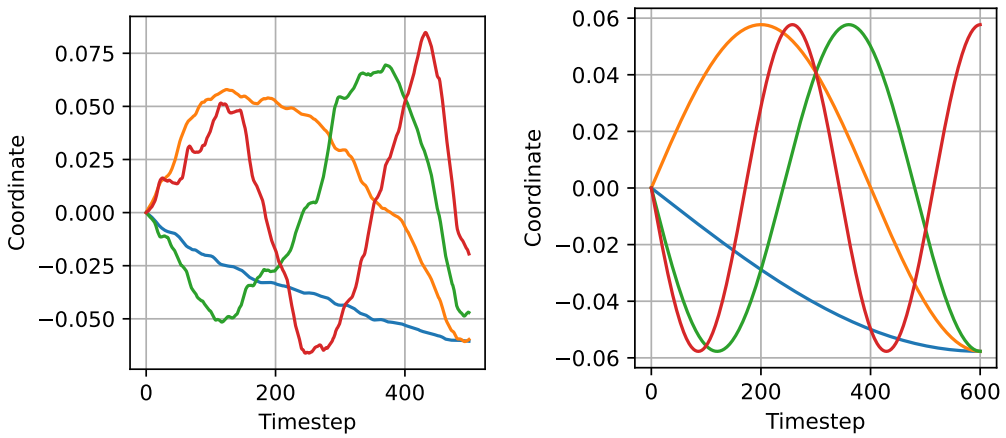
and the covariance kernel can be computed from:

$$\begin{aligned} R(s,t) &= \mathbb{E}[(X_t - \mathbb{E}[X_t])(X_s - \mathbb{E}[X_s])] \\ &= \sum_{k=1}^{+\infty} \sum_{l=1}^{+\infty} \frac{\mathbb{E}[\beta_k \beta_l]}{(k-1/2)^2 \pi^2} e_k(t) e_l(s). \end{aligned}$$



(a) (b)

Figure 3.11: Prediction of covariance kernel by RYIM model with different idle time distributions: (a) $\Gamma(\alpha = 75, \beta = 0.05)$, (b) $\Gamma(\alpha = 175, \beta = 0.25)$.



(a) Observed **(b)** Prediction

Figure 3.12: (a) KL basis obtained by numerically eigendecomposing the empirical covariance kernel (b) KL basis predicted by the RYIM model. Functions that are inverted vertically are the same since the corresponding coefficients can be replaced with an opposite sign.

Proof. Consider the RAIM given by Eq. (3.3):

$$X_t = \mathbb{E}[X_t] + \int_{s=0}^{t} \int_{u=0}^{s} (A_u \mathrm{d}u)\, \mathrm{d}s. \tag{3.17}$$

Restricting to $t \in [0,1]$, and since A_t and B_t are similar, we can reuse Equation (3.16):

$$X_t = \mathbb{E}[X_t] + \int_0^1 \left(Z_0 + \sum_{k=1}^{+\infty} \beta_k \sqrt{2} \sin\left(\left(k - \frac{1}{2}\right)\pi t\right)\right) \mathrm{d}t. \tag{3.18}$$

Without any loss of generality, we can assume $Z_0 = 0$. A direct integration gives:

$$X_t = \mathbb{E}[X_t] + \sum_{k=1}^{+\infty} \frac{-\beta_k}{k - \frac{1}{2}\pi} \sqrt{2} \left(\cos\left(\left(k - \frac{1}{2}\right)\pi t\right) - 1\right). \tag{3.19}$$

While (3.19) is a series expansion of X_t, it is far from being the eigenfunctions (or the KLT). This is clear from the fact that $\{\cos((k-1/2)\pi t) - 1\}_k$ do not even form an orthogonal basis. To avoid cluttering of notations, we denote:

$$e_k(t) = \sqrt{2}\left(\cos\left(\left(k - \frac{1}{2}\right)\pi t\right) - 1\right).$$

Then, (3.19) can be rewritten as:

$$X_t = \mathbb{E}[X_t] + \sum_{k=1}^{+\infty} \frac{-\beta_k}{k - \frac{1}{2}\pi} e_k(t).$$

The covariance function can be calculated directly as:

$$R(s,t) = \mathbb{E}[(X_t - \mathbb{E}[X_t])(X_s - \mathbb{E}[X_s])] \tag{3.20}$$

$$= \sum_{k=1}^{+\infty} \sum_{l=1}^{+\infty} \frac{\mathbb{E}[\beta_k \beta_l]}{(k-1/2)^2 \pi^2} e_k(t) e_l(s), \tag{3.21}$$

and independence of β_k leads to:

$$R(s,t) = \sum_{k=1}^{+\infty} \frac{\mathbb{E}[\beta_k^2]}{(k-1/2)^2 \pi^2} e_k(t) e_k(s).$$

Using (3.16), we can further simplify the equation to:

$$R(s,t) = \sum_{k=1}^{+\infty} \frac{1}{(k-1/2)^4 \pi^4} e_k(t) e_k(s). \tag{3.22}$$

The KL basis is solution to the Fredholm equation:

$$\int_{t=0}^{1} R(s,t)\phi(t)\mathrm{d}t = \lambda\phi(s), \tag{3.23}$$

which is a special case of the Fredholm integral equation of the second kind with separable kernel. In general, there are infinitely many solutions corresponding to the zero eigenvalue of the integral operator of kernel $R(s,t)$. But it is of our interest to find solutions with the largest positive eigenvalues which are written under the form:

$$\phi(t) = \sum_{k=1}^{+\infty} \frac{c_k}{\pi^2(k-1/2)^2} e_k(t). \tag{3.24}$$

Substitution into (3.23) gives:

$$\begin{aligned}
\lambda\phi(s) &= \int_{t=0}^{1} \left(\sum_{k=1}^{+\infty} \frac{1}{(k-1/2)^4\pi^4} e_k(s)e_k(t) \right) \\
&\left(\sum_{j=1}^{+\infty} \frac{c_j}{\pi^2(j-1/2)^2} e_j(t) \right) \mathrm{d}t \\
&= \sum_{k=1}^{+\infty}\sum_{j=1}^{+\infty} \frac{e_k(s)}{(k-1/2)^2\pi^2} c_j \int \frac{e_k(s)}{(k-1/2)^2\pi^2} \frac{e_j(s)}{(j-1/2)^2\pi^2}\mathrm{d}t.
\end{aligned}$$

Let:

$$\begin{aligned}
A_{kj} &= \int \frac{e_k(s)}{(k-1/2)^2\pi^2} \frac{e_j(s)}{(j-1/2)^2\pi^2}\mathrm{d}s \\
&\triangleq \frac{e_k(s)}{(k-1/2)^2\pi^2} \frac{e_j(s)}{(j-1/2)^2\pi^2}.
\end{aligned}$$

Then:

$$\lambda \sum_{k=1}^{+\infty} \frac{c_k}{\pi^2(k-1/2)^2} e_k(t) = \sum_{k=1}^{+\infty}\sum_{j=1}^{+\infty} A_{kj} \frac{e_k(s)}{(k-1/2)^2\pi^2} c_j. \tag{3.25}$$

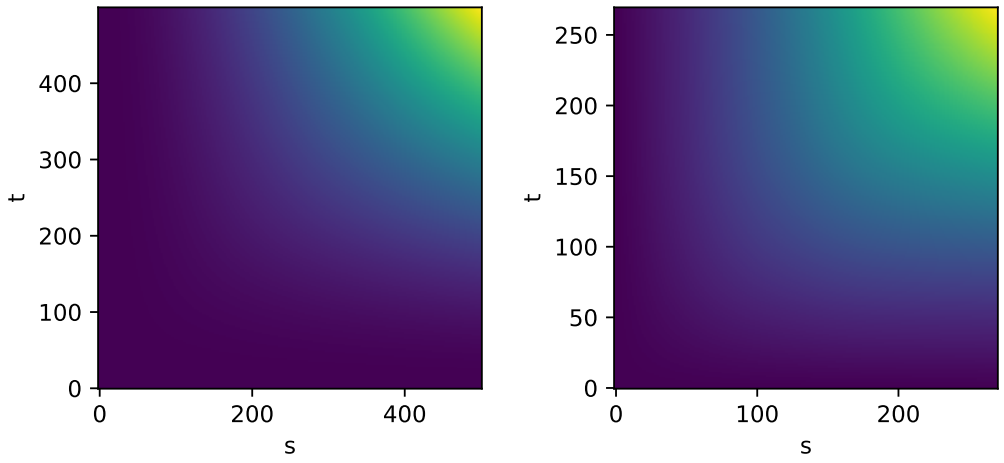
Because $e_k(t) \neq 0$, we can match the coefficients of $e_k(t)$:

$$Ac = \lambda c, \tag{3.26}$$

where $A = [A_{kj}], c = [c_1, c_2, \dots]^\top$. Hence, an eigendecomposition of A should reveal the eigenvectors of coefficients c and the KLT basis is given by (3.24). The coefficients of the first 3 KL basis functions are given in Table 3.1. □

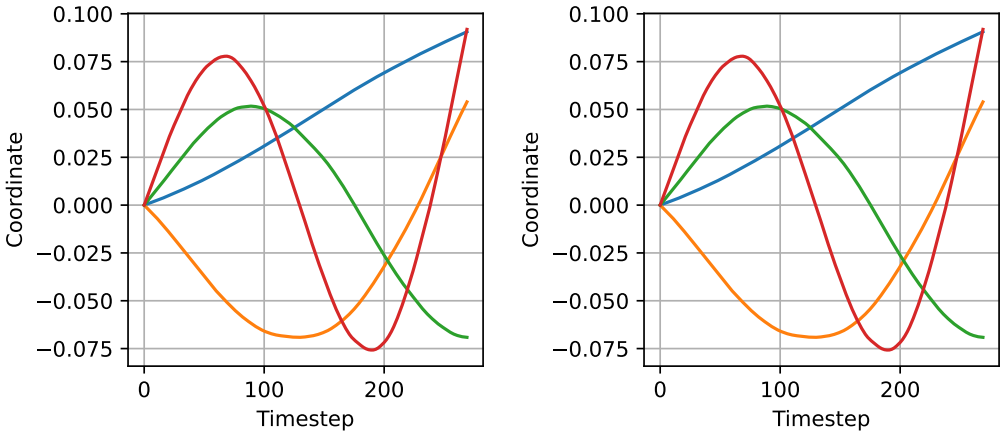
Table 3.1: 7 first coefficients of the first 3 KL basis to be used in Equation (3.24).

	c_1	c_2	c_3	c_4	c_5	c_6	c_7
ϕ_1	-9.99e-01	-3.21e-02	-1.78e-03	-8.45e-04	-2.05e-04	-1.28e-04	-5.03e-05
ϕ_2	-9.42e-01	3.30e-01	5.48e-02	4.71e-03	3.85e-03	9.49e-04	8.11e-04
ϕ_3	-0.72	0.55	-0.39	-0.10	-0.01	-0.01	0



(a) Prediction **(b)** Observed

Figure 3.13: The covariance kernels from (a) RAIM and (b) NGSIM dataset.



(a) Prediction **(b)** Observed

Figure 3.14: (a) KL basis obtained from RAIM (b) KL basis obtained numerically from NGSIM dataset. Note that some components in (a) and (b) were inverted, but they are the same since we can invert the sign of the corresponding score component.

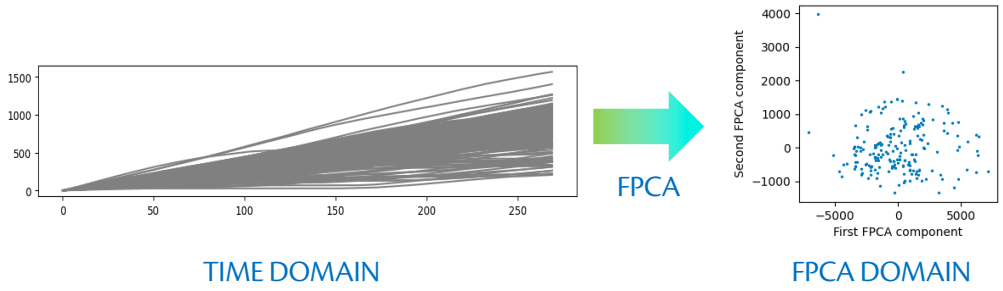


Figure 3.15: In the decomposition of trajectories using FPCA, each original time series is treated as a functional—equivalent to a $\mathbb{R}^{150}$ vector in the discrete domain—as depicted by a curve in the left graph. Each of these curves is subsequently transformed into a 2D point, which can be thought of as equivalent to a $\mathbb{R}^2$ vector. The horizontal axis is β_1, and the vertical axis is β_2.

The striking similarity between the covariance kernels predicted by RAIM and those numerically computed from the datasets is readily observable (Fig. 3.13). Their eigenfunctions also display a strong similarity (Fig. 3.14). These observations provide robust support for the assumptions presented in Proposition 3.1, but to a significantly greater degree than the lateral component.

3.4 FUNCTIONAL PCA (FPCA) APPLICATIONS

Dimensionality reduction, a cornerstone of machine learning, promises innovative solutions in predictive modeling, anomaly detection, and classification tasks. Among these methods, PCA stands out. Extensively utilized in fields like biology and engineering, PCA effectively mitigates the curse of dimensionality, enhancing the performance of inference methods in low-dimensional space.

It turns out that the eigenfunctions introduced in Eq. (3.16) and (3.24) are also the basis functions required to perform dimensionality reduction (Fig. 3.15) [16, 74]. In particular, let $f(t)$ be the time series that is either the longitudinal or lateral component of the trajectory, and $e_k(t)$ representing the k-th eigenfunction denoted in (3.16). Then:

$$\beta_k = \int_t e_k(t) f(t) \mathrm{d}t \tag{3.27}$$

Then, given some $N \in \mathbb{N}^*$, $f(t) \in \mathbb{R}^\infty$ can be approximated by a *truncated series*:

$$f(t) \approx \tilde{f}_N(t) \triangleq \sum_{k=1}^{N} \beta_k e_k(t), \tag{3.28}$$

in other words, $f(t)$ can now be represented by a vector $\beta = [\beta_1, \ldots, \beta_N]$ with a finite dimension of N.

We illustrate multiple applications of this dimensionality reduction method, encompassing data compression, identifying anomalies, and completing trajectories, providing comparisons whenever feasible.

3.4.1 Dimensionality Reduction

Definition 3.4 (N-terms basis restriction error and Normalized mean Squared Error). Let the *N-terms approximated functional* of $f(t)$ be given by (3.28), we define the **N-terms basis restriction error as:**

$$\epsilon_N = \int_{t=-\infty}^{+\infty} \left(f(t) - \tilde{f}_N(t) \right)^2 \mathrm{d}t, \tag{3.29}$$

and the related *normalized mean squared error*:

$$\tilde{\epsilon}_N = \frac{\epsilon_N}{\int_{-\infty}^{+\infty} f(t)^2 \mathrm{d}t} \tag{3.30}$$

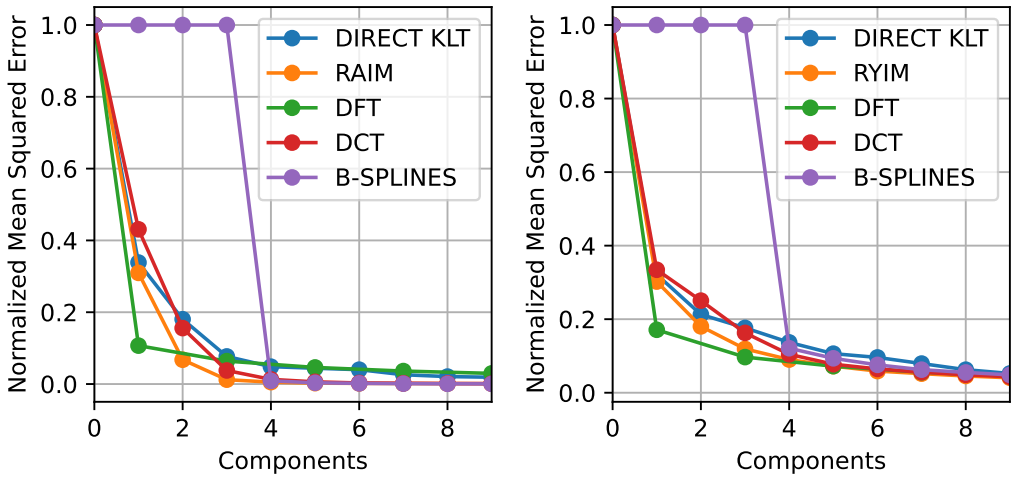
Then, it has been shown in [75] that the FPCA decomposition whose score coefficients given by (3.27) minimize the N-terms basis restriction error.

Typically, in the traditional FPCA workflow, the basis functions must be computed from a sample set, referred to as the *learning dataset*, and then applied to carry out the decomposition. However, according to Proposition 3.1, these basis functions remain invariant provided the prescribed assumptions are maintained.

To demonstrate the approximation power, we compare the normalized mean squared error when performing the decomposition using the analytical basis functions (KLT transform) provided by (3.16) and (3.24), the basis functions computed numerically from the learning set, and other harmonic linear transforms such as Discrete Fourier Transform (DFT) and Discrete Cosine Transform (DCT). We also include the Cubic Splines Basis for reference due to its popularity, even though B-spline Transform is not typically considered as a linear transform.

The remarkable efficacy of the RAIM and RYIM models is clearly demonstrated in Fig. 3.16, where they exhibit the quickest error convergence for the longitudinal component of the trajectories. Moreover, they rank second for the speediest convergence on the lateral component.

Tables 3.2 and 3.3 present a more detailed comparison of the performance of the KLT, as predicted by the RAIM and RYIM models, against other harmonic transforms. This includes the numerical KLT, DFT, and DCT. It is clear that RAIM not only delivers better performance in terms of approximation error, but also exhibits consistency across all trajectories. This consistency is reflected in the smaller standard deviations for all values, including the number of components, as well as the maximum and minimum squared error. For RYIM model, DFT is



(a) Longitudinal (b) Lateral

Figure 3.16: Average N-terms basis restriction error of longitudinal component (a) and lateral component (b). RAIM and RYIM symbolize the predicted basis functions from the RAIM and RYIM models respectively.

quicker to converge and also more consistent than RYIM's eigenfunctions. This is thought to be attributed to the non-compliance of the ideal convergence criteria of Proposition 3.1.

It's noteworthy to mention that this error convergence rate was even faster than when using the empirical basis numerically computed from the covariance kernel. This phenomenon can be linked to the inherent numerical errors and variances resulting from working with a sampled training set. However, it's anticipated that this discrepancy will diminish as more data is incorporated into the training set.

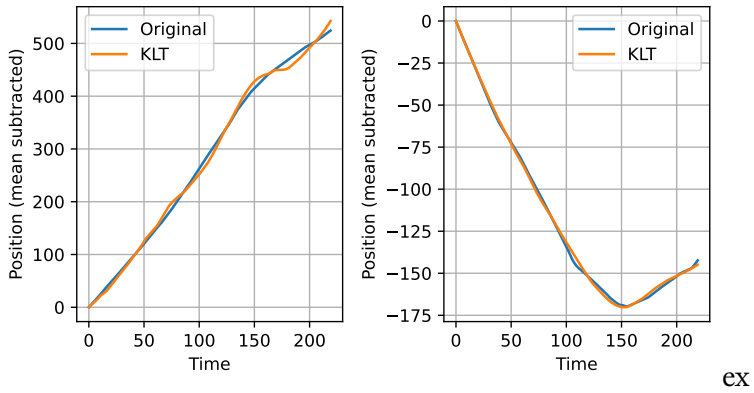
Figures 3.17 and 3.18 demonstrate the approximation of trajectories using only nine FPCA scores. It is immediately obvious that FPCA is comparable to the

Table 3.2: Approximation power of various harmonic transforms for longitudinal component of vehicle trajectories. Values for AVG, STD, MAX, MIN are in normalized square error. ***Abbreviations:*** *N*: number of components, D. KLT: direct KLT.

	AVG			STD			MAX			MIN		
N	3	6	9	3	6	9	3	6	9	3	6	9
RAIM	**0.05**	**0.00**	**0.00**	0.10	**0.00**	**0.00**	**0.55**	**0.01**	**0.00**	0.00	0.00	0.00
D. KLT	0.17	0.05	0.02	0.15	0.03	0.02	0.60	0.14	0.08	0.00	0.00	0.00
DFT	0.07	0.03	0.02	0.05	0.02	0.01	0.18	0.08	0.05	0.00	0.00	0.00
DCT	0.14	0.01	0.00	0.19	0.01	0.00	0.82	0.07	0.01	0.00	0.00	0.00

Table 3.3: Approximation power of various harmonic transforms for lateral component of vehicle trajectories. Values for AVG, STD, MAX, MIN are in normalized square error. ***Abbreviations:*** *N*: number of components, D. KLT: direct KLT.

	AVG			STD			MAX			MIN		
N	3	6	9	3	6	9	3	6	9	3	6	9
RYIM	0.15	0.06	0.04	0.17	0.08	0.07	0.80	0.54	0.50	0.02	0.01	0.00
D. KLT	0.20	0.10	0.06	0.19	0.10	0.07	0.85	0.63	0.38	0.03	0.03	0.01
DFT	**0.09**	**0.04**	**0.02**	**0.10**	**0.05**	**0.03**	**0.61**	**0.26**	**0.18**	0.02	0.01	0.00
DCT	0.23	0.07	0.05	0.17	0.09	0.07	1.00	0.63	0.53	0.03	0.01	0.00



ex

Figure 3.17: Approximation with 9 components of the longitudinal component of two sample trajectories. The mean component was subtracted for better visualization.

role of DFT and DCT in image processing where the truncated basis similarly reveals a denoising ability. There's a direct relationship between the number of components used in the transformation and the smoothness of the trajectory - fewer components result in smoother trajectories.

3.4.2 Data Compression

Drawing parallels with renowned digital data formats such as MP3 and JPEG, the KLT carries strong potential akin to the DCT for data compression. This potential is particularly important in the context of vehicular network. For instance, instead of transmitting vehicle coordinates for each of the 150 time steps, it becomes feasible to transmit only 2-3 FPCA score coefficients, resulting in a far more efficient scheme. Details can further be found in our previous work [76], where we successfully demonstrated the implementation of such transform in devising a data compression strategy for the Cooperative Awareness Message (CAM) transmission within the V2X network setting.

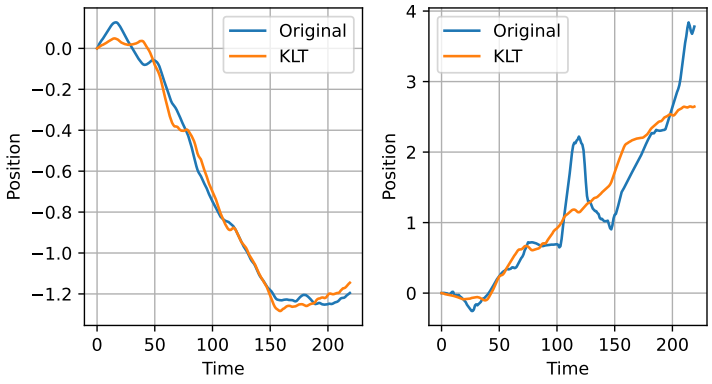


Figure 3.18: Approximation with 9 components of the lateral component of two sample trajectories.

The earlier implementation, however, has some by practical challenges. Specifically, it required continuous collection of trajectories in the learning set and recurrent computation of the basis functions. This chapter addresses these challenges and offers a significant improvement: we demonstrate that a single, unique basis set can be used, which simplifies the process considerably. This development makes the approach more amenable for hard coding on embedded platforms.

3.4.3 Trajectory Prediction and Completion

FPCA simplifies the dataset by reducing its dimensions, thereby facilitating a more intuitive and straightforward inference process. To this, we propose a Bayesian inference framework as an example. This framework is designed to "fill in the blanks" of the unobserved part of the longitudinal trajectory, using partial observations obtained up to a specific timestep.

Formally, let $y_{1:n}$ be the set of n noisy observations identically and independently obtained from a vehicle's trajectory whose measurement model is $y = \hat{y} + \epsilon_y$ where $\epsilon_y \sim \mathcal{N}(0, \sigma_y^2)$ and $\mathcal{D} = \{y^i_{1:q}; 1 \leq i \leq m\}$ be the training set of m trajectories whose length is $q > n$. We would like to find the best estimate for $y_{n+1:q}$. Furthermore, let $a \in \mathbb{R}^N$ be the corresponding FPCA score vector of the trajectory $y_{1:q}$ computed from (3.27).

The general idea is to fit a distribution (presumably Gaussian in this example) on the FPCA score vectors corresponding to the trajectories of $\mathcal{D}$, giving us the *prior distribution* $p(a) = \mathcal{N}(0, \Sigma_a)$. Then, we compute the posterior (Fig. 3.19).

$$p(a|y_{1:n}) = \frac{p(y_{1:n}|a)p(a)}{p(y_{1:n})} \propto_a \prod_{i=1}^{n} p(y_i|a)p(a) \tag{3.31}$$

In the context that all distributions are Gaussians, the maximum-a-posteriori coincides with the minimum mean square (MMSE) estimate of a, is obtained from Algorithm 3.1 Figure 3.20 provides an example of the completion of the trajec-

Algorithm 3.1: Trajectory Representation Estimation

$\phi \leftarrow [e_k[0], 1 \leq k \leq q]$
$\mu' \leftarrow 0_q \triangleq [0, 0, \ldots, 0]$ (q zeroes)
$\Sigma' \leftarrow \Sigma_a$
for $i = 1$ to n **do**
    $\Sigma' \leftarrow (\Sigma_a^{-1} + 1/\sigma_y^2(\phi_i \otimes \phi_i))^{-1}$
    $\mu' \leftarrow \mu' + 1/\sigma_y^2(y_i - \mu_i)\phi_i^\top$
    $a^* \leftarrow \Sigma'\mu'$
end for

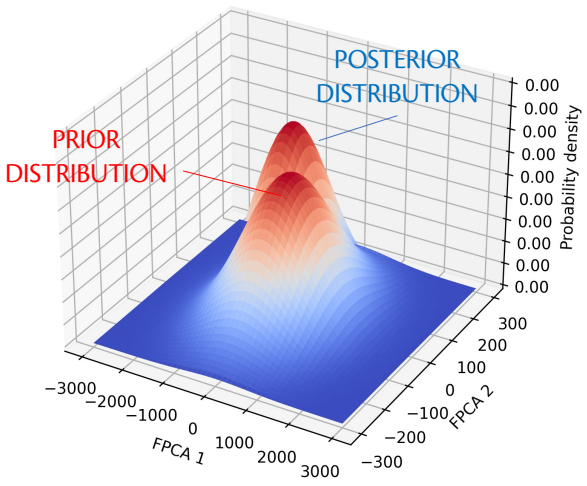


Figure 3.19: Illustration of the estimation process: the observations provide information to "narrow down" the distribution of the value of a in the FPCA domain, which explains the more "pointy" look of the posterior distribution.

tory's longitudinal component. Remarkably, Fig. 3.21 reveals that the prediction error remains within acceptable bounds (with error tolerance ϵ=5ft) for as many as 90 timesteps, equivalent to 9 seconds. This is in contrast to the average of only 20 timesteps predicted by the Constant Velocity model. It's also noteworthy that our FPCA-based method begins to exhibit divergence in error from timestep 100 onwards. We attribute this divergence to the finiteness of the time series, a phenomenon similar to performing a Fourier Transform on a windowed signal.

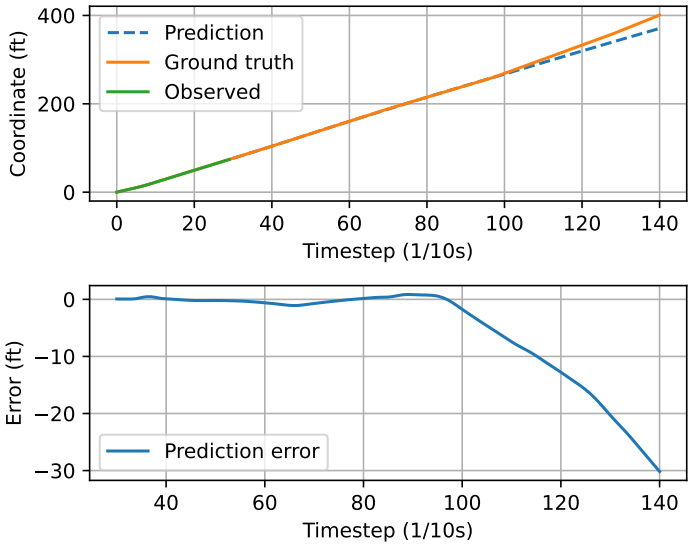
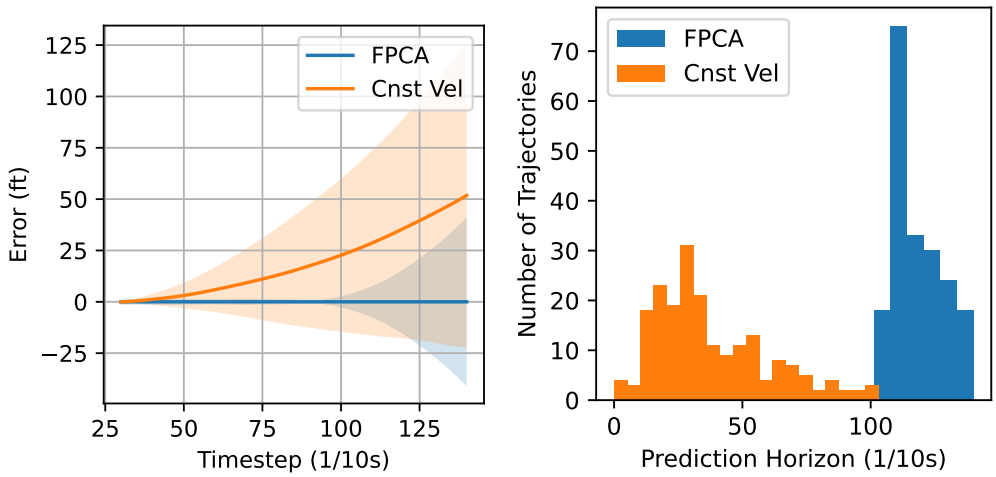
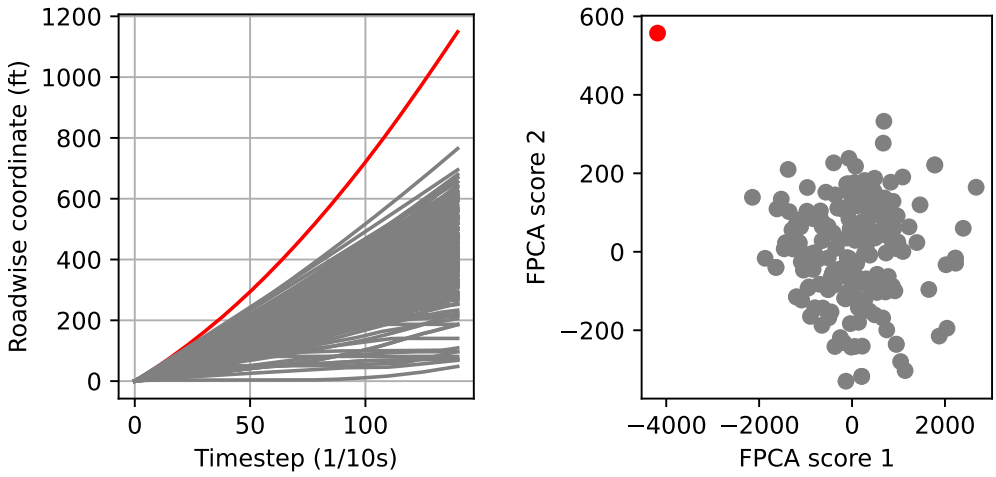


Figure 3.20: The trajectory completion is illustrated for observations up to timestep 30. The upper graph presents the completed trajectory, while the lower one displays the error between the actual and predicted trajectories.



(a) Prediction Error **(b)** Prediction Horizon with $\epsilon = 5$ft

Figure 3.21: A comparison was conducted between our Random Impulses FPCA estimate and the Constant Velocity models. This included the mean and standard deviation of the prediction error (a), and a histogram of the prediction horizon (b) for 100 test trajectories sampled from the NGSIM dataset. The prediction error tolerance was set at ϵ=5ft, and 30 timesteps of observations were made available for prediction.



(a) (b)

Figure 3.22: (a) The speeding vehicle's longitudinal component (b) The corresponding FPCA scores. KNN could easily single out the FPCA representation of this speeding vehicle from the rest.

3.4.4 *Anomaly Detection*

Anomaly detection tasks are also straightforward as algorithms such as K-nearest-neighbor (KNN) and clustering can be effectively performed on this domain (Fig. 3.22). While this concept has been partially explored in previous studies like [30, 77], the utilization of the ***same*** basis functions for different traffic scenarios to decompose trajectories is a novel finding. This feature, akin to data compression applications, allows us to bypass the learning process, favoring a hard-coded transformation, making these results more impactful.

3.5 DISCUSSION

The validation of the Random Impulses Model demonstrates that the Markovian assumption regarding random acceleration and yawing impulses is reasonable, although the assumption is more valid for the longitudinal component than the lateral. The models further reveal that the discrepancy originates from the less frequent regeneration of yawing impulses compared to acceleration impulses.

The models also offer a more intuitive approximation of the underlying physical processes. Rather than modeling the driver's reactions as functions of various external factors, they represent cumulative distance-keeping behavior using a single component, namely the mean trajectory. The residual component reflects the

variations in inputs, leading to a significantly simpler model. However, despite this simplicity, the model still maintains the powerful characteristics seen in car-following models, such as a long prediction horizon.

With the validation of these models and the availability of covariance kernels, the applications of FPCA can be extended. Conventionally, FPCA requires the derivation of basis functions from a *training set*, raising the question of the *robustness* of these functions in relation to time and traffic conditions. This chapter introduced a novel finding from Proposition 3.1, asserting that the basis functions remain invariant under mild conditions of impulse regeneration time and amplitude samples from a stationary distribution. These assumptions appear valid for highway scenarios, and we speculate that they likely hold for most traffic scenarios, provided conditions do not rapidly change, such as with the sudden presence of reckless vehicles.

The implications of these findings are significant; the learning phase can be entirely bypassed, and the same basis functions can be hard-coded into signal processors. This can lead to considerable memory and cost savings for vehicular and ITS applications.

Further refinement of the models is necessary if multiple modes of traffic are available. Thirdly, akin to the Fourier Transform, it is crucial to account for the window effect on the time domain signal transform. This helps to prevent the divergence of error towards the end of the signal.

3.6 CONCLUSION

In conclusion, this chapter introduced a kinematic statistical model for road vehicle trajectory and investigated its potential in various applications. These include FPCA, as well as the approximated forms for probabilistic filtering and smoothing of trajectories derived from aerial video extraction.

The model's assumptions were validated through the computation of covariance kernels, while a convergence study illuminated the approximation power of the RYIM and RAIM models. The simplicity of these models, combined with the theoretical developments presented in this chapter, advocate for increased confidence in the use of the derived FPCA basis functions within ITS and vehicular embedded platforms.

Ultimately, this research not only contributes to the current understanding of vehicle trajectory modeling, but also paves the way for further exploration and application in this field.

4

Probabilistic Methods for Real-time Unsupervised Anomalous Trajectory Detection

Abstract

Anomaly detection plays a crucial role in numerous Intelligent Transportation Systems (ITS) applications, as unusual or anomalous behaviors often carry significant safety implications. Particularly in road traffic, it is of utmost importance to be able to detect anomaly in real-time so that alert about anomalous trajectories of surrounding vehicles can be issued soon after sufficient evidence to substantiate the possibility emerges.

This chapter presents a probabilistic framework including likelihood and Bayesian methods to enable real-time hypothesis testing for anomaly detection as observations accumulate. Operating in an unsupervised manner, the framework eliminates the need for explicit anomaly modeling. The time series data undergoes dimensional reduction through Functional Principal Component Analysis (FPCA) and is then fitted to obtain a prior distribution of FPCA scores. After this, the likelihood of time series observations stemming from this prior distribution is computed in real-time.

We conducted a series of numerical experiments to demonstrate the effectiveness and responsiveness of the proposed methods in detecting common road hazards such as wrong-way driving, overspeeding, and sudden hard-braking. The results showed successful detection of all presented anomalies and a rapid response rate, ranging from 40ms to 160ms delay. The framework produced a significantly lower rate of false alarms compared to the Local Outlier Factor (LOF) method. Additionally, the framework is capable of calibrating to accommodate different

sensor types and capabilities (Figure 4.1). It also demands low computational resources, making it readily implementable across various embedded Advanced Driver Assistance Systems (ADAS) platforms.

Contents

4.1 INTRODUCTION

With the roll-outs of radar and V2X equipped vehicles on the market, much traffic information is expected to be captured. It is presumed that this information, in addition to many other already available information sources such as security cameras and traffic crowd-sourcing applications, can deliver more significant benefits not only for traffic authorities but also for road participants thanks to the popularity of ADAS and autonomous driving systems [78] on newer vehicles. However, the availability of data does not necessarily translate to better safety because crucial information is often buried under an overabundance of irrelevant data. It is typically observed that accidents usually follow abnormal traffic behavior. Hence, identifying these situations are crucial to the success of any safety system.

A version of this chapter, titled "Probabilistic Methods for Real-time Unsupervised Anomalous Trajectory Detection", has been submitted for review to the *IEEE Transactions on Intelligent Transportation Systems* in 2021. The authors of this submission are Thinh Hoang, Vincent Martinez, Pierre Maréchal, and Daniel Delahaye.

Anomaly detection, a matured field of research, boasts a diversity of methodologies suitable for univariate and multivariate time series [79]. These strategies range from simple parametric models [80] to comprehensive deep learning frameworks [81], spanning supervised [82, 83], unsupervised [84, 85], and hybrid learning approaches. Such detection techniques are fundamental to ITS applications, including driver alert systems, collision avoidance, and autonomous driving. Very often, an anomaly detector consists many distinct detectors for each road hazard type, such as wrong-way driving [86, 87], overspeeding [88], and aberrant braking [89] detectors. Alternatively, unsupervised machine learning techniques have facilitated the creation of a new class of algorithms, which can identify abnormal behaviors based solely on training examples. These techniques have the advantage of being highly adaptable, which may be important for rapid adaptation to different road scenarios.

These methods typically construct a representation of normal behavior trajectories through a dictionary [90], a parametric distribution, or histogram bins [91]. Many such approaches can achieve state-of-the-art online performance; however, they often demand significant computational resources, a requirement typically unmet by onboard embedded platforms. Additionally, deep learning techniques also require large volumes of training data, and their opaque nature inhibits the derivation of analytical insights that might inform performance guarantees, such as the system's adaptability to sensors of differing quality. Furthermore, these techniques often assume the complete availability of trajectories (e.g., from traffic security cameras), yet onboard sensors typically yield fragmented trajectories due to frequent occlusions by other road objects. Additionally, safety-oriented systems must operate under strict time constraints to allow for the identification of escape routes, which may be executed by the driver or an autonomous driving system.

Such considerations have motivated this chapter to introduce an anomaly detection framework that is capable of real-time trajectory classification, computationally efficient, adaptable, and simple that enables analyzing of performance guarantees and accommodating various sensor types and capabilities. The primary contributions of this chapter are as follows:

1. We introduce an unsupervised anomalous trajectory detection framework, with the trajectories represented in FPCA scores, and two methods-likelihood-based and Bayesian-based, to perform classification of abnormality of the time series in real-time, as data arrives. The algorithms are recursive, so they are highly efficient with memory consumption and computational complexity.
2. We perform thorough analytical analysis of the mathematical properties of this framework to reveal its characteristics, types of trajectory anomalies suitable for detection.

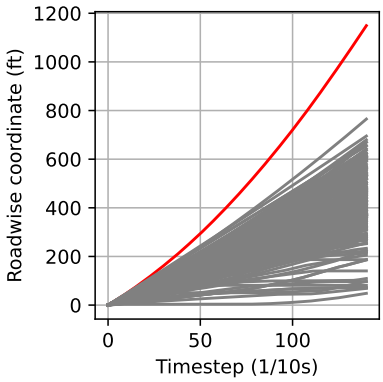


Figure 4.1: A vehicle traveling at excessive speeds can be identified as an anomalous time series. Analyzing its longitudinal trajectory reveals that the average speed of such a vehicle significantly exceeds that of others. An alert should be issued immediately upon detection of this anomaly, typically identifiable around the 20th timestep. However, if the detection system is overly sensitive, it could potentially trigger false alarms. This can occur when the detector mistakenly attributes erratic measurements, which could simply be due to noise, to genuine anomalies.

3. We demonstrate that many common road hazards such as wrong-way driving, overspeed and abarrent braking can be detected using the same framework.

The organization of this chapter is as follows: Section 4.2 offers a comprehensive review of the existing literature on anomalous trajectory detection. Section 4.3 introduces essential background information on FPCA. Section 4.4 outlines our proposed anomaly detection framework, which includes two distinct methodologies: likelihood-based and Bayesian-based. Following this, in Section 4.5, we display a variety of numerical simulations to demonstrate the performance of our algorithm, comparing it to other recognized anomaly detection methods like the Local Outlier Factor (LOF) and Isolation Forest. A deeper interpretation and discussion of these results can be found in Section 4.6. The chapter concludes with Section 4.7.

4.2 RELATED WORKS

In [92], a Bayesian network was proposed to predict the behavior of the driver based on curvilinear coordinates of the vehicle on a highway road. Trained on the NGSIM US-101 dataset [93], the network predicted the driver would proceed on either keeping or changing lanes. In [94], a measure based on graph centrality function was proposed to measure the deviation or similarity of one vehicle to the

neighboring vehicles. It was discovered that the various centrality functions, like degree centrality and closeness centrality, can reflect the aggressiveness in driving style, which may be harmful to other traffic participants. In [95], authors used game theory to model the behavior by benefits of taking specific actions against other drivers. Several datasets with labels on human driving behavior are given in [96]. A human evaluated driving style using LASSO for feature selection and prediction by a simple yet efficient linear regression model was constructed in [97].

On the deep learning side, [98] proposed a framework based on generative adversarial network (GAN) to detect anomalies in time series with application to mobility data. In [99], a GAN was proposed to mimic the driving style of the drivers. Although quite powerful, these methods require a very large training dataset, and a powerful hardware to run in real-time.

Reconstruction-based Anomaly Detection is another popular approach built upon the observation that similar subjects (trajectories) can be represented by a much simpler object (in sparse coding context, a very sparse vector) that yields minor reconstruction error. In [100], the approach was proposed to design an online algorithm that detects anomalies in the video stream. Spatial-temporal points of interest from video sequences formed cuboids that resulted in sparse vectors when represented in a dictionary. Unusual events were noted when reconstruction error exceeded a predefined threshold. [83] presented a similar approach but opted for the ℓ_0 norm to regularize sparsity. In [80], the authors presented a parametric method that learned a Gaussian mixture model to identify anomalous behaviors. In contrast, [101] suggested a Bayesian nonparametric framework for detection of collisions and nearby passes at intersections based on the Hidden Markov Model (HMM). Due to the nature of sequential data, HMMs are also popular, such as in [102] where trajectories were separated into sub-trajectories where a Dirichlet process combined with HMM look for the anomaly. It is noteworthy that although the majority of these approaches work with video sequences as input, they are general enough to work with a variety of other data types, including time-series and entropy [103] or energy functions [104] [30].

Aside from learned basis, other natural bases for time series such as Fourier basis, Spline basis were also explored [105]. In [106] [107], PCA was used to reduce the dimension of the feature vector for more robust performance of classification algorithms. PCA in the functional domain (or FPCA), particularly with energy function, was successfully adopted for the identification of anomalous landing trajectories in [30]. Under appropriate representations, it is possible to use various machine learning techniques to classify anomalous trajectories. For example, [108] uses a Self-Organizing Map Neural Network, [42] used a K-means clustering on Fuzzy membership function. Various distances between observed trajectory to other nominal trajectories can also be used: Hausdorff distance [109] or Dynamic Time Warping (DTW) [110].

Despite the variety of approaches in the literature, existing methods still encounter several challenges. For example, strategies that depend on transforms, such as the one referenced in [30], require a complete trajectory for detection, rendering them unsuitable for real-time applications where observations are continuously gathered over time. The common solution of using a sliding window encounters its own issues: the autocorrelation feature used for detection is lost between subsequent time windows, and observations must entirely fill the sliding window before classification can begin.

In this dissertation, we propose a comprehensive framework capable of performing real-time hypothesis testing for representations transformed by Functional Principal Component Analysis (FPCA) basis functions. However, its application is not restricted to this use. The framework is compatible with all linear transforms, including Fourier and Discrete Cosine Transform (DCT), among others. The goal is to enable real-time hypothesis testing, which allows for immediate alerts once enough evidence has been gathered to confirm abnormal behavior.

4.3 ANOMALY DETECTION FRAMEWORK

The framework is structured into two distinct stages: the learning phase and the detection phase, as depicted in Fig. 4.2. The learning phase applies FPCA to reduce the dimensionality of time series trajectory data. This process yields a simplified, low-dimensional representation of the trajectories, facilitating the subsequent fitting of distribution parameters that will be used during the detection or operation phase.

The detection phase involves real-time hypothesis testing, primarily revolving around the null hypothesis, represented as H_0: observations originate from a "nominal" support of the representation space.

The key objective of this detection framework is to determine whether there is enough evidence to reject the null hypothesis.

4.3.1 Learning Phase

Consider a specific road patch, denoted as P, which is presumed to be a rectangular section with dimensions $w \times l$. It's important to note that within this patch, a single traffic mode exists at any given time, implying that the traffic pattern remains homogeneous. An illustrative example of such a patch could be a segment of a carriageway, where vehicles are mandated to maintain the same direction and adhere to a uniform speed limit (refer to Fig. 4.3). A patch containing two lanes moving in opposite directions would be deemed invalid.

Trajectories of vehicles traversing the designated patch P could be extracted

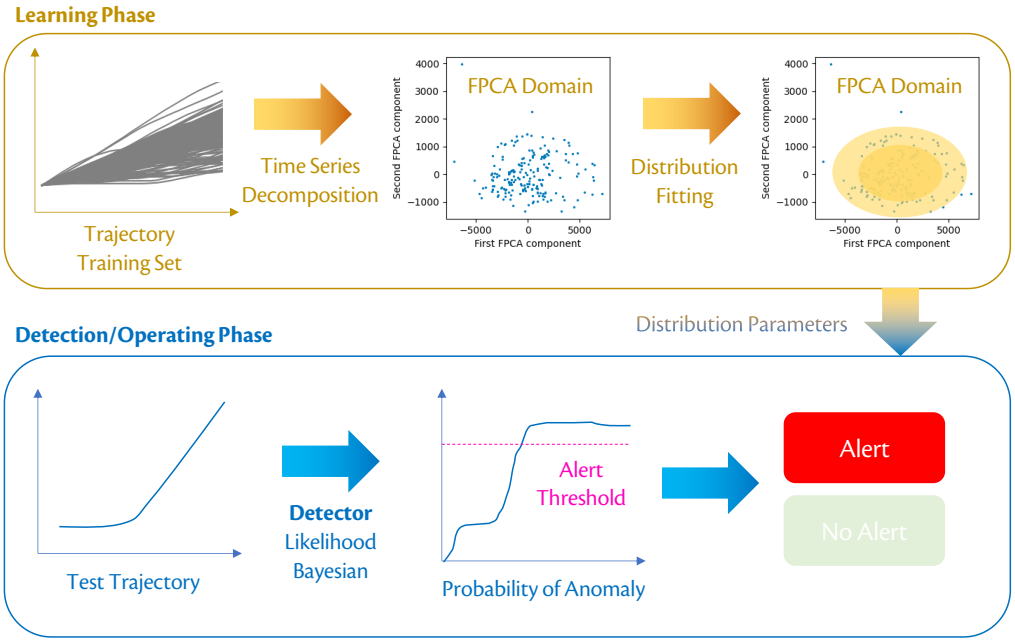


Figure 4.2: Outline of the two phases of our detection framework for anomalous time series. The learning phase uses dimensionality reduction for efficient learning of the representation distribution. The detection phase performs real time hypothesis testing of anomalous trajectory.

from a security camera like the NGSIM-101 dataset [111] or V2X Cooperative Awareness Message (CAM) traces [112]. This data forms the *training set* $\mathcal{S} = (x_k^i, y_k^i), i \in 0, 1, \ldots, n, k \in 1, 2, \ldots, N_v$. It's worth noting that vehicles may pass through the patch P at various speeds, resulting in a significant variance in the lengths of the trajectories. To address this, we analyze the distribution of these lengths, selecting its mode value as n. We then adjust the trajectories accordingly: those shorter than n are extended using cubic splines, while longer ones are truncated. The moment a vehicle enters the patch is defined as $i = 0$.

Consider a *training set* of vehicle trajectories with a consistent temporal length n denoted as $\mathcal{S}_x = \{\boldsymbol{x}_k \in \mathbb{R}^n, 1 \leq k \leq N_v\}$ for the lateral (lane changing) component, and $\mathcal{S}_y = \{\boldsymbol{y}_k \in \mathbb{R}^n, 1 \leq k \leq N_v\}$ for the longitudinal component. We can use Discrete FPCA (as detailed in section 2.3) to determine the FPCA scores (also referred to as *representations*) for each individual trajectory [113, 114]. In essence, FPCA converts a time-domain trajectory vector $\mathbb{R}^n$ into a representation vector also of dimension $\mathbb{R}^n$. However, only the first few elements carry significant weight, with the rest approaching zero. Consequently, we only consider the first $m << n$ coefficients to derive the sets of representations $\mathcal{P}_x = \{\boldsymbol{z}_{x,k} \in \mathbb{R}^m, 1 \leq k \leq N_v\}$ and $\mathcal{P}_y = \{\boldsymbol{z}_{y,k} \in \mathbb{R}^m, 1 \leq k \leq N_v\}$ for the lateral and longitudinal

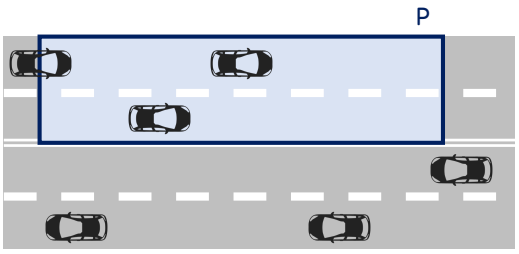


Figure 4.3: Illustration of the road patch of interest P. All vehicles should be moving in the same direction within the road patch.

components of the trajectory, respectively.

The final step involves fitting a Gaussian mixture, denoted as $p_{\mathfrak{M}} \triangleq p(a|\mathfrak{M})$, onto these representation sets $\mathcal{P}_x, \mathcal{P}_y$ using the EM algorithm. Each Gaussian component, represented as p_{a_i} within the mixture, is associated with a distinct nominal traffic mode. Consider a situation where a road patch includes a highway ramp, vehicles may decide to proceed straight or make a right turn to follow the ramp off the highway. In such instances, two distinguishable clusters might emerge in the representation space, leading the mixture to comprise two components.

4.3.2 Detection Phase

We formalize our problem as follows. An ego-vehicle E traveling near a potential outlier vehicle O and records a series of state observations from O denoted as $\boldsymbol{x}_{1:n} = [x_1, x_2, \ldots, x_n]^\top$, $n < t_{max}$. The random variable x_k denotes the state observation at time $T_{obs} + (k-1)T_s$, given that observation starts at T_{obs} and the sampling time is T_s. For convenience, suppose $T_{obs} = kT_s, k \in \mathbb{N}$. We also denote x^i as the random variable of observation at iT_s. In the following, two detection methods will be presented, one based on *likelihood computation* and another on posterior distribution computation, a *Bayesian method*.

Likelihood Method

Given a latent representation a, like a vector of FPCA scores, assume that each measurement of the trajectory is independent, the *likelihood distribution* can be written as:

$$p(x^{1:n}|a) = \prod_{i=1}^{n} p(x^i|a) \tag{4.1}$$

The learning phase has given us the prior distribution of latent representation $p(a)$ hence it is possible to compute the *marginal likelihood*:

$$\mathcal{L}_{\mathfrak{M}} \triangleq p(x^{1:n}) = \int_a p(x^{1:n}|a)p(a)da = \int_a \prod_{i=1}^{n} p(x^i|a)p(a)da \tag{4.2}$$

The marginal likelihood can be used to test the hypothesis, H_0. As will be obvious later, the marginal likelihood converges to a value proportional to the distance from the origin to the true latent representation, a^*, which generates the observations.

We consider a particular class of observations as functionals that admit a unique linear decomposition:

$$\boldsymbol{x}|a = \sum_{k=0}^{m} a_k \phi_k(x), \quad m < +\infty \tag{4.3}$$

and that $\{a_k\} \in \mathbb{R}^m$ is a Gaussian random representation vector that follows $\mathcal{N}(a; \mu_a; \Sigma_a)$. Hence, in the discrete domain, a segment of vehicle trajectory could be written as:

$$x^i = \mu^{T_{obs}+i} + \sum_{k=1}^{m} a_k \phi_k^{T_{obs}+i} \tag{4.4}$$

where $\epsilon^i \sim \mathcal{N}(0; {\sigma_\epsilon}^2)$ (i.i.d) is related to the sensor measurement error.

Note that because T_{obs} is unknown, [115] suggested replacing T_{obs} with an unknown T_{obs} that maximizes $\mathcal{L}_{\mathfrak{M}}$:

$$\max \mathcal{L}_{\mathfrak{M}} = \max_{T_{obs}} \int_a \prod_{i=1}^{n} p(x^i|a)p(a)da \tag{4.5}$$

where x^i follows (4.4). In the following, we assume that T_{obs} is known, since it can otherwise be found with a simple line search procedure.

Recursive analytical solutions. Closed-form solution for problem (4.5) exist when $p(a)$ is a simple Gaussian distribution (or even a Gaussian mixture, due to the linearity of integral). For the moment, suppose the measurement noise $\epsilon^i = 0$. Consider the term:

$$\begin{aligned} \mathcal{L}_{\mathfrak{M}} = \int_a \prod_{k=1}^{n} p(x^k|a)p(a)da = \frac{1}{(\sigma_\epsilon\sqrt{2\pi})^n} \frac{1}{(2\pi)^{m/2}\sqrt{det(\Sigma_a)}} \\ \int_a \exp\left[-\frac{1}{2}(a-\mu_a)^\top \Sigma_a^{-1}(a-\mu_a)\right. \\ \left. -\frac{1}{2\sigma_\epsilon^2}\sum_{k=1}^{n}\left(x^k - \mu^{T_{obs}+k} - \sum_{j=1}^{m} a_j \phi_j^{T_{obs}+k}\right)^2\right] da \end{aligned} \tag{4.6}$$

Denote $\Phi^{T_{obs}+k} = [\phi_1^{T_{obs}+k}, \phi_2^{T_{obs}+k}, \dots]^\top$, $\mathcal{L}_{\mathfrak{M}}$ becomes:

$$\begin{aligned}\mathcal{L}_{\mathfrak{M}} \propto \int_a \exp\Bigg[-\frac{1}{2}\left(a^\top \Sigma_a^{-1} a - 2a^\top \Sigma_a^{-1}\mu_a + \mu_a^\top \Sigma_a^{-1}\mu_a\right) \\ -\frac{1}{2\sigma_\epsilon^2}\sum_{k=0}^{n-1}\Bigg(\left(x^k\right)^2 + \left(\mu^{T_{obs}+k}\right)^2 + \left(\Phi^{T_{obs}+k^\top} a\right)^2 \\ -2x^k\mu^{T_{obs}+k} - 2\mu^{T_{obs}+k}\Phi^{T_{obs}+k^\top} a - 2x^k\Phi^{T_{obs}+k^\top} a\Bigg)\Bigg]\, da\end{aligned} \tag{4.7}$$

We further let $C_\times^{T_{obs}+k} = \frac{1}{\sigma_\epsilon^2}(\Phi^{T_{obs}+k} \otimes \Phi^{T_{obs}+k})$ where $\otimes$ being the outer product, and $C_\dagger^{T_{obs}+k} = \frac{1}{\sigma_\epsilon^2}(-\mu^{T_{obs}+k} + x^{T_{obs}+k})\Phi^{T_{obs}+k^\top}$, then equation (4.7) can be rewritten as:

$$\begin{aligned}\mathcal{L}_{\mathfrak{M}} \propto \int_a \exp\Bigg[-\frac{1}{2}\Bigg[a^\top\left(\Sigma_a^{-1} + \sum_{k=0}^{n-1} C_\times^{T_{obs}+k}\right)a \\ -2\left(\Sigma_a^{-1}\mu_a + C_\dagger^{T_{obs}+k}\right)^\top a + \left(\mu_a^\top\Sigma_a^{-1}\mu_a + \sum_{k=0}^{n-1}\iota^{T_{obs}+k}\right)\Bigg]\Bigg] da\end{aligned}$$

where we have replaced:

$$\iota^{i+k} = \frac{1}{\sigma_\epsilon^2}(x^k - \mu^{T_{obs}+k})^2 \tag{4.8}$$

The exponent, after completion of squares, bares the form of a Gaussian distribution's exponent, with the new covariance and mean:

$$\Sigma_a^{*-1} = \Sigma_a^{-1} + \sum_{k=0}^{n-1} C_\times^{T_{obs}+k} \tag{4.9}$$

and

$$\mu_a^* = \left(\Sigma_a^{-1} + \sum_{k=0}^{n-1} C_\times^{T_{obs}+k}\right)^{-1}\left(\Sigma_a^{-1}\mu_a + \sum_{k=0}^{n-1} C_\dagger^{T_{obs}+k}\right) \tag{4.10}$$

Equation (4.9) reflects the gain of information from a bare hypothesis, sequentially with time as new measurements arrive. The mean shift is shown in (4.10). Finally, to perform the marginalization integral in (4.5), we use the Euler-Poisson integral to yield:

$$\begin{aligned}\mathcal{L}_{\mathfrak{M}} = \frac{1}{(\sigma_\epsilon\sqrt{2\pi})^n}\sqrt{\frac{det(\Sigma_a^*)}{det(\Sigma_a)}} \\ \exp\left(-\frac{1}{2}\left(-\mu_a^{*\top}\Sigma_a^{*-1}\mu_a^* + \mu_a^\top\Sigma_a^{-1}\mu_a + \sum_{k=0}^{n-1}\iota^{T_{obs}+k}\right)\right)\end{aligned} \tag{4.11}$$

It is noteworthy that (4.9), (4.10) can be used for the recursive calculation of $\mathcal{L}_{\mathfrak{M}}$. It will become obvious later that these expressions also allow for the calculation of the posterior mean and covariance in the Bayesian method. The details are shown in Algorithm 4.1.

Algorithm 4.1: Recursive Computation of Posterior Distribution Parameters and Marginal Likelihood.

initialize $\Sigma_a^{*-1} \leftarrow \Sigma_a^{-1}$, $\mu_a^{*'} \leftarrow \Sigma_a^{-1}$.
for $T_{obs} \in \mathbb{N}$ **do**
  if data arrives **then**
    $\Sigma_a^{*-1} \leftarrow \Sigma_a^{*-1} + \frac{1}{\sigma_\epsilon^2}(\Phi^{T_{obs}+k} \otimes \Phi^{T_{obs}+k})$
    $\mu_a^{*'} \leftarrow \mu_a^{*'} + \frac{1}{\sigma_\epsilon^2}(-\mu^{T_{obs}+k} + x^{T_{obs}+k})\Phi^{T_{obs}+k^\top}$
    $\mu_a^{*} \leftarrow \Sigma_a^{*}\mu_a^{*'}$
    $\mathcal{L}_{\mathfrak{M}}(T_{obs}) \leftarrow$ Equation (4.11)
  end if
end for
return $\max_{T_{obs}} L_{\mathfrak{M}}$

Bayesian method

From Bayes Theorem:

$$p(a|x_{1:n}) = \frac{p(x_{1:n}|a)p(a)}{\int_a p(x_{1:n}|a)p(a)da} \propto_a p(x_{1:n}|a)p(a), \tag{4.12}$$

which is nothing but the integrand of $\mathcal{L}_{\mathfrak{M}}$ in (4.6). Because a Gaussian form is exhibited in the integrand of equation (4.3.2), it can be deduced that (4.9) and (4.10) are the covariance and mean of the posterior distribution, respectively.

The likelihood method yields a single real number, which decreases over time as more observations are incorporated. In contrast, the Bayesian method generates a distribution, wherein the eigenvalues of the covariance matrix decrease with time, suggesting a more reliable posterior estimate. In the subsequent section, the results are presented, enabling the use of these values for hypothesis testing.

4.3.3 Real-time Hypothesis Testing with the Marginal Likelihood

Suppose that the trajectories learned in the learning phase came from a ranging sensor that can be modeled with the following measurement model:

$$x^i = \hat{x}^i + \epsilon^i, \tag{4.13}$$

where $\epsilon_i \sim \mathcal{N}(0, \sigma_\epsilon^2)$. For this marginal likelihood method, we also assume that $\hat{x}^i$ satisfies (4.4). The usual way to define anomaly in an unsupervised manner is as follows. Recall that the KL decomposition of a functional is:

$$f_i(t) = \sum_{j=0}^{\infty} a_{ij} v_j(t), \quad \text{where } \mathbb{E}[a_{ij}] = 0. \tag{4.14}$$

Definition 4.1. A trajectory is defined as *abnormal* if the Euclidean distance from its FPCA representation a^* in (4.14) to the centroid of the training set's FPCA representations is larger than some threshold $\hat{d}$:

$$||a^* - \mathbb{E}[a]||_2 \geq \hat{d}, \tag{4.15}$$

and from the Karhunen-Loeve theorem, it is known that $\mathbb{E}[a] = 0$, thus the definition is equivalent to:

$$||a^*||_2 \geq \hat{d}. \tag{4.16}$$

Given $p\left(x^i|a\right) = \frac{1}{\sqrt{2\pi}\sigma_\epsilon} e^{-\frac{\left(x^i - \mu^i - \Phi^{i\top} a\right)^2}{2\sigma_\epsilon^2}}$ and $x^i = \Phi^{i\top} a^* + \mu^i + \epsilon^i$. Then:

$$\begin{aligned} p\left(x^i|a\right) &= \frac{1}{\sqrt{2\pi}\sigma_\epsilon} e^{-\frac{\left(\Phi^{i\top}(a^* - a) + \epsilon^i\right)^2}{2\sigma_\epsilon^2}} \\ &= \frac{1}{\sqrt{2\pi}\sigma_\epsilon} e^{-\frac{\left(\Phi^{i\top}(a^* - a)\right)^2}{2\sigma_\epsilon^2}} e^{-\frac{\left(\epsilon^{i2} + 2\Phi^{i\top}(a^* - a)\epsilon^i\right)}{2\sigma_\epsilon^2}} \end{aligned} \tag{4.17}$$

Substituting into (4.2) we have:

$$\begin{aligned} \mathcal{L}_{\mathfrak{M}} = \int_a & \frac{1}{(2\pi)^{\frac{n+m}{2}} \sigma_\epsilon^n \sqrt{\det(\Sigma_a)}} \\ & e^{-\frac{\sum_{i=1}^n \left(\Phi^{i\top}(a^* - a)\right)^2}{2\sigma_\epsilon^2} - \frac{1}{2}(a - \mu_a)^\top \Sigma_a^{-1}(a - \mu_a)} \\ & e^{-\frac{\sum_{i=1}^n \left(\epsilon^{i2} + 2\Phi^{i\top}(a^* - a)\epsilon^i\right)}{2\sigma_\epsilon^2}} da \end{aligned}$$

Defining the variables $b = a - \mu_a$, $b^* = a^* - \mu_a \Rightarrow a^* - a = b^* - b$ and $da = db$, we have:

$$\begin{aligned} \mathcal{L}_{\mathfrak{M}} = \int_a & \frac{1}{(2\pi)^{\frac{n+m}{2}} \sigma_\epsilon^n \sqrt{\det(\Sigma_a)}} \\ & e^{-\frac{\sum_{i=1}^n \left(\Phi^{i\top}(b^* - b)\right)^2}{2\sigma_\epsilon^2} - \frac{1}{2} b^\top \Sigma_a^{-1} b} \\ & e^{-\frac{\sum_{i=1}^n \left(\epsilon^{i2} + 2\Phi^{i\top}(b^* - b)\epsilon^i\right)}{2\sigma_\epsilon^2}} db \end{aligned}$$

The term $1/[(2\pi)^{\frac{n+m}{2}}\sigma_\epsilon^n\sqrt{\det(\Sigma_a)}]$ can be recognized simply as a normalization constant. To condense and streamline the calculation steps, it can be denoted as $C(n)$. Now, by completing the squares:

$$\mathcal{L}_{\mathfrak{M}} = C(n)\int_a \exp\Bigg[-\frac{1}{2\sigma_\epsilon^2}\sum_{i=1}^{n}\left(\Phi^{i\top}b^*\right)^2 + \frac{1}{\sigma_\epsilon^2}\sum_{i=1}^{n}\left(\Phi^{i\top}b^*\Phi^{i\top}b\right) - \frac{1}{2}b^\top\left(\frac{1}{\sigma_\epsilon^2}\sum_{i=1}^{n}\Phi^i\Phi^{i\top} + \Sigma_a^{-1}\right)b\Bigg] \exp\left[-\frac{\sum_{i=1}^{n}\left(\epsilon^{i2} + 2\Phi^{i\top}\left(b^* - b\right)\epsilon^i\right)}{2\sigma_\epsilon^2}\right]db$$

Taking the Poisson-Euler integral and get the log of both sides:

$$\begin{aligned}\log\mathcal{L}_{\mathfrak{M}} &= \log C(n) + \frac{m}{2}\log(2\pi) + \frac{1}{2}\log\det\Sigma^* \\ &+\frac{1}{2\sigma_\epsilon^2}\Bigg[\left(\sum_i \Phi^{i\top}b^*\Phi^{i\top} + \sum_i \Phi^{i\top}\epsilon^i\right) \\ &\frac{\Sigma^*}{\sigma_\epsilon^2}\left(\sum_i \Phi^i b^{*\top}\Phi^i + \sum_i \Phi^i\epsilon^i\right) - \sum_i\left(\Phi^{i\top}b^* + \epsilon^i\right)^2\Bigg]\end{aligned} \tag{4.18}$$

where

$$\Sigma^* = \left(\frac{1}{\sigma_\epsilon^2}\sum_{i=1}^{n}\Phi^i\Phi^{i\top} + \Sigma_a^{-1}\right)^{-1}$$

For convenience, we let $\log D(n) = \log C(n) + \frac{m}{2}\log(2\pi) + \frac{1}{2}\log\det\Sigma^*$ and notice that this term is independent of b. As a result, computation of $D(n)$ can be bypassed if we let $b^* = 0$ to obtain the marginal likelihood curve $\log D(n)$, then subtract $\log D(n)$ from $\mathcal{L}_{\mathfrak{M}}$.

Perfect Measurement Case

We solve the simple case of perfect measurements (no noise, or $\epsilon^i = 0$). Then, (4.18) becomes:

$$\log\mathcal{L}_{\mathfrak{M}} = \log D(n) + \frac{1}{2\sigma_\epsilon^2}\Bigg[\left(\sum_i \Phi^{i\top}b^*\Phi^{i\top}\right)\frac{\Sigma^*}{\sigma_\epsilon^2} \left(\sum_i \Phi^i b^{*\top}\Phi^i\right) - \sum_i\left(\Phi^{i\top}b^*\right)^2\Bigg]$$

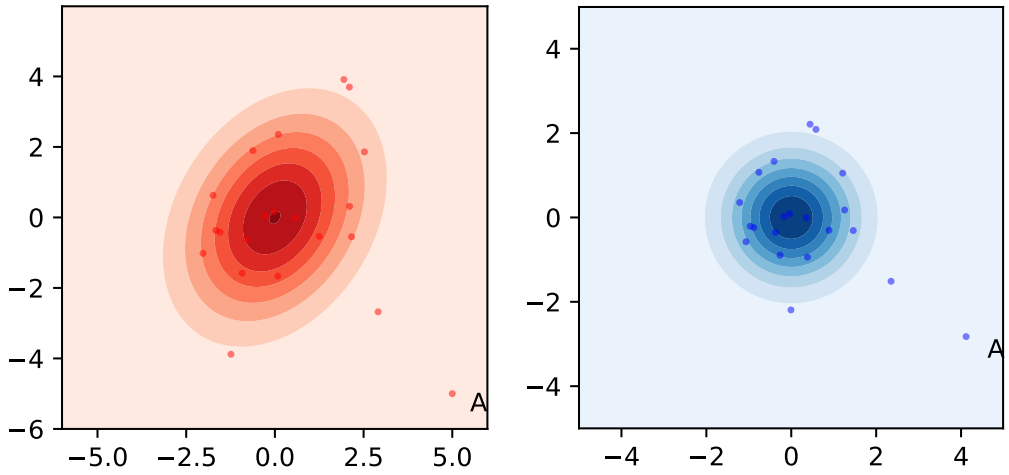


Figure 4.4: Density function of two 2D Gaussian latent distributions (a) $p(b^*)$ and (b) $p(R^{-1/2}b^*)$. The latent representation A is identified as an anomaly due to its substantial distance from the cluster's centroid, and the anomaly detection problem is the same in both latent distributions.

The objective of this framework is to determine whether the latent representation of the observed trajectory significantly deviates from the cluster's centroid, which is also the origin point. However, using Euclidean distance as a direct measure of this deviation poses challenges, as the interpretation of distance may vary across different directions. This variation is linked to a greater spread of data points along the eigenvectors of the covariance matrix, meaning that a data point A at a given distance d in this direction may still have more neighboring points than another data point B, at the same distance d but in a different direction (Fig. 4.4).

Mathematically, we would like to find a matrix R such that $b^* = Rc^*$ where $c^* \sim \mathcal{N}(0; \boldsymbol{I}_{m\times m})$. Then, the distance from b^* to $\mathbf{0}$ can be straightforwardly defined with $||c^*||_2$. One straightforward choice is:

$$R = \Sigma_a^{1/2} \tag{4.19}$$

Proposition 4.2. Without measurement noise ($\epsilon^i = 0$), the marginal likelihood $\mathcal{L}_{\mathfrak{M}}$ defined in (4.2) is bounded below by:

$$T^i = \lambda_1 \left(-U^\top \frac{\Sigma^*}{\sigma_\epsilon^2} U + V \right) d, \tag{4.20}$$

if the Euclidean distance from the transformed latent representation $c^* = \Sigma_a^{-1/2} b^*$

to the mean of the transformed prior distribution $p(\Sigma_a^{-1/2}a)$ is d, where U, V are defined in (4.21) and $\lambda_1(M)$ denotes the largest eigenvalue of the matrix M.

Proof. The log likelihood expression can be rewritten as:

$$\log \mathcal{L}_{\mathfrak{M}} = \log D(n) + \frac{1}{2\sigma_\epsilon^2}\left[\left(\sum_i {\Phi^i}^\top R c^* {\Phi^i}^\top\right)\frac{\Sigma^*}{\sigma_\epsilon^2}\left(\sum_i \Phi^i (Rc^*)^\top \Phi^i\right) - \sum_i \left({\Phi^i}^\top R c^*\right)^2\right]$$

Using the dot product property, with some transformations:

$$\sum_i {\Phi^i}^\top R c^* {\Phi^i}^\top = \sum_i \left(R^\top \Phi^i\right)^\top c^* {\Phi^i}^\top = \sum_i c^{*\top}\left(R^\top \Phi^i {\Phi^i}^\top\right)$$

$$\sum_i \left({\Phi^i}^\top R c^*\right)^2 = \sum_i c^{*\top} R^\top \Phi^i {\Phi^i}^\top R c^*$$

Hence:

$$\log \mathcal{L}_{\mathfrak{M}} = \log D(n) + \frac{1}{2\sigma_\epsilon^2}\left[\left(c^{*\top}\left(\sum_i \left(\Phi^i {\Phi^i}^\top R\right)^\top\right)\right)\frac{\Sigma^*}{\sigma_\epsilon^2}\left(\sum_i \left(\Phi^i {\Phi^i}^\top R\right)\right)c^* - c^{*\top}\left(\sum_i R^\top \Phi^i {\Phi^i}^\top R\right)c^*\right]$$

Let:

$$\begin{aligned} U &= \sum_i \Phi^i {\Phi^i}^\top R = \Phi R \\ V &= \sum_i R^\top \Phi^i {\Phi^i}^\top R = R^\top \Phi R \end{aligned} \tag{4.21}$$

We have:

$$\log \mathcal{L}_{\mathfrak{M}} = \log D(n) - \frac{1}{2\sigma_\epsilon^2}\left[c^{*\top}\left(-U^\top \frac{\Sigma^*}{\sigma_\epsilon^2} U + V\right)c^*\right]$$

which is a quadratic form. Note that $-U^\top \frac{\Sigma^*}{\sigma_\epsilon^2} U + V$ is positive definite, hence:

$$\log \mathcal{L}_{\mathfrak{M}} \geq \log D(n) - \frac{1}{2\sigma_\epsilon^2}\left[\lambda_1\left(-U^\top \frac{\Sigma^*}{\sigma_\epsilon^2} U + V\right)||c^*||^2\right] \tag{4.22}$$

where λ_1 indicates the maximum eigenvalue of $-U^\top \frac{\Sigma^*}{\sigma_\epsilon^2} U + V$. □

Proposition 4.3. The bound of (4.22) is tight. Assume that Q is full-rank. In case of infinite observations, the lower bound converges to:

$$T^{\infty} = -\frac{1}{2}||c^*||_2^2 \tag{4.23}$$

Proof. We rename Φ to $S_n = \Phi = \sum_{i=1}^{n} \Phi^i \Phi^{i\top}$ for emphasis on the evolution of Φ with time. From (4.21), $U = S_n R, V = R^T S_n R$.

$$\begin{aligned}\frac{1}{\sigma_\epsilon^2}\left(U^\top \frac{\Sigma^*}{\sigma_\epsilon^2} U - V\right) &= \frac{1}{\sigma_\epsilon^2} R^\top \left(S_n^\top \Sigma^* \frac{1}{\sigma_\epsilon^2} - I\right) S_n R \\ &= \frac{1}{\sigma_\epsilon^2} R^\top \left(S_n^\top \left(\frac{1}{\sigma_\epsilon^2} S_n + \Sigma_a^{-1}\right)^{-1} \frac{1}{\sigma_\epsilon^2} - I\right) S_n R\end{aligned}$$

As Σ_a^{-1} is positive definite, we can perform an eigen-decomposition $\Sigma_a^{-1} = Q\Lambda Q^\top$ and use the matrix inversion lemma:

$$\begin{aligned}\left(\frac{1}{\sigma_\epsilon^2} S_n + \Sigma_a^{-1}\right)^{-1} &= \left(\frac{S_n}{\sigma_\epsilon^2}\right)^{-1} - \left(\frac{S_n}{\sigma_\epsilon^2}\right)^{-1} \\ Q\left(\Lambda^{-1} + Q^\top \left(\frac{S_n}{\sigma_\epsilon^2}\right)^{-1} Q\right)^{-1} Q^\top &\left(\frac{S_n}{\sigma_\epsilon^2}\right)^{-1}\end{aligned}$$

Hence:

$$\frac{1}{\sigma_\epsilon^2}\left(U^\top \frac{\Sigma^*}{\sigma_\epsilon^2} U - V\right) = -RQ\left(\Lambda^{-1} + Q^\top \sigma_\epsilon^2 S_n^{-1} Q\right)^{-1} Q^\top R$$

and as $S_n \to \infty$, combined with Q being full rank:

$$\frac{1}{\sigma_\epsilon^2}\lambda_1\left(U^\top \frac{\Sigma^*}{\sigma_\epsilon^2} U - V\right) \to \lambda_1(-R\Sigma_a^{-1}R) = -1,$$

by definition of $R = \Sigma_a^{1/2}$. Proposition 4.3 follows. □

Noisy Measurement Case

In this section, we derive the likelihood of anomaly when measurements are contaminated with Gaussian white noise, that is $\epsilon^i \neq 0$. Rewriting (4.18) as follows:

$$\log \mathcal{L}_{\mathfrak{M}} = \log D(n) + \frac{1}{2\sigma_\epsilon^2}\Bigg[- c^{*\top}\left(-U^\top \frac{\Sigma^*}{\sigma_\epsilon^2}U + V\right)c^* \\ +2\left(\sum_i \epsilon_i \Phi^{i\top}\right)\frac{\Sigma^*\Phi R}{\sigma_\epsilon^2}c^* + \left(\sum_i \epsilon^i \Phi^{i\top}\right)\frac{\Sigma^*}{\sigma_\epsilon^2}\left(\sum_i \Phi^i \epsilon^i\right) \\ - \sum_i \left(\epsilon^{i2} + 2\Phi^{i\top} R c^* \epsilon^i\right)\Bigg]$$

which can be rewritten as:

$$\begin{aligned}\log \mathcal{L}_{\mathfrak{M}} &= \log D(n) + \frac{1}{2\sigma_\epsilon^2}\Bigg[- c^{*\top}\left(-U^\top \frac{\Sigma^*}{\sigma_\epsilon^2}U + V\right)c^* \\ &\quad +2\left(\sum_i \epsilon^i \Phi^{i\top}\right)\left(\frac{\Sigma^*\Phi}{\sigma_\epsilon^2} - I\right)Rc^* \\ &\quad + \left(\sum_i \epsilon^i \Phi^{i\top}\right)\frac{\Sigma^*}{\sigma_\epsilon^2}\left(\sum_i \Phi^i \epsilon^i\right) - \sum_i \epsilon^{i2}\Bigg] \\ &= \log C(n) \; + \frac{1}{2\sigma_\epsilon^2}[A + B + C - D]\end{aligned}$$

Term A is the same as in the perfect measurement case. For term B, let us define $\gamma = \sum_i \phi^i \epsilon^i$ (note $\mathbb{E}\gamma = 0$). Define G and use the matrix inversion lemma:

$$\begin{aligned}G &= \frac{\left(\frac{1}{\sigma_\epsilon^2}\Phi + \Sigma_a^{-1}\right)^{-1}\Phi}{\sigma_\epsilon^2} - I = \left(I + \left(\frac{\Sigma_a\Phi}{\sigma_\epsilon^2}\right)^{-1}\right)^{-1} - I \\ &= -\left(\frac{\Sigma_a\Phi}{\sigma_\epsilon^2} + I\right)^{-1}\end{aligned}$$

Then:

$$B = 2\gamma^\top G R c^*$$

Since $\frac{\Sigma_a\Phi}{\sigma_\epsilon^2}$ is positive definite, there exists an eigen-decomposition $Q\Lambda Q^T$:

$$G = -\left(Q\Lambda Q^T + QQ^T\right)^{-1} = -Q(\Lambda + I)^{-1}Q^T$$

Hence, the eigenvectors of G are also the eigenvectors of $\frac{\Sigma_a \Phi}{\sigma_\epsilon^2}$. In a general FPCA decomposition, the eigenvalues of G can potentially be very large, thus the measurement noise is amplified. However, it is worth noting that this only causes a false alarm if the anomalous vehicle is whose latent representation of the trajectory c^* is parallel to v_1 where v_1 is the eigenvector corresponding to the largest eigenvalue of G, and γ hits the same direction as well. Given the randomness of γ, $B \approx 0$.

For C, as $\left(\frac{1}{\sigma_\epsilon^2}\Phi + \Sigma_a^{-1}\right)^{-1} \to 0$ and at the beginning of the detection, $||\gamma||$ is small, C is usually small throughout the whole detection process. That leaves us with:

$$\begin{aligned} \log \mathcal{L}_{\mathfrak{M}} &\approx \log D(n) \\ &+ \frac{1}{2\sigma_\epsilon^2}\left[-c^{*\top}\left(-U^T\frac{\Sigma^*}{\sigma_\epsilon^2}U + V\right)c^* - \sum_i \epsilon^{i2}\right] \end{aligned} \tag{4.24}$$

It should be noted that, according to Proposition 4.3, the first term converges to the distance between the latent representation c^* and the origin. This effectively represents the degree of anomaly in the current stream of observations. Therefore, this term can be interpreted as the "signal" component of the marginal likelihood. On the other hand, the second term, $\sum_i \epsilon^{i2}$, introduces noise that disrupts the signal, thereby adding complexity to the detection process.

Observe that σ_ϵ^2 is a parameter of the detector. Since the first term converges to $||c^*||^2$ irrespective of σ_ϵ^2, we can use this variable to optimize the signal-to-noise ratio. Let's consider the case where $Var[\epsilon^i] = \sigma_{\epsilon,m}^2$:

$$\log \mathcal{L}_{\mathfrak{M}} - \log D(n) \to -\frac{1}{2}||c^*||^2 - \frac{\sigma_{\epsilon,m}^2}{2\sigma_\epsilon^2}\sum_i \frac{\epsilon^{i2}}{\sigma_{\epsilon,m}^2} \tag{4.25}$$

Then, it is obvious that for best performance, one should keep the ratio $\sigma_{\epsilon,m}^2/\sigma_\epsilon^2 \approx 0$, or σ_ϵ^2 sufficiently large.

Because $\sum_i \epsilon^{i2}$ follows a χ^2 distribution, (4.24) suggests a way to test the hypothesis H_0 with the marginal likelihood $\mathcal{L}_{\mathfrak{M}}$. Let P_n be the *nominal probability*, i.e., probability that observations are coming from the transformed latent representation that is less than a distance of d from the origin:

$$\begin{aligned} P_n = F_{\chi^2}\Bigg[&- 2\frac{\sigma_\epsilon^2}{\sigma_{\epsilon,m}^2}(\log \mathcal{L}_{\mathfrak{M}} - \log D(n)) \\ &+ \frac{1}{\sigma_{\epsilon,m}^2}\left(-c^{*\top}\left(-U^T\frac{\Sigma^*}{\sigma_\epsilon^2}U + V\right)c^*\right)\Bigg] \end{aligned} \tag{4.26}$$

where F_{χ^2} denotes the cumulative distribution function (CDF) of a chi-squared random variable.

4.3.4 Bayesian Anomaly Detection

The noise term $\sum_{i=1}^{n} \epsilon^{i2}$, as derived from equation (30), may dominate the signal term under certain conditions, including: (1) when n is large, (2) when the generative noise ϵ^i values do not represent independent samples from a Gaussian distribution, and (3) when the approximation provided by equation (4.3) is unsatisfactory. Consequently, these factors can significantly diminish the performance of an anomaly detector relying on marginal likelihood.

A potential solution is to rely on the Bayesian view to design an alternative anomaly detector. Let $Q = R^{-1}$. Conditioned on past observations $x_{1:n}$, we have the posterior estimate of $a^* \sim \mathcal{N}(\mu_a^*, \Sigma_a^*)$ where μ_a^* and Σ_a^* are computed recursively from Algorithm 4.1. Like before, we introduce a transformed latent representation $c^* = R^{-1}a^*$ whose prior is the multivariate normal distribution $\mathcal{N}(0, \boldsymbol{I})$:

$$c^* \sim \mathcal{N}(Q\mu_a^*, Q\Sigma_a^* Q^\top)$$

We further define:

$$Q^* = (Q\Sigma_a^* Q^\top)^{1/2}$$

then the random variable:

$$\nu = Q^{*-1}(c^* - Q\mu_a^*)$$

is a normal random vector. Equivalently:

$$c^* = Q^*\nu + Q\mu_a^*,$$

which leads to:

$$\begin{aligned} c^{*\top}c^* &= (Q^*\nu + Q\mu_a^*)^\top(Q^*\nu + Q\mu_a^*) \\ &= \nu^\top Q^{*\top}Q^*\nu + 2\mu_a^{*\top}Q^\top Q^*\nu \\ &\quad + \mu_a^{*\top}Q^\top Q\mu_a^* \end{aligned} \tag{4.27}$$

This expression bears the form of a Generalized Chi-Squared random variable $x^\top Q_2 x + \boldsymbol{q_1}^\top x + q_0$ and a closed form analytical solution generally does not exist. However, note that as $\Sigma_a^* \to 0$ and Q is full rank, the second order term converges to zero faster than the first order term. Hence, as Σ_a^* is sufficiently small:

$$c^{*\top}c^* \approx 2\mu_a^{*\top}Q^\top Q^*\nu + \mu_a^{*\top}Q^\top Q\mu_a^* \tag{4.28}$$

which is simply a Gaussian distribution with parameters:

$$\mu_p = \mu_a^{*\top}Q^\top Q\mu_a^* \tag{4.29}$$

$$\sigma_p^2 = 4\mu_a^{*\top}Q^\top Q^* Q^{*\top}Q\mu_a^* \tag{4.30}$$

and the nominal p-value is:

$$p_n = F_{\mathcal{N}}\left(\frac{d^2 - \mu_p}{\sigma_p}\right) \tag{4.31}$$

where $F_{\mathcal{N}}$ is the CDF of a normal distribution.

4.4 NUMERICAL SIMULATIONS

4.4.1 Data Preprocessing

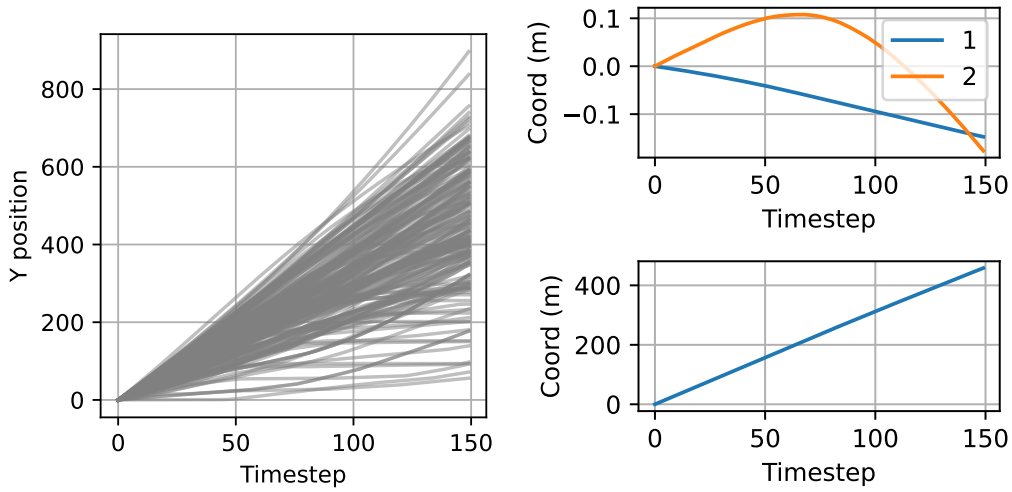
The NGSIM US-101 dataset [93] comprises genuine vehicle trajectories collected from traffic cameras installed along a section of the US-101 highway in California. Each trajectory boasts a temporal resolution of up to 0.1 seconds, which is sufficiently accurate for macroscopic traffic modeling. The dataset includes labels that extend beyond mere vehicle positions and timestamps to encompass vehicle dimensions (length and width), the lane the vehicle occupies, and measurements such as headway, headway distance, and time with respect to leading vehicles.

We defined the road patch P whose width spans all the lanes and length is 900ft ($\approx$ 274m) and randomly collected 200 vehicle trajectories passing through P. Let t_M be the mode of the distribution of trajectories length in $\mathcal{S}$. Because the Discrete FPCA required a time series of equal length, the outliers significantly longer or shorter than t_M were removed. Those that fell within the acceptable range of ± 10 time steps were either trimmed or extrapolated with a second-order cubic spline to meet the requirement (see Fig. 4.5a).

4.4.2 Learning the Nominal Distribution

The discrete FPCA process, described in Section 2.3 revealed that the first two principal components (Fig. 4.5b) explained up to 99.67% of variance in the dataset. This allowed us to assume that the first two FPCA components were sufficient to reconstruct $\mathcal{S}$ as well as to discriminate against anomalies. However, it is noteworthy that although theoretically on the functional domain C^∞, FPCA will yield an "almost exact" reconstruction of any functional $f \in C^\infty$ but realistically, the discrete implementation leads to increasing error towards the end of the trajectories. For this reason, we proposed to only perform detection on the first 90% segment of the trajectory, leaving the remaining 10% as a safety margin.

Each time series decomposed with the two principal components above will become a 2D vector in the representation domain. Since the movement is unimodal in this road patch, we opted for fitting a Gaussian distribution $p(a)$ over the set of representation vectors $\mathcal{P}$ (Fig. 4.6).



(a) (b)

Figure 4.5: All longitudinal trajectories of vehicles from the NGSIM dataset (a), and (b) the FPCA components f_1, f_2 (top) the mean component (bottom) used to dimensionally reduce the trajectories to 2D representations in Fig. 4.6.

4.4.3 Noiseless Detection using the Marginal Likelihood

In the absence of measurement noise, the anomaly threshold boundary is defined by Proposition 4.3. Fig. 4.7 presents a total of six trajectories, numbered from 0 to 5. Their corresponding representation form, c^*, as described in Proposition 4.2, is provided in the second column of Table 4.1. We categorize trajectories with latent representations located within the unit disc as nominal trajectories, that is, those corresponding to $d = 1$. Under this classification, approximately 68% (equivalent to 1 standard deviation) of trajectories are considered nominal.

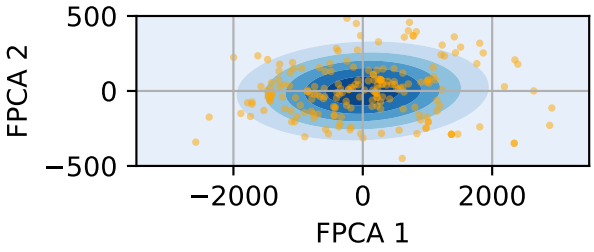
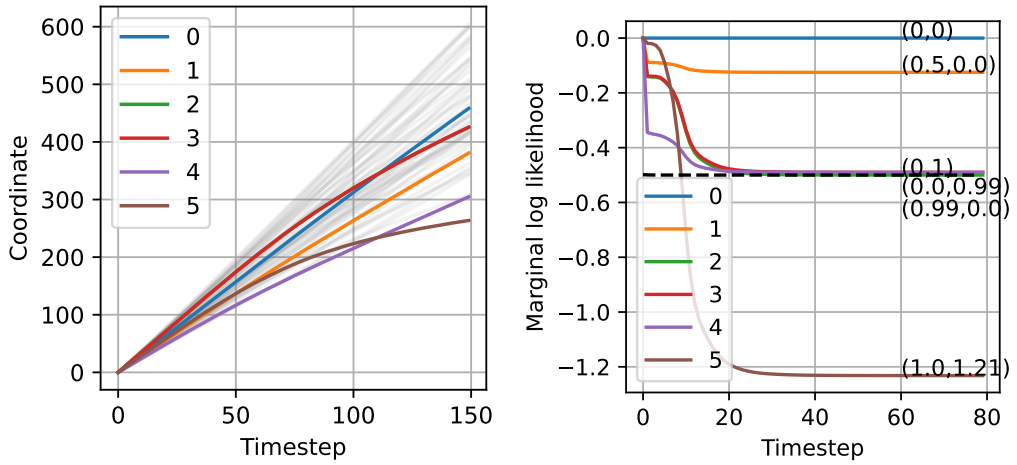


Figure 4.6: Decomposition of time series in the training set to representation forms. Data points are yellow scattered dots. The fitted Gaussian distribution contours are also shown in blue.



(a) (b)

Figure 4.7: On the left-hand side, six sample trajectories are depicted, with the "nominal trajectories" - those with transformed latent representations residing within the unit disc - are shown in gray. On the right-hand side, the real-time computation of the marginal likelihood curves that correspond to these six sample trajectories. The threshold corresponding to $d = 1$ is the dashed black line. Marginal likelihood curves below this line will trigger the alarm of anomaly, which mark significant deviation from "nominal trajectories".

As anticipated, curves with $||c^*|| \leq 1$ did not trigger any alarms. Alarm was only raised for curve 5, where $||c^*|| \approx 1.57$, after only 14 of 150 observations. This highlights a significant distinction between our methods and other comparison techniques like sparse representation or sliding window-based techniques. The latter requires the window to be fully populated before further analysis can occur, resulting in additional delays.

4.4.4 Detection of Anomaly in The Case of Noise-Contaminated Observations Using the Marginal Likelihood

The marginal likelihood detector detailed in (4.26) is showcased as an unsupervised tool to detect aberrant driving behavior through the demonstration of four test trajectories depicted in Fig. 4.8(a). These trajectories are representative of: (1) a typical driving trajectory from the dataset, (2) a vehicle exceeding the average traffic speed by up to two-fold, notably from timestep 25, (3) a vehicle coming to an unexpected halt at the 40th timestep, and (4) a vehicle proceeding in an incorrect direction. The simulation is further refined by the addition of zero-mean Gaussian White Noise (GWN) with a variance of $1/\sigma^2_{\epsilon,m} = 1$, mimicking sensor noise. The marginal likelihoods associated with these trajectories are illustrated in

Table 4.1: Latent Representation c^* and the Responsiveness of the Detection Algorithm with Detection Threshold $d = 1$. Only the fifth curve has the Euclidean distance to the origin larger than d, thus triggering an alarm at $t = 14$.

Curve	Latent Represent. $c^* = Rb^*$	Alarm Delay
0	(0, 0)	∞
1	(0.5, 0)	∞
2	(0, 1)	∞
3	(0, 0.99)	∞
4	(0.99, 0)	∞
5	(1, 1.21)	14

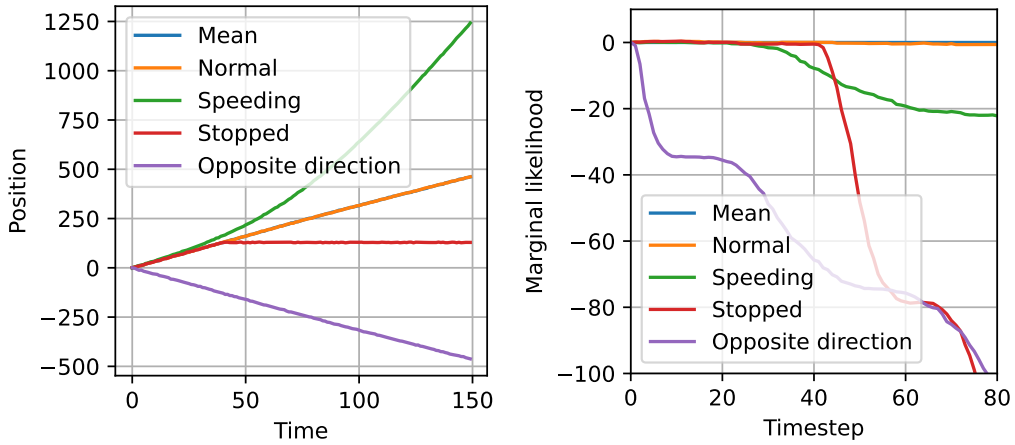


Figure 4.8: (a) Typical anomalous trajectories. (b) The marginal likelihood computed by Algorithm 4.1. The mean trajectory in (a) was hidden behind the Normal curve, and was added for visualization purpose only.

Fig. 4.8(b).

The anomaly p-value for each trajectory, calculated using equation (4.26), is depicted in Fig. 4.9. In this configuration, we have set the variance $1/\sigma_\epsilon^2$ to 0.08 and the detection threshold d to 1. The black dotted line represents the alarm threshold set at $p = 0.05$. Within these examples, all abnormal behaviors were accurately classified. The alarm for the speeding vehicle was triggered at $t = 35$, while the alarm for the halted vehicle was activated at $t = 46$, representing a delay of just 6 timesteps.

To underscore the findings, we compare the results with those obtained using the well-established Local Outlier Factor (LOF) and Isolation Forest (IF) meth-

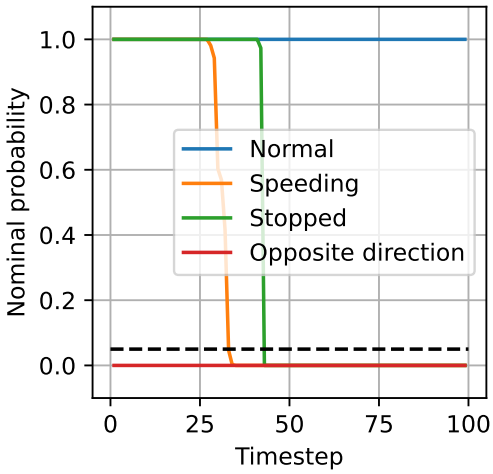


Figure 4.9: Marginal likelihood method: probability that the trajectory is nominal. The dotted black line is the alarm threshold.

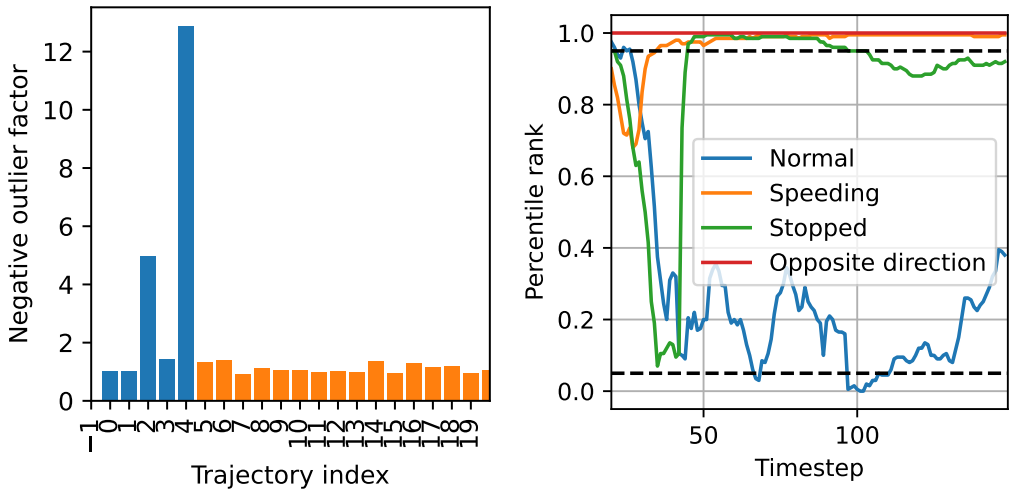
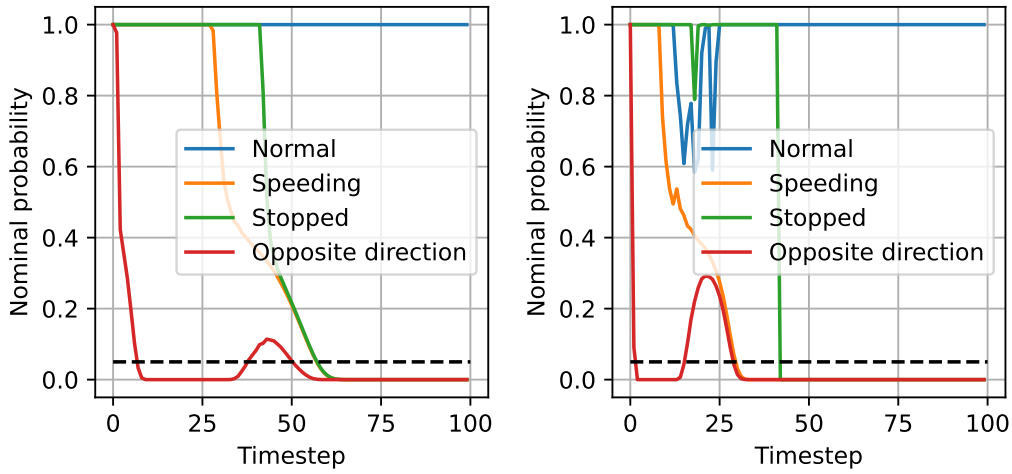


Figure 4.10: Computation of Negative LOF with $N_{neighbors} = 5$: (a) The LOF (Local Outlier Factor) for all trajectories, with the first five representing test trajectories and the remaining ones constituting the training set. (b) The LOF's progression over time corresponding to partial observations. The dotted lines represent the thresholds for the 5% and 95% percentiles.



(a) Sensitivity $1/\sigma_\epsilon^2 = 0.08$ (b) Sensitivity $1/\sigma_\epsilon^2 = 8$

Figure 4.11: Bayesian method: probability that the trajectory is nominal with two different values of sensitivity $1/\sigma_\epsilon^2$. The dotted black line is the alarm threshold set corresponding to $p = 0.05$.

ods for anomaly detection. Our results expose significant limitations within LOF specifically, but these shortcomings are speculated to be pervasive within the family of local density anomaly detection techniques as well. Despite having access to all measurements, the LOF method only accurately identified the second (speeding) and fourth (wrong-way driving) trajectories (Fig. 4.10), failing to detect the anomaly of a vehicle that suddenly stopped. Moreover, the progression of LOF curves as measurements became sequentially available indicated a susceptibility to false alarms with this approach. For instance, even during normal trajectory, false alarms were triggered at timesteps 65 and 96 when the percentile rank curve crossed the 5% threshold line.

4.4.5 Anomaly Detection with Bayesian Method

To illustrate the efficacy of the proposed Bayesian anomaly detector (referenced in Eq. (4.31)), we employ the same test trajectories outlined in the preceding section. Fig. 4.11(a) depicts the time-varying p-value. It is obvious that, upon receiving complete observations, all anomalies were accurately classified. Contrastingly, a comparison with Fig. 4.9 indicates a relatively reduced reactivity of the Bayesian method, and increase in susceptibility to false alarms. Nonetheless, the performance is significantly better than the baseline LOF, marking a substantial improvement.

For example, the vehicle made an abrupt stop at timestep 40. While the confidence level started to decline approximately at the same time, the system didn't trigger an alarm until timestep 56, resulting in a delay of 16 timesteps. This delay is notably longer than the 6 timestep delay demonstrated by the marginal likelihood method. The identification of a vehicle driving in the wrong direction also experienced a 4 timestep delay, along with a false negative period spanning from timestep 36 to timestep 50.

Performance trends become apparent when comparing various values of σ_ϵ^2. Decreasing σ_ϵ^2 is entailed by faster convergence of Σ_a^* to zero. This, in turn, allows the distance between the transformed latent representation and the origin to be estimated more accurately, resulting in a more reactive detector. However, this sensitivity increases the likelihood of false alarms, as can be observed from Fig. 4.11(b), where there are notable dips in confidence early in the time series, and the false negative period of the wrong-way driving trajectory lasts longer and hits higher confidence peak.

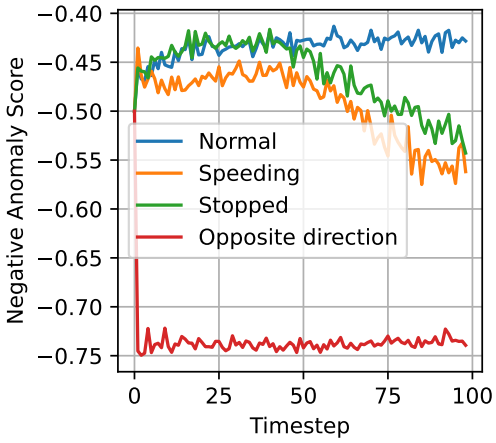


Figure 4.12: Anomaly scores computed in real-time with Isolation Forest

The Isolation Forest algorithm (Fig. 4.12) generally exhibits good detection performance alongside a reduced false alarm rate. However, the initial overlapping anomaly score values suggest that the initial dozens of timesteps can be disregarded. Interestingly, the system accurately identified instances of opposite direction driving from the onset. Furthermore, it correctly classified a suddenly stopped vehicle with approximately a 25 timestep delay, at around timestep 65. Regarding the detection of a speeding vehicle, the anomaly was recognized as early as timestep 10. This performance significantly outpaces the marginal likelihood and Bayesian detectors, which only activated at timesteps 35 and 56 respectively.

Table 4.2 provides a comprehensive comparison of the selected methods. Gen-

Table 4.2: Comparisons between different anomaly detection methods. AT: alarm time, FD: maximum false negative time period.

Method	Normal		Speeding		Stopped		Opposite	
	AT	FD	AT	FD	AT	FD	AT	FD
Marginal Likelihood	∞	0	35	0	46	0	1	0
Bayesian	∞	0	56	0	56	0	10	14
LOF	65; 96	4	25	0	48	50	1	0
Isolation Forest	∞	0	10	0	65	0	1	0

erally, both the marginal likelihood and Bayesian methods exhibit excellent detection rates. However, the Isolation Forest method is better than the others in terms of responsiveness. It's essential to remember that the FPCA-based detector incorporates measurement noise into its computation, requiring a greater number of observations for a reliable testing of hypothesis H_0.

4.5 DISCUSSION

This chapter introduces a comprehensive unsupervised framework designed for identifying abnormal road vehicle trajectories. The framework draws on two probabilistic methods: the marginal likelihood and the Bayesian posterior estimate. Detailed formulas pertaining to the Gaussian prior latent representation case and the Gaussian white noise measurement model, were presented. The entailed in-depth mathematical analysis provided grounds for a general anomaly detection framework for PCA and other representational methods, such as [30]. This complete the existing literature by offering a framework that can carry out real-time, online anomaly classification, in contrary to using a sliding window. Numerical simulations underscored the efficacy of the system in promptly detecting common road anomalies like speeding or wrong-way driving with a mere 4-16 time steps delay.

The marginal likelihood method and the Bayesian method operate within two distinct domains. The former approximates nominal behaviors and reconstructs them within the time domain, performing anomaly classification there. Given that the FPCA transformation is linear, this can be considered as equivalent to the marginal likelihood method detecting anomalies throughout the entire $\mathbb{R}^n$ space. Conversely, the Bayesian method seeks to classify anomalies within the representation domain. This is akin to projecting observations onto a subspace $\mathbb{R}^m, m < n$, and computing the anomaly score subsequently. Through this interpretation, it becomes clear that the marginal likelihood method offers quicker responsiveness

and is potentially less susceptible to false alarms, provided that the approximation or reconstruction (4.3) remains valid.

The framework presented in this chapter is not confined solely to the trajectories of road vehicles. Instead, it can be adapted to a variety of time series data, including aircraft trajectories in aviation or the detection of malicious attacks in computer networking.

Nevertheless, the framework has several identifiable shortcomings. One such limitation is that the formulas provided are only applicable in cases where there's a Gaussian prior over the latent representations. While this is a reasonable assumption for situations with a singular traffic mode, like highways, more complex scenarios such as road merges or intersections, may require modeling with a Gaussian mixture. The unsupervised nature of the system may also be perceived as a potential limitation. While the detector can discriminate if an observed trajectory deviates from what it learned from the training set, it lacks the capacity to explicitly categorize anomalies into specific groups such as wrong-way driving or speeding. It is projected that a machine learning classifier could be instrumental in addressing this scenario. Moreover, similar to the Fourier Transform, FPCA experiences the "window effect" when applied to finite time series, which means the approximation power diminishes near the trajectory's endpoint. This factor should be duly considered when constructing the road patch P.

4.6 CONCLUSION

In this chapter, we have introduced a novel unsupervised anomaly detection framework and provided an in-depth exploration of its mathematical properties. Our numerical simulations have convincingly demonstrated that this method exhibits a robust capability for detecting a wide array of anomalous driving behaviors, without the need for explicit teaching. Despite its current limitation to unimodal data, the framework offers significant potential in identifying anomalies within time series data.

Recognizing this limitation to unimodal data, our future work will be devoted to addressing this issue to extend the applicability of the framework. The proposed anomaly detection framework is sufficiently versatile to be relevant to a diverse range of other disciplines, such as aviation and computer networking. It is particularly pertinent to fields associated with human safety, where the requirement for a rapid, responsive detection system is paramount.

5

Spherical Codec for V2X Trajectory Compression: A Preliminary Study

Abstract

V2X holds enormous potential in augmenting road traffic safety by broadcasting information such as the position and velocity of the vehicle to others, rendering itself visible to the network even if it is occluded or still far away. However, regular broadcasting of information by many stations may heavily impact the V2X channel, leading to a high packet error rate due to collisions, reduced broadcasting range, delays, and compromises in the readiness of the channel to deliver critical safety information in time. Therefore, compressing these data is critically needed, especially in areas with high traffic density. This chapter presents an introduction to V2X and a novel algorithm, Spherical Codec, to help compress trajectory data. The codec introduces new data transmission schemes, adaptable to different implementations. By leveraging inference on the functional domain, it is possible to reduce the transmission frequency while sacrificing a similar amount of accuracy to the current ITS-G5 redundancy mitigation standard. As a numerical experiment, the Gaussian-based version of the codec demonstrated a reduction of up to 2 times improvement in total bytes sent, halved average channel loads, and marginal improvement in Packet Delivery Ratio at all transmission distances.

Contents

5.1 V2X

A derivative of this chapter, bearing the title "Spherical Codec for V2X Cooperative Awareness Trajectory Compression: A Preliminary Study," was officially published in the *Proceedings of the IEEE Vehicular Technology Conference (VTC) Spring* in 2023. The named contributors to this publication are Thinh Hoang, Vincent Martinez, Pierre Maréchal, and Daniel Delahaye. Replication of content is conducted under the rights accorded to the authors.

Portions of the material in this chapter have been permitted for disclosure in accordance with Patent Number EP22306821.4, titled "Optimization of Message Generation Frequency and Compression of V2X Data," and are represented herein under the applicable rights.

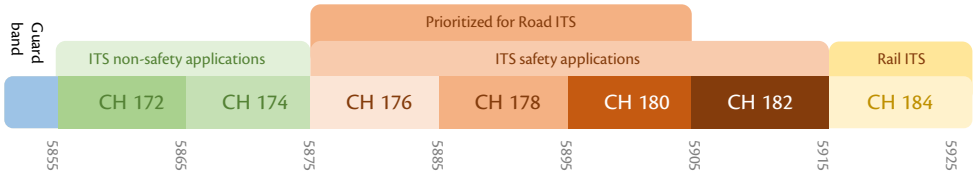


Figure 5.1: Allocated V2X channels and their uses, as of current standardization in 2023.

5.1.1 Introduction

The concept of V2X refers to any communication technology between a vehicle and another element of the ITS, including road-side units, smart traffic lights, or other urban traffic facilities. There are many technologies available; some compete, while others complement existing technologies. These operate on various frequency bands, follow different standardization processes, and have been independently tested and deployed in many different countries. Overall, these wireless communication technologies can be divided into two categories: short-range (DSRC), and cellular.

By enabling the exchange of information, safety systems can operate with a more extensive base of knowledge about the traffic environment, thus creating opportunities for better decision-making. It has been estimated that the enabling of V2X can eliminate 80% of current road crashes [116]. Despite recent enhancements in cellular network technology, DSRC allows an ad-hoc communication scheme without the need for the availability of a nearby base station. This makes DSRC more prepared for deployment and enables it to function even in rural and remote areas where cellular coverage is scarce. In this thesis, we will focus strictly on DSRC and, more specifically, the European standard of DSRC also known as ITS-G5, standardized by the European Telecommunications Standards Institute (ETSI).

5.1.2 ETSI ITS-G5 V2X

In Europe, the frequency spectrum dedicated to DSRC ranges from 5470 MHz to 5925 MHz and is designed based on IEEE 802.11p, using carrier sense multiple access with collision avoidance (CSMA/CA). The day-one deployment of ITS-G5 consists of three distinct groups of operations operating on different bands (Figure 5.1):

- **5470 MHz - 5725 MHz (ITS-G5C) [117]:** Allocated for V2I communications,
- **5795 MHz - 5815 MHz:** Comprising four 5 MHz channels, dedicated to road

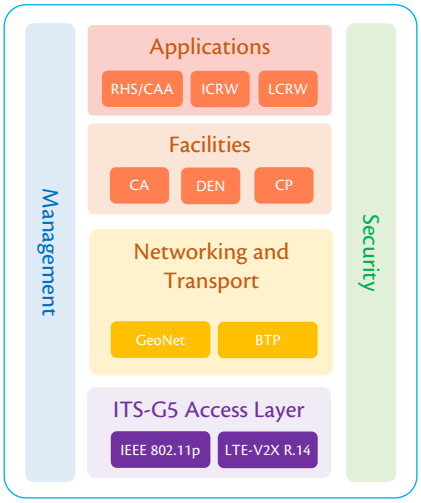


Figure 5.2: Architecture of the ETSI ITS-G5 stack. **Abbreviations:** RHS: Road Hazard Signaling, ICRW: Intersection Collision Risk Warning, LCRW: Longitudinal Collision Risk Warning, CA: Cooperative Awareness, DEN: Decentralized Environmental Notification, CP: Collective Perception.

transport and traffic telematics,

- **5855 MHz - 5925 MHz (ITS-G5A and ITS-G5B):** Comprising a 10 MHz control channel and six other service channels of the same bandwidth. Most ITS applications will operate on this band. Non-safety applications will occupy the band from 5855-5875 MHz, while safety applications will operate on 5875-5905 MHz. The rest is dedicated to future applications [118].

Figure 5.2 depicts the fundamental architecture of ETSI ITS-G5. It begins with the access layer at the lowest level, incorporating both the physical and data link layers. The stack architecture is agnostic to the access layer, meaning that all services are guaranteed to perform in the same manner whether the underlying access technology is based on IEEE 802.11p or 3GPP LTE.

All ITS stations are required to comply with the basic set of services located at the facility layer, including the generation and handling of cooperative awareness messages (CAMs) and Decentralized Environmental Notification Messages (DENMs), which are emitted on the 5855-5925 MHz band. In essence, all ITS stations are required to continuously beacon their position and status to the network, as well as have the capability to emit and handle notification messages regarding potential road hazards. This dissertation focuses specifically on Cooperative Awareness Messages and attempts to solve several problems pertaining to the current way CAMs are handled according to the standards [119].

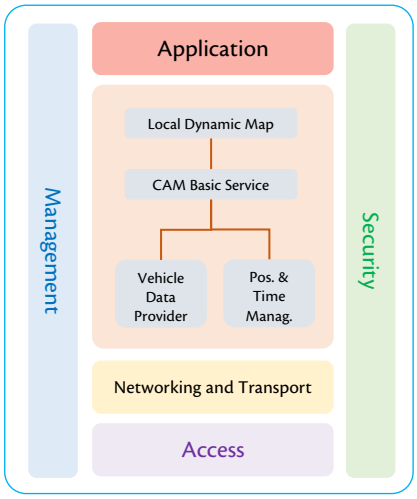


Figure 5.3: Architecture of the CA service, located in the Facilities Layer.

5.1.3 *Cooperative Awareness Message (CAM)*

As suggested by their name, CAMs are standardized messages exchanged between various road participants in the ITS, with the goal of advertising their presence. Through CAMs, other entities on the road become aware of each participant, facilitating cooperation between them. The exact content of a CAM depends on the specific type of station, whether it's a vehicle or a roadside unit, as well as its sensing capabilities. Nevertheless, most CAMs will convey basic information such as the ID, time of emission, motion status (position, speed, acceleration), as well as the station's physical attributes such as dimensions, vehicle type, role, and the characteristics of equipped sensors.

The CA service (Figure 5.3) organizes the generation of CAMs in the ETSI stack. By design, the CA service does not interact directly with any application from the layer above. Instead, the perception of road objects is relayed through the Local Dynamic Map (LDM). The LDM serves as an endpoint for the assimilation of all sensor data, including data from one's own sensors, as well as from V2X messages. The CA service retrieves information about the vehicle status from the Vehicle Data Provider (VDP). The Position and Time Management (POTI), on the other hand, provides the synchronized time and position of the vehicle.

The CA service can be further decomposed into four functional blocks which perform separate functions:

1. **Encode CAM:** to correctly encode the message according to the standard, described in the corresponding ASN.1 data structure, per the Packed Encoding Rules (PER) [120],

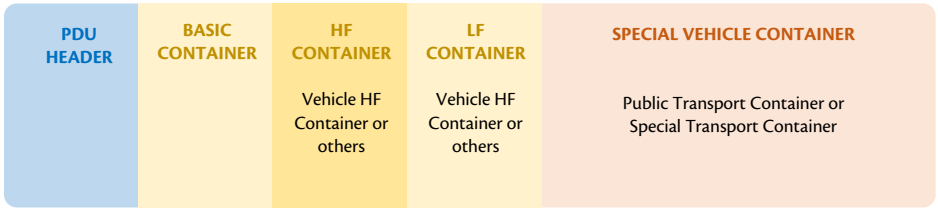


Figure 5.4: General format of a CAM.

2. **Decode CAM:** to decode the received CAM,
3. **CAM Transmission Management:** this function manages all aspects related to the generation of CAMs, including the beacon frequency, as well as initialization and termination processes,
4. **CAM Reception Management:** this function serves the decoded CAM to the LDM or other applications at the application layer that are interested in CAMs. It may also verify the validity of CAMs, as well as handle all errors and exceptions, though this is not strictly required by the standard.

More specifically, CAMs should only be emitted on the control channel and be single-hop only. CAMs should never be forwarded to other stations. As long as the station is functional, CAMs should be transmitted with respect to the following limits:

- CAMs should be generated less frequently than the minimum interval of 100 ms, or 10Hz,
- CAMs should be generated more frequently than the maximum interval of 1000 ms, or 1Hz.

The precise interval should be determined according to the specific use case, which could depend on the station type, the dynamics of the sensed data, as well as the channel congestion status (CBR, for instance). DCC will generally set an upper bound for the message generation frequency, and an additional condition can be used to check for the triggering of a CAM generation, including:

- The difference in heading should exceed 4^o,
- The difference in position should exceed 4m,
- The absolute difference in speed should exceed 0.5m/s.

There are generally two versions of CAMs that can be generated. A shorter one contains only mandatory and highFrequencyContainer information as specified below, while a longer one also includes a lowFrequencyContainer. The long CAM generation interval should be at least 500ms. This helps prioritize important

information and prevents the channel from being overwhelmed with information that is not deemed time-critical, such as information about a vehicle's lighting status (Figure 5.4). In the following, we will briefly describe the information field required by each container for the case of a vehicle.

1. **PDU header:** containing protocol version, type of message and the ID of the station that generates the message,
2. **Basic container:** containing the type of originating station (vehicle or roadside unit), and the geographical position of the station. This container is mandatory for all stations participating in the network,
3. **Vehicle high-frequency container (for vehicle stations):** also mandatory for all vehicles, containing fast-changing information like heading and speed,
4. **Vehicle low-frequency container (for vehicle stations):** optional, containing slow-changing or not so important information such as vehicle status or lighting status. It was also specified in [119] that the LF container may also contain many past data trajectory data points, with each account for 8-9 bytes of the packet size [121]. It will be obvious later that the Spherical Codec makes the transmission of these past data points redundant,
5. **Special Vehicle container:** reserved for special vehicles like buses, trams, vehicles requiring prioritization such as ambulance, police, or transporting dangerous goods.

A more detailed description of ETSI CAM can be found in [119].

5.1.4 Collective Perception (CP)

With Vehicle-to-Everything (V2X) Cooperative Awareness (CA), a multitude of road actors, including vehicles, roadside units, and traffic infrastructures, can disseminate information and collaborate wirelessly. Nevertheless, despite the comprehensive range of use cases the CA encompasses, inherent technical complexities have propelled researchers to extend this concept further. For instance, roadside units can identify pedestrians and other entities not equipped with V2X technology, a capacity that is paralleled by radar-equipped vehicles. Consequently, a requirement arises to distinguish the information transmitted by the vehicle itself from that relating to road objects it can detect, thereby stimulating the advent of Collective Perception (CP).

The impetus for CP also stemmed from evidence suggesting that fatalities often originate from collisions involving vehicles and pedestrians or cyclists, as opposed to vehicular crashes [122]. On one hand, it remains highly impracticable to mandate pedestrians, cyclists, or motorcyclists to deploy V2X communication

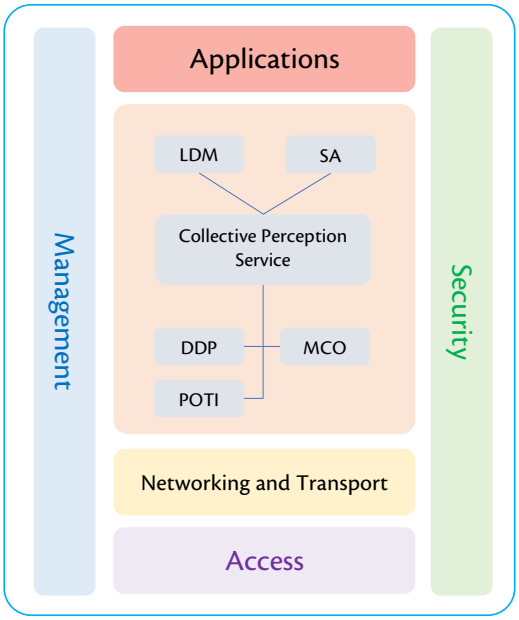


Figure 5.5: Architecture of the CPS [2].

devices, thus leaving these road entities virtually invisible to the V2X network. Conversely, most modern or prospective vehicles, such as the Tesla 3, already possess an intricate radar system capable of detection, classification, and tracking with high precision. It is logical for these radar hosts to relay information concerning these road entities to the V2X network akin to traditional CA objects. Nevertheless, with sensed information, noise and correlations are innately linked to the estimate. Furthermore, duplication could occur if an object is sensed by multiple hosts, which exacerbates the complexity of the situation. Consequently, it is unfeasible to extend the current CA service for the transmission of this information. It is evident, therefore, that a novel form of V2X service must be developed.

5.1.5 *Collective Perception Service*

As shown in [2], the goal of CPS is to "share information about perceived objects (such as vehicles, pedestrians, animals and other collision relevant objects) and perception regions (road regions that allow receiving ITS-Ss to determine unoccupied regions) in the local environment."

Like CA, the operation of CP is managed by CPS, which is in charge of CP messages (CPM) generation and reception. The general architecture of the CPS in the ITS-G5 stack is shown in Figure 5.5. The key components of CPS include:

1. **Local Dynamic Map (LDM):** Acting as an information reservoir, the LDM

gather data from an array of sources including CAMs, CPMs, and radar measurements. Central to the operation of the LDM is the concept of state estimation. Furthermore, the LDM functions as the pivotal node for various Advanced Driver Assistance Systems (ADAS) components, as well as the Human Machine Interface (HMI), thereby facilitating the display of pertinent information to the human operator.

2. **Device Data Provider (DDP):** This module provides a comprehensive set of state information, encompassing aspects such as the vehicle's position (latitude, longitude), velocity, acceleration, and status of lighting and braking mechanisms. While a duplicate of this data is likewise found in CAMs, the generation of CPMs requires a number of derived quantities, such as relative velocity, serving as intermediate variables.
3. **Position and Time Management (POTI):** As the moniker suggests, this module is responsible for delivering real-time positional data, in addition to synchronized time utilized for timestamping the CPMs.
4. **Service Announcement (SA):** This module's objective is to signal other stations regarding the host station's capability to generate CPMs.
5. **Multichannel Operation (MCO_FAC):** This pertains to the implementation of multi-channel operation to maximize the effective utilization of channel bandwidth.

5.1.6 Illustrated Use Cases for CP

As expounded upon in [3], the objective of CPS is to enhance the perceptual capacity of the CA Service by integrating data collected from vehicular sensors, such as radars and video cameras. This section encapsulates the scenarios in which CP proves beneficial through multiple illustrations.

The standard use case of CP, illustrated in Figure 5.6, primarily aims to disseminate sensor-derived information regarding non-connected road entities, such as pedestrians and cyclists, among other V2X participants. This information typically includes basic attributes like object size and position, as well as dynamic parameters such as speed and acceleration. It should be noted that the inclusion of road objects capable of beaconing CAMs in the CPMs is contingent on the specific implementation of the data fusion module.

Figure 5.7 presents a similar scenario. Here, CPMs serve to render inanimate road impediments visible to the V2X network. Detection encompasses diverse road objects, including but not limited to potholes, puddles, tree branches, traffic cones, road barriers, fences, speed bumps, guardrails, and streetlights. Advanced recognition systems installed on equipped vehicles facilitate the detection of these

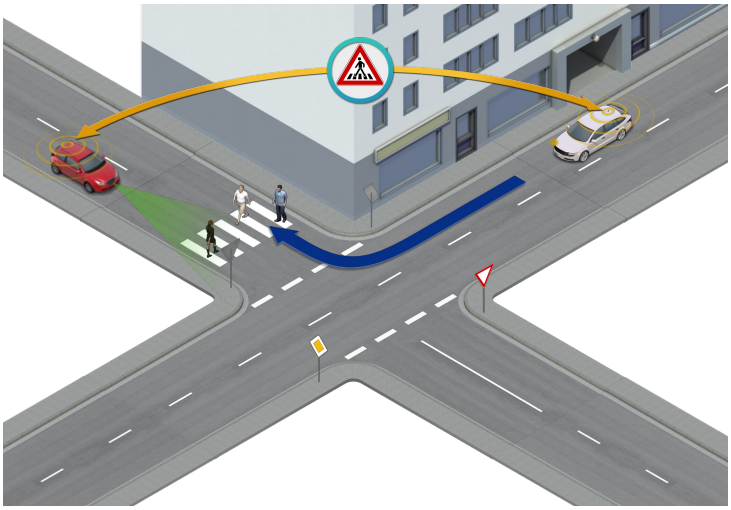


Figure 5.6: Awareness of non-connected objects. In this scenario, pedestrians are not equipped with a V2X transmitter, but their presence can be detected with radars from a vehicle. This information is relayed to other network participants via the sending of a CPM [1].

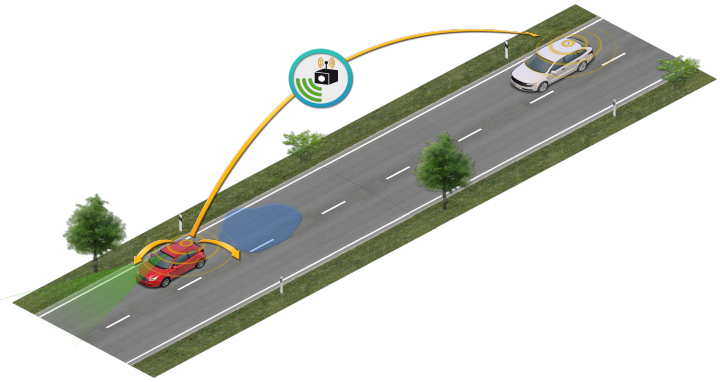


Figure 5.7: CPMs also convey information regarding hazardous road objects such as potholes, puddles, tree branches or other road equipment [1].

objects, and the resultant benefits are accessible to regular vehicles equipped merely with a simple V2X transmitter.

As depicted in Figure 5.8, the considered use case involves a roadside unit, specifically affixed atop traffic lights, incorporating radars and a V2X transmitter, and demonstrating the capacity to receive CAMs. This unit gather data obtained from CAMs and its own radar-derived measurements, thereby enhancing the precision of state estimates encompassing position, velocity, and acceleration. These so-called "aggregate stations" act as decentralized nodes for data fusion.

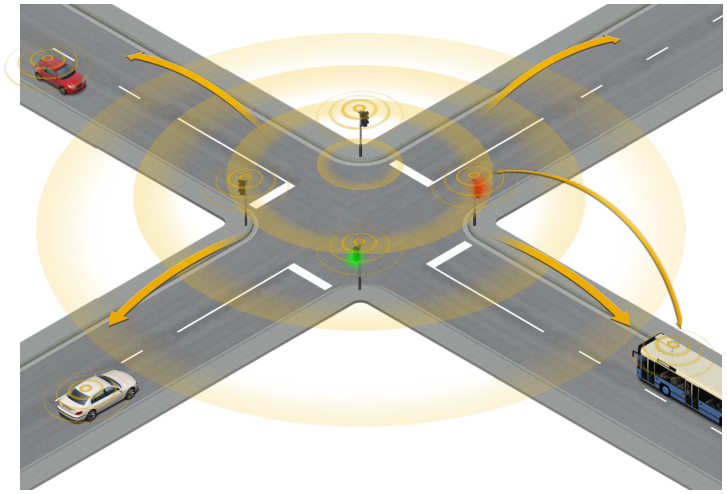


Figure 5.8: CPMs can also be broadcast by aggregating information, including from CAMs and proprietary sensors. This is usually done at RSUs stationed at locations that usually witness complex traffic situations such as intersections where the position and dynamics of vehicles needed to be accurately tracked [1].

5.1.7 Data Feeding

The previous discussion put great emphasis on data fusion in the transmission and reception of CPMs. In fact, data fusion prepares the information for CAM/CPM generation, and processes the raw data from received CAMs/CPMs. It is customary, but is almost always by design that the data fusion module will update the LDM database, from which information will serve different endpoint modules, including displaying to users, generation of new CPMs, or other safety applications such as dangerous driving behavior detection (Figure 5.9).

5.1.8 Standardized ITS-G5 CPM Format

In the following section, we offer a short introduction to the types and formats of data typically found in a generic CPM, thus providing insight into the nature of information that is transmitted within the V2X network via CPM. Detailed specifications can be found in [2].

Every CPM must contain the Management Container, while all other fields are optional (Fig. 5.10). The inclusion of relevant data fields is determined based on the specific use case, and whether the host is a vehicle or a roadside unit.

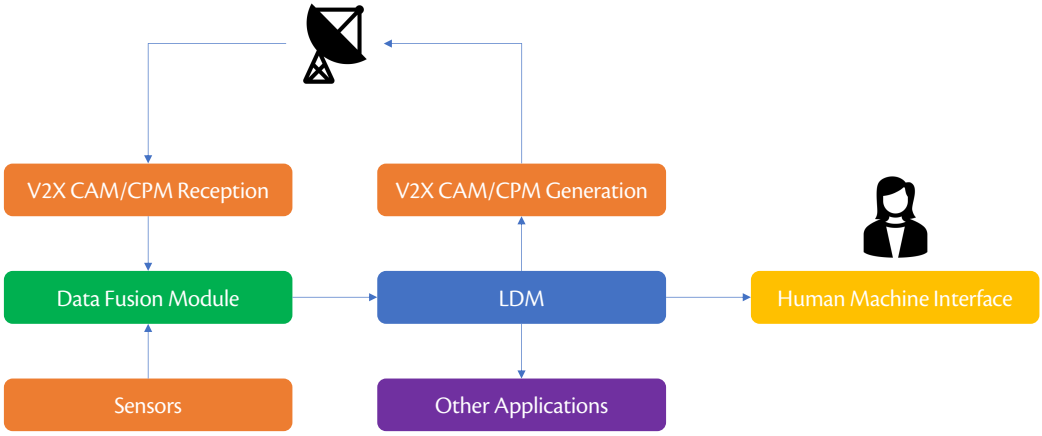


Figure 5.9: Functional relations between data fusion, LDM and CPS in most designs. Alternative designs exist, for example, separate perception database for data fusion can be used when it is not helpful to incorporate own sensor data into improving the estimates from V2X messages.

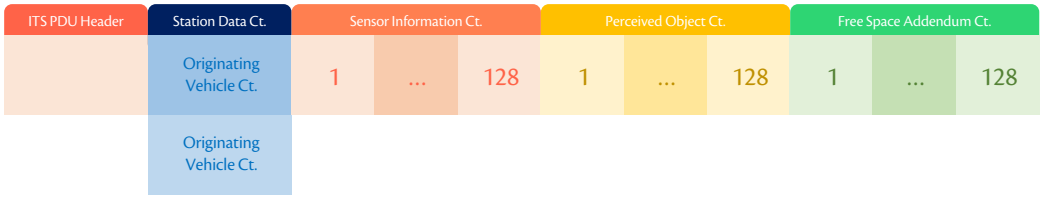


Figure 5.10: Format of a general CPM [2].

Management Container

This sequence contains 3 fields:

1. **stationType:** Specifies whether the station is a vehicle or a roadside unit,
2. **perceivedObjectContainerSegmentInfo:** Denotes the number in the sequence, applicable when the original CPM is excessively long, necessitating its division into multiple segments,
3. **referencePosition:** Refers to the host's reference position, which is generally the center of the vehicle.

Station Data Container

The host's identity, as a vehicle or a roadside unit, dictates the appropriate container to be used. We will primarily concentrate on the vehicle scenario.

1. **heading:** Represents the vehicle's heading in relation to the magnetic North.

2. **driveDirection:** Preset as forward. If the vehicle is driving in reverse, it is indicated here.
3. **acceleration:** Recorded via longitudinal, lateral, and vertical acceleration value fields.
4. **yawRate:** Documents the rate of yaw.
5. **roll, pitch angles:** Supplementary to determine the vehicle's attitude. These fields are optional.
6. **vehicleLength, vehicleWidth, vehicleLength:** Used to describe the dimensions of the vehicle within a 3D space.

Sensor Information Container

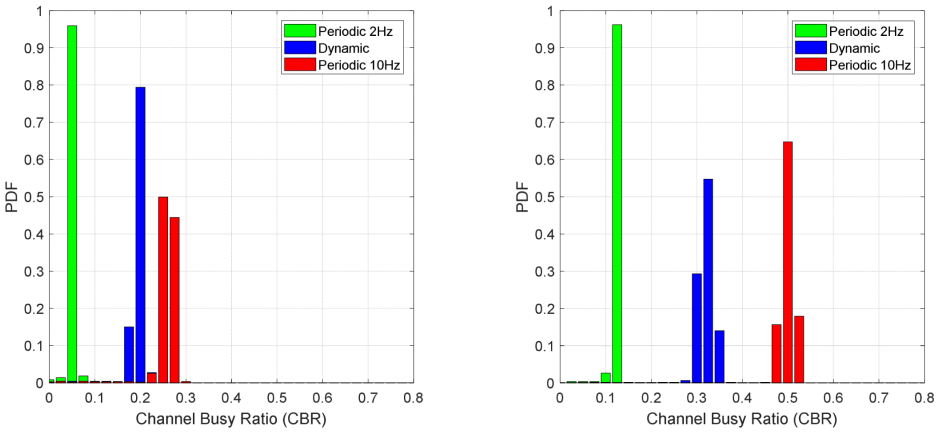
This sequence encompasses fields illustrating the vehicle's sensing capability, which includes the sensor type, detection range, and shape of the view area.

1. **sensorID:** Specifies the unique ID of the sensor.
2. **type:** Distinguishes between radars, cameras, and so forth.
3. **detectionArea:** Describes the sensor's area of detection and its range.
4. **freeSpaceConfidence:** Addressing a common issue with sensor data, which is the ambiguity of whether a non-detected object was absent or outside the sensor's perception range, the freeSpaceConfidence container contributes additional information regarding the area where the sensor can confidently report the absence of any object.

Perceived Object Container

This particular container constitutes a significant proportion of each CPM's overall size. Some selected fields are:

1. **objectID:** The unique identifier allocated to the detected object,
2. **timeOfMeasurement:** The time at which the object was sensed, offering other parties a gauge of whether the information remains current,
3. **objectConfidence:** The spatial confidence interval signifying the uncertainty inherent in the measurements,
4. **xDistance, yDistance, zDistance, xSpeed, ySpeed, zSpeed, xAcceleration, yAcceleration, zAcceleration, yawAngle:** These represent the position and dynamic parameters of the sensed object.



(a) Low traffic density **(b) High traffic density**

Figure 5.11: Simulated Channel Busy Ratio (CBR) of Highway Scenarios with CPM Transmission set to different beacon frequency values [3].

Free Space Addendum

In a similar vein to the freeSpaceConfidence, this addendum imparts additional details concerning areas devoid of any object. A more comprehensive explanation of the format is accessible in [3].

5.1.9 *Message Generation Paradigm*

A central challenge with Collective Perception is its substantial bandwidth requirements for standard operations. This is anticipated considering that, unlike a CAM, which only conveys information regarding the host, a CPM transmits roughly equivalent (or more) information pertaining to each road object - and such objects are typically numerous within the field of view. Consequently, channel congestion has emerged as a critical concern.

Figure 5.11 visually elucidates the aforementioned problem. In a standard highway context, CPS was found to exert a considerable strain on channel resources, driving the CBR near to 0.5 at the typical beaconing frequency of 10Hz [3]. This burden is somewhat alleviated by the DCC's directive for CPS to decrease the message rate. However, other potential solutions to mitigate this issue encompass:

1. Modifying the message rate for a specific object if it has been referenced frequently enough on the channel (Frequency-based Redundancy).,
2. Modifying the message rate for a specific object if the dynamic state of the

object displays minimal variance from its previous value (Dynamics-based Redundancy & Distance-based Redundancy),

3. Modifying the message rate for a specific object if the local estimate of the object's state is not superior to what has been transmitted (Confidence-based Redundancy).
4. Modifying the message rate for a specific object if the information increment is stagnant (Entropy-based Redundancy).

and these are on top of the strict requirements: "The minimum time elapsed between the start of consecutive CPM generation events should be equal to or larger than T_GenCpm. T_GenCpm is limited to $T_{GenCpmMin} \leq T_{GenCpm} \leq T_{GenCpmMax}$, where $T_{GenCpmMin}$ = 100 ms and $T_{GenCpmMax}$ = 1000 ms."

However, simulation results have indicated a potentially severe issue with CP, namely, its significant consumption of channel resources. Given the challenge of increasing the allocated spectrum for ETSI ITS-G5, this issue warrants a comprehensive investigation into strategies aimed at further reducing message transmissions, thereby saving bandwidth and resources. In addition, there is a need for advanced data compression techniques to maintain the quality of service for numerous road participants, especially as V2X adoption continues to rise.

The following sections of this chapter, while mainly focusing on the transmission and reception of CAM, will demonstrate that the methods used for transmission rate reduction and data compression can be similarly implemented in CPM with minimal modifications. The root of this adaptability lies in the fundamental implementation process, which involves the application of the codec to each individual road object encapsulated within the CPM. The Spherical Codec emerges as a solution attempting to concurrently address both the challenge of congestion mitigation and the requirement of data compression.

5.1.10 Related Works about Congestion Mitigation for ITS-G5

Standardization Efforts

As CAMs and DENMs are transmitted on the same privileged channel for road safety, known as the Control Channel (CCH), the problem of channel congestion becomes significant, especially in high vehicle-density scenarios. As the channel becomes increasingly crowded, the risk of packet loss increases, potentially having serious safety implications. Various approaches have been proposed, and standardization efforts have evolved since 2011, with the inclusion of cross-layer Decentralized Congestion Control (DCC) described in the ETSI TS 102 687 standard [123].

Figure 5.12: ETSI DCC states. Each state can only be reached by its neighboring state.

Overall, DCC comprises multiple modules and affects many different layers of the ITS-G5 stack. In the access layer, DCC can limit the transmit power, transmit rate (TPC, TRC, and TDC respectively). In particular, the transmit rate is modified by targeting the average transmit power per packet. TRC, on the other hand, attempts to modulate the ratio of time the node is in transmission mode over the total operation time. Lastly, the data rate is the control signal for TDC, which uses a queue and assigns priority to individual packets.

In the new release of TS 102 687 [123], several new concepts were introduced in DCC, including the division of the channel status into different states corresponding to the level of congestion, measured by the Channel Busy Ratio (CBR), which is the amount of time the channel was sensed to be busy (Figure 5.12). The busy state indication is made available to other components of the DCC, including those located at the facility layer such as the CAM service. As a result, the CAM generation rate can be adjusted accordingly to the CBR measure.

In another proposal, called adaptive DCC, the CBR estimate is smoothed using a moving average filter, then the parameter δ representing the amount of time that the ITS station can occupy the channel is computed. This operational mode borrows ideas from LIMERIC, which will be presented later. The goal of adaptive DCC is to introduce a feedback control mechanism to maintain the CBR at a specific predefined target. Packets will be withheld at the network layer and cannot be delivered to the access layer if the fraction of time utilization of the channel exceeds δ.

Although DCC has addressed the problem of channel congestion to a large extent, many researchers have identified several significant limitations associated with very high-density traffic scenarios.

Limitations of Standardized DCC

In [124], it was demonstrated that the standardized DCC is plagued by two major categories of drawbacks: unfairness and oscillation. Unfairness describes a scenario where two different vehicles operating on the same channel can make disparate decisions concerning the choice of transmission parameters, specifically in terms of power and data rate. This phenomenon was investigated in [125], where a series of experiments were conducted with each vehicle independently transmitting ran-

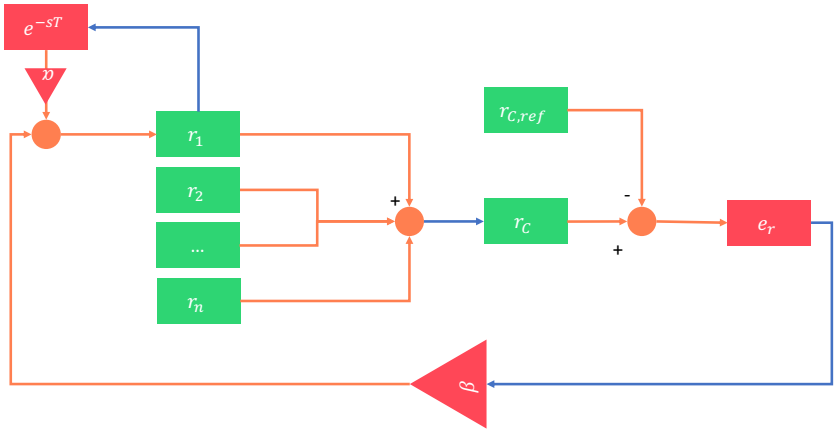


Figure 5.13: With feedback mechanism, LIMERIC can use the individual message rate r_k to control the CBR to the target reference value $r_{C,ref}$ using the error feedback $e_r = r_C - r_{C,ref}$. The gains α, β help the system converge, as well as stabilizing the estimate of the CBR.

dom CAMs at different moments, within a synchronized time window across all vehicles. The study found that the width of this time window led to unfairness. Thus, the phenomenon can be attributed to the asynchronous measurement of the CBR, resulting in one vehicle preemptively switching to a more conservative transmission setting to resolve the issue for both. Unfairness not only led to erratic transmission settings but also throttled the messages more than necessary [126].

Oscillation refers to the phenomenon associated with proactive DCC design, in which the sensed CBR is used to segregate the congestion control operation into several discrete states. The rate control creates a dynamic equilibrium in which the state can oscillate between less and more restrictive settings. [127] and [128] demonstrated that the combined effects of these two phenomena could result in detrimental impacts on the timely dissemination of crucial traffic information or the range of safety messages. In restrictive settings, notifications may not reach vehicles beyond their immediate neighbor.

Other Congestion Control Techniques

In this section, we only review key ideas in the development and evolution of DCC designs described in the comprehensive review of [124]. Perhaps the most influential DCC algorithm is LIMERIC [129] (Figure 5.13), where the CBR is a target value controlled by the adjustment of the message rate. This adjustment is achieved through the introduction of a feedback loop and a moving average filter on CBR estimation. Evaluations on the Inter Packet Gap (IPG) and the Tracking Error (TE) in simulations show a remarkable improvement over ETSI's proactive DCC.

In [130], a simulation scenario was created in which there is a mixture of ETSI proactive DCC vehicles and LIMERIC enabled DCC vehicles. This simulation revealed that the former group of vehicles might experience a significant deterioration in performance across all metrics. A workaround was proposed by adopting the Channel Busy Percentage (CBP), which correlates with the vehicle density, instead of using a fixed value of CBR.

Like ETSI DCC, the LIMERIC algorithm could potentially suffer from the problem of unfairness. To address this issue, [131] proposed a Periodically Updated Load Sensitive Adaptive Rate Control algorithm (PULSAR). The algorithm, like LIMERIC, adjusts the transmission rate based on the error between the reference CBR and the current CBR, but it introduces an additional two-hop piggybacking mechanism to enforce collaboration between stations located within the Carrier Sense range. The same line of thought was dedicated to designing an extension of LIMERIC, where an additional constraint regarding the fair share of the channel was added.

Further study of the problem revealed that the unfairness issue shares a similar cause with the hidden terminal problem [132]. This realization motivated another variant of LIMERIC, called cooperative LIMERIC. However, simulation results have demonstrated negligible improvement over the Probability of Packet Reception (PPR).

Another issue that has arisen with LIMERIC is the problem of scalability. It has been discovered that the convergence of LIMERIC is not guaranteed in scenarios with high vehicle density. This was attributed to the rigidity of the selected parameters, such as the feedback gain. This prompted the proposal of a Self-Weighted Rate Control [133].

An alternative approach involves regulating the data rate, as opposed to the transmission rate. In simpler terms, data rate control aims to shorten messages, thus reducing the amount of time they occupy the channel during transmission. The standardized CA service [119] has already proposed transmitting two versions of CAMs: one containing only essential safety information, such as position, heading, and velocity; and a full version that is broadcast less frequently but provides more detailed information about the vehicle's state, such as lighting status. In [134], a DCC algorithm was proposed that uses not just the CBR, but also the packet count number. Simulation results have shown promise, as this approach managed to outperform LIMERIC.

In [135], a scheme called Distributed Fair Power Adjustment for Vehicular Environment (D-FPAV) was derived to adjust the broadcasting power and to introduce an adaptive retransmission scheme with the goal of increasing the reachability of important messages while keeping channel congestion under control. However, this approach is generally considered challenging due to the rapid evolution of network topology [124].

Recently, improvements in machine learning have shed new light on the problem of congestion control. In [136], the authors proposed allocating more channel resources to vehicles that have larger link weights, which are quantified based on the number of neighboring vehicles and the quality of connections. These quantities mainly depend on the line-of-sight that the "ego" vehicle establishes with its neighbors. Non-line-of-sight status is detected using a Support Vector Machine (SVM). Another approach involves employing reinforcement learning for resource allocation, the details of which can be found in [137].

5.2 DATA COMPRESSION

While DCC generally operates at a lower level and is typically insensitive to the specific type of data being transmitted, dictating only the transmission rate, there is a clear need for a more efficient scheme. Just as video data should be transmitted differently than text or images, considering the nature of the data being transmitted in ITS (primarily trajectory data) could significantly help in the design of a better communication protocol. In the next section, we offer a brief overview of data compression, as well as a review of compression methods for trajectory data.

5.2.1 *Primer*

Data compression (DC) aims to reduce the storage size needed for data or decrease channel utilization in the context of communication. By minimizing the space required for data storage and streamlining transmission speed, DC lessens the burden on storage mediums and simplifies data management.

The field of DC is well-established and robust, rooted in a rigorous mathematical foundation derived from Information Theory. For successful data compression, there needs to be either statistical redundancy, irrelevance, or both within the data. Essentially, raw data must exhibit some repetitive attributes, or some redundancy that allows a portion of information to be omitted without significantly affecting the reconstruction process. For instance, image compression is achievable because typical photographs contain recurring patterns, such as uniformly colored objects or distinct edges separating the elements within the scene. Irrelevance usually pertains to noise, a common issue with sensors. Eliminating this undesired noise reduces the data's size and enhances the quality of the necessary data.

Data compression techniques can be categorized using various criteria [138]. For instance, they can be classified as either lossy or lossless (Figure 5.14), depending on whether the reconstructed data exactly replicates the original signal. The MP3 data format is an example of a lossy compression technique, which introduces slight distortion to the original sound during playback. However, this difference

Original Photo (Raw) | Lossless Reconstruction (PNG Format) | Lossy Reconstruction (JPEG With CR = 4)

Figure 5.14: Lossy and lossless reconstruction of the original signal (an image) after compression. Notice that in lossy compression, the quality of the signal degrades; while in lossless compression, the quality stays the same.

or error is often too subtle for human detection. Conversely, text and spreadsheet compression must be lossless, as any alteration in context could have significant implications.

Various coding schemes underpin numerous data compression algorithms, including Huffman [139], Arithmetic [140], Lempel Ziv [141], and Burrows Wheeler Transform [142], among others. Almost all coding schemes aim to assign shorter codewords, which are simply binary strings, to more frequently observed patterns. Huffman coding, the optimal prefix code, is widely used in all lossless data compression, with numerous variations available. Each variation strives to optimize a different aspect of the original code, such as compression/decompression speed or memory usage.

The scenario becomes more complex with lossy data compression, as there is usually a need for a metric to gauge how closely the reconstructed data being similar to the original. Numerous metrics for images and sounds have been proposed, including L^1, L^2, and Total Variation. However, it remains unclear which metric best aligns with human perception. Coding schemes can also be categorized into predictive or transformative. Predictive schemes derive a predictor from a mathematical model integrated in both the encoder and decoder, and only the discrepancy between the model's prediction and the actual data is encoded. Naturally, the more accurately the model can predict, the less information remains to be encoded. In contrast, transformative schemes depend on a transformation (typically Discrete Wavelet Packets or Discrete Cosine Transform [143–145], as with MP3, MP4, JPEG, etc.) to represent the data in a lower dimensional space. Subsequently, the representation is quantized, and the encoding process ensues.

Given its close association with statistical inference, dimensionality reduction, and unsupervised learning, this chapter addresses the compression problem in a

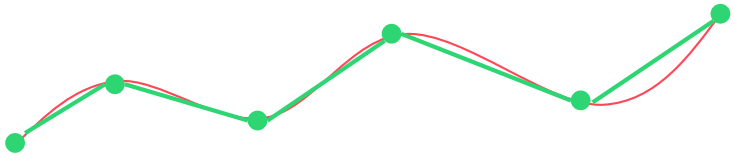


Figure 5.15: The Douglas-Peucker approximates any curve (represented by a discrete high dimensional vector - in red) with a set of points such that the lines (in green) connecting these points lie closest to the original curve.

broader context, covering both the predictive and transformative branches of the coding scheme.

5.2.2 *Trajectory Compression*

Earliest attempt to solve the line generalization problem can be traced back to the problem defined by Stone: given a continuous function $g(x)$, find the constants a_k, b_k and the points of division $u_1, \cdots, u_{N-1}$ such that the function $J = \sum \int (g(x) - a_k - b_k x)^2$ is minimized. Bellman came up with the solution to the problem using dynamic programming, which has a computational complexity of $O(N^3)$ [146]. In 1973, another popular trajectory compression algorithm was proposed by Douglas and Peucker [147]. The idea was to start from the first and last points of the trajectory in the set. Then, gradually, one point at a time is added to the set, resulting in an increasing number of segments so that the trajectory is "closest" to the original trajectory in a perpendicular sense (Figure 5.15). However, these techniques do not consider the temporal dimension; thus, Meratnia and Rolf [148] introduced another algorithm using an "opening window" to enclose as many points of the original trajectory as possible. The spatial distance metric was replaced with Synchronized Euclidean Distance (SED) to account for the temporal dimension. Another approach was to use dead reckoning from velocity [149]. The algorithm had lower complexity of $O(n)$ and was capable of online compression. In [150], a priority queue was introduced to optimize the SED metric locally by dropping points with the lowest SED when the buffer is full. Due to the limited points being taken into account in each optimization attempt, the algorithm has lower complexity of $O(\log n)$. The method was extended in [151] to allow control of both the compression ratio λ and the error ϵ. These approaches' drawback is that they work best with trajectories close to linear. However, real-world trajectories are often smooth curves, depending on road topology and multiple other factors; hence, line generalization often incurs a large error to reach a target compression ratio.

Another class of compression algorithms for trajectories is called delta compression. A predictor is usually used to extrapolate from the current point, and only

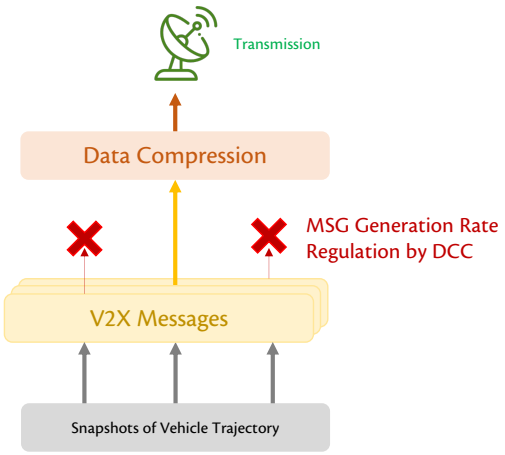


Figure 5.16: Usual approach found in the literature to reduce channel utilization of trajectory information transmission.

the difference against the extrapolated value is encoded. The difference is zero for linear trajectories, and for general trajectories, the difference should be close to zero. This number is encoded by a leading zero encoding scheme and can be lossy or lossless. Two influential papers in this category include TrajStore [152] and Trajic [153]. Unlike the line generalization class, delta compression usually achieves a better compression ratio at the same error tolerance level. The performance of delta compression depends strongly on the predictor, with various linear predictors proposed: [152] used a constant predictor, [153] used a temporally-aware linear predictor to account for different sampling rates. Nonlinear predictors such as splines [154] have also been studied, but computational complexity is often higher than linear class ones.

In [155], the authors conducted a series of experiments to study the effect of compression on CAMs, CPMs, and CMMs. Entropy coding with the Shannon-Fano algorithm on 12-bit symbols reduced the message size to 25% on average for all types and 50% for CPMs in particular. Adaptive Dictionary methods such as GZIP had an adversary effect on the message size. It was also found that Channel Busy Ratio (CBR) also improved by activating compression, and so was Packet Delivery Ratio (PDR). However, the algorithms presented attempted to compress standard V2X messages rather than the trajectory data. Thus, the predictability of trajectory was overlooked. Because compression took place on the Facility Layer, the compression had little effect on the header size.

A thorough literature review on the functioning of congestion control mechanisms and trajectory compression reveals that these two concepts seemingly address the same problem, but from distinct perspectives (Figure 5.16). This is interesting considering that they share a common mathematical framework as their

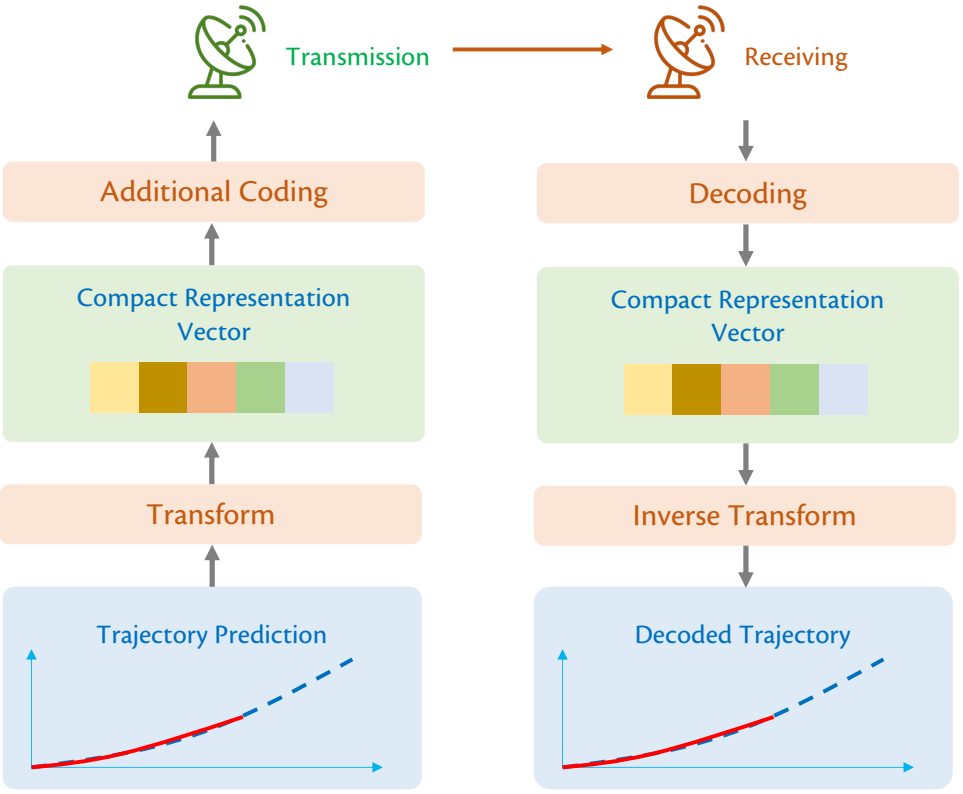


Figure 5.17: The new approach proposed in this dissertation serves a dual function. The predictive compression scheme also controls congestion. Information is only required to be retransmitted if the deviation from the prediction is significant.

base. However, it is important to note that while statistical redundancies within the message content can be eliminated with a data compression algorithm, as outlined in [155], the substantial redundancy inherent in trajectory data itself has not been effectively exploited.

Specifically, the strong correlation among various variables characterizing the trajectory – such as position, speed, and heading values at neighboring time steps – hasn't been sufficiently leveraged. Current V2X messages only deliver a "snapshot" of the available information for transmission at any given moment, and the receiving end doesn't take further steps to decrease the workload on the transmitting end. This disparity underscores the substantial gap between these two approaches. This chapter seeks to bridge this gap by introducing two novel codecs that function both as congestion controllers and data compressors, thereby brings about the convergence of congestion control and data compression.

5.3 INTRODUCTION TO SPHERICAL CODEC

To our knowledge, the Spherical Codec presented this chapter is the first attempt to design a data compression algorithm that explicitly exploits the fact that trajectories inherently lie in a low-dimensional subspace to target both the compression and redundancy mitigation problems at the same time. We also join [155] in discussing the potential benefits of applying compression techniques to V2X messages in general. The rest of the chapter is organized as follows: we describe the Spherical Codec in detail in Section 5.4 and give two numerical simulations in Section 5.5, one illustrates the codec's characteristics and the other studies the impact on the ITS-G5 V2X channel. We discuss some avenues for implementations and optimizations in future work in Section 5.6 and conclude the chapter in Section 5.7.

5.4 INFERENCE AND SPHERICAL CODEC

5.4.1 Overview of the Codec

All effective compression schemes must exploit some statistical redundancy in the data. In the particular case of trajectories, there typically exhibits a predictable feature since moving road vehicles must be constrained under physical laws such as inertia. As a result, we can split the data into two parts, one being "predictable" and the remaining being "unpredictable." The message recipients can readily predict the former, and only the "unpredictable" part needs to be transmitted. This explains why delta compression is much more effective than other techniques since it only encodes the "unpredictable" part of the data. The Spherical Code's starting point is from building an effective "predictor" on a reduced dimensional domain, which trades accuracy for a lower number of dimensions. Because estimates from the predictor converges in time: $\boldsymbol{a}_n \xrightarrow{n\to\infty} \boldsymbol{a}$, it follows that $\boldsymbol{a}_{n+1} - \boldsymbol{a}_n \xrightarrow{n\to\infty} 0$. This property can be used to design a messaging scheme such that the broadcasting frequency and the message sizes reduce over time. At a glance, the algorithm comprises two major components:

1. **An inference module** that continuously refines the posterior distribution of prediction $p(x_n, x_{n+1}, \dots | x_1, x_2, \dots, x_{n-1})$ based on past observations up to the current moment.
2. **A delta compression scheme** that only encodes the change in the representation $\Delta\theta_n$ that corrects the message recipients' prediction within a predefined error tolerance.

Because a large header is included in each V2X message, the best compres-

sion is achieved when no data is transmitted. In the Spherical Codec, no further communications will be made if the road object's trajectory does not deviate much from what the message recipients can predict accurately. If the transmitter observes a deviation that the clients should know about, it will transmit the change of θ_n (called $\Delta\theta_n$) to the recipient. Intuitively, the scheme works best if the initial choice of θ_1 matches the trajectory that the vehicle will travel in the future, which is likely if we choose the prior distribution p_{θ_1} carefully enough.

5.4.2 Trajectory Completion

Functional decomposition. For a detailed discussion about trajectory completion, please see section 3.4.3. In what follows, we let $\mathcal{D}$ be the set of vehicle trajectories:

$$\mathcal{D} = \{\boldsymbol{x} \in \mathbb{R}^{\infty}\}$$

One may also approximate the trajectories with cubic cardinal splines $\{B_{i,4}(t), i \in \{1, 2, \ldots, n_b\}\}$ as basis functions with n_b control points $\{a_i\}$:

$$\hat{x}(t) = \sum_{i=1}^{i=n_b} a_i B_{i,4}(t) \tag{5.1}$$

Efficient algorithms to construct $\{B_{i,4}\}$ exist, such as [156]. The number of basis functions n_b was selected so that:

$$\epsilon_{\mathcal{D}} = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[\max_t |x(t) - \hat{x}(t)|] < E_a \tag{5.2}$$

where E_a is some predefined tolerance error for approximation. Then, for each $\boldsymbol{x} \in \mathcal{D}$, we can calculate its corresponding control $\boldsymbol{a} = \{a_i, i \in \{1, 2, \ldots, n_b\}\}$ points following [157]. The process above could be done in a discrete domain. Hence, Equation (5.1) can be rewritten as:

$$\hat{x}[k] = \sum_{i=1}^{i=n_b} a_i \phi_i[k] \tag{5.3}$$

This equation can be leveraged for an inference process, since it is in a linear form of $\boldsymbol{a}$ and the basis functional vector $\phi[\cdot] = [\phi_i[k]]_i^{\intercal}$:

$$\hat{x}[k] = \boldsymbol{a}^{\intercal}\phi[\cdot] \tag{5.4}$$

Instead of working on the time domain, we will perform inference on the representation domain - that is, we are interested in the distribution of the representation coefficients $p(a)$ rather than $p(x_n)$. The reason is that the representation

vector $\boldsymbol{a}$ usually has a much smaller number of dimensions than $\boldsymbol{x}$, making inference on this domain much more efficient.

We can use any distribution fitting technique to find the prior distribution $p(a)$ from $\mathcal{D}$. To be as general as possible, we may perform a Kernel Density Estimation (KDE) to learn $p(a)$:

$$p(a) = \frac{1}{Nh} \sum_{i=1}^{N} K\left(\frac{a - a_i}{h}\right) \tag{5.5}$$

where K is the Epanechnikov kernel function, the bandwidth h is obtained from leave-one-out cross-validation. However, if the trajectories are close to realizations of the Gaussian Process, we can approximate $p(a)$ by $\mathcal{N}(\mu_a, \Sigma_a)$. In this particular case, a simple analytical method exists to calculate the maximum-a-posteriori estimate of the representation vector.

Bayesian inference based on Sequential Importance Resampling (SIR). Given $p(a)$ in the form of (5.5), inference could be difficult. Using Bayes formula:

$$\begin{aligned} p(a|x_{1:n}) &= p(a|x_1, x_2, \ldots, x_n) \\ &= \frac{p(x_{1:n}|a)p(a)}{\int_{a'} p(x_{1:n}|a')p(a')da'} \end{aligned} \tag{5.6}$$

If we use an empirical approximation of $p(a)da$ as $M \to \infty$:

$$\begin{aligned} p(a|x_{1:n}) &\simeq \frac{p(x_{1:n}|a) \sum_{j=1}^{M} \delta(da - a_j)}{\int_{a'} p(x_{1:n}|a')p(a')da'} \\ &= \sum_{j=1}^{M} \frac{p(x_{1:n}|a_j)}{\sum_{j=1}^{M} p(x_{1:n}|a_j)} \delta(da - a_j) \end{aligned} \tag{5.7}$$

This is an equivalent form of a Boostrap Particle Filter, with no state dynamics for a. In particular, we represent the posterior $p(a|x_{1:n})$ by $\{(a_{i,n}, w_{i,n}), 1 \leq i \leq M, 1 \leq n \leq N\}$ where $w_{i,n}$ is the weight for the particle $a_{i,n}$ at time n. Instead of sampling from $p(a|x_{1:n})$, we can sample from a proposal distribution $\pi(a)$:

$$p(a|x_{1:n}) \approx \sum_{i=1}^{M} w_{i,n} \delta(a - a_{i,n}) \tag{5.8}$$

where the weight:

$$w_{i,n} \propto \frac{p(x_{1:n}|a)p(a)}{\pi(a)} \tag{5.9}$$

we note that if we choose the proposal distribution $\pi(a)$ to be $p(a)$, we have a recursive formula:

$$w_{i,n} \propto p(x_{1:n}|a) = p(x_{1:n-1}|a)p(x_n|a) \propto w_{i,n-1}p(x_n|a) \tag{5.10}$$

This is the same form as (5.7). The weights are normalized at the end of each iteration. However, due to the increased variance of the estimate, resampling is required every once in a while. In implementation, we used adaptive resampling where we perform resample of $x_{i,n}$ when the effective number of samples $n_{eff} = \sum_{i=1}^{M}(w_{i,n})^{-2}$ is too small.

The particle filter undoubtedly will require heavy use of the available computational resource. However, in practice, there should be very little need to re-implement this filter because necessary inputs are already available from other applications such as radar filtering, tracking [158] or anomalous detection algorithms [159]. The maximum-a-posteriori (MAP) estimate of $p(a|x_{1:n})$ is the particle with the highest weight before resampling. The MAP coincides with the mean when no observation is received and will be sequentially refined as observations arrive.

Inference over Gaussian prior of representations. In the particular case that a multivariate Gaussian distribution can approximate $p(a)$, the maximum-a-posteriori estimate can be analytically calculated similarly to [159]. We outlined the algorithm in Algorithm 3.1.

5.4.3 Delta Communications

Following the discussion in Section 2.3, functional decomposition has provided a way to represent a (high-dimensional) vehicle trajectory using a compact, low-dimensional representation vector. While this effectively compresses the whole trajectory, there remains a question of how to do compression in real-time, as the data required is only available in the future. We propose a combined inference scheme that is intuitively similar to betting. In particular, we use some "money" (message bits to be transmitted) to bet on a particular value of a - the representation of the whole trajectory. If the vehicle we observe in the future turns out to have its trajectory KLT transformed to a, we "win" much money (saving many re-transmissions). If our initial a is not good enough, we will rebroadcast Δ_a so that our recipients can correct their value of a.

Quantization of the functional domain. To introduce the reconstruction error tolerance, we define a $(n_b - 1)$-sphere with diameter l such that any two points

within the same sphere, when inverse transformed to the time domain, will have a bounded maximal error. In particular, we choose l such that:

$$\max_k ||(a_1 - a_2)^\intercal \phi[k]||_2 \leq E_q \tag{5.11}$$

where E_q is the quantization error tolerance we choose. For orthonormal basis $\phi[k]$ like cubic B-splines or KLT, $l = E_q$. We also denote the transmitting vehicle as Alice's and the receiving vehicle as Bob's. For the sake of simplicity, we assume that there is no packet loss and because Alice is broadcasting, Alice certainly knows the exact information that Bob receives.

Alice runs the inference module introduced from the previous section to estimate, from her own trajectory $x_{1:n}$, what would likely be best value of a (the maximum-a-posteriori estimate) that represents her complete trajectory $x_{1:N} = \{x_1, x_2, \ldots, x_n, x_{n+1}, \ldots, x_N\}$ that she will complete in the future:

$$\hat{a}[n] = \arg\max_a p(a|x_{1:n}) \tag{5.12}$$

Deriving the Delta Message. Because Alice knows Bob's estimate b of the representation of her trajectory, she can perform an inverse Karhunen-Loève transform from his estimate b to yield Bob's belief of her trajectory $y[k]$:

$$y[k] = \sum_{l=1}^{n_b} b_k \phi_l[k] \tag{5.13}$$

Re-transmission is necessary if Alice's trajectory deviates an amount of E_d from Bob's belief, that is:

$$\texttt{TRANSMIT} = \begin{cases} 0 & \text{if } |y[k] - x[k]| < E_d \\ 1 & \text{if } |y[k] - x[k]| \geq E_d \end{cases} \tag{5.14}$$

We define the *Delta Vector*:

$$\delta_n = \hat{a}[n] - b \tag{5.15}$$

The goal is to encode δ_n so that our recipients can reconstruct $\hat{a}_n$ from δ_n and b with the accuracy given by (5.11). The first step of the spherical code is to convert δ into spherical coordinates. One choice is:

$$
\begin{aligned}
r &= ||\delta||_2 \\
\phi_1 &= \arccos \frac{\delta_1}{\sqrt{\delta_1^2 + \ldots + \delta_n^2}} \\
\phi_2 &= \arccos \frac{\delta_2}{\sqrt{\delta_2^2 + \ldots + \delta_n^2}} \\
&\ldots \\
\phi_n &= \begin{cases} \arccos \frac{\delta_n}{\sqrt{\delta_{n-1}^2 + \delta_n^2}}, \delta_n \geq 0 \\ 2\pi - \arccos \frac{\delta_n}{\sqrt{\delta_{n-1}^2 + \delta_n^2}}, \delta_n < 0 \end{cases}
\end{aligned}
$$

Quantization of the radius r is straightforward:

$$r_q = \lfloor r/l \rfloor$$

We define the number of angular lattices as $N_l(\rho)$ as:

$$N_l(\rho) = \lfloor \frac{2\pi\rho}{l} \rfloor + 1 \tag{5.16}$$

then each angular lattice will encompass a segment less than l of a circle whose radius is ρ. In other words, if we quantize the angle ϕ_i as

$$\phi_{q,i} = \lfloor \frac{2\pi r_q}{N_l} \phi_i \rfloor = \lfloor \frac{\phi l}{r_q} \rfloor \tag{5.17}$$

then the point $(r_q, \phi_{i,q}) \in \mathcal{B}_{E_q}(r, \phi_i)$ (see Fig. 5.18), then by (5.11), the quantization process does not induce an error larger than E_q.

Entropy of the Delta message. The number of bit required for coding is related to the entropy of r_q and $\phi_{q,i}$:

$$H \approx \log_2 r_q + \log_2 N_l(r_q) \approx 2 \log_2 r_q - \log_2 l + const \tag{5.18}$$

It is obvious from (5.18) that the maximum limit of message size depends greatly on r_q, hence the expression explains why delta compression is so effective for trajectory data. As the MAP estimate converges towards a^*, $r_q \to 0$. In other words, the payload size and the number of re-transmissions will decay with time.

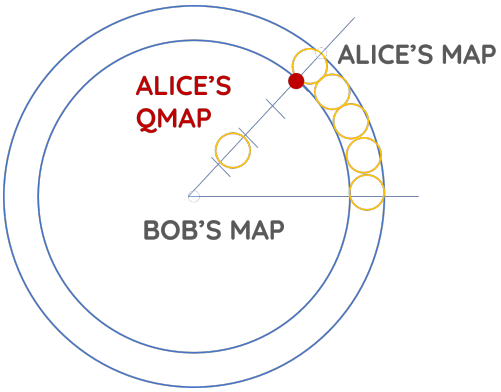


Figure 5.18: Illustration of the quantization process for Delta Message derivation in $\mathbb{R}^2$. Each small quantization sphere (yellow) has the diameter of l. Alice's quantized MAP (QMAP) is determined by $r_q \times l$, and stays within the same quantization sphere as Alice's original MAP.

5.5 IMPLEMENTATION AND NUMERICAL SIMULATIONS

So far, we have only described the "core features" of the Spherical Codec. The current design has yet to address the issue that new recipients that just come into transmit range or those that miss just one of the Delta messages, will be unable to decode the correct trajectory of the transmitting vehicles. To help remedy the problem, we introduce one particular codec implementation in this section, but many other different implementations are possible.

It is also important to stress beforehand that despite our attempt to compare the spherical codec with the conventional CA here, the results should not be interpreted as for justification of competition or replacement of the already implemented technology. This is due to the fact that the two are very different in mechanism. We envision that the spherical codec would only help in the special cases of congestion for which only position information is required.

5.5.1 *Compression and Mitigation Characteristics*

In this simulation, we set out to study the "core performance" of the Spherical Codec, reflected through the frequency of rebroadcasts and message entropy as calculated in (5.18). 200 vehicle trajectories in NGSIM dataset captured from 8:00 to 8:15 on the US-101 highway through the state of California are visualized in Fig. 5.19. We choose $n_b = 5$ basis components to achieve an approximation tolerance error of $E_a = 1ft$ and quantization tolerance $E_q = 3$ft for the representation tolerance error. Conversion into B-spline basis results in a set of vectors of $\mathbb{R}^5$ whose 3 first components were visualized in Fig. 5.20.

The SIR process introduced in Section 5.4.2 on a test trajectory, is shown in

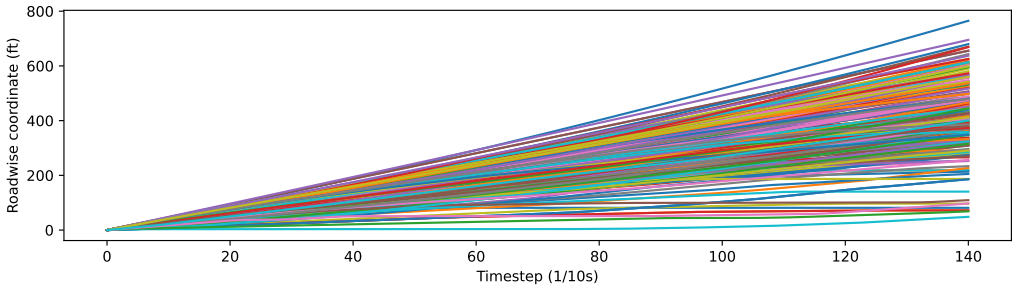


Figure 5.19: Roadwise coordinate graph of 200 vehicles from the NGSIM-101 dataset.

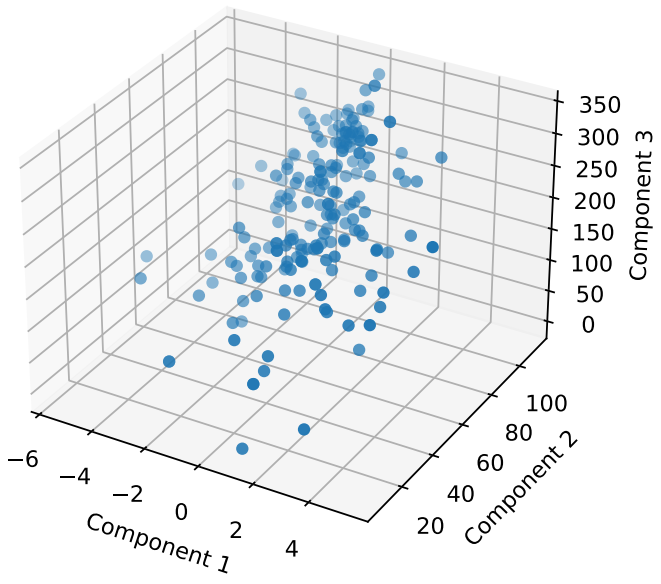


Figure 5.20: First three components of the functional domain $\mathcal{A}$.

Fig. 5.21. In the beginning, with no observations, the MAP estimate is at the mean of $p(a)$. The estimate gets better with time and eventually converges to the true (latent) representation vector a that generated this trajectory after 10s of observations.

However, with the tolerance of 3ft of error, it is only necessary to perform a total of 10 transmissions. The values are shown as individual dots in Fig. 5.22. Compared to sending 140 messages individually, we have already achieved im-

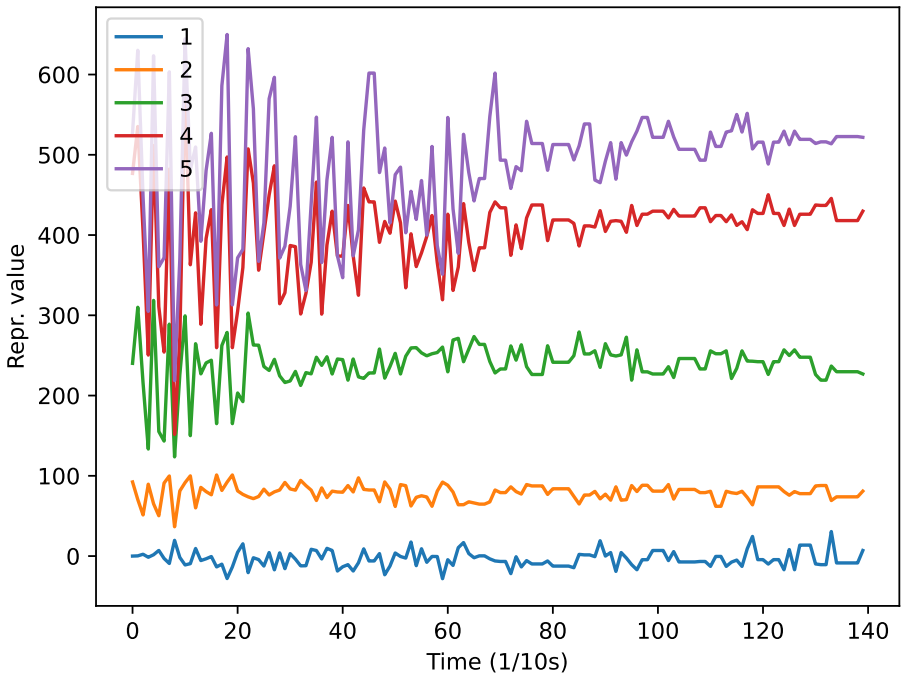


Figure 5.21: SIR to infer the five components of the representation vector a for a series of observations from a test trajectory. Note the convergence of the components of the representation vector with respect to time.

pressive broadcast saves. The reconstructed trajectory by message recipients and the error compared to the ground-truth trajectory is shown in Fig. 5.23, which confirms our design of restricting the deviation error between predicted and actual coordinates to be less than 3ft.

5.5.2 *Spherix: the Gaussian prior based implementation of the Spherical Codec*

To integrate the Spherical Codec into the current ETSI ITS-G5 Cooperative Awareness Service [160], we implemented a reference library for the Artery framework [161], an OMNET++ based discrete event simulation framework for ETSI ITS-G5 V2X, including an inference module based on Algorithm 4.1, along with the message encoder and decoder. The library is standalone, which means that possible integration into other simulation framework can be considered.

To address the problem of missing just one delta broadcast leading to the inability to decode the entire trajectory, we introduce two other messages: the `SHOUT`

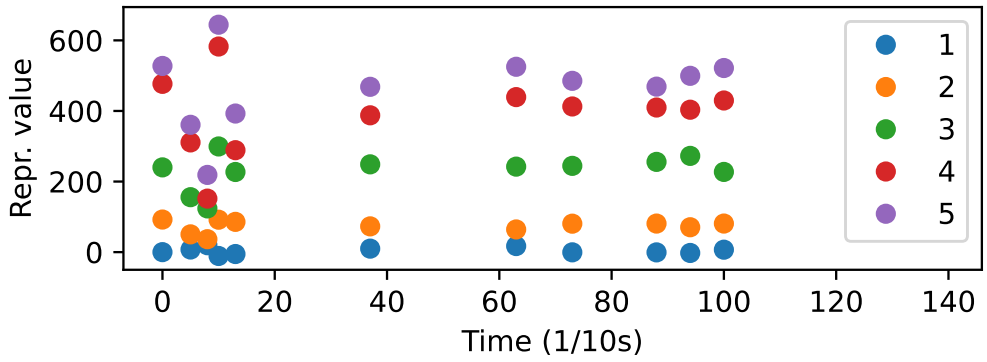


Figure 5.22: Representation values broadcast in subsequent messages, with error tolerance 3ft.

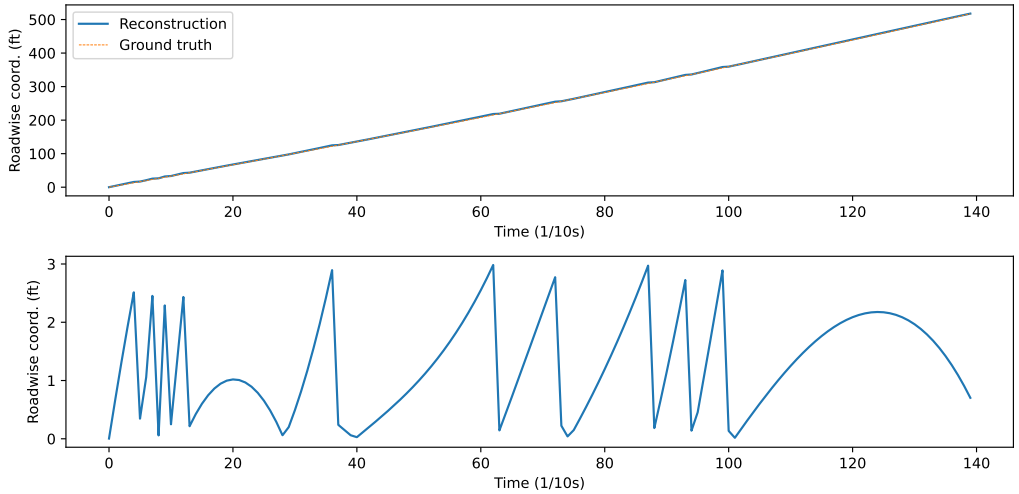


Figure 5.23: Reconstruction of the trajectory by message recipients, and absolute difference with ground-truth trajectory.

and the CAMLITE message formats. The SHOUT message periodically repeats the current estimate $\hat{a}[n]$ to all the recipients so that new recipients can begin predicting the transmitting vehicle's trajectory and announces the presence or disappearance of the vehicle so that recipients' local perception can be updated accordingly. Finally, the CAMLITE message is a dialled-down version of the current CA message [160], with many fields filled in with a default value. We also propose a "fallback mechanism" based on Algorithm 5.1, that automatically switches back to the conventional real-time CAM broadcasts if the prediction from the inference module consecutively churns out inaccurate estimates of $\hat{a}$. This may happen due to

the transmitting vehicle's exotic trajectory, which is a kind of trajectory rarely seen on the dataset. Continuing broadcasting in the Spherical Codec could potentially lead to an even higher channel utilization than the conventional way.

Algorithm 5.1: Dynamic Switching between Spherical CA Service and Legacy CA Service

Require: n_t: transmitted Deltas, n_s: saved messages, t_d: time between switching attempts, τ_s: message rate tolerance to enter Spherical CA Service, τ_t: message saving rate threshold for Spherical CA Service.
Ensure: S: switching decision variable: $S = 2$ if Legacy CA Service is used, $S = 1$ is Spherical CA Service is used
for every t_d time **do**
  if $|y[k] - x[k]| < E_d$ **then**
    $n_s \leftarrow n_s + 1$
  else
    $n_t \leftarrow n_t + 1$
  end if
  if $n_t/t_d > \tau_t$ and $S = 1$ **then**
    $S \leftarrow 2$
    $n_s \leftarrow 0$
  else if $n_s/t_d > \tau_s$ and $S = 2$ **then**
    $S \leftarrow 1$
    $n_t \leftarrow 0$
  end if
end for

5.5.3 Highway Simulation with Spherical Codec on the Artery Framework

We consider a simulation using SUMO [162] about a highway section whose length is 500m that consists of 5 lanes. All vehicles on the road were equipped with ITS-G5 V2X, and only the Cooperative Awareness Service was active. All stations were configured to transmit at 200mW power on the 5.9 GHz band. In three scenarios, traffic density casually built up from 0 vehicle to a maximum value of 140, 200, 250, 320 and 380 vehicles respectively with an average speed of 55km/h. We particularly chose this traffic scenario to achieve the desired traffic densities while the vehicle dynamics were not changing too quickly. For the Codec, the `SHOUT` interval was set to 3s, with switching parameters $\tau_s = \tau_n = 0.01$. We also used Nakagami's model [163] for both cases to simulate the channel fading.

Table 5.1 shows the key performance difference between the CA service based

Table 5.1: Key Performance Metrics **for each vehicle**.

Vehicle Density	140		200		250		320		380	
Codec	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON
Packets Sent	162	**33**	126	**34**	113	**34**	103	**34**	87	**33**
Bytes Sent	65 579	**12 780**	50 647	**13 051**	45 067	**12 374**	41 668	**13 218**	35 215	**12 441**
Packets Received	11 588	**2 339**	14 048	**3 626**	16 758	**5 294**	19 485	**7 155**	20 366	**8 054**
Max Bytes Received	4 659 079	**858 949**	5 650 559	**1 347 144**	6 743 476	**1 937 774**	7 843 798	**2 634 315**	8 201 731	**2 944 864**
Channel Load	0.2221	**0.0394**	0.2590	**0.0610**	0.2762	**0.0780**	0.3281	**0.1075**	0.3626	**0.1279**

on Spherical Codec and the current implementation across different vehicle density values. As traffic built up, the Decentralized Congestion Control (DCC) reduced the interval between CAM messages (Fig. 5.24), leading to a reduction in the number of packets sent at each station. With spherical codec, however, the vehicle's position was provided at the nominal frequency of 10Hz for all values of vehicle densities. It is obvious from Fig. 5.25 that the spherical codec helped significantly reduce the number of packets exchanged between stations and the channel load while ensuring the approximation error stays bounded by a predefined value.

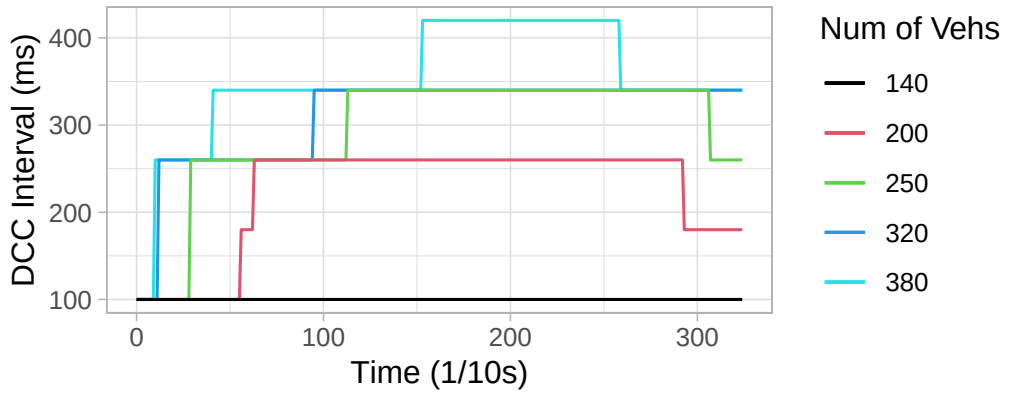
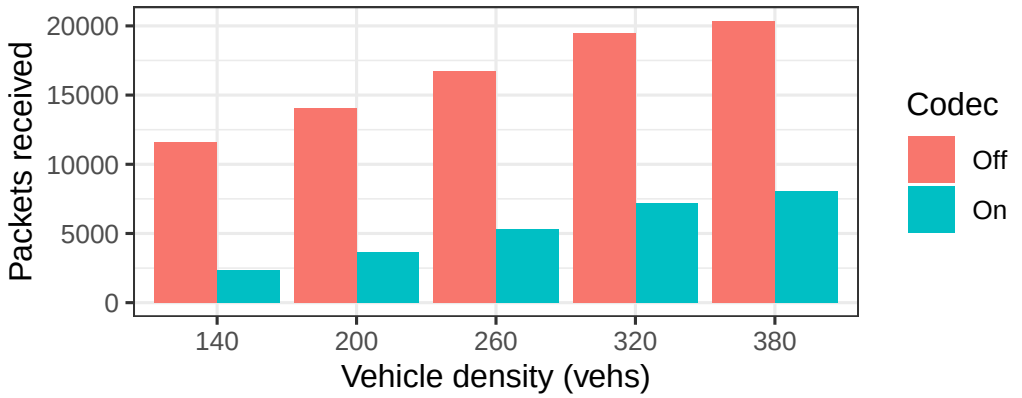


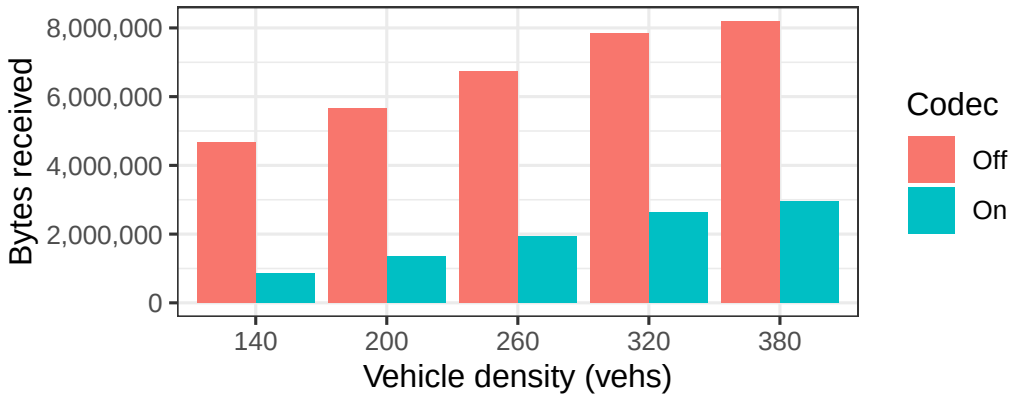
Figure 5.24: The CAM broadcasting frequency was regulated by the DCC.

5.6 DISCUSSION

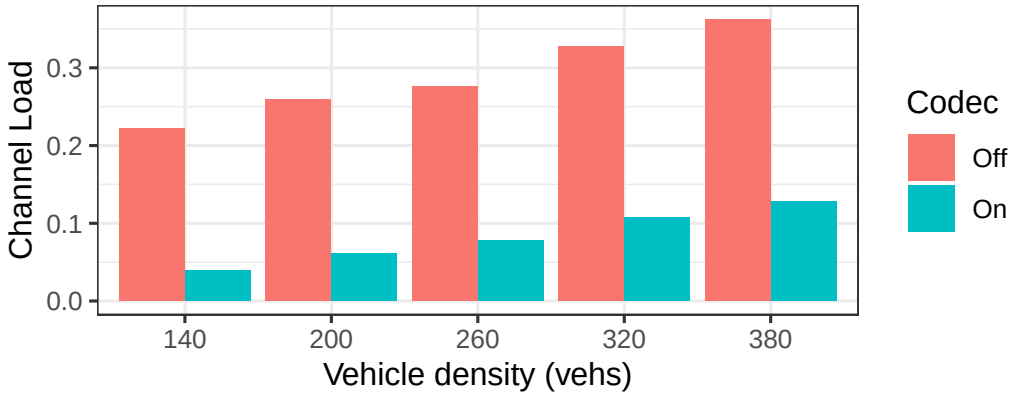
The name "spherical codec" stems from the fact that codewords are assigned to the parts of the volumetric n-sphere. Thanks to a different compression mechanism, Spherical Codec could work with conventional data compression methods, such as those proposed in [155] for a better compression ratio. However, it is also important to stress that this nature of the codec also opens a wide avenue for different implementations, and further optimization could yield even better results. For



(a) Number of packets received by a single station during the whole trajectory, with and without the spherical codec.



(b) Number of bytes received by a single station during the whole trajectory, with and without the spherical codec.



(c) Time-averaged channel load observed by a single station, with and without the spherical codec.

Figure 5.25: Visualization of key performance metrics from the simulation. Spherical Codec helped reduce the channel load and the number of bytes broadcast and received by all stations.

instance, the codec's parameters are currently selected through a trial-and-error process, and an adaptation mechanism could have significant impacts. Ideally, the SHOUT rebroadcasting frequency should depend on how regularly vehicles appear and disappear from the transmission horizon. Road scenarios could also be essential to choose the switching threshold $\tau_{(\cdot)}$. It was demonstrated in [164] that highway scenarios involving traffic with fast-changing dynamics are ideal situations to use Spherical Codec. In this way, the conventional and new methods complement each other rather than competing and replacing one another. Finally, we are aware that CA messages deliver not only the vehicle's trajectory but also additional information about the vehicle's state, such as lighting and status. Nevertheless, that information is usually packed in a container that is broadcast at long intervals. Hence, the SHOUT messages, perhaps, could carry this information along.

The Spherical Codec's current design will work with any time series data in general and does not account for information related to the first and higher order derivatives. However, it is believed that for vehicle trajectories, the velocity (first-order derivative) also contains a lot of statistical redundancy that has not been exploited in this design so a more effective communication scheme could be possible. This is an almost obvious fact by observing that vehicles traveling on highway rarely changes velocity and is currently under study as future work.

5.7 CONCLUSION

In this chapter, we have presented a novel data compression technique that specifically targets the V2X Cooperative Awareness Messages. The algorithm exploits the inherent low-dimensional nature of trajectories to combine inference and coding to "compress and mitigate redundancy". Numerical simulations have shown a significant gain in all performance metrics, with open avenues to further develop and extend the codec to other aspects of the current V2X standards.

6

Solving the Optimal Retransmission Problem with Linear Extrapolation over Mean Subtracted Residual

Abstract

In this chapter, we present a theoretical study of the trajectory data transmission redundancy mitigation problem (OTD-TP), framing it within a mathematical model. The Random Impulses models, outlined in Chapter 3, serve as the foundation upon which we prove optimality results relating to the solution for the minimum expected time until retransmission. We demonstrate that the solution to this problem suggests the design of an optimal message rate control algorithm. We also provide bounds regarding the expected stopping time.

Contents

6.1 THE OPTIMAL TRAJECTORY DATA TRANSMISSION PROBLEM (OTD-TP)

6.1.1 Background

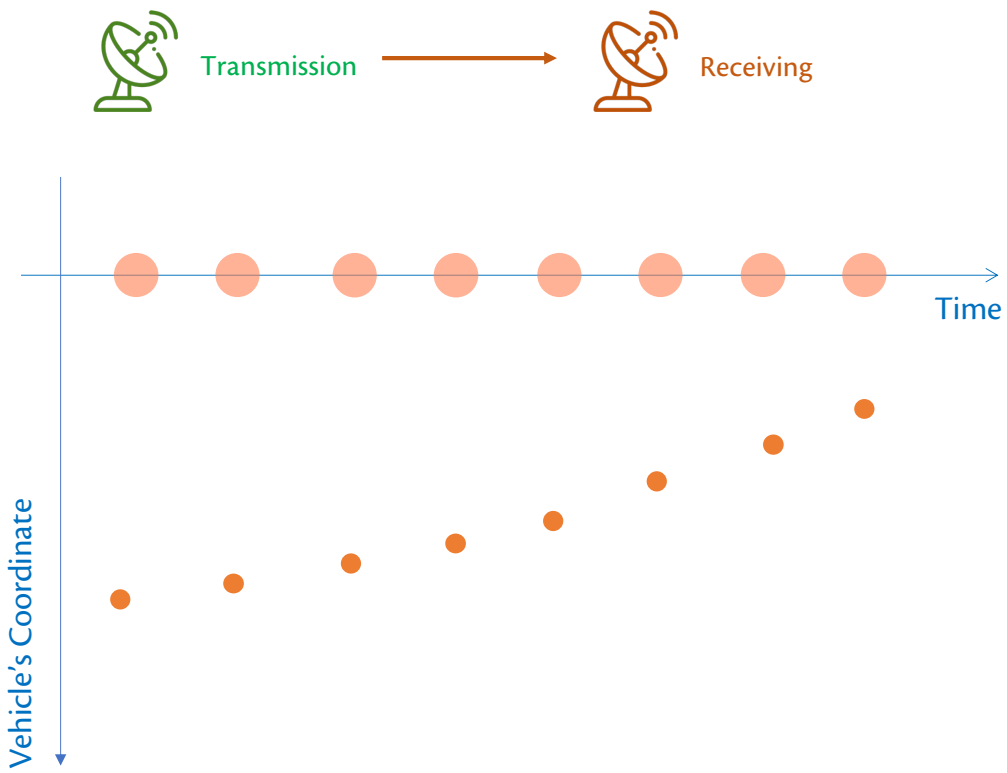


Figure 6.1: The conventional way of transmitting trajectory data: data is continuously sent over the channel at periodic intervals. The recipient may need to interpolate between data points, but this is usually not necessary.

In the usual data transmission scheme, the transmitter continuously beacon the trajectory data to the recipient. The recipient does not need to perform any complicated trajectory prediction task (Figure 6.1).

In the Silent Inference scheme introduced in Chapter 5, however, the transmitter will not beacon its trajectory information all the time, but will only transmit a message with enough information, then stays silent on the channel. The recipient will make efforts in predicting the trajectory information during the silent periods, based on the last message it received. Since the prediction methodology is also

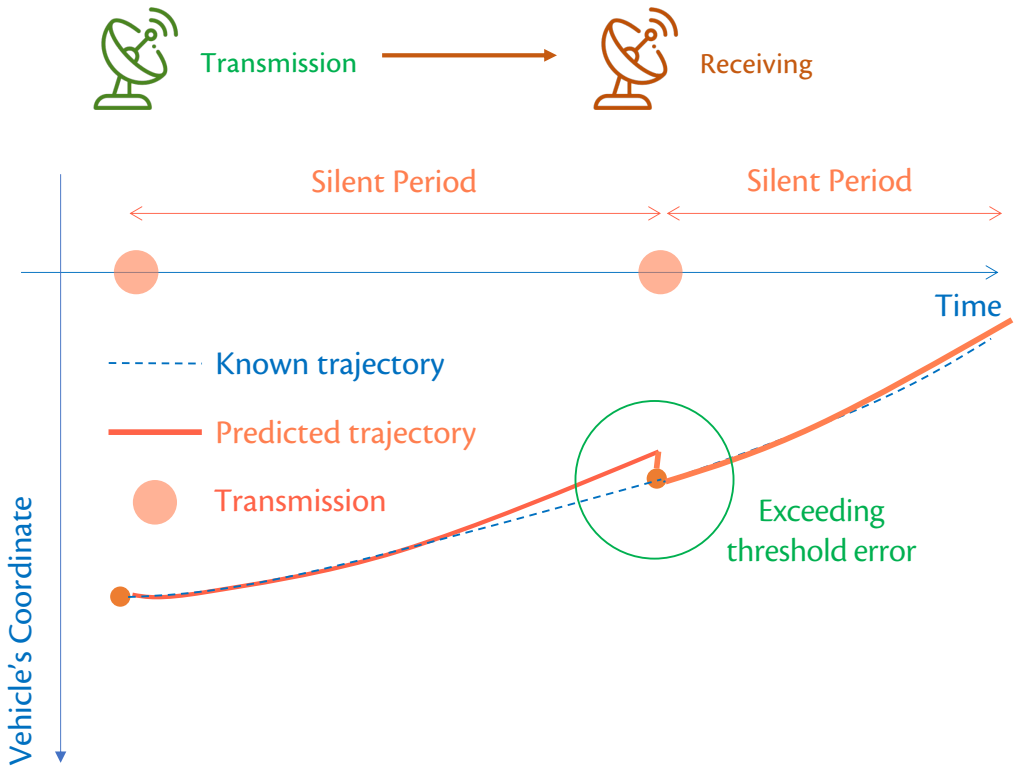


Figure 6.2: In the silent inference scheme, the recipient will have to make efforts predicting the trajectory when the transmitter is not transmitting (i.e, during *silent periods*). The transmitter also performs the same prediction task as the recipient to check if the recipient's prediction is correct or not. If it deviates more than some threshold value, the transmitter will transmit another message.

known to the transmitter, the transmitter can anticipate when the recipient's prediction fails (i.e., deviating significantly enough from the reality), and transmits another message and stays silent (Figure 6.2). The time between subsequent transmission attempts of the transmitter is called *expected time between retransmissions*.

6.1.2 Motivation

The goal of this chapter is to answer the following two questions:

1. In the inference scheme, which data should be exchanged to maximize the expected time between retransmissions? In other words, does one suppose to include the position, the velocity, the acceleration, or a transformed representation vector like the Spherical Codec?
2. Can we get the expected time between retransmissions? Or at least, can we

give a lower bound of it in the worst case scenario if that's too difficult?

6.1.3 Problem Statement

Define a probability space $(\Omega, \mathcal{F}, P)$ and a measurable space $(\mathbb{R}, \mathcal{B})$ where $\mathcal{B}$ denotes the Borel sigma algebra. Denote the continuous time index set as $T = \mathbb{R}_+$, we assume that each individual vehicle trajectory is an independent path sample $X(t, \cdot)$ of the stochastic process:

$$X(t, \omega) : (T \times \Omega) \to \mathbb{R}, \tag{6.1}$$

and that $X(t, \omega)$ is $\mathcal{B}$-measurable.

From some deterministically given time $t_0 \geq 0$, define $\hat{X}_t$ the deterministic prediction of X_t at time $t > t_0$. We call the discrepancy between the prediction $\hat{X}_t$ and the actual value of X_t the *residual*:

$$R_t = X_t - \hat{X}_t, \quad t \geq t_0, \tag{6.2}$$

and that the *retransmission event* A_t will happen when the trajectory residual exceeding some predefined threshold value δ:

$$A_t = \{\omega \in \Omega : |R(t)| \geq \delta\}, \quad t \geq t_0. \tag{6.3}$$

Let $(\Omega, \mathcal{F}, \mathcal{F}_t, P)$ be the filtered probability space where $\mathcal{F}_t$ is the natural filtration. Then, the random variable T, defined as:

$$T = \inf\{t \in \mathbb{R}_+ : |R(t)| \geq \delta\} \tag{6.4}$$

is a stopping time. Obviously, $\{\omega \in \Omega : T \leq t\} = \{\omega \in \Omega : \sup_{t' \leq t} |R(t')| \geq \delta\}$ is $\mathcal{F}_t$-measurable. To see this, note that:

$$\{\sup_{t' \leq t} |R(t')| \geq \delta\} = \cup_{s \in \mathbb{Q}, 0 \leq s \leq t} \{|R(s)| \geq \delta\}, \tag{6.5}$$

and that $\{|R(s)| \geq \delta\} \in \mathcal{F}_t$ if $\sigma(\hat{X}_t) \subset \mathcal{F}_t$. To avoid infinity, we also define a sequence of bounded stopping times:

$$T_n = T \wedge n, \quad n \in \mathbb{N}_+ \tag{6.6}$$

The goal of OTD-TP is to solve for the optimal estimate:

$$\hat{X}_t = \phi(t, X_{t_0}) \tag{6.7}$$

such that $\hat{X}_t$ maximizes the expected time until retransmission:

$$\lim_{n \to \infty} \mathbb{E}[T_n] = \sum_{n=1}^{+\infty} nP[\{T = n\}]. \tag{6.8}$$

Obviously by definition, $\sigma(\hat{X}_t) \subset \mathcal{F}_0 \subset \mathcal{F}_t$ which satisfies the condition for T to be a stopping time.

6.2 THE LONGITUDINAL OTD-TP (LOTD-TP)

The Random Impulses Models introduced in Chapter 3 assumed the independence between lateral and longitudinal components. If the RAIM model (3.3) is used in (6.1), we obtain the LOTD-TP.

Let $\boldsymbol{\rho} = \{\rho_0, \rho_1, \dots\}$ be the set of regeneration points from 1 to 0 of the acceleration activity process $\mathcal{L}_t$ defined in (3.4), then we can extract from X_t at $t \in \boldsymbol{\rho}$ a discrete stochastic process X_{ρ_k}. Then, the associated A_{ρ_k} given by (3.5) is Markov, since the k-th acceleration impulse $A_k = f(\omega_{2k+1} - \omega_{2k}, M_k)$ where $\omega_{2k+1} - \omega_{2k} = I_{1,k}$ and M_k are assumed to be independent samples from the respective distributions.

We define an auxiliary variable:

$$f_k = \frac{\omega_{2k+1} - \omega_{2k}}{\omega_{2k+2} - \omega_{2k}}, \tag{6.9}$$

which indicates the proportion of time the process spends in $\mathcal{L}_t = 1$ between two consecutive regenerations from 1 to 0. Consequently, f_k is also independently identically distributed, and independent from $I_{1,k}$. We also define:

$$\tilde{X}_t = X_t - \mu_{x,t}, \tag{6.10}$$

the residual of X_t when subtracted the mean trajectory $\mathbb{E}[X_t]$ in (3.2). Likewise, the residual velocity is defined as:

$$\tilde{v}_t = \frac{\mathrm{d}}{\mathrm{d}t}\left(X_t - \mathbb{E}[X_t]\right). \tag{6.11}$$

Lemma 6.1. Given the velocity longitudinal position at time ρ_0 being v_{ρ_0} and X_{ρ_0} respectively, the position and velocity at time ρ_n can be written as:

$$\begin{aligned}\tilde{X}_{\rho_n} = \tilde{X}_{\rho_0} &+ \tilde{v}_{\rho_0}(\rho_n - \rho_0) \\ &+ \sum_{k=0}^{n-2} M_k(\rho_{k+1} - \rho_k) f_k(\rho_n - \rho_{k+1}) \\ &+ \frac{1}{2}\sum_{k=0}^{n-1} M_k f_k^2(\rho_{k+1} - \rho_k)^2,\end{aligned} \tag{6.12}$$

and:

$$\tilde{v}_{\rho_n} = v_{\rho_0} + \sum_{k=0}^{n-1} M_k f_k(\rho_{k+1} - \rho_k) \tag{6.13}$$

Proof. The base case for $n = 2$ is trivial. Suppose that (6.12) and (6.13) hold for n. We will show that the expressions hold for $n + 1$.

By the RAIM model, we have:

$$\tilde{v}_{\rho_{n+1}} = \tilde{v}_{\rho_n} + \int_{\rho_n}^{\rho_{n+1}} A_s \mathrm{d}s \tag{6.14}$$

From (3.5), notice that:

$$A_s = \begin{cases} M_n \sim F_a(\cdot), & \rho_n \leq s \leq \rho_n + f_n(\rho_{n+1} - \rho_n) \\ 0, & \text{otherwise.} \end{cases}$$

Hence, (6.14) becomes:

$$\tilde{v}_{\rho_{n+1}} = \tilde{v}_{\rho_n} + \int_{\rho_n}^{\rho_n + f_n(\rho_{n+1} - \rho_n)} M_n \mathrm{d}s = \tilde{v}_{\rho_n} + M_n f_n(\rho_{n+1} - \rho_n)$$
$$= v_{\rho_0} + \sum_{k=0}^{n} M_k f_k(\rho_{k+1} - \rho_k).$$

Moving onto $\tilde{X}_{\rho_n}$:

$$\tilde{X}_{\rho_{n+1}} = \tilde{X}_{\rho_n} + \int_{u=\rho_n}^{\rho_{n+1}} \tilde{v}_u \mathrm{d}u,$$

where

$$\tilde{v}_u = \tilde{v}_{\rho_n} + \int_{s=\rho_n}^{u} A_s \mathrm{d}s.$$

This implies:

$$\tilde{X}_{\rho_{n+1}} = X_{\rho_n} + \tilde{v}_{\rho_n}(\rho_{n+1} - \rho_n) + \int_{\rho_n}^{\rho_{n+1}} \int_{\rho_n}^{u} A_s \mathrm{d}s \mathrm{d}u$$
$$= \tilde{X}_{\rho_n} + \tilde{v}_{\rho_n}(\rho_{n+1} - \rho_n) + \int_{\rho_n}^{\rho_n + f_n(\rho_{n+1} - \rho_n)} M_n(u - \rho_n) \mathrm{d}(u - \rho_n)$$
$$= \tilde{X}_{\rho_n} + \tilde{v}_{\rho_n}(\rho_{n+1} - \rho_n) + \frac{1}{2} M_n f_n^2 (\rho_{n+1} - \rho_n)^2$$

Substitution of $\tilde{X}_{\rho_n}$ and $\tilde{v}_{\rho_n}$ from the induction hypothesis we gets:

$$\tilde{X}_{\rho_{n+1}} = \tilde{X}_{\rho_0} + \tilde{v}_{\rho_0}(\rho_n - \rho_0) + \sum_{k=0}^{n-2} M_k(\rho_{k+1} - \rho_k) f_k(\rho_n - \rho_{k+1}) + \frac{1}{2} \sum_{k=0}^{n-1} M_k f_k^2 (\rho_{k+1} - \rho_k)^2$$
$$+ v_{\rho_0}(\rho_{n+1} - \rho_n) + \sum_{k=0}^{n-1} M_k f_k(\rho_{k+1} - \rho_k)(\rho_{n+1} - \rho_n) + \frac{1}{2} M_n f_n^2 (\rho_{n+1} - \rho_n)^2$$

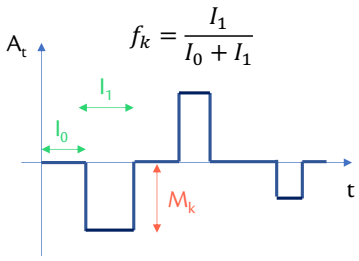


Figure 6.3: Illustration of various random variables in $\tilde{X}_{\rho_n}$, including the acceleration impulse magnitude M_k, the idle interval $I_{0,k}$, the active interval $I_{1,k}$.

By collecting the terms $\tilde{v}_{\rho_0}$, $M_k f_k(\rho_{k+1} - \rho_k)$ and $M_k f_k^2$, we obtain:

$$\tilde{X}_{\rho_{n+1}} = \tilde{X}_{\rho_0} + \tilde{v}_{\rho_0}(\rho_{n+1} - \rho_0) + \sum_{k=0}^{n-1} M_k f_k(\rho_{k+1} - \rho_k)(\rho_{n+1} - \rho_{k+1}) + \frac{1}{2}\sum_{k=0}^{n} M_k f_k^2(\rho_{k+1} - \rho_k)^2$$

□

Figure 6.3 illustrates various random variables of Equation (6.12). Finding the stopping time T is essentially finding the first passage time, a problem that has been well studied for Wiener process [73] with most solutions using the reflective principle. However, since $\tilde{X}_{\rho_k}$ is not Markov, the reflective principle cannot be applied. Moreover, unlike Markov processes which tend to "forget" the distant past, $\tilde{X}_{\rho_k}$ actually sees the effect of past acceleration impulses reinforced over time.

Nevertheless, under some special assumption about $M_k \sim F_a(\cdot)$, it is still feasible to find the optimal estimate $\phi(\tilde{X}_{\rho_n}, \tilde{v}_{\rho_n})$ that maximizes the expected time until retransmissions T.

Like the stopping time T, we can also define the stopping time τ for the extracted stochastic process $\tilde{X}_{\rho_k}$:

$$\tau = \inf\{n \in \mathbb{N}_+ : |\tilde{X}_{\rho_n} - \hat{X}_{\rho_n}| \geq \delta\}. \tag{6.15}$$

With a slight abuse of notation, we let $\tilde{X}_{\rho_k} \triangleq \tilde{X}_k$.

Lemma 6.2. The expected time until retransmissions for the extracted process $\tilde{X}_{\rho_k}$ is:

$$\mathbb{E}[\tau] = \sum_{n=0}^{\infty} (1 - P[\tau \leq n]) \tag{6.16}$$

Proof. We have:

$$1 - P[\tau \leq n] = \sum_{k=n+1}^{\infty} P[\tau = k],$$

hence:

$$\sum_{n=0}^{\infty}(1 - P[\tau \leq n]) = \sum_{n=0}^{\infty}\sum_{k=n+1}^{\infty} P[\tau = k].$$

Because $\{[n+1, +\infty], n \in [0, +\infty]\}$ covers the same space as $\{[0, n], n \in [0, +\infty]\}$, by swapping the sums:

$$\sum_{k=0}^{\infty}\sum_{n=0}^{k} P[\tau = k] = \sum_{k=0}^{\infty} kP[\tau = k] = \mathbb{E}[\tau].$$

□

We define the *optimal estimate* of a random variable X with regard to some predefined threshold $d > 0$ as:

$$\hat{X} = \arg\min_{c} \left(P[X \leq c - d] + P[X > c + d]\right) \tag{6.17}$$

In the general case, it would require knowing the entire distribution function of X in order to maximize or minimize the probability of some event E that ties to a random variable X. However, in the case of a symmetric distribution, we can get around that requirement via the following lemma:

Lemma 6.3. If X is a random variable whose distribution is symmetric, i.e.,

$$\exists \mu : f_X(\mu - x) = f_X(\mu + x) \quad \forall x \in \mathbb{R},$$

then the following statements are true:

(i). The expectation $\mathbb{E}[X] = \mu$,

(ii). The *optimal* estimate of X is $\mathbb{E}[X]$. the sufficient condition for optimality

Proof. It is trivial to show (i). To show (ii), consider the first derivative:

$$\frac{\mathrm{d}}{\mathrm{d}c}\left(F_X(c - d) + 1 - F_X(c + d)\right) = f_X(c - d) - f_X(c + d) = 0,$$

which gives $f_X(c - d) = f_X(c + d)$. If the $f_X(\cdot)$ is symmetric, then $\mathbb{E}[X]$ satisfies the equation. □

The following proposition attempts to give answer to the LOTD-TP under some assumptions:

Theorem 6.4. Let a vehicle trajectory have its longitudinal component following the RAIM model as described in Equation 3.3. Suppose that the distribution of acceleration impulses M_k is zero-mean and symmetric. Then, the following estimate:

$$\hat{X}_n = X_0 + \mu_{x,n} - \mu_{x,0} + (v_0 - \mu_{v,0})(\rho_n - \rho_0) \tag{6.18}$$

solves the LOTD-TP problem (6.8).

We need another lemma before proving Theorem 6.4.

Lemma 6.5. Define $R_n = \hat{X}_n - \tilde{X}_n$ as the estimate error at time ρ_n, and with a slight abuse of notation, $\mathcal{R}_n \triangleq \{\omega \in \Omega : |R_n| \leq \delta\}$, the event that the error is *acceptably* bounded. Then, the distribution of $R_n | \mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_{n-1}$ is symmetric.

Proof. The goal is to show that $\forall \lambda \geq 0, \forall \epsilon \geq 0$, the measure of:

$$P\left[\{R_n \in [\lambda, \lambda + \epsilon]\} \cap \mathcal{R}_1 \cap \mathcal{R}_2 \cap \cdots \cap \mathcal{R}_{n-1}\right]$$

is equal to:

$$P\left[\{R_n \in (-\lambda - \epsilon, -\lambda]\} \cap \mathcal{R}_1 \cap \mathcal{R}_2 \cap \cdots \cap \mathcal{R}_{n-1}\right]$$

For convenience, we define the event:

$$\Lambda = \{\omega \in \Omega : |R_1(\omega)| \leq \delta, |R_2(\omega)| \leq \delta, \ldots, |R_{n-1}(\omega)| \leq \delta, R_n(\omega) \in [\lambda, \lambda + \epsilon]\},$$

and because $\Lambda \in \sigma(R_n)$, by definition:

$$R_n = \sum_{k=0}^{n-1} M_k f_k (\rho_{k+1} - \rho_k)(\rho_{n+1} - \rho_{k+1}) + \frac{1}{2} \sum_{k=0}^{n} M_k f_k^2 (\rho_{k+1} - \rho_k)^2, \tag{6.19}$$

and we have $\Lambda \in \sigma(R_n) \subset \sigma(M_k, f_k, I_k), 0 \leq k \leq n-1$. The sample space Ω can be written as:

$$\Omega = \prod_{k=0}^{n-1} \left(\Omega_{M_k} \times \Omega_{f_k} \times \Omega_{I_k}\right),$$

which implies that each $\omega \in \Lambda$ can be written as a vector:

$$\omega = [\omega_{M_k}, \omega_{f_k}, \omega_{I_k}] = [\omega_{M_0}, \omega_{f_0}, \omega_{I_0}, \omega_{M_1}, \ldots], \quad 0 \leq k \leq n-1$$

We define a mapping $W : \Omega \to \Omega, \omega \mapsto \omega^-$ where $\omega^- = [\omega^-_{M_k}, \omega^-_{f_k}, \omega^-_{I_k}], \quad 0 \leq k \leq n-1$ defined as follows:

$$\begin{aligned} \omega^-_{M_k} &= \omega'_{M_k}, \quad 0 \leq k \leq n-1 \\ \omega^-_{f_k} &= \omega_{f_k}, \quad 0 \leq k \leq n-1 \\ \omega^-_{I_k} &= \omega_{I_k}, \quad 0 \leq k \leq n-1 \end{aligned}$$

and ω'_{M_k} is defined such that:

$$M_k(\omega'_{M_k}) = -M_k(\omega_{M_k}), \quad 0 \leq k \leq n-1.$$

We know that this choice is possible because M_k is $\mathcal{B}(\mathbb{R})$ measurable. Define the image of Λ through the mapping W as Λ^-. From (6.19), it is evident that:

$$R_k(\omega^-) = -R_k(\omega), \quad \forall\, \omega^- \in \Lambda^-, \omega \in \Lambda, \forall k : 0 \leq k \leq n \tag{6.20}$$

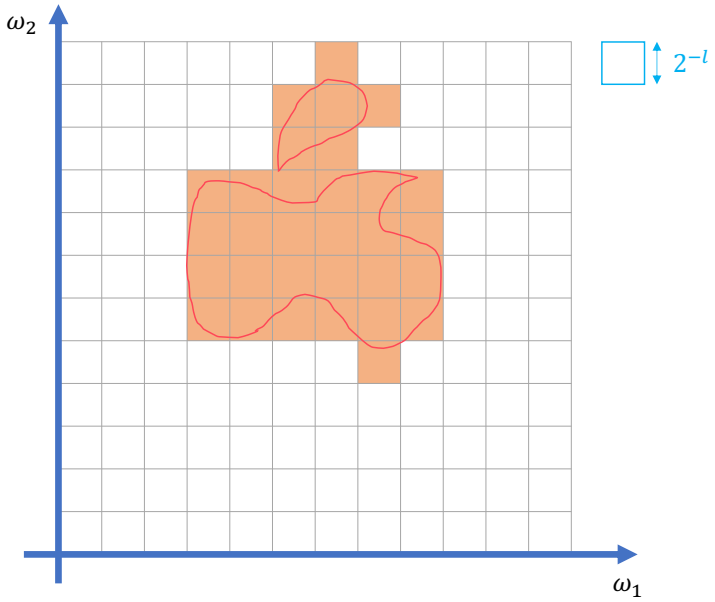


Figure 6.4: Discretization π of the exemplary 2D sample space Ω and the identity mapping 1_π that maps B onto the cells whose size is 2^{-l}.

And because $\Lambda \in \mathcal{R}_k = \{\omega \in \Omega : |R_k(\omega)| \leq \delta\}$, $\Lambda^- = \{\omega^- \in \Omega : |R_k(\omega^-)| = |-R_k(W^{-1}(\omega^-))| \leq \delta\}$, which implies that $\Lambda^- \in \mathcal{R}_k$ too for $0 \leq k \leq n-1$. It also follows (6.19) that $R_n(\omega^-) = -R_n(\omega)$ where $\omega^- = W(\omega)$, hence $R_n(\Lambda^-) = (-\lambda - \epsilon, \lambda]$. Hence, $\Lambda^- \in \sigma(R_n)$, which implies that a probability measure of Λ^- is possible.

Intuitively, this result roughly says that in essence, for every sample path that reaches λ while staying bounded all the time to $[-\delta, \delta]$, there is another sample path that reaches $-\lambda$ while staying bounded all the time to $[-\delta, \delta]$ too. We expect that these two sample paths have the same probability.

We define a *partition* $\pi_l(\Omega)$ of $\Omega = \prod_k \left(\Omega_{M_k} \times \Omega_{f_k} \times \Omega_{I_k}\right)$, $l \in \mathbb{N}_+$ such that:

$$\pi_l(\Omega) = \prod_k \left(\pi_{l,M_k} \times \pi_{l,f_k} \times \pi_{l,I_k}\right), \tag{6.21}$$

where π_{l,M_k}, π_{l,f_k} and π_{l,I_k} are disjoint unions of intervals whose length is 2^{-l}:

$$\pi_{l,\cdot} = \bigcup_{n\in\mathbb{N}} \left[n2^{-l}, (n+1)2^{-l}\right]. \tag{6.22}$$

Denote $\boldsymbol{n} = [n_{M_k}, n_{f_k}, n_{I,k}]$. We further define an identity mapping:

$$\mathbf{1}_{\pi,l} : \Omega \to \pi_l(\Omega),$$

$$\omega \mapsto \bigcup_{\boldsymbol{n}:\mathbf{1}_{\boldsymbol{n}}(\omega)=1} \prod_k \left(n_{M_k}2^{-l}, (n_{M_k}+1)2^{-l}\right] \times \left(n_{f_k}2^{-l}, (n_{f_k}+1)2^{-l}\right] \times \left(n_{I_k}2^{-l}, (n_{I_k}+1)2^{-l}\right],$$

where:

$$\mathbf{1}_{\boldsymbol{n}}(\omega) = \begin{cases} 1, & \text{if } \omega_{M_k} \in (n_{M_k}2^{-l}, (n_{M_k}+1)2^{-l}] \text{ and } \omega_{f_k} \in (n_{f_k}2^{-l}, (n_{f_k}+1)2^{-l}] \\ \text{and } \omega_{I_k} \in (n_{I_k}2^{-l}, (n_{I_k}+1)2^{-l}]; \\ 0, & \text{otherwise.} \end{cases}$$

Figure 6.4 provides an example of the discretization of a 2D sample space, and the action of the identity mapping in choosing the "cells" that contains a Borel set B, outlined in red. $\mathbf{1}_{\pi,l}(B)$ results in the shaded cells.

Consider $\Lambda_l = \mathbf{1}_{\pi,l}(\Lambda)$ and ${\Lambda^-}_l = \mathbf{1}_{\pi,l}(\Lambda^-)$. For each of them, we define two associated events:

$$M_\Lambda = \bigcup_{\substack{\boldsymbol{n}:\mathbf{1}_{\boldsymbol{n}}(\omega)=1 \\ \omega\in\Lambda}} \prod_k \left(n_{M_k}2^{-l}, (n_{M_k}+1)2^{-l}\right] \times \Omega_{f_k} \times \Omega_{I_k} \tag{6.23}$$

$$N_\Lambda = \bigcup_{\substack{\boldsymbol{n}:\mathbf{1}_{\boldsymbol{n}}(\omega)=1 \\ \omega\in\Lambda}} \prod_k \Omega_{M_k} \times \left(n_{f_k}2^{-l}, (n_{f_k}+1)2^{-l}\right] \times \left(n_{I_k}2^{-l}, (n_{I_k}+1)2^{-l}\right], \tag{6.24}$$

and

$$M_{\Lambda^-} = \bigcup_{\substack{\boldsymbol{n}:\mathbf{1}_{\boldsymbol{n}}(\omega)=1 \\ \omega\in\Lambda^-}} \prod_k \left(n_{M_k}2^{-l}, (n_{M_k}+1)2^{-l}\right] \times \Omega_{f_k} \times \Omega_{I_k} \tag{6.25}$$

$$N_{\Lambda^-} = \bigcup_{\substack{\boldsymbol{n}:\mathbf{1}_{\boldsymbol{n}}(\omega)=1 \\ \omega\in\Lambda^-}} \prod_k \Omega_{M_k} \times \left(n_{f_k}2^{-l}, (n_{f_k}+1)2^{-l}\right] \times \left(n_{I_k}2^{-l}, (n_{I_k}+1)2^{-l}\right], \tag{6.26}$$

Then, it is obvious that:

$$M_\Lambda \cap N_\Lambda = \Lambda_l$$
$$M_{\Lambda^-} \cap N_{\Lambda^-} = {\Lambda^-}_l$$

Moreover, notice that M_Λ and $M_{\Lambda^-} \in \sigma(M_k), 0 \le k \le n-1$; while N_Λ and $N_{\Lambda^-} \in \mathcal{P} \setminus \sigma(M_k), 0 \le k \le n-1$. By independence of M_k from the RAIM model:

$$P[\Lambda_l] = P[M_\Lambda]P[N_\Lambda]$$
$$P[{\Lambda^-}_l] = P[M_{\Lambda^-}]P[N_{\Lambda^-}]$$

However, due to the symmetry of M_k distribution:

$$P[M_\Lambda] = P[M_{\Lambda^-}],$$

and by construction:

$$N_\Lambda \equiv N_{\Lambda^-} \Rightarrow P[N_\Lambda] = P[N_{\Lambda^-}],$$

this leads to:

$$P[\Lambda_l] = P[\Lambda^-{}_l]$$

and letting $l \longrightarrow +\infty$, we obtain:

$$P[\Lambda] = P[\Lambda^-].$$

So far, the mapping $R_n \circ W \circ R_n^{-1}$ maps every Borel sets in the positive half-line to Borel sets in the negative half-line. In particular, for every sample path ends up in $[\lambda, \lambda + \epsilon)$, we find another sample path ends up in $(-\lambda - \epsilon, -\lambda]$ with the same probability.

It should be obvious that $R_n \circ W \circ R_n^{-1}$ is a bijective mapping. It is a surjection because $\forall B = (-\lambda - \epsilon, -\lambda], (R_n \circ W \circ R_n^{-1})([\lambda, \lambda + \epsilon)) = B$ so the mapping codomain covers the entire negative half-line. To show that it is an injection, suppose $\exists B_1, B_2 \subset [0, +\infty), B^* \subset (-\infty, 0] : R_n \circ W \circ R_n^{-1}(B_1) = R_n \circ W \circ R_n^{-1}(B_2) = B^*$. Because $R_n \circ W \circ R_n^{-1}(B) = -B$, we have:

$$-B_1 = -B_2 \Rightarrow B_1 = B_2$$

As a result, the conditional distribution of R_n conditioned on $\mathcal{R}_k, 0 \leq k \leq n-1$ is symmetric:

$$\begin{aligned} P\left[\{R_n \in [\lambda, \lambda + \epsilon]\} \cap \mathcal{R}_1 \cap \mathcal{R}_2 \cap \cdots \cap \mathcal{R}_{n-1}\right] \\ = P\left[\{R_n \in (-\lambda - \epsilon, -\lambda]\} \cap \mathcal{R}_1 \cap \mathcal{R}_2 \cap \cdots \cap \mathcal{R}_{n-1}\right] \end{aligned}$$

□

We may now prove Theorem 6.4.

Proof of Theorem 6.4. Consider the probability:

$$\begin{aligned} 1 - P[\tau \leq n] = P[\tau > n] = P[\{\sup_{k \leq n} |\tilde{X}_k - \hat{X}_k| < \delta\}] \\ = P\left[\bigcap_{k \leq n} \{|\tilde{X}_k - \hat{X}_k| < \delta\}\right] \end{aligned}$$

Thus, from (6.16):

$$\mathbb{E}[\tau] = \sum_{n=0}^{\infty} P\left[\bigcap_{k\leq n}\{|\tilde{X}_k - \hat{X}_k| < \delta\}\right] = P\left[\{|\tilde{X}_0 - \hat{X}_0| < \delta\}\right] + P\left[\{|\tilde{X}_0 - \hat{X}_0| < \delta\} \cap \{|\tilde{X}_1 - \hat{X}_1| < \delta\}\right] + \dots \quad (6.27)$$

Let $\hat{X}_n = X_0 + \mu_{x,n} - \mu_{x,0} + (v_0 - \mu_{v,0})(\rho_n - \rho_0) + C_n$, we will show that $C_n = 0$ is necessary to maximize $\mathbb{E}[\tau]$, and this will result in the estimate of (6.39). Let $\boldsymbol{C} = \{C_k, 0 \leq k \leq n-1\}$. From (6.27), we may choose C_0 that maximizes $P\left[\{|R_0 - C_0| < \delta\}\right]$, which happens to be $C_0 = 0$ according to Lemma 6.3. Then we may choose C_1 that maximizes $P\left[\{|R_1 - C_1| < \delta\} \cap \mathcal{R}_0\right]$, which happens also to be $C_1 = 0$. This procedure can be repeated for $k = 2, 3, \dots$. □

Corollary 6.6. The deterministic optimal prediction rule (6.39) can be approximated as:

$$\hat{X}_n = X_0 + \mu_{x,n} - \mu_{x,0} + (v_0 - \mu_{v,0})\,\mathbb{E}[\rho_n - \rho_0], \quad (6.28)$$

which can further be rewritten into:

$$\hat{X}_t = X_0 + \mu_{x,t} - \mu_{x,0} + (v_0 - \mu_{v,0})t. \quad (6.29)$$

These results are the convergence bounds of (6.39) when the acceleration activity process $\mathcal{L}_t$ regenerates fast enough, and by the law of large numbers.

Remark. The equation (6.29) reveals the three components that affect the prediction of the future longitudinal position of a vehicle:

1. X_0 is the last known position of the vehicle.
2. $\mu_{x,n} - \mu_{x,0}$ is the average (most commonly observed) distance traveled by all the vehicles.
3. $(v_0 - \mu_{v,0})t$ is the additional distance traveled by the host vehicle, when compared to the average velocity.

Although bearing some similarities to the Constant Velocity (CV) model, the appearance of the second term related to the average distance traveled by all vehicles is crucial to ensuring the extended period of time in which the prediction remains valid. It is related to the self-regulatory effect of distance-keeping when vehicles traveled together.

6.3 LOWER BOUND OF THE EXPECTED STOPPING TIME FOR LOTD-TP

In this section, we attempt to give a conservative estimate on the first passage time T to the extracted longitudinal position process:

$$R_n = \tilde{X}_n - \hat{X}_n = \sum_{k=0}^{n-1} M_k f_k (\rho_{k+1} - \rho_k)(\rho_{n+1} - \rho_{k+1}) + \frac{1}{2} \sum_{k=0}^{n} M_k f_k^2 (\rho_{k+1} - \rho_k)^2$$

The first challenge associated with the analysis of this process is that it is not Markov. Furthermore, the more distant the impulses M_k are in the past, the larger their effects become. As a result, R_n is fundamentally different from the Wiener process, and the reflective principle could not be applied. The second challenge is that it is not known (and proven) whether $\mathbb{E}[T] < \infty$ or not.

Define the tail-sum:

$$S_k = \begin{cases} \sum_{l=k}^{n-1} I_k, & 0 \le k \le n-1 \\ 0, & k = n \end{cases} \tag{6.30}$$

and the following process that bounds R_n from below:

$$\begin{aligned} R_n^* &= \sum_{k=0}^{n-2} M_k^* f_k I_k (S_{k+1}) + \frac{1}{2} \sum_{k=0}^{n-1} M_k^* f_k^2 I_k^2 \\ &= \sum_{k=0}^{n-1} M_k^* \left(f_k I_k S_{k+1} + \frac{1}{2} f_k^2 I_k^2 \right) \end{aligned} \tag{6.31}$$

where M_k^* is a distribution of random acceleration impulse values with only positive support. The distribution of M_k^* can be obtained by transporting the negative part of the density function to the value of zero. In other words, we assume that if A_n gives a negative sign impulse, we replace it with zero. Figure 6.5 illustrates the density function of M_k and M_k^*.

Define T^* the first passage time of R_n^* to δ:

$$T^* = \inf\{n \in \mathbb{N}_+ : R_n^* \le \delta\} \tag{6.32}$$

then, it should be obvious that T^* is a stopping time, $T^* \le T$ because R_n can both increase and decrease, while R_n^* can only increase. Hence, T^* can be regarded as the lower bound of the passage time (or time between retransmissions) of the original process R_n.

Proposition 6.7. The stopping time T^* has finite expectation:

$$\mathbb{E}[T^*] < \infty$$

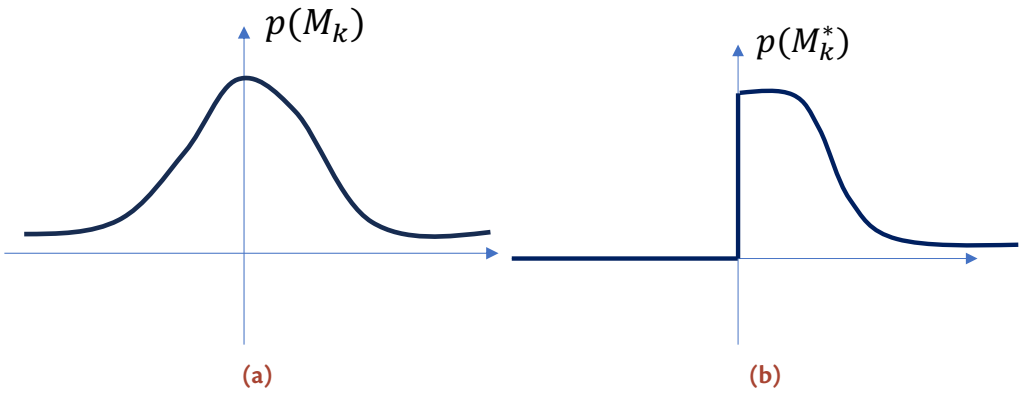


Figure 6.5: The symmetric acceleration amplitude distribution of M_k and the one-sided with positive support of acceleration amplitude distribution M_k^*.

Proof. We have:

$$R_n^* = \sum_{k=0}^{n-1} M_k^* \left(f_k I_k S_{k+1} + \frac{1}{2} f_k^2 I_k^2 \right) \geq \frac{1}{2} \sum_{k=0}^{n-1} M_k^* f_k^2 I_k^2,$$

so if one define an alternative process $R_{lb,n}^* = \frac{1}{2} \sum_{k=0}^{n-1} M_k^* f_k^2 I_k^2$, then it is obvious that the associated first passage time to δ: $T_{lb,n}^* \geq T^*$.

Notice that $R_{lb,n}^*$ exhibits the *Strong Markov Property*. Let p denote the probability that $M_k^* f_k^2 I_k^2 \geq 1$, and we know that $p > 0$. We define another process by flooring the increment of $R_{lb,n}^*$:

$$R_{llb,n}^* = \begin{cases} R_{lb,0}^*, & n = 0; \\ R_{llb,n-1}^* + 1, & \text{if } M_n^* f_n^2 I_n^2 \geq 1 \\ R_{llb,n-1}^*, & \text{otherwise.} \end{cases} \tag{6.33}$$

Again, it should be obvious that $R_{llb,n}^* \leq R_{lb,n}^* \forall n \in \mathbb{N}_+$. We denote $T_{llb|s}^*$ the first passage time when $R_{llb,n}^*$ starts from s (i.e., $R_{llb,0}^* = R_{lb,0}^* = s$), then:

$$\begin{aligned} T_{llb|s}^* &= 1 + pT_{llb|s+1}^* + (1-p)T_{llb|s}^* \\ -1/p &= T_{llb|s+1}^* - T_{llb|s}^*, \end{aligned} \tag{6.34}$$

and the solution for this finite difference equation always exists. The conclusion follows from the fact that $T_{llb|n}^* \geq T_{lb,n}^* \geq T^*$. □

We define an auxiliary random variable:

$$W_n = \frac{2R_n^*}{n\,\mathbb{E}[M_k^*]\,\mathbb{E}[I_k]^2\,\mathbb{E}[f_k]} - n$$

Proposition 6.8. With respect to the natural filtration $\mathcal{F}_n$ of R_n^*, under the assumption that all distributions of f_k, M_k^*, I_k W_n have finite second-order moments, W_n approximates a martingale as $n \to \infty$.

Proof. It is sufficient to show that:

$$\mathbb{E}[W_{n+1}|\mathcal{F}_n] - \mathbb{E}[W_n|\mathcal{F}_n] = 1,$$

which is equivalent to:

$$\frac{2}{\mathbb{E}[M_k^*]\,\mathbb{E}[f_k]\,\mathbb{E}[I_k]^2}\left(\frac{R_{n+1}^*}{n+1} - \frac{R_n^*}{n}\right) \tag{6.35}$$

We have:

$$R_{n+1}^* = R_n^* + \sum_{k=0}^{n-1} M_k^* f_k I_k\,\mathbb{E}[I_n] + \frac{1}{2}\,\mathbb{E}[M_k^*]\,\mathbb{E}[f_k^2]\,\mathbb{E}[I_k^2].$$

A direct substitution to (6.35) gives:

$$= \frac{2}{\mathbb{E}[M_k^*]\,\mathbb{E}[f_k]\,\mathbb{E}[I_k]^2}\frac{1}{n(n+1)}\left(n\left(R_n^* + \sum_{k=0}^{n-1} M_k^* f_k I_k\,\mathbb{E}[I_n] + \frac{1}{2}\,\mathbb{E}[M_k^*]\,\mathbb{E}[f_k^2]\,\mathbb{E}[I_k^2]\right) - (n+1)R_n^*\right)$$

$$= \frac{2}{\mathbb{E}[M_k^*]\,\mathbb{E}[f_k]\,\mathbb{E}[I_k]^2}\frac{1}{n(n+1)}\left(\sum_{k=0}^{n-1} M_k^* f_k I_k n\,\mathbb{E}[I_k] + \frac{1}{2} n\,\mathbb{E}[M_k^*]\,\mathbb{E}[f_k^2]\,\mathbb{E}[I_k^2] - R_n^*\right),$$

where we have used independence of random variables where needed to simplify the expressions. By substituting R_n^* from (6.31) and splitting $n\,\mathbb{E}[I_k]$ into $(k+1)\,\mathbb{E}[I_k] + (n-k-1)\,\mathbb{E}[I_k] = (k+1)\,\mathbb{E}[I_k] + \mathbb{E}[S_{k+1}]$:

$$= \frac{2}{\mathbb{E}[M_k^*]\,\mathbb{E}[f_k]\,\mathbb{E}[I_k]^2}\left(\frac{1}{n(n+1)}\sum_{k=0}^{n-1} M_k^* f_k I_k\left(-S_{k+1} + \mathbb{E}[S_{k+1}]\right)\right.$$

$$\left. + \frac{1}{2n(n+1)}\sum_{k=0}^{n-1}\left(-M_k^* f_k^2 I_k^2 + \mathbb{E}[M_k^* f_k^2 I_k^2]\right) + \frac{1}{n(n+1)}\sum_{k=0}^{n-1} M_k^* f_k I_k (k+1)\,\mathbb{E}[I_k]\right)$$

If we invoke the Law of Large Numbers as $n \to \infty$, it is obvious that the first two terms converge to zero. The last term converges to $1/2\,\mathbb{E}[M_k^*]\,\mathbb{E}[f_k]\,\mathbb{E}[I_k]^2$, which is then normalized by the first product term $2/\mathbb{E}[M_k^*]\,\mathbb{E}[f_k]\,\mathbb{E}[I_k]^2$. □

It is our belief that the regeneration rate for $\mathcal{L}_t$ of the RAIM model is high enough, as evidenced from Chapter 3, for Proposition 6.8 to hold. The following theorem gives the estimate of the expectation of the stopping time for R_n^*:

Theorem 6.9. If the conditions in Proposition 6.8 holds, the expected time between retransmissions T can be bounded below (i.e., pessimistically) by:

$$\mathbb{E}[T^*] = \sqrt{\frac{2\delta}{\mathbb{E}[M_k^*]\,\mathbb{E}[f_k]\,\mathbb{E}[I_k]^2}}, \tag{6.36}$$

for the extracted process R_k^*, and:

$$\mathbb{E}[T^*] = \sqrt{\frac{2\delta}{\mathbb{E}[M_k^*]\,\mathbb{E}[f_k]}} \tag{6.37}$$

for the original process R_t^*.

Proof. The results can simply be obtained by invoking the Optional Stopping Theorem for W_n, which is approximately a martingale for n large enough:

$$\mathbb{E}[W_n] = \mathbb{E}[W_0] = 0 \Rightarrow \mathbb{E}[T^*] = \frac{2\delta}{\mathbb{E}[T^*]\,\mathbb{E}[M_k^*]\,\mathbb{E}[f_k]\,\mathbb{E}[I_k]^2},$$

which gives (6.36). Equation (6.37) is simply the result of applying the Wald identity to (6.36). □

Algorithm 6.1: Transmission of Longitudinal Trajectory Data by Linear Extrapolation on Mean-subtracted Residual (LEMONS).

Require: Synchronized mean trajectory $\mu_{x,t}$ data of all vehicles participating in the network.
Require: Initialization δ - the tolerable prediction error; last-known position x_{t_0}, velocity v_{t_0}, time index variable t_0.
for measurement x_t arrives **do**
  $\hat{x}_t = x_{t_0} + \mu_{x,t} - \mu_{x,t_0} + (v_{t_0} - \mu_{v,t_0})$ ▷ Prediction of longitudinal position at time t.
  if $|x_t - \hat{x}_t| \geq \delta$ **then**:
    Send a message including the new position x_t, velocity v_t and the time index t. ▷ These values will become x_{t_0}, v_{t_0}, t_0 on the client's side.
  end if
end for

Theorem 6.4 suggested the data transmission scheme described in Algorithm 6.1 is optimal in maximizing the expected time between retransmissions, and Theorem 6.9 provides a pessimistic estimate of this time.

6.4 THE LATERAL OTD-TP (LATD-TP)

The LATD-TP problem is quite similar to LOTD-TP, but is less challenging due to its similarity with random walk problems. In particular, by substituting the RYIM model (3.6) into (6.1), we obtain:

$$X_t = X_{\rho_0} + \int_{\rho_0}^{t} B_t \mathrm{d}t,$$

which can be further rewritten as:

$$X_{\rho_n} = X_{\rho_0} + \sum_{k=0}^{n-1} R_k f_k I_k. \tag{6.38}$$

In Chapter 3, we have discussed the similarity between (6.38) and a random walk. The goal of LATD-TP is to find the prediction rule $\hat{X}_k \triangleq \hat{X}_{\rho_k}$ such that the expected time between transmissions is also maximized.

We claim that the *best prediction* for X_{ρ_n} is $\hat{X}_{\rho_n} = X_{\rho_0}$.

Since the following results can be proven in a similar way to previous results, we state them without proofs. Like before, we define $\mathcal{R}_k = \{\omega \in \Omega : |X_{\rho_n} - \hat{X}_{\rho_n}| \leq \delta\}$.

Lemma 6.10. $X_{\rho_n} - \hat{X}_{\rho_n}$ is a symmetric distribution, and it is also symmetric when conditioned on $\mathcal{R}_1, \mathcal{R}_2, \ldots$, given the distribution of yawing impulses R_k are symmetric.

Applying Lemma 6.3, we obtain the following theorem:

Theorem 6.11. Let a vehicle trajectory have its lateral component following the RYIM model as described in Equation 3.6. Suppose that the distribution of acceleration impulses R_k is zero-mean and symmetric. Then, the following estimate:

$$\hat{X}_n = X_0 \tag{6.39}$$

solves the LATD-TP problem (6.8).

In other words, the best prediction of a vehicle's lateral position is its last known lateral position.

6.5 STATISTICS OF THE RETRANSMISSION TIME FOR LATD-TP

For convenience, let $X_{\rho_k} \triangleq X_k$. From (6.38), it should be obvious that if R_k, f_k, I_k are independent random variables, the process X_k is constructed in a very similar way to the Wiener process, thus we can adapt the *reflection principle* to obtain the distribution of the retransmission time.

Lemma 6.12. The expected time until retransmission T of the extracted process X_k corresponding to the RYIM model is almost surely finite.

Proof. To simplify the notations, we define $\phi_k \triangleq R_k f_k I_k$. This is a product of three distributions corresponding to the random variables R_k, f_k and I_k. The proceeding is reminiscent of the proof that a Wiener process is unbounded [165].

Noticing that the lateral component of a vehicle's trajectory must be continuous, it is then sufficient to show that X_n grows unbounded almost surely as $n \to \infty$. This is equivalent to showing:

$$P\left[\bigcup_{n\geq 0}\bigcap_{m\geq n}\{|X_m - \hat{X}_m| \leq c\}\right] = 0 \tag{6.40}$$

Let p be the probability that $\phi_k \geq 2c$. It is obvious that with probability p, starting from some n with $|X_n - \hat{X}_n| \leq c$, $|X_{n+1} - \hat{X}_{n+1}| > c$. Hence, the probability:

$$P\left[\bigcap_{n\leq m\leq n+l}\{|X_m - \hat{X}_m| \leq c\}\right] = p^l \to 0, \text{ as } l \to \infty, \forall n \in \mathbb{N}.$$

Because $X_n - \hat{X}_n$ grows unbounded almost surely, $X_t - \hat{X}_t$ grows unbounded almost surely too. At $t = 0$, $X_t - \hat{X}_t = 0$ so from continuity, $X_t - \hat{X}_t$ must cross δ in finite time, almost surely. □

The expected time between retransmissions can be computed quite easily via the following theorem.

Theorem 6.13. The expected time between retransmissions T for the extracted process X_n is:

$$\mathbb{E}[T] = \frac{\delta^2}{\mathbb{E}[R_k^2 f_k^2 I_k^2]}, \tag{6.41}$$

and for the original process X_t is:

$$\mathbb{E}[T] = \frac{\delta^2}{\mathbb{E}[R_k^2 f_k^2 I_k]} \tag{6.42}$$

Proof. The result follows the fact that $X_n^2 - n\,\mathbb{E}[R_k^2 f_k^2 I_k^2]$ is a martingale, so by the Optional Stopping Theorem:

$$\mathbb{E}\left[\frac{X_T^2}{\mathbb{E}[R_k^2 f_k^2 I_k^2]} - T\right] = \mathbb{E}[X_T^2] - \mathbb{E}[T]\,\mathbb{E}[R_k^2 f_k^2 I_k^2] = 0$$

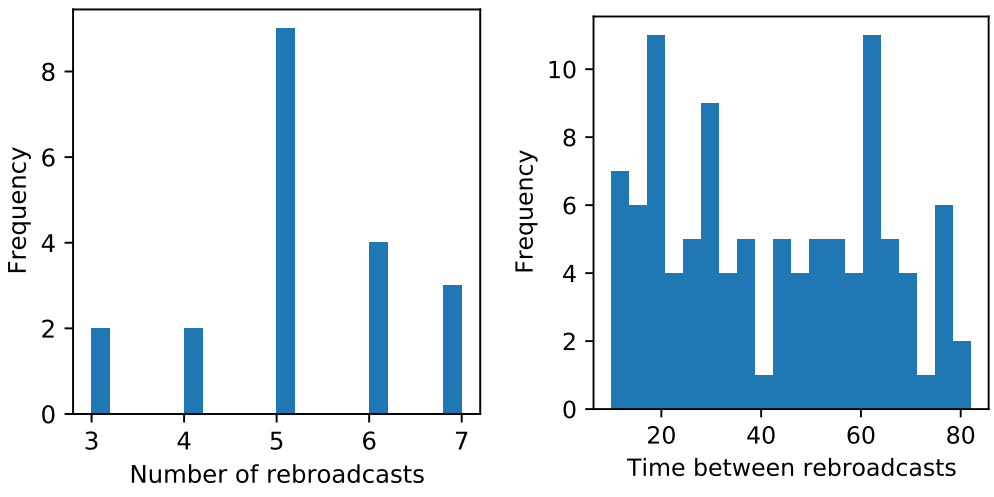
$$\Rightarrow \mathbb{E}[T] = \mathbb{E}\left[\frac{X_T^2}{\mathbb{E}[R_k^2 f_k^2 I_k^2]}\right] = \frac{\delta^2}{\mathbb{E}[R_k^2 f_k^2 I_k^2]}$$

□

6.6 NUMERICAL SIMULATIONS

In the following, we present the simulated transmission results for the Spherical Codec (Chapter 5) and the LEMONS algorithm (this chapter), based on 20 trajectories from the NGSIM-101 dataset. Due to the faster regeneration of the longitudinal component of the trajectory compared to the lateral component, the longitudinal component exerts a dominant influence on the number of rebroadcasts and the expected time between retransmissions T. We will thus focus exclusively on this component.

In both cases, the maximum tolerable error was set to $\epsilon = 3$ft. The Spherical Codec's setup adheres to that described in Chapter 5, whereas the LEMONS algorithm assumes that the mean trajectory, $\mu_{x,t}$, is the sample average derived from ten random trajectories in the dataset and is known to all road participants. It is important to note that, in a realistic implementation, this information would need to be synchronized among all parties, for example, through V2X communication.

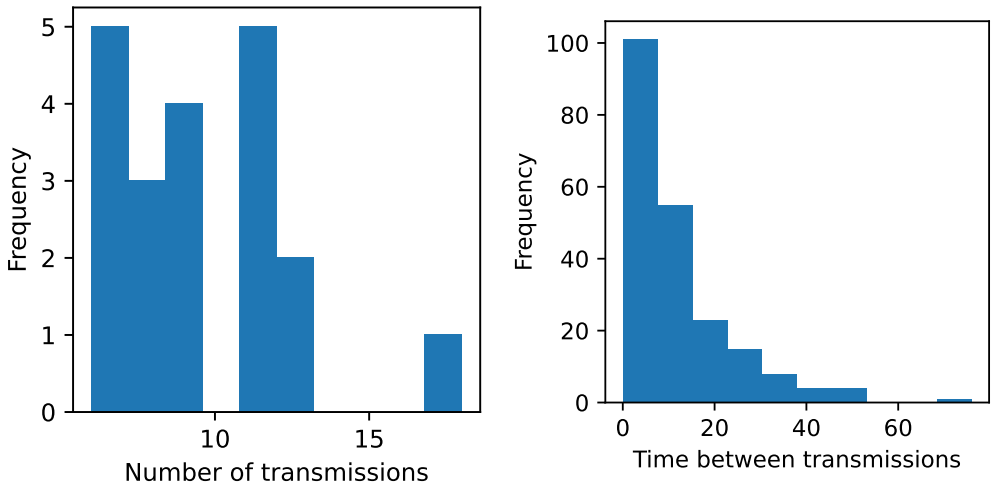


(a) (b)

Figure 6.6: Histogram for (a) the number of rebroadcasts (lower is better), and (b) the time between transmissions for LEMONS (higher is better).

As indicated in Figure 6.6, LEMONS demonstrated significantly better performance compared to Spherical Codec (Figure 6.7), with 5.2 ± 1.12 transmissions per trajectory as opposed to 9.55 ± 2.87, representing an improvement factor of 1.8 in retransmission attempts.

In terms of the expected time between retransmissions, LEMONS also offers a significantly longer prediction horizon, with 42.17 ± 20.84 timesteps of validity as opposed to 11.70 ± 11.48; an improvement factor of 3.6.



(a) (b)

Figure 6.7: Histogram for (a) the number of rebroadcasts (lower is better), and (b) the time between transmissions for Spherical Codec (higher is better).

These results confirm the optimality of LEMONS as a transmission scheme that maximizes the expected time between retransmissions. Furthermore, LEMONS is less computationally demanding than Spherical Codec due to the absence of a particle filter. Combined with its simplicity, LEMONS is a compelling choice for implementation on embedded platforms.

However, the implementation of LEMONS should be considered in the context of other dynamic data that must be transferred as well, such as velocity, acceleration, and vehicle status. Consequently, LEMONS does not aim to replace conventional V2X CAMs or CPMs, but rather contributes to the range of solutions that could be considered when communication channels are approaching their saturation points.

Similar to Spherical Codec, LEMONS also has the limitation that vehicles entering the reception range during silent periods do not receive any position information. Therefore, positions and status information still need to be broadcast frequently, albeit at a reduced rate.

6.7 CONCLUSION

In this chapter, we have developed an optimal trajectory data retransmission algorithm by conducting a rigorous analytical investigation of the RAIM and RYIM models, as previously introduced in Chapter 3. These findings underscore the broad applicability of these models and provide a comprehensive answer to the

question of optimality within the context of the OTD-TP problem. In light of the numerous potential data fields available for selection–such as position, velocity, and acceleration in conventional V2X messages, or FPCA scores as seen in the Spherical Codec—the results presented in this chapter robustly demonstrate that the mean trajectory, coupled with residual dynamic information like residual position and velocity, are sufficient to ensure the maximal expected time between retransmissions. It is obvious that, without the Random Impulses model, deriving these conclusions would have been unfeasible.

Future research directions may include refining the Random Impulses model and deriving more accurate theoretical bounds and approximations for the expected time between retransmissions. The significance of these results not only solidifies our understanding of the OTD-TP problem as well as the Random Impulses models, but also sets the stage for other engineering applications related to vehicular networking.

7

Filtering and Smoothing of Vehicle Trajectories Extracted from Aerial Videos

Abstract

This chapter extends the application of the Random Impulses Models, as detailed in Chapter 3, to detect and extract trajectory data of vehicles from aerially-captured video streams. We implement a Deep Neural Network for vehicle detection and use the Gaussian Mixture Kalman Filter and Smoother for data smoothing, based on the Random Impulses Model. Post-smoothing results demonstrate a substantial improvement in velocity estimation accuracy, with a 50% increase in the confidence interval.

Contents

A version of this chapter, titled "Exploring the Random Impulses Vehicle Trajectory Model for Dimensionality Reduction and Motion Extraction from Aerial Videos", has been submitted for review to the *IEEE Transactions on Intelligent Transportation Systems* in 2023. The authors of this submission are Thinh Hoang, Vincent Martinez, Pierre Maréchal, and Daniel Delahaye.

7.1 INTRODUCTION

In the domain of extracting high-resolution vehicle trajectories, established tracking frameworks such as the Simple Online and Realtime Tracking (SORT) [13] commonly involve methods such as Kalman or Particle Filter for data assimilation, in conjunction with the Hungarian algorithm to associate measurements with their respective targets. SORT, for instance, presupposes the following *process model*:

$$x_{k+1} = x_k + \epsilon_k, \epsilon_k \sim_{i.i.d.} \mathcal{N}(0, Q) \tag{7.1}$$

where $x_k = [u, v, s, r, \dot{u}, \dot{v}, \dot{s}]^\top$; here, s and r symbolize the scale, and u, v signify the horizontal and vertical coordinates of the target center, respectively. However, the model makes some questionable assumptions. For instance, the assumption about the nature of the model being perturbed by random walk noise is not adequately justified. Additionally, the characterization of Q as a full-rank covariance matrix is not fully appropriate.

We now propose a novel, comprehensive filtering and smoothing framework based on the previously validated RAIM and RYIM models. Despite extensive research on filtering, smoothing has been relatively under-explored. To the best of our knowledge, this is the first study to suggest a smoothing framework for this application.

7.2 RELATED WORKS

High-resolution trajectory datasets have become essential tools in the study of macroscopic traffic phenomena such as car-following, lane switching, and lane-merging [27, 166, 167]. Typically, these datasets are created through a comprehensive pipeline of image processing from primary sources, predominantly traffic CCTVs and aerial videos captured by drones. Post-processing steps such as object tracking and trajectory filtering or smoothing are important for achieving the most accurate estimations (Figure 7.1). Each of these stages presents unique technical challenges that influence the overall performance, including the reliability of the object detector given varying lighting conditions and diverse vehicle appearances. Moreover, tracking fast-moving objects and those crossing paths are inherently sensitive to error [168], leading to incomplete or corrupted trajectory tracks.

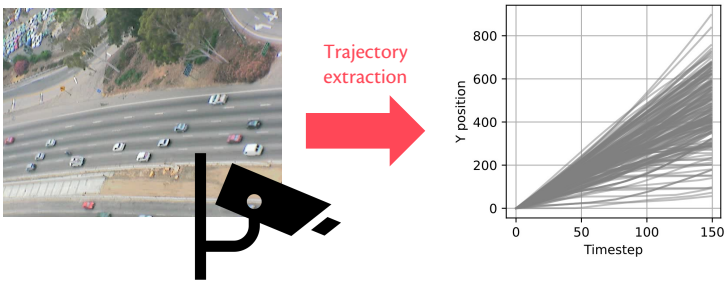


Figure 7.1: The goal of trajectory extraction is to create a set of vehicle trajectories for further studying.

The NGSIM-101 [93] was one of the earliest and remains one of the most popular trajectory datasets. As drone usage became more widespread, it was suggested that a similar processing pipeline could be adapted for traffic videos captured from a drone. This idea brought forth new technical challenges, such as the need to calibrate the viewing angle, position, and motion of the camera during capture. In [169], a general trajectory extraction framework was proposed in which vehicles were detected with a Canny edge detector and the detected edges were merged into a cohesive region representing the vehicle. The KCF tracker and wavelet transform were then used to denoise the resultant trajectories. Other methods include the use of Scale Invariant Feature Transform (SIFT) features for vehicle detection [170], optical flow for tracking [171], and a neural network for object detection [172, 173]. The latter also used a Rauch-Tung-Striebel smoother with a constant acceleration model for trajectory smoothing.

7.3 METHODOLOGY

7.3.1 Approximated Form for Lateral Component Dynamics

Let $\mu_{x,k}$ be the mean of the lateral component of the trajectory at time k. The RYIM model suggests the following approximated model:

$$\begin{aligned} x_{k+1} &= x_k + \mu_{x,k+1} - \mu_{x,k} + \eta_{x,k} \\ y_k &= y_k + \zeta_{x,k} \end{aligned} \tag{7.2}$$

where:

$$\begin{aligned} \zeta_{x,k} &\sim_{i.i.d.} \mathcal{N}(0, \sigma_x^2) \\ \eta_{x,k} &\sim_{i.i.d.} \alpha_l \mathcal{N}(\mu_{v,l}, \sigma_{v,l}^2) + \alpha_c \mathcal{N}(0, \sigma_{v,c}^2) + \alpha_r \mathcal{N}(\mu_{v,r}, \sigma_{v,r}^2) \end{aligned}$$

In other words, $\eta_{x,k}$ samples from a Gaussian Mixture composing of three modes corresponding to switching to left lane, lane-keeping and switching to right lane (Fig. 3.7 (b)), rather than from a unimodal Gaussian distribution. By definition:

$$\mathbb{E}[\eta_{x,k}] = 0 \Rightarrow \mu_{v,l} = -\mu_{v,r} \tag{7.3}$$

and

$$\alpha_l + \alpha_c + \alpha_r = 1 \tag{7.4}$$

7.3.2 Lateral Component Filtering

In the following, the filtering expressions for the particular case of RYIM model will be presented. For a more general case of Gaussian Mixture process and measurement models, we refer to [174]. Recall the fundamental equations of the Bayesian filtering problem are:

$$p(x_k|y_{1:k}) = \frac{p(y_k|x_k)p(x_k|y_{1:k-1})}{p(y_k|y_{1:k-1})} \tag{7.5}$$

$$p(x_{k+1}|y_{1:k}) = \int p(x_{k+1}|x_k)p(x_k|y_{1:k})\mathrm{d}x_k \tag{7.6}$$

The equations can be developed into a recursive algorithm, with q being the length of the trajectory and $p(x_1)$ being the prior distribution of the vehicle's lateral position. The algorithm consists of two steps: in the measurement assimilation step, we go from:

$$p(x_k|y_{1:k-1}) = \sum_{l=1}^{N_{k|k-1}} w_{k|k-1}^l \mathcal{N}(\hat{x}_{k|k-1}^l, {\sigma_{k|k-1}^l}^2) \tag{7.7}$$

to:

$$p(x_k|y_{1:k}) = \sum_{l=1}^{N_{k|k}} w_{k|k}^l \mathcal{N}(\hat{x}_{k|k}^l, {\sigma_{k|k}^l}^2) \tag{7.8}$$

by using the Kalman update for each individual component l of the predictive mixture $p(x_k|y_{1:k-1})$. In the prediction step, we go from the previous posterior distribution $p(x_k|y_{1:k})$ to the predictive distribution:

$$p(x_{k+1}|y_{1:k}) = \sum_{l=1}^{N_{k+1|k}} w_{k+1|k}^l \mathcal{N}(\hat{x}_{k+1|k}^l, {\sigma_{k+1|k}^l}^2) \tag{7.9}$$

Given that the process model's Gaussian mixture consists of three components, the quantity of components in the posterior will increase exponentially as iterations

progress. To mitigate this, we implement the Kullback-Leibler (KL) GMM Reduction, as outlined in Algorithm 1 of [175]. The main idea is to pairwise merge the components that result in the KL divergence staying bounded to some threshold limit. The filtering procedure for RYIM model is depicted in Algorithm 7.1.

7.3.3 Smoothing of Lateral Component

For a more precise estimation, the distribution of the state vector can be tailored based on future observations, rather than solely relying on the observations up until the estimation time. This leads to a *smoother*, which is usually formulated as:

$$p(x_k|y_{1:q}) = p(x_k|y_{1:k}) \int \frac{p(x_{k+1}|x_k)p(x_{k+1}|y_{1:q})}{p(x_{k+1}|y_{1:k})} \mathrm{d}x_{k+1} \tag{7.10}$$

The conventional form described is used by the renowned *Rauch-Tung-Streibel (RTS) Smoother*. However, as evidenced in [176], a closed-form solution doesn't exist when a Gaussian mixture is involved as in the case of RYIM model. Consequently, an alternative method known as the *Two-Filter approach* should be applied:

$$p(x_k|y_{1:q}) = \frac{p(y_{k+1:q}|x_k)p(x_k|y_{1:k})}{p(y_{k+1:q}|y_{1:k})} \tag{7.11}$$

where

$$p(y_{k+1:q}|x_k) = \int p(y_{k+1}|x_{k+1})p(y_{k+2:q}|x_{k+1})p(x_{k+1}|x_k)\mathrm{d}x_{k+1} \tag{7.12}$$

The precise formulas of the Two-Filter, applied to a general Gaussian mixture for both process and measurement cases, are detailed in [175]. A standout feature is the information form to circumvent computational singularity tied to the uniform distribution prior. In Algorithm 7.2, we introduce the expressions for the Backward Information Filter, specific to the RYIM model, which noticeably incorporates the mean trajectory as the control input of the process model. Owing to this unique aspect, we've named this algorithm *Smoothing over Expectation*, or SmoothiE.

The notations used in Algorithm 7.2 are as follows: $p(y_{k+2:q}|x_{k+1})$ is the Gaussian mixture whose lth component is characterized by the information $L^l_{k+1|k+2}$, $s^l_{k+1|k+2}$ is the information vector and $t_{k+1|k+2}$ is a helpful variable for the weight normalization.

Algorithm 7.1: GMKF. Abbreviations: LC - Left Change, LK - Lane Keeping, RC - Right Change, Comp - Mixture Component.

Initialize $N_{1|0} = 1, \hat{x}^1_{1|0} = x_1, p^1_{1|0} = \sigma^2_{x_1}$ ▷ A unimodal prior of the lateral position $\mathcal{N}(x_1, \sigma^2_{x_1})$

for $1 \leq k \leq q$ **do**

▷ Measurement assimilation (Eq. 7.5)

$N_{k|k} = N_{k|k-1}$ ▷ Unimodal measurement model

for $1 \leq l \leq N_{k|k-1}$ **do**

$e^l_k = x_k - \hat{x}^l_{k|k-1}$ ▷ Prediction error

$\sigma^{l\,2}_k = \sigma^{l\,2}_{k|k-1} + \sigma^2_x$

$K^l_k = \sigma^{l\,2}_{k|k-1} / \sigma^{l\,2}_k$ ▷ Kalman gain

$\hat{x}^l_{k|k} = \hat{x}^l_{k|k-1} + K^l_k e^l_k$ ▷ Kalman update

$\sigma^{l\,2}_{k|k} = (1 - K^l_k)\sigma^{l\,2}_{k|k-1}$ ▷ Posterior comp. var

$\overline{w}^l_{k|k} = \frac{\overline{w}^l_{k|k} \exp\left(-1/2 e^{l\,2}_k / \sigma^{l\,-2}_k\right)}{\sqrt{2\pi\sigma^l_k}}$

Normalize $\{\overline{w}^l_{k|k}\}$ ▷ Posterior comp. weight

end for

▷ Prediction (Eq. 7.6)

for $1 \leq s \leq N_{k|k}$ **do**

for $1 \leq j \leq 3$ **do** ▷ 3 modes: LC, LK, RC.

$N_{k+1|k} = 3N_{k|k}$

$l = 3(s-1) + j$

$\hat{x}^l_{k+1|k} = \mu_{x,k+1} - \mu_{x,k} + \hat{x}^l_{k|k}$ ▷ Position for LC, LK, RC respectively

$\sigma^{l\,2}_{k+1|k} = \sigma^{l\,2}_{k|k} + \sigma^2_p$ where $\sigma^2_p = \sigma^2_{v,l}$ if $j = 1$, $\sigma^2_{v,c}$ if $j = 2$ and $\sigma^2_{v,r}$ if $j = 3$

$w^l_{k+1|k} = w^s_{k|k}\alpha_p$ where $\alpha_p = \alpha_l$ if $j = 1$, α_c if $j = 2$ and α_r if $j = 3$

end for

end for

Perform Mixture Component pruning using KL Divergence (see Algorithm 1 of [175])

end for

7.3.4 Approximated Form for Longitudinal Component Dynamics

Let z_k represent the longitudinal position and let $\mu_{y,k}$ stand for the longitudinal mean time series value at timestep k.

$$\begin{aligned} z_{k+1} &= z_k + v_k \\ z_{k+1} - \mu_{y,k+1} &= z_k - \mu_{y,k+1} - \mu_{y,k} + \mu_{y,k} + v_k \end{aligned}$$

Let $\tilde{z}_k \triangleq z_k - \mu_{y,k}$:

$$\begin{aligned} \tilde{z}_{k+1} - \tilde{z}_k &= \mu_{y,k} - \mu_{y,k+1} + v_k \\ \tilde{v}_k &= -\overline{v}_k + v_k \end{aligned}$$

where $\overline{v}_k$ is the numerical derivative of the mean trajectory, or the mean velocity timeseries. The RAIM model then assumes:

$$\tilde{v}_{k+1} = \tilde{v}_k + \eta_{v,k}, \quad \eta_{v,k} \sim_{i.i.d.} \mathcal{N}(0, \sigma_v^2) \tag{7.13}$$

As the result, the full process model for the longitudinal state vector z is:

$$\begin{cases} z_{k+1} = z_k + v_k \\ v_{k+1} = \overline{v}_{k+1} + \tilde{v}_k + \eta_{y,k} = (\overline{v}_{k+1} - \overline{v}_k) + v_k + \eta_{v,k} \end{cases} \tag{7.14}$$

With a slight abuse of notation of the measurement being y_k for the *longitudinal position* at time k:

$$y_k = z_k + \zeta_{y,k}, \quad \zeta_{y,k} \sim_{i.i.d.} \mathcal{N}(0, \sigma_y^2) \tag{7.15}$$

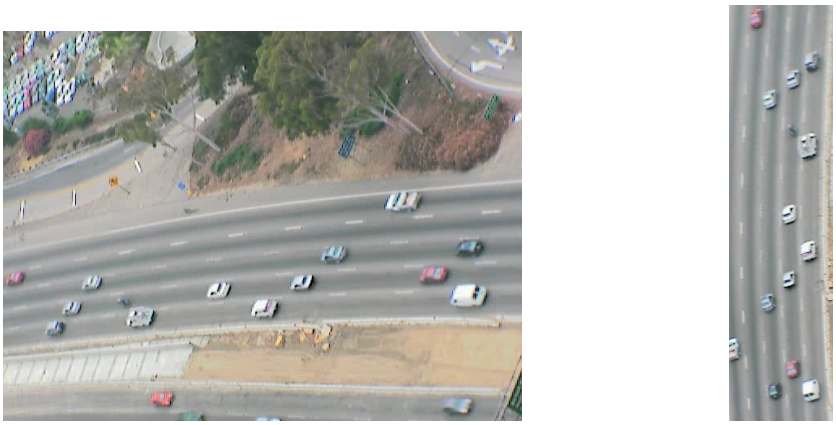
Obviously, the longitudinal component is much simpler than the lateral component since all distributions are unimodal Gaussians. This results in a familiar state-space model:

$$\begin{bmatrix} z_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z_k \\ v_k \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} (\overline{v}_{k+1} - \overline{v}_k) + \begin{bmatrix} 0 \\ \eta_{v,k} \end{bmatrix} \tag{7.16}$$

$$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} z_k \\ v_k \end{bmatrix} + \zeta_{y,k} \tag{7.17}$$

7.3.5 Filtering and Smoothing of Longitudinal Component

The aforementioned state-space model enables conventional implementation of the Kalman Filter [177] for the longitudinal component. When considering smoothing, one might tempt to apply the *Two-Filter Approach*, as detailed in preceding sections.



(a) (b)

Figure 7.2: (a) An original frame from the NGSIM 101 dataset and (b) the frame whose perspective is corrected.

Equation (7.16) implies a process model covariance matrix of

$$Q = \begin{bmatrix} 0 & 0 \\ 0 & \sigma_v^2 \end{bmatrix} \tag{7.18}$$

which does not have full-rank, complicating calculations like O^s_{k+1} in Algorithm 7.2. We suggest using the RTS smoother to avoid this, as it doesn't require inversion of the process covariance matrix.

7.3.6 Experimental Results

We derive vehicle trajectories from the NGSIM 101 dataset's aerial footage. Due to unavailable camera calibration information, we manually rectified perspective (Fig. 7.2), using pixel coordinates directly. While this doesn't yield measurements in meters, it is sufficient to demonstrate the proposed filter and smoother's effectiveness.

The full trajectory extraction framework is presented in Fig. 7.3. The car positions and bounding boxes on the perspectived transformed frame are detected with YOLOv8 [178] trained with 200 manually collected examples. The Precision-Recall Curve is shown in Fig. 7.4.

Raw trajectory extracted from the CSRT tracker, a direct application of Algorithm 7.1 and 7.2 for the lateral component, along with the KF and RTS smoother for the longitudinal component reveal the results shown in Fig. 7.5-7.7. Here the trajectories are represented in the nominal unit of pixel (px) since the metric unit requires camera calibration settings, which were unfortunately unavailable from

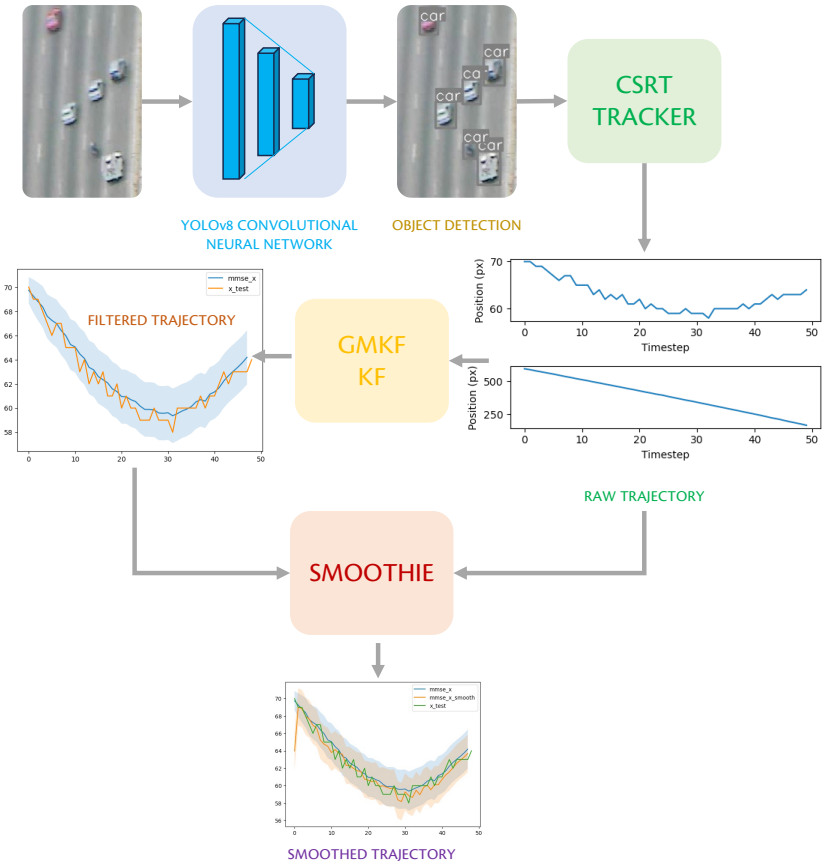


Figure 7.3: Overview of the trajectory extraction framework: the GMKF and SMOOTHIE are the novel filter and tracker proposed by using the approximated Random Impulses as dynamics models. The neural network detects cars from the video frames, while the CSRT tracker yields raw trajectories from the detected objects.

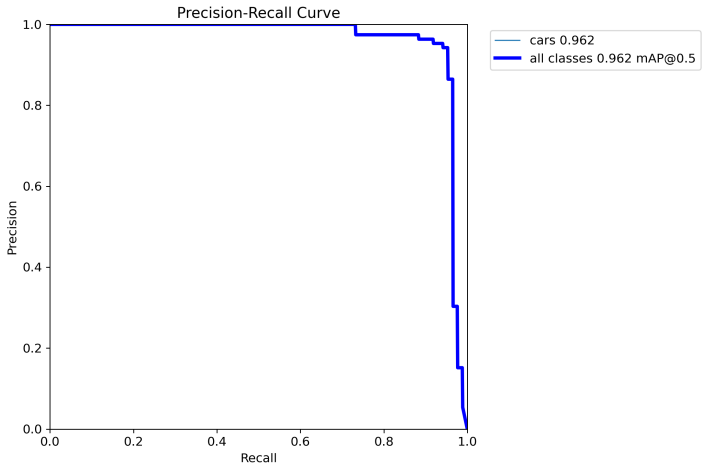


Figure 7.4: Precision-recall characteristics of the deep convolutional neural network for detection of cars in video frames.

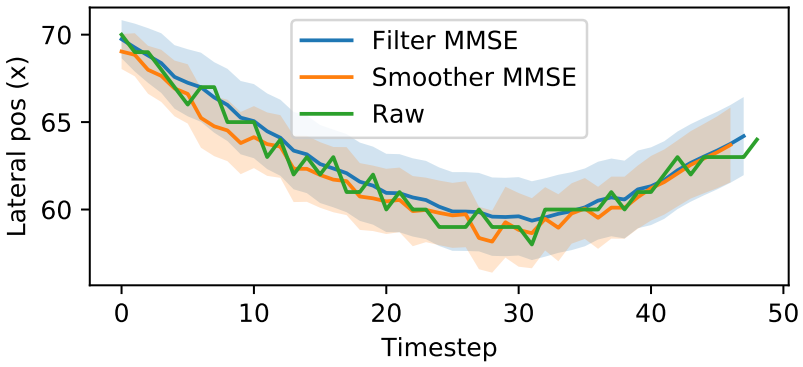


Figure 7.5: Estimation of lateral position x.

the dataset. However, regardless of the unit, the principle of operation for the tracker and the smoother remain the same. Here we have used the following parameters: $\sigma_x = \sigma_y = 0.3, \sigma_v = 0.2, \alpha_l = 0.05, \mu_l = -1.5, \sigma_{v,l} = 0.1, \alpha_{v,c} = 0.9, \sigma_{v,c} = 0.1, \alpha_{v,r} = 0.05, \mu_{v,r} = 1.5, \sigma_{v,r} = 0.1$.

The data clearly demonstrates that both the filter and the smoother produced similar outcomes concerning the lateral and longitudinal position of the vehicle. Nonetheless, the smoother's performance displayed a significant enhancement with approximately a twofold decrease in the confidence interval of the velocity. The method's uniqueness lies in the provision of the smoothed MMSE, coupled with the availability of the confidence interval, distinguishing it from conventional denoising techniques where a confidence interval is not usually obtainable. In sce-

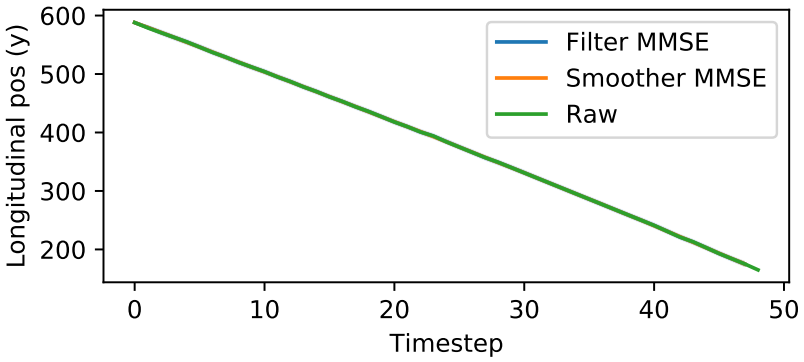


Figure 7.6: Estimation of longitudinal position y. The difference between estimates are negligible.

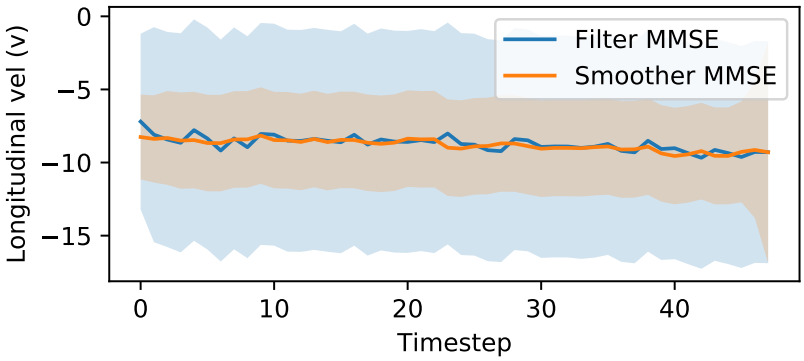


Figure 7.7: Estimation of velocity v.

narios where the vehicle's velocity does not factor into the application, using the filtered result could be sufficient for the sake of simplicity.

7.3.7 Estimation of the Mean Trajectory

As a prerequisite to both RAIM and RYIM models, the mean trajectory has to be computed for both the lateral and longitudinal components.

Proposition 7.1. The unbiased estimate of the mean trajectory is the empirical mean of the trajectory samples. The growth rate of the sample set size is proportional to n^2, where n signifies the length of the mean trajectory.

Proof. From the approximated RYIM model (7.2):

$$x_{k+1} = x_k + \mu_{x,k+1} - \mu_{x,k} + \eta_{x,k} + \zeta_{x,k}$$

then, with n being the length of the trajectory, we have:

$$x_n - x_1 = \mu_{x,n} - \mu_{x,1} + \sum_{k=1}^{n-1} \left(\eta_{x,k} + \zeta_{x,k}\right)$$

Suppose the set of samples for $x_n - x_1$ is $\{x_n^{(i)} - x_1^{(i)}\}_{i=1}^m$, using Central Limit Theorem (CLT):

$$\frac{\sum_{i=1}^m (x_n^{(i)} - x_1^{(i)}) - m(\mu_{x,n} - \mu_{x,1})}{\sqrt{m}}$$
$$\rightarrow_D \mathcal{N}(0, (n-1)(Var[\eta_{x,k}] + Var[\zeta_{x,k}]))$$

or:

$$\frac{1}{m}\left(\sum_{i=1}^m (x_n^{(i)} - x_1^{(i)}) - (\mu_{x,n} - \mu_{x,1})\right)$$
$$\rightarrow_D \mathcal{N}\left(0, \frac{n-1}{m}(Var[\eta_{x,k}] + Var[\zeta_{x,k}])\right)$$

where $\rightarrow_D$ signifies convergence in distribution.

In other words, the size of the dataset required for estimation of the mean lateral component is proportional to the length of the trajectory:

$$m = O(n) \tag{7.19}$$

For the longitudinal component, we have:

$$y_n - y_1 = \sum_{k=1}^{n-1} v_k$$

We also have:

$$v_k - v_1 = \mu_{v,k} - \mu_{v,1} + \sum_{j=1}^{k=1} \eta_{v,j}$$

Then, it is not hard to see that:

$$y_n - y_1 = \sum_{k=1}^{n-1}(v_1 + \mu_{v,k} - \mu_{v,1}) + \sum_{k=1}^{n-1}\sum_{j=1}^{k-1} \eta_{v,j} + \zeta_{y,n}$$

Notice that $\mathbb{E}\, v_1 = \mu_{v,1}$, and that both v_1 and $\zeta_{y,n}$ are dominated by the $\sum_{k=1}^{n-1}\sum_{j=1}^{k-1} \eta_{v,j}$ term if n is sufficiently large, using CLT again:

$$\frac{1}{m}\left((y_n^{(i)} - y_1^{(i)}) - \sum_{k=1}^{n-1} \mu_{v,k}\right)$$
$$\rightarrow_D \mathcal{N}\left(0, \frac{(n-2)(n-1)}{2m} Var[\eta_v]\right)$$

Consequently, the size of the sample set m grows with the square of the length of the vehicle trajectory:

$$m = O(n^2) \tag{7.20}$$

□

7.4 DISCUSSION AND CONCLUSION

This chapter presents a comprehensive probabilistic framework for filtering and, more importantly, smoothing of trajectories from aerial videos. While the position estimates of the vehicles don't differ substantially between filtering and smoothing, velocity estimation proves to be much more effective, as indicated by a confidence interval reduction of approximately 50%. This chapter also provides the theoretical background, justification, and reference implementation for the use of random walks as process noise models for vehicle tracking. While the application is currently limited to trajectory extraction from aerial videos, we believe these principles could be applicable in various automotive contexts, such as radar tracking and antenna beamforming.

However, we acknowledge several unaddressed shortcomings that warrant further research. Firstly, a more detailed experimental or numerical study that compares filtering and smoothing results against other common trackers is needed. Secondly, different mixture components pruning strategy may be studied to speed up the filter and smoother's performance.

Algorithm 7.2: Smoothing of lateral component (SMOOTHIE).

Initialize $L^1_{q|q+1} = 0, s_{q|q+1} = 0, t_{q|q+1} = 0, N_q = 1$
for $q - 1 \geq k \geq 1$ **do**

▷ Measurement assimilation (Eq. 7.11)

  for $1 \leq l \leq N_{k+1}$ **do**
    $L^l_{k+1|k+1} = L^l_{k+1|k+2} + 1/\sigma_x^2$
    $o_{k+1} = -y_{k+1}$
    $s^l_{k+1|k+1} = s^l_{k+1|k+2} + o_{k+1}/\sigma_x^2$
    $t^l_{k+1|k+1} = t^l_{k+1|k+2} + o^2_{k+1}/\sigma_x^2 + \ln(2\pi\sigma_x^2)$
  end for

▷ Backward prediction (Eq. 7.12)

  for $1 \leq l \leq N_{k+1}$ **do**
    for $1 \leq i \leq 3$ **do**
      $s = 3(l-1) + i$
      $\sigma_p^2 = \sigma^2_{v,l}$ if $i = 1$, $\sigma^2_{v,c}$ if $i = 2$ and $\sigma^2_{v,r}$ if $i = 3$
      $\alpha_p = \alpha_l$ if $i = 1$, α_c if $i = 2$ and α_r if $i = 3$
      $O^s_{k+1} = \frac{1}{L^l_{k+1|k+1} + 1/\sigma_p^2}$
      $L^s_{k|k+1} = \frac{1}{\sigma_p^2}(1 - O^s_{k+1}/\sigma_p^2)$
      $b^i_k = \mu_{x,k+1} - \mu_{x,k} + \mu_{v,l}$ if $i = 1$, $\mu_{x,k+1} - \mu_{x,k}$ if $i = 2$, $\mu_{x,k+1} - \mu_{x,k} + \mu_{v,r}$ if $i = 3$

$$s^s_{k|k+1} = \frac{1}{\sigma_p^2}\left(b^i_k + O^s_{k+1}\left(s^l_{k+1|k+1} - \frac{b^i_k}{\sigma_p^2}\right)\right)$$

$$t^s_{k|k+1} = t^l_{k+1|k+1} - O^s_{k+1}{s^l_{k+1|k+1}}^2 + \ln\sigma_p^2 - 2\ln\alpha_p$$
$$- \ln|O^s_{k+1}| + b^i_k/\sigma_p^2\left(b^i_k + 2O^s_{k+1}s^l_{k+1|k+1} - O^s_{k+1}b^i_k/\sigma_p^2\right)$$

    end for
  end for

▷ Convert to state-covariance form

  for each comp l of the predictive mixture $k|k+1$ **do**
    ${\sigma^l}^2 = 1/L^l$
    $\mu^l = -{\sigma^l}^2 s^l$
    $w^l = 0.5(\ln|2\pi{\sigma^l}^2| + L^l{\mu^l}^2 - t^l)$
  end for
  Perform Mixture Component pruning using KL Divergence (see Algorithm 1 of [175])

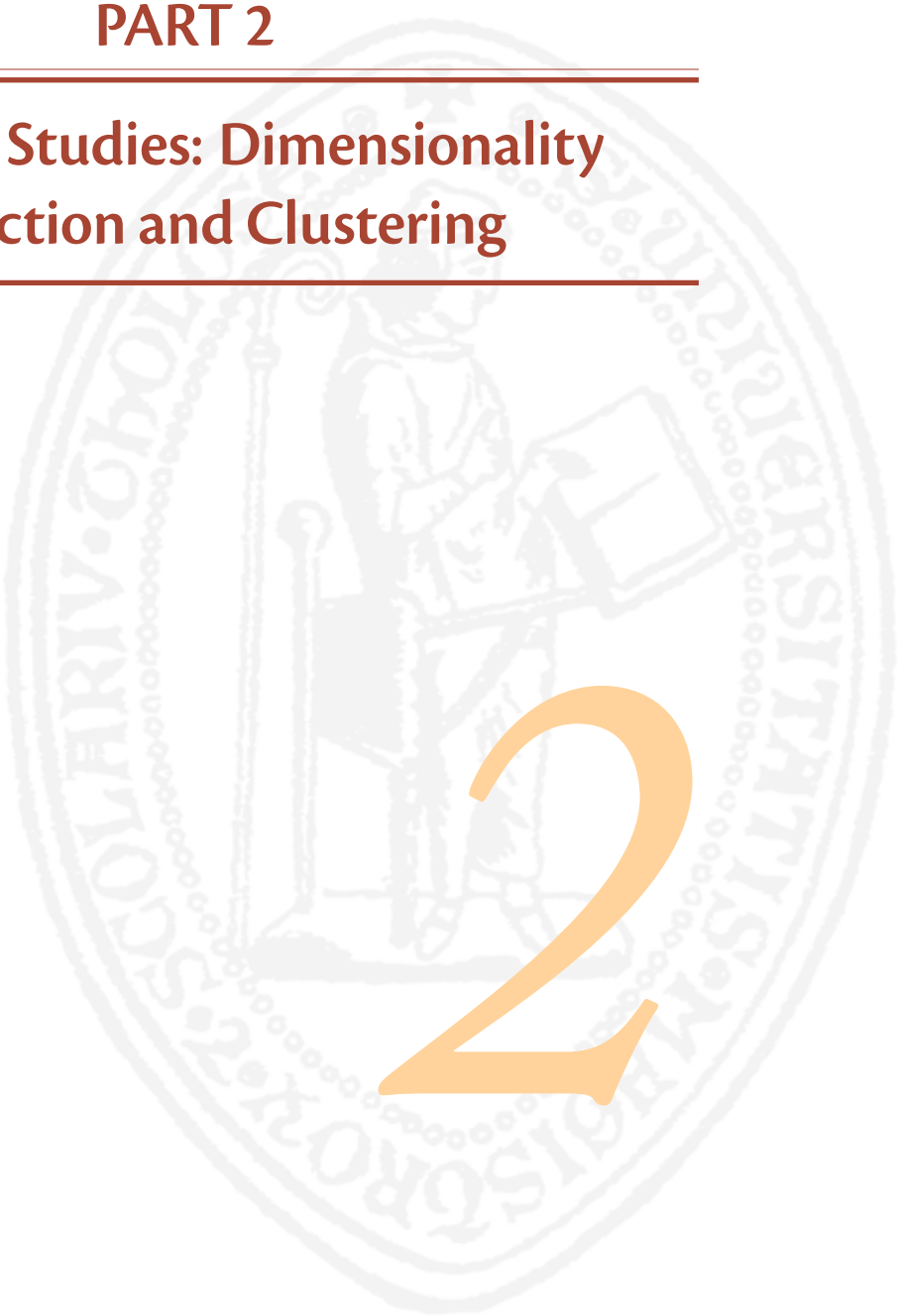
▷ Convert back to information form

Algorithm 7.2: Smoothing of lateral component (SMOOTHIE) - continued.

for each comp l of the predictive mixture $k|k+1$ **do**
$L^l = 1/\sigma^{l^2}$
$s^l = -L^l \mu^l$
$t^l = \mu^{l^2}/\sigma^{l^2} + \ln|2\pi\sigma^{l^2}| - 2w^l$
end for
end for

PART 2

Ancillary Studies: Dimensionality Reduction and Clustering



8

Dual Dirichlet Processes for Unsupervised Segmentation of Functional Data with Application to Lane-changing Behavior Characterization

Abstract

Many functional data types in traffic research, engineering, and finance are multi-modal, that is to say, containing multiple segments generated from the respective independent stochastic processes. Identifying the presence of modes in these data generally requires solving the two interlocked problems of change point detection and segment clustering at the same time. Available methods often require specifying the prior distribution based on the duration between change points and the number of modes or clusters in advance. Little is known about this information; hence in this chapter, we present a novel algorithm for unsupervised segmentation of such data that uses the Hierarchical Dirichlet Process as the prior for the number of change points and the membership function of each segment. The algorithm, in theory, might support an infinite number of change points and clusters but will try its best to explain the observations using as few clusters and change points as we like through the control of two hyper-parameters. Simulations on the lane-changing behavior dataset are given to demonstrate the algorithm.

Contents

8.1 INTRODUCTION

With the deployment of a new generation of road vehicles and instruments, the enormous amount of data generated by this intelligent transportation infrastructure can enable smarter decision makings, safer and better driving experiences. Time series are very commonly found in such system, hence analyzing this kind of data is extremely important. Despite the fact that functional data analysis is a majored field, there are still many hurdles to overcome since a common assumption of the time series being stationary rarely applies, for example, lateral vehicle trajectory data often harbors interleaved periods of lane-keeping where vehicles move in an almost straight trajectory, and lane-changing where vehicles deviate from the current lane to switch to the left or right adjacent lane. The underlying data generation processes are often multi-modal: either the dynamics will change with time, or the parameters associated with the model will shift. More than often, the change could be abrupt (such as in lateral trajectories) than smooth (in longitudinal trajectories). As a result, recognition of these changes may have strong implications on applications such as fault detection, monitoring and in designing fault-tolerant safety systems. This problem is called change point detection, where the change point indicates the moment when the segment of the time series after the change point is no longer generated by the same one before it. In other words, the change point indicates an abrupt change in the model, often associated with a significant event.

A common use case of change point detection is to detect the change of statistics such as the mean, variance or the slope of the time series (see Fig. 8.1). Applications have been found in detecting faults in the Tennessee Eastman Process [179], climate change models [180], calculation of financial returns [181] and the effects of the OPEC oil embargos on the U.S. stock market exchange [182]. A

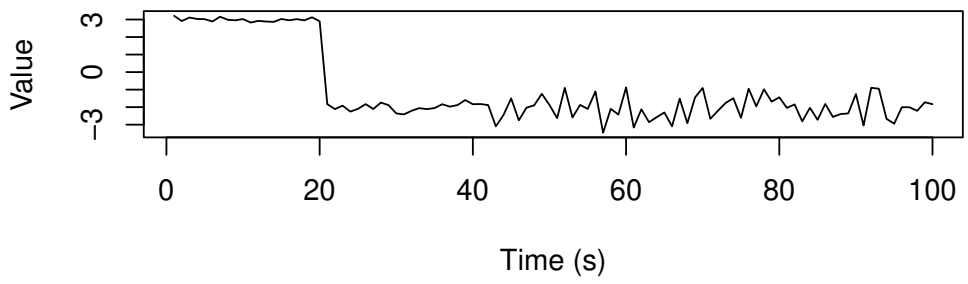


Figure 8.1: A sample time series with 2 change points: at $t = 20$ where the mean of the process changes, and at $t = 40$ where the variance of the process changes.

change point detection algorithm is often formulated as an optimization problem that aims to maximize the maximum likelihood from the product distribution where data is often assumed to be independently drawn from a distribution such as Gaussian [183] or Poisson [184]. Although exact solution to the optimization problem is possible, the exponential growth of the search space with the number of change points makes detection impossible from the practical point-of-view. Dynamic programming was introduced to circumvent the issue by breaking down the problem of finding optimal n change points into finding one optimal change point, given $n-1$ previous optimal change points for all substrings of the time series were given.

In many cases, change point detection algorithms require specification of the number of change points in advance, or a prior distribution on the duration between change points, which in all cases, is not a trivial task. On the other hand, the problem about unknown the number of states has been partially addressed in non-parametric statistics, notably the use of Dirichlet process as prior distribution for the number of hidden states of the Markov Chain [185] and the number of hidden Markov Chains in speaker diarization [186].

In this chapter, we present a novel algorithm that performs simultaneous change point detection and classification (clustering) of segments of similar slopes, which characterize the lane-keeping and lane-changing behaviors in the vehicle trajectory data (Fig. 8.2). Because Dirichlet processes are used as prior for both the number of change points and clusters, finding the precise distribution of change points is unnecessary but instead, two concentration parameters are sufficient to control the number of change points and clusters expected to be present in the dataset. Results from the algorithm are valuable for analysis of lane-change be-

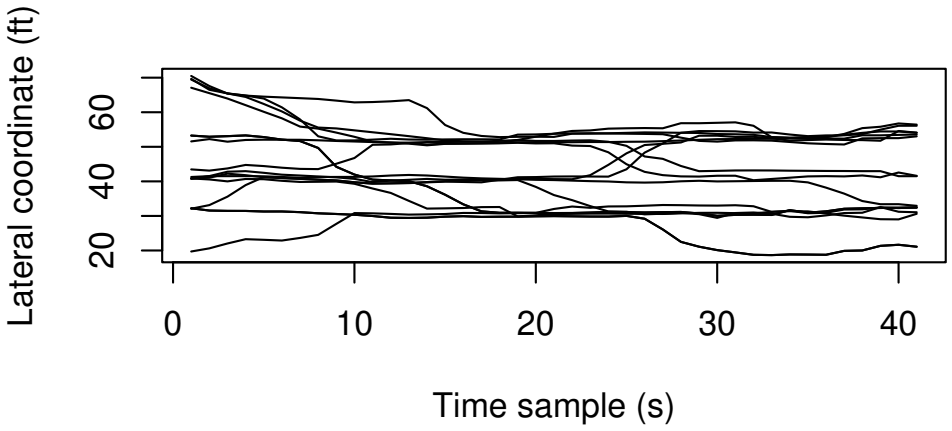


Figure 8.2: Sample lateral trajectories from the NGSIM-101 dataset.

haviors, trajectory prediction as well as anomaly detection systems. The rest of the chapter is organized as follows: an overview of the literature about change point detection is given in section 8.2. In section 8.3, we formulate the problem in a rigorous mathematical framework and derive the Gibbs sampler. In section 8.4, we provide a simulation results on the lateral trajectories extracted from the NGSIM-101 [187] dataset and make comparison with some of the popular Bayesian change point detection methods. The chapter is concluded in section 8.5.

8.2 RELATED WORK

Change point detection is a large and majored field with a vast number of approaches. According to [188], the algorithms can be divided based on the model of the cost function, the search method and the type of constraints present. Due to the length limit, we shall only focus on algorithms close to our approach, which involves maximizing the likelihood as the cost function, as well as having an unknown number of change points presenting in the time series. The problem is typically formulated under product partition model [189], where the change points are searched to maximize the likelihood assuming independence of segments between change points. The emission distribution is frequently assumed to be Gaussian, Poisson or belonging to the exponential family [190], with applications in financial stock market [191], climate models [192, 193].

The algorithm must know the exact number of change points in order to optimize their placements. As a result, the problem of unknown number of change

points is reduced to either placing a prior distribution on the discrete domain $\mathbb{N}$ or adding a constraint into the cost function, both of which are equivalent from the mathematical point of view. Historically, ℓ_0, ℓ_1 norms were used [188, 194, 195]. In parallel, Dirichlet process prior was successfully applied to constrain the number of clusters in the Gaussian Mixture Model [196]. The formulation allows one to account for potentially unlimited number of clusters with a gradual build up process on the number of clusters, thus prevents one from having to place an upper limit of clusters and calculate the likelihood for all number of clusters less than the predefined limit. The model was expanded for grouped data by nesting two Dirichlet process together, often referred to as Hierarchical Dirichlet Process, that accounts for unknown number of states in the Hidden Markov Chain, and unknown number of groups in the dataset at the higher level [186,197]. The model has demonstrated state of the art performance in speech recognition, speaker diarization and recognition of human motion data [198].

8.3 METHODOLOGY

8.3.1 Problem Formulation

The basic intuition is based on two observations of the lateral trajectories dataset:

1. Although it is not known about the exact number of change points in each time series, it is reasonable to assume that the number of change points corresponding to each vehicle passing through the same area is sampled from an invariant (latent) discrete distribution.

2. The lane-changing dynamics should be similar, whether it is in the same trajectory of one vehicle, or between two lane changing attempts of two different vehicles.

With these two observations, we can therefore define the change point detection problem in a rigorous mathematical framework:

Definition 8.1 (Time Series and Dataset). Let a time series be denoted as $\boldsymbol{x} = \{x[t], t \in \{1, 2, \ldots, T\}\} \in \mathbb{R}^T$. We also define the collection of time series in the dataset as $X = \{\boldsymbol{x}_i\}, i \in \{1, 2, 3, \ldots, m\}$.

In the definition above, a dataset contains multiple time series $\boldsymbol{x}_i$. Each time series contains a set of proprietary change points C_i. Let $x_{i:j} = \{x[i], x[i{+}1], \ldots, x[j]\}$. Following the product partition model [189]:

Definition 8.2 (Change point). A change point c of a time series $\boldsymbol{x}$ containing only one change point is the index of the time step c such that: $p(\boldsymbol{x}) = p(x_{1:c})p(x_{c:T})$.

Likewise, a time series $\boldsymbol{x}$ is said to possess a set of change points $C_i = \{c_1 = 1, c_2, \ldots, c_M = T\}$ if $\max_{c_{ij}} p(\boldsymbol{x}|c_{ij}) = p(x_{1:c_1})p(x_{c_1:c_2}) \ldots p(x_{c_{M-1:c_M}})$. We call each $x_{c_m:c_{m+1}}$ a segment of the time series $\boldsymbol{x}$.

In the change point model above, each segment $x_j = x_{c_m:c_n}$ is generated from a stationary process associated with some parameter θ_j. Between the change points, the parameter θ_j will abruptly change from one value to another and the later segment is assumed to be independent of all the earlier ones. In other words, we assume that the time series will "forget" about all of its history after the change point.

For each time series $\boldsymbol{x}_i$ in the dataset X, we would like to find the placements of the change points c_{ij} such that:

$$c_{ij} = \arg\max_{c_{ij}} p(\boldsymbol{x}_i|c_{ij}) \tag{8.1}$$

where $p(\boldsymbol{x}_i|c_{ij})$ is the probability of observing the time series $\boldsymbol{x}_i$ from the model defined as in Definition 8.2:

$$p(\boldsymbol{x}_i|c_{ij}) = p(x_{1:c_1-1})p(x_{c_1:c_2-1}) \ldots p(x_{c_{M-1}-1:c_M}) \tag{8.2}$$

in which, the probability of observing a segment $p(x_{c_m:c_{m+1}})$ is defined as follows:

Definition 8.3 (Homogeneity - Typicality - Popularity (HTP) model). Define Θ as the set of all hyper-parameters, i_m the segment indicator variable for the segment $x_{c_m:c_{m+1}}$, the joint probability of the indicator variable and the observation is modeled as:

$$p(x_{c_m:c_{m+1}}, i_m; \Theta) = p(x_{c_m:c_{m+1}}|a^*)p(a^*|i_m; \Theta)p(i_m; \Theta) \tag{8.3}$$

We now explain in details the HTP model. The reason why the three terms on the right hand side of Equation (8.3) are respectively called homogeneity, typicality and popularity will be made clear in subsequent discussions.

Homogeneity term. We assume each segment can be decomposed by W basis functions:

$$x[t] = \sum_{w=1}^{W} a_w \phi_w[t] \tag{8.4}$$

and $a^* = \{a_w\}$ coefficients are the least-square solutions to the approximation problem:

$$a^* = \arg\max_{a_w} \|x[t] - \sum_{w=1}^{W} a_w \phi_w[t]\|_2 \tag{8.5}$$

We recall the trivial result that the analytical solution to the above problem can be written as $(\Phi^T\Phi)^{-1}\Phi^T X$. If the basis $\Phi[k] = k$ is used, the a^* corresponding to the slope of the classic linear regression problem. Assuming the Gaussian observation model:

$$p(x_{i:j}|a^*) = \prod_{k=i}^{j} \frac{1}{\sqrt{2\pi}\sigma_\epsilon} \exp\left[-\frac{(x_k - a^{*\intercal}\Phi[k])^2}{2\sigma_\epsilon^2}\right] \tag{8.6}$$

Because this probability is small if the least square approximation of the original time series is poor and vice-versa, the term describes the "homogeneity" of the segment: whether it is well-explained by its approximation coefficients.

Typicality term. Let a^* be clustered by a Gaussian Mixture Model (GMM):

$$p(a^*) = \sum_{N_c=1}^{N_{c,max}} \pi_{N_c} \mathcal{N}(a^*; \mu_{N_c}, \Sigma_{N_c}) \tag{8.7}$$

and i_m the indicator variable of the segment: $i_m \in \{1, 2, \ldots, N_{c,max}\}$. We have:

$$p(a^*|i_m) = \mathcal{N}(a^*; \mu_{i_m}, \Sigma_{i_m}) \tag{8.8}$$

This quantity depends on how close a^* is to its assigned cluster's mean.

Popularity. The final term represents the likelihood of the cluster itself, which is nothing more than just the mixing proportion π_{N_c} in Equation (8.7).

8.3.2 Inference Model

Figure 8.3 illustrates the intuition behind the algorithm. In essence, we want to place the change points at places such that the slope of each segment will fit best into one of the available clusters of "slopes". We look for the solution such that both the number of change points and the number of clusters are constrained - which in our case, thanks to the Dirichlet Process prior - to prevent overfitting, which is the phenomenon where the algorithm decides to open too many clusters and change points, resulting in very short segments but achieving very high likelihood.

While i_m can be marginalized out of Equation (8.3), without the number of change points M, one can hardly solve the problem in Equation (8.1). To this aspect, we rely on other time series in X and the assumption that the number of change points across different time series should follow a similar distribution whose prior is a Dirichlet Process with rate α_{cp}:

$$M_i|\alpha_{cp} \sim Dirichlet\left(\cdot|\frac{\alpha_{cp}}{K}, \ldots, \frac{\alpha_{cp}}{K}\right) \tag{8.9}$$

α_{cp} will control the rate of increasing the number of change points. The number of change points decides on the number of segments in the dataset:

$$M = \sum_i M_i m \tag{8.10}$$

and the clustering of segments is similar to a Dirichlet Process - Gaussian Mixture Model [185, 196]:

$$\begin{aligned} \pi|\alpha &\sim Dirichlet(\cdot|\alpha/K, \ldots, \alpha/K) \\ i_{ij}|\pi &\sim Discrete(\pi) \\ x_{ij}|c_{ij} = k; \Theta &\sim \mathcal{N}(\mu_k, \Sigma_k) \\ \mu_k &\sim \mathcal{N}(\mu_0, \Sigma_0) \\ \Sigma_k &= \Sigma_1 \forall k \end{aligned} \tag{8.11}$$

We will also consider the following model:

$$\begin{aligned} \pi|\alpha &\sim Dirichlet(\cdot|\alpha/K, \ldots, \alpha/K) \\ i_{ij}|\pi &\sim Discrete(\pi) \\ x_{ij}|c_{ij} = k; \Theta &\sim \mathcal{N}(\mu_k, \Sigma_k) \\ \mu_k &\sim \mathcal{N}(\mu_0, \Sigma_k/\kappa_0) \\ \Sigma_k &\sim Inverse - Wishart_{v_0}\left(\Lambda_0^{-1}\right) \end{aligned} \tag{8.12}$$

Model (8.11) is different from model (8.12) only about the assumption of the cluster variance Σ_k. The former one assumes a constant variance across all clusters, which may help considerably in computational complexity and facilitate a simpler implementation.

The entire inference dependency graph is shown in Figure 8.4. We propose a two-step algorithm to solve the problem (8.1). The algorithm alternates between the change point optimization phase, which uses dynamic programming to solve for the exact location of change points and the segment clustering phase that follows attempts to classify the slopes of the curve into clusters (Figure 8.5). Cluster means and variances are fed back to the Change point Optimizer to be employed in the HTP model (8.3).

8.3.3 Determining the number of change points

Traditionally, the probabilities are derived in the Gibbs sampling framework. Define $n_{-i}^{(M)}$ as the number of time series, except for the ith time series, in X such that its number of change points is M, the exchangeability property of the Dirichlet Process allows us to write the probability that the ith time series contains M change points:

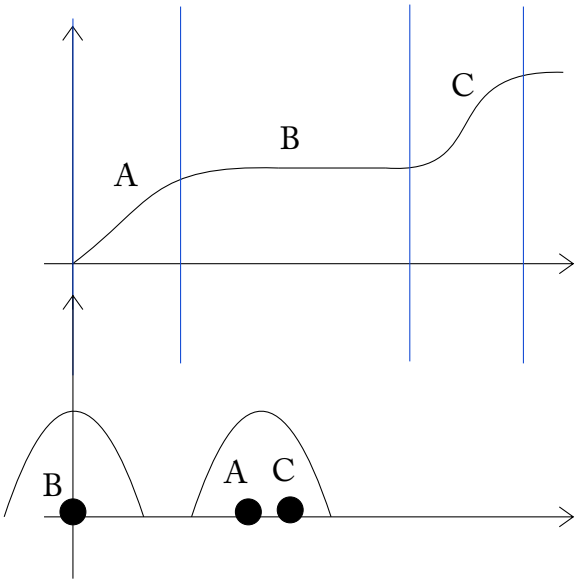


Figure 8.3: The algorithm performs simultaneous clustering of "curve slopes" (in the case of a linear basis) and change point detection. The upper graph shows the time series and the positions of change points, the lower graph shows the slopes of each segment A, B, C, along with a Gaussian Mixture fitted to show two clusters of slopes, representing lane keeping and lane changing behavior respectively.

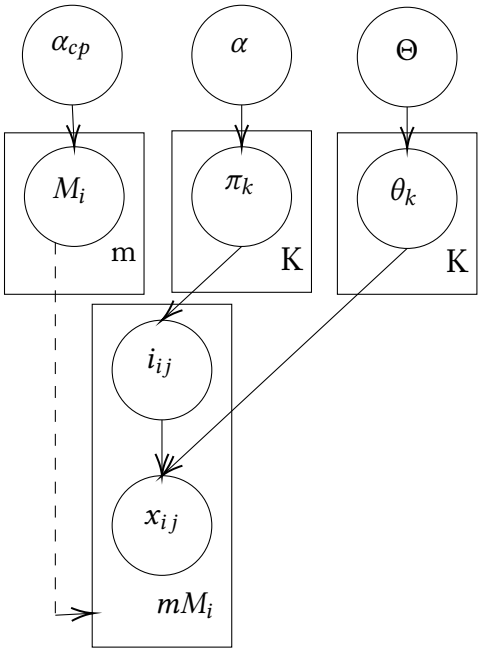


Figure 8.4: Bayesian Graph for the Unsupervised Segmentation Problem.

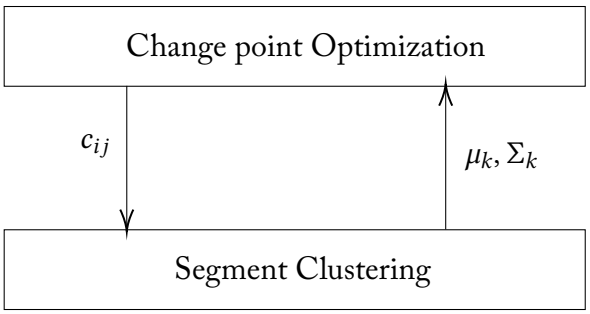


Figure 8.5: Two modules of the algorithm

Number of change points for which $n_{-i}^{(M)} > 0$:

$$p(M_i = M|X) = p(M_i = M|M_{-i}^{(M)}, \alpha_{cp}) \tag{8.13}$$

$$= \frac{n_{-i}^{(M)}}{m - 1 + \alpha_{cp}} \tag{8.14}$$

All other number of change points combined:

$$p(M_i \neq M_{i'} \text{ for all } i \neq i'|X) \tag{8.15}$$

$$= p(M_i \neq M_{i'} \text{ for all } i \neq i'|M_{-i}^{(M)}, \alpha_{cp}) \tag{8.16}$$

$$= \frac{\alpha_{cp}}{m - 1 + \alpha_{cp}} \tag{8.17}$$

For each time series $1 \leq i \leq m$ brought into consideration, equations (8.14) and (8.17) provide a discrete distribution that can be sampled to yield the number of change points for $\boldsymbol{x}_i \in X$:

$$M_i \sim p(M_i = k|X), k \in \{1, 2, \dots\} \tag{8.18}$$

8.3.4 Optimizing Change points

With the number of change points available, many methods can be used to find the location of the change points c_{ij}. In this case, we opted for exact method using dynamic programming [188, 199].

From (8.3), we can integrate out the indicator variables:

$$p(x_{c_m:c_{m+1}};\Theta) = \sum_{k=1}^{k=+\infty} p(x_{c_m:c_{m+1}}, i_m = k;\Theta) \tag{8.19}$$

$$= \sum_{k=1}^{k=+\infty} p(x_{c_m:c_{m+1}}|a^*)p(a^*|i_m = k;\Theta)p(i_m = k;\Theta) \tag{8.20}$$

$$= p(x_{c_m:c_{m+1}}|a^*) \sum_{k=1}^{k=+\infty} p(a^*|i_m = k;\Theta)p(i_m = k;\Theta) \tag{8.21}$$

To avoid cluttering of notations, we will call the jth segment of the time series under consideration as x_j. (8.21) can be rewritten as:

$$p(x_j;\Theta) = p(x_j|a^*) \sum_{k=1}^{k=+\infty} p(a^*|i_j = k;\Theta)p(i_j = k;\Theta) \tag{8.22}$$

Assuming there are $M-1$ change points in the time series, from (8.2) we have:

$$\log p(\boldsymbol{x}_i|c_{ij}) = \sum_{j=1}^{j=M} \log p(x_j) \tag{8.23}$$

$$= \log p(x_{1:c_1-1}) + \log p(x_{c_1:c_2}) + \dots \tag{8.24}$$

Hence, the problem (8.1) can be rewritten in the recursive form:

$$\begin{aligned} &\min_{c_j} \sum_{j=1}^{j=M} \log p(x_j) \\ = &\min_{t \le T-M} \left(\log p(x_{1:t}) + \min_{c_j, j \ge 2} \sum_{j=2}^{j=M} \log p(x_j) \right) \end{aligned} \tag{8.25}$$

The equation suggests that the problem of finding M change points can be decomposed into solving for $M-1$ optimal change points for all subsequences of the original time series starting at different time. It was shown that the complexity of this algorithm is $O(KT^2)$ [200].

8.3.5 Gibbs sampling

Returning to Equation (8.22) and suppose one segment x_{ij} (of one particular time series $\boldsymbol{x}_i$) is considered.

$$p(x_{ij}|X_{-ij};\Theta) = p(x_{ij}|X_{-ij},a^*)\sum_{k=1}^{k=+\infty} p(a^*|i_{ij}=k,X_{-ij};\Theta)$$
$$p(i_{ij}=k|X_{-ij};\Theta) \tag{8.26}$$
$$= p(x_{ij}|a^*)\sum_{k=1}^{k=+\infty} p(a^*|a^{*(k)}_{-ij};\Theta)p(i_{ij}=k|i^{(k)}_{-ij};\Theta)$$

It is clear that the two terms under summation are posterior predictive distribution. Similarly to the Dirichlet Process prior on the number of change points, define $n^{(k)}_{-ij}$ the number of segments in X, apart from the segment under consideration, that are assigned to cluster (of "slopes") k. We consider the last term:

Clusters for which $n^{(k)}_{-ij} > 0$:

$$p(i_{ij}=k|i^{(k)}_{-ij};\Theta) = p(i_{ij}=k|i^{(k)}_{-ij};\alpha) = \frac{n^{(k)}_{-ij}}{(\sum mM_i) - 1 + \alpha} \tag{8.27}$$

All others combined:

$$p(i_{ij} \neq i_{i'j'} \text{ for all } ij \neq i'j'|i^{(k)}_{-ij};\alpha) = \frac{\alpha}{(\sum mM_i) - 1 + \alpha} \tag{8.28}$$

The middle term of (8.26) depends on the specific model of inference we choose. If model (8.11) is used, one may have to sample for the posterior mean first [196]:

$$\mu_k|a^{*(k)}_{-ij} \sim \mathcal{N}\left(\frac{\bar{a_k}n_k\tau_k + \mu_0\tau_0}{n_k\tau_k + \tau_0}, \frac{1}{n_k\tau_k + \tau_0} + \sigma^2_\epsilon\right) \tag{8.29}$$

n_k is the number of segments assigned to cluster k, and τ_k, τ_0 are the precision (inverse of variance) of the kth cluster and the prior value respectively. Derivation details can be found in [196]. The value of the middle term then follows Equation (8.8).

On the other hand if model (8.12) is chosen, conjugacy allows a closed-form of the predictive posterior distribution:

$$a^*|a^{*(k)}_{-ij};\Theta \sim t_{\nu_n - D+1}(\mu_n, \Lambda_n(\kappa_n+1)/(\kappa_n(\nu_n - D + 1))) \tag{8.30}$$

where $t(\cdot)$ is the p.d.f for the multivariate Student distribution [201]. Let $\bar{a_k}$ be the mean of all a^* assigned to cluster k, the parameters are calculated as follows:

$$\begin{aligned}
\mu_n &= \frac{\kappa_0}{\kappa_0 + n_k} + \frac{n_k}{\kappa_0 + n_k}\bar{a_k} \\
\kappa_n &= \kappa_0 + n_k \\
\nu_n &= \nu_0 + n_k \\
\Lambda_n &= \Lambda_0 + S + \frac{\kappa_0 n_k}{\kappa_0 + n_k}(a_k - a_0)(a_k - a_0)^\intercal \\
S &= (a_k - \bar{a_k})^\intercal (a_k - \bar{a_k})
\end{aligned} \tag{8.31}$$

It is worth mentioning that the coupling between Σ_k and the variance for the cluster mean μ_k can be troublesome. However, many sampling schemes were proposed to circumvent this problem which can also be adopted here [185].

Using Bayes formula:

$$\begin{aligned}
p(i_{ij} = k | i_{-ij}^{(k)}, X) &\propto p(X | i_{ij} = k, i_{-ij}^{(k)}) p(i_{ij} = k | i_{-ij}^{(k)}) \\
&= p(x_{ij} | i_{ij} = k, i_{-ij}^{(k)}) p(i_{ij} = k | i_{-ij}^{(k)})
\end{aligned} \tag{8.32}$$

Both of these two terms are already given by Equation (8.27), (8.28), (8.29) and (8.26). The pseudo-code for the algorithm described in section 8.3 is shown in Algorithm 8.1.

8.4 EXPERIMENTS

We performed unsupervised segmentation on a small extract of lateral trajectories from the NGSIM-101 dataset [187], specifically belonging to the chunk of trajectories collected from 8:00 AM to 8:15 AM, to demonstrate the functioning of the method. Among over 200 trajectories confined to the first 40 seconds, we filtered out 15 trajectories that surely contain at least 1 change point ($|x_{t=40} - x_{t=0}| > 7ft$). The trajectories are visualized in Fig. 8.2.

The hyper-parameters are summarized in Table 8.1.

Algorithm 8.1 was implemented in R 4.1.2 `aarch64-apple-darwin20`, running on macOS 12.1 (Apple Macbook Pro, SoC Apple M1, 16GB Unified Memory). One big iteration costs around 20 seconds to complete.

To measure the performance, all the 15 time series were human labeled and the number of change points, the location of change points as well as characteristics of the slopes of the segments are compared.

The learned number of change points were shown in Table 8.2. Because the number of change points is quite invariant across all time series, we see our algorithm matches the ground truth values for most of the time series (between 2 and 3). The third time series has an abnormally high number of change points, with longer than usual lane changing time (Fig. 8.6), and our algorithm failed to detect

Algorithm 8.1: Lateral Trajectory Unsupervised Segmentation

$N_{cp} \leftarrow 1$ ▷ Max. n. change points
$N \leftarrow 1$ ▷ Max. n. clusters
while $iter < iter\_max$ **do**
  for $1 \leq i \leq m$ **do**
    for $1 \leq k \leq N_{cp}$ **do**
      $p(M_i = k|\cdot) \leftarrow$ Eq. (8.14)
    end for
    $p(M_i = N_{cp} + 1|\cdot) \leftarrow$ Eq. (8.17)
    $M_i \sim p(M_i|\cdot)$ ▷ Sample from $p(M_i|\cdot)$
    Run dynamic programming to solve (8.25) for c_{ij}.
  end for
  Get segments from the change points c_{ij} and corresponding slope values a^* from (8.5).
  while $iter2 < clustering\_iter$ **do**
    for each segment **do**
      for $1 \leq k \leq N$ **do**
        $p(i_{ij} = k|\cdot) \leftarrow$ Eq. (8.27)
      end for
      $p(i_{ij} = N + 1|\cdot) \leftarrow$ Eq. (8.28)
    end for ▷ $p(i_{ij}|\cdot)$ is the second term of (8.32)
    Calculate the first term of (8.32).
    Sample i_{ij} from $p(i_{ij}|\cdot)$.
    Merge consecutive segments belonging to the same cluster.
  end while
end while

Table 8.1: Hyper-parameters used in the algorithm

Parameter	Value
μ_0	0
Σ_0	0.1
σ_ϵ^2	0.05
α	0.1
α_{cp}	1e-126
ν_0	1
κ_0	5
Λ_0	0.5
max_iter	3
clustering_iter	750
basis	$[1, 2, \ldots, 30]$

Table 8.2: Detected number of change points of 13 time series (TS). GT are ground truth values, US are values obtained by the algorithm. Time series 11 and 15 were not shown since they correspond to 2 vehicles joining the highway from a frontage road.

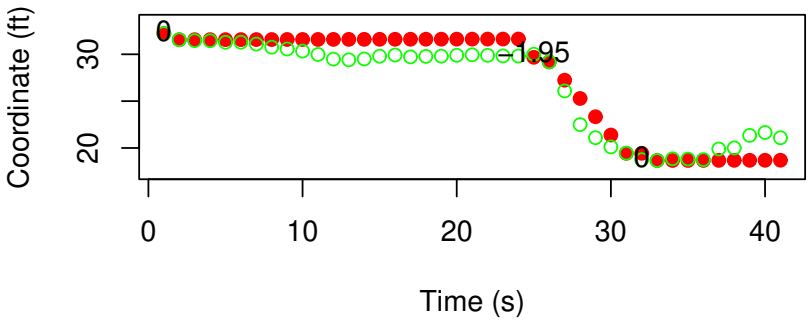
TS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
GT	2	2	**6**	2	2	3	2	2	3	3	-	2	2	2	-
US	2	2	**2**	2	2	3	2	2	3	3	-	2	2	2	-

the lane changing attempts. This is due to the assumption that all time series share similar distribution of the number of change points, a constraint enforced by the Dirichlet Process prior on the number of change points.

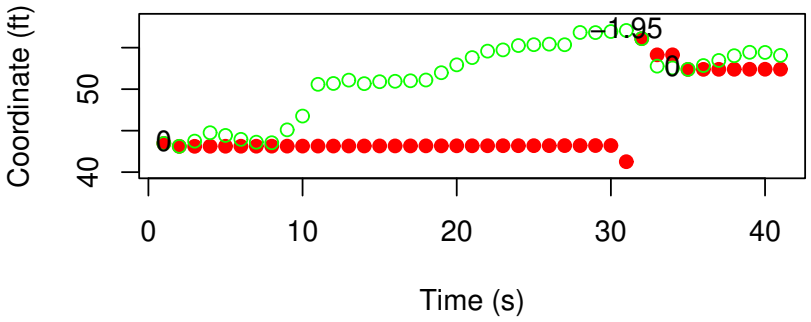
The algorithm revealed three clusters whose means are 0.003501827, −1.951116 and 2.152781. These correspond to the lane keeping behavior, lane changing to the left and lane changing to the right respectively, with the numerical value indicating the average lateral velocity (in ft/s). Fig. 8.6 and 8.7 shows the change points and the mean of the assigned clusters of the segments.

Fig. 8.8 shows the histogram of the slopes of the segments, where the three clusters can clearly be seen. Overall, the lane changing clusters have a much larger variance than the lane-keeping cluster, but it is important to note that because some time series might be incorrectly segmented (such as Time series 3), the results might not fully reflect outlier behaviors.

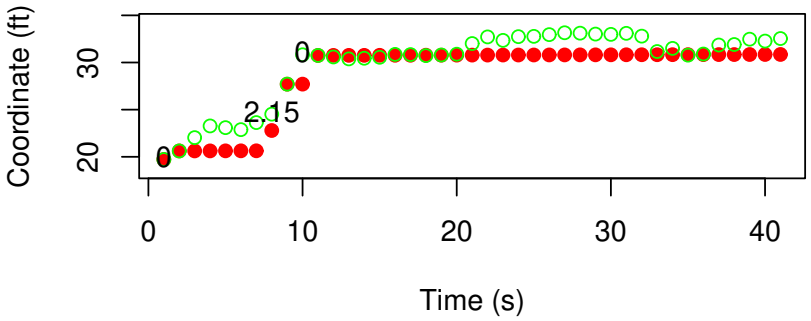
There is a caveat to this analysis: the small sample size prevents generalization of these results, but nevertheless, the algorithm has exhibited very good segmentation results, with varying number of change points and unknown clusters of "slopes" exhibited in the time series. Generalization of the method to other types of change point such as changes in variance should be straightforward. The



(a) Time series 1

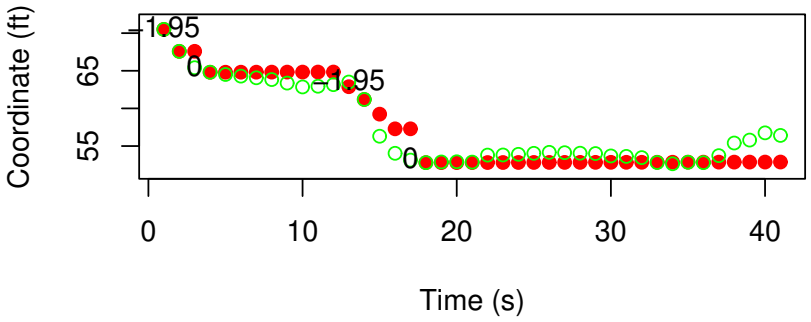


(b) Time series 3

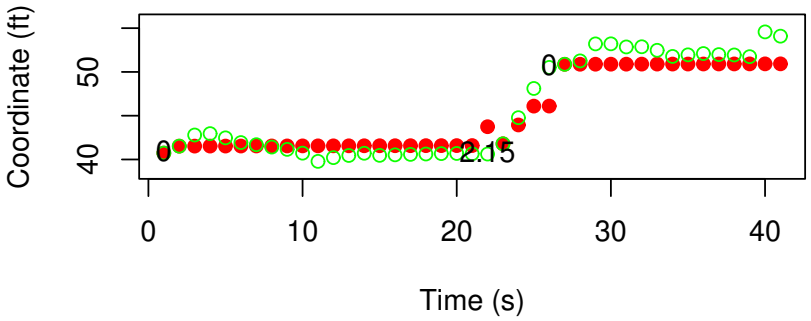


(c) Time series 5

Figure 8.6: Green dots are original trajectory data, and red dots are reconstructed trajectory data. The graphs show change points (breaks in red dots curve) and the cluster means (text at change point) of several time series in the dataset.



(a) Time series 11



(b) Time series 14

Figure 8.7: Change points (breaks in red dots curve) and the cluster mean (text at change point) of several time series in the dataset.

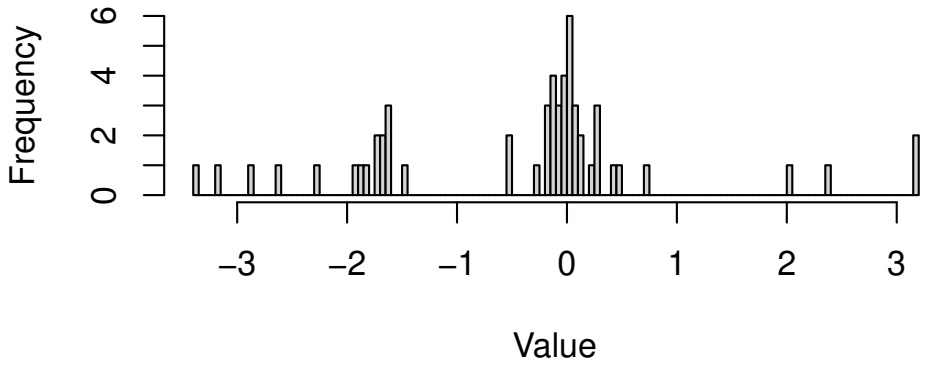


Figure 8.8: Histogram of the segment slopes.

results from the segmentation process can also be used to design a lane change detector which can help sending appropriate V2X messages [202], automatic activation of blinkers and help save redundant message transmission in Collective Perception [11].

8.5 CONCLUSION

In this chapter, we have presented a novel algorithm that performs simultaneous change point detection and clustering of similar segments. The latter provides information about what kind of information for the change point detector to look for, while both are constrained on the complexity by the Dirichlet Process as priors. Good segmentation results were achieved on a small extract of the NGSIM dataset, and the yielded cluster characteristics were able to revealed the average lane-changing velocity, as well as three corresponding modes: lane-keeping, lane-changing to the left and lane-changing to the right. These results, albeit simple, could in principle be generalized to more complex traffic situations without the need to hard-code the traffic laws and road scenarios, as well as paving the way to further applications in Intelligent Transporation Systems, such as in automatic blinker activation, turn warning systems as well as in various V2X applications.

9

Multidimensional Scaling: Hidden Gems in Trajectory Clustering and Traffic Comparison in Similar Thunderstorm Days

Abstract

In this chapter, we address two related problems related to air traffic pattern comparison on similar weather days: (1) In trajectory clustering, the result heavily depends on the algorithm's parameters. The sub-optimal choice of these parameters could lead to sub-optimal results that are often difficult to interpret as the relationships between clusters may not be clear. Is there a way to visualize and anticipate these clusters in advance as a sanity check? The other problem is that (2) given two periods of time, is there any way to determine if the air traffic situations are similar and if not, is there a quick way to tell where the differences are? Multidimensional Scaling (MDS) is found to be a common solution to these two problems by presenting the data in lower dimensions, allowing visualization of the data structure. However, traditional MDS is usually an underdetermined problem, meaning multiple solutions may exist. To this end, we propose a two-step MDS method that places several flight path embeddings first, then uses these "training" embeddings to constrain the embeddings of the remaining flight paths in the dataset. As an example application, we find two dates with similar thunderstorm developments at Dallas Fort Worth (DFW) airport and use MDS to compare the trajectory embeddings. Compared to other dimensionality reduction techniques such as PCA, MDS gives valid low-dimensional embeddings for many machine learning tasks such as clustering. Visualization of the embeddings not

only allows the detection of anomalies but also the localization of the impact on arrival patterns.

Contents

9.1 INTRODUCTION

As demand for air travel returns following the outbreak of the COVID-19 pandemic, it is reasonable to expect that the rate of growth returns to the pre-pandemic level, which was 9.7% annually [203]. Automation could help lighten the load for air traffic controllers while increasing the safety and efficiency of the system. As a result, a number of different theories and tools were developed to help better manage the situation in the context of weather uncertainty, increasing operational efficiency and travelers' comfort.

While Standard Instrument Departure (SID) and Standard Instrument Arrival (STAR) are available for most terminal areas, the inherently complex nature

This work was conducted in collaboration with the Intelligent Aerospace Systems Lab (IASL) at George Washington University in Washington, D.C., USA.

and a significant number of flight constraints prevent many aircraft from taking standard departure or arrival procedures. In addition, the presence of convective weather or thunderstorms in the terminal area further complicates decision-making, leading to diverse maneuvers such as holding and diversion for pilots and air traffic controllers. This gives rise to the problem of identifying these maneuvers from historical aircraft trajectories, as well as establishing relationships among weather, airport status and trajectory patterns. While the latter is considered central to the development of assistive decision-making tools, the value of the former cannot be overlooked and is typically achieved by using a trajectory clustering algorithm. Any sub-optimal choice of clustering parameters will lead to results that are difficult to interpret and use, and a lot of trajectory plotting is required to make sense of the relationships between the clusters.

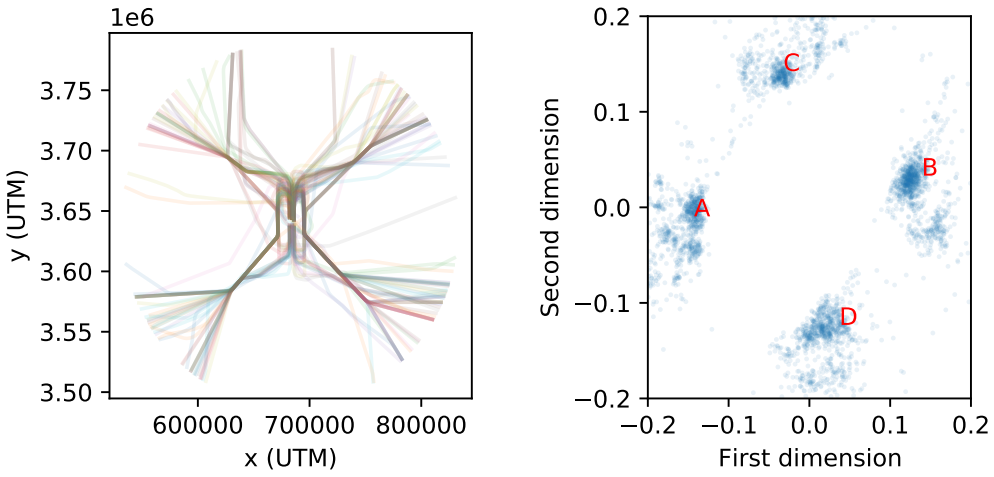
Multidimensional Scaling (MDS) is a class of unsupervised learning methods that represent a set of objects from similarity (or dissimilarity) measures between pairs of objects in low-dimensional space. The hidden structure of the dataset is then revealed and can be investigated visually, which is often not possible when considered in the original high-dimensional domain (see Fig. 9.1).

In this context, MDS can be considered as a data visualization tool, and is similar to the first step of all clustering algorithms: revealing the underlying data structure but different in that **it does not yet associate the data with clusters**. For example, in Fig. 9.1 if the parameters are not chosen carefully enough, one might end up with four large clusters of trajectories. This result, while still meaningful, does not reveal the subtle features hidden in the "small satellite clusters" around the dense centroids of these 4 clusters. These correspond to different selections of the Final Approach Fixes and could have significant implications as these could explain the impact of thunderstorms on traffic. Clearly, MDS gives us more insights into the data than simply looking at the clustering results from any algorithm such as DBSCAN. As an application, we give an example of how MDS can be used to visualize flight tracks at Dallas Fort-Worth (DFW) airport with similar thunderstorm development. We also give details about how these two dates can be found with a pruned tree-search method from the calculation of image distance. Overall, the contributions of the chapter include:

- Showing that MDS gives appropriate low-dimensional embeddings for machine learning tasks such as clustering (in contrary to PCA for instance [204]).

- Through a variation of MDS (called CMDS, presented below), the underdetermined problem of MDS becomes determinable, and thus a unique solution is obtained.

- As an example application, MDS/CMDS is used to interpret the trajecto-



(a) Trajectories **(b)** Embeddings

Figure 9.1: (b) The results of Multidimensional Scaling (MDS) of all aircraft trajectories landing at DFW on various dates in 2020 and 2021 (a), represented in UTM coordinates, with $q = n_{dim} = 2$. The four clusters A, B, C, and D correspond to four main approaches from the South-West (SW), North-East (NE), North-West (NW) and South-East (SE) directions. There are smaller "subclusters" around each main cluster which correspond to different Final Approach Fixes that were taken. Sub-optimally tuned trajectory clustering usually omits this subtle feature.

ries and understand the difference in air traffic situations in the presence of similar thunderstorm developments.

The rest of the chapter is organized as follows. In Section 9.2, we will review some related works about trajectory clustering and attempt to distinguish between common approaches (e.g., DBSCAN, K-means clustering,...) and clustering of the output of the embeddings by MDS. Section 9.3 is devoted to the derivation of the solution to the Constrained MDS problem, while the interpretation of trajectory embeddings is presented in Section 9.4. We present a method to recognize similar thunderstorm developments across different dates and times in Section 9.5 and perform a comparison of trajectories using their embeddings resulting from Constrained MDS. We conclude the chapter with Section 9.6.

9.2 RELATED WORK

Dimensionality reduction is an umbrella term for many techniques developed throughout the field of statistics and machine learning that aim to find a more parsimonious representation of the data for various tasks such as inference, denoising, and compression. Several examples include Principal Component Analysis (PCA) [205], Fisher's Linear Discriminant Analysis (LDA) [206], Locality Preserving Projections (LPP) [204]... Among them, MDS has a tight connection to clustering algorithms since under some constraints, clustering on the original high-dimensional space is exactly the same as clustering on the low-dimensional MDS embeddings.

Due to the attractiveness of computational cost, the idea of performing clustering on low-dimensional embeddings has been pursued several times. In [31], a waypoint-based trajectory clustering framework was proposed to identify anomalous aircraft trajectories and compute airspace complexity. Some of these ideas were explored again in [30], where Functional Principal Component Analysis was applied to learn the linear operator that projects aircraft trajectories onto a reduced dimension Euclidean space and a sliding window was proposed to perform real-time identification of anomalous trajectories. However, as noted in [204], while PCA yields the most optimal linear operator in terms of mean-square error, the locality and hence, manifold-related structural information may not be present on the low-dimensional embeddings of the trajectories. Consequently, it was suggested that clustering on PCA embeddings should be avoided. However, we argue in this chapter that this idea is still valid as long as the embeddings were obtained from MDS. However, as will be demonstrated later, the success of the clustering algorithms depends heavily on how the residual errors of the embeddings' placement.

While not being the key part of the chapter, we also present several works of literature regarding the application problem of studying meteorological effects on air-

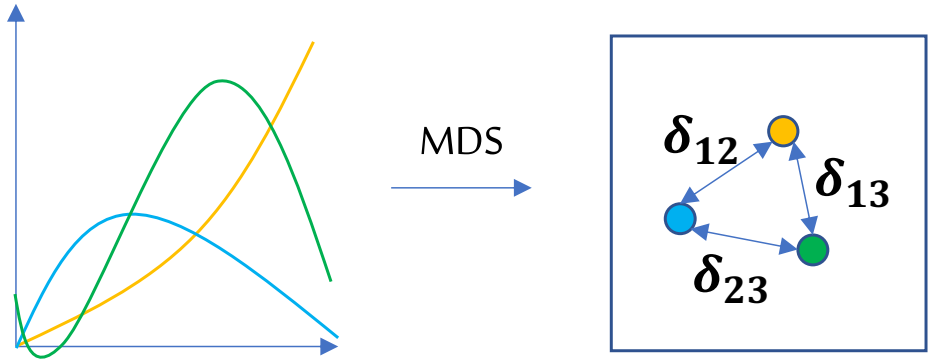


Figure 9.2: Illustration of the MDS. The embeddings, which are the positions of the 3 points on the right side were computed such that the distances between those 3 points are similar to the distances between the 3 original trajectories, given by the Δ matrix.

craft trajectories. In [207], a clustering algorithm of en-route Weather Impacted Traffic Index (WITI) based on minimum variance optimization was proposed to identify common meteorological scenarios over the U.S. national airspace. [208] proposed multiple methods to retrieve historical meteorological data to determine the presence of a Ground Delay Program (GDP) while [209] presented the training of an ensemble of decision trees with Bootstrap aggregation (BDT) to predict the days of restrictive miles-in-trails (MIT) constraints. Likewise, in [210] the authors showed the construction of a logistic model tree (LMT) to classify weather scenarios associated with GDP. Finally, [211] presented a neural network to predict the average delay and number of flights impacted from direct parsing of Meteorological Aviation Routine Weather Report (METAR) data.

9.3 MDS OF AIRCRAFT TRAJECTORIES

9.3.1 *Background on MDS*

The problem of MDS involves representing n objects (could be aircraft trajectories, people, animals...), usually in high-dimensional space $\mathbb{R}^p$ by n points in low-dimensional space $\mathbb{R}^q$ with $q << p$. While p could potentially be infinite in the case of functional data, a time-sampled vector could be used in place [212] (Figure 9.2). We define $\mathcal{S}$ to be the set of original objects: $\mathcal{S} = \{x_i \in \mathbb{R}^{N\times f}, \quad 1 \leq i \leq n\}$ and S to be the set of embeddings: $S = \{y_j \in \mathbb{R}^q, \quad 1 \leq j \leq n\}$. Let $\Delta = [\delta_{ij}] \in \mathbb{R}^{n\times n}, \quad 1 \leq i, j \leq n$ be the matrix of distance between objects pairwise with

$$\delta_{ij} = d(x_i, x_j), \quad 1 \leq i, j \leq n, \tag{9.1}$$

where $d(\cdot,\cdot)$ is a distance function between original objects. If $d(\cdot,\cdot)$ is a metric, then obviously Δ is symmetric. Examples include Euclidean, Frechet. In case of the Euclidean distance, with $z_i, z_j \in \mathbb{R}^{N\times f}$,

$$d(z_i, z_j) = \sqrt{\sum_{l=1}^{f}\sum_{k=1}^{N}\left(z_i[k,l] - z_j[k,l]\right)^2} \tag{9.2}$$

and $d(Z), Z \in \mathbb{R}^{q\times n}$ is defined by

$$d(Z) = [d(Z_{\cdot,i}Z_{\cdot,j})]_{i,j}, \quad 1 \le i,j \le n \tag{9.3}$$

where $Z_{\cdot,i}$ denotes the i^{th} column of Z, and $Z_{i,\cdot}$ denotes the i^{th} row of Z. We will keep these notations consistent throughout the remainder of the chapter.

Finding the embeddings is equivalent to minimizing the *raw stress function* [213]:

$$\begin{aligned} Y^* &= \arg\min_{Y\in\mathbb{R}^q} \sigma_r(Y) \\ \text{where } \sigma_r(Y) &= \sum_{i<j} w_{ij}\left(d_{ij}(Y) - \delta_{ij}\right)^2 \end{aligned} \tag{9.4}$$

where w_{ij} denotes the importance of the distance between the pair x_i and x_j and $d_{ij}(Y) = [d(Y)]_{i,j}$. Practically, we may assign $w_{ij} = 0$ if the distance between x_i and x_j is not available. One particular case is that either x_i or x_j is missing, and the other case is in unfolding [212]. Usually, the problem of (9.4) is solved through the SMACOF algorithm by iterating the Guttman transform [214], though the convergence rate is quite slow [215]. It is worth mentioning that faster variations are available [216, 217].

9.3.2 Data Source

Trajectory data was provided by OpenSky [218], which aggregated ADS-B packets from various volunteer receivers deployed worldwide. Naturally, ADS-B packets could be corrupted and affected data points were replaced by linear interpolation values of two nearest time points. We also took out trajectories that originated inside the 150km radius and resampled all the series at the rate of 1s. A total number of 64 thunderstorms were collected according to the criteria described below, with the earliest one occurring on 4th March 2020, and the latest one on 24th October 2022. Corresponding to these thunderstorm time periods, a total number of 3,539 aircraft trajectories were admitted to the analysis.

For thunderstorm detection, Echo Top and Vertically Integrated Liquid (VIL) are typically used. Echo Top is the highest level that the radar can detect a minimum reflectivity of 16dBz, which usually coincides with the height of precipitation. On the other hand, VIL estimates the total precipitation in the cloud. [219]

conjectured that the weather impact on the pilot's decisions can be quantified by evaluating these two parameters. Additionally, VIL Density, defined as VIL over Echo Top was proposed. This was done to resolve some inherent problems with the VIL, including identifying and assessing the severity of thunderstorms, especially if large hail was present. VIL was used in [219] to calculate the weather avoidance field (CWAP), which contains significant diversion areas due to the presence of severe convective weather activities. In this chapter, the public database of meteorological information High-Resolution Rapid Refresh (HRRR) [220] was employed to extract the VIL, Echo Top and VIL Density for an area of radius 150km from Dallas Fort-Worth Airport (DFW). HRRR contains real-time meteorological parameters with resolution up to 3km updated every 30 minutes. A severe thunderstorm is considered as present in the terminal area if within the 150km radius, the peak VIL density exceeds three, which is associated with the possibility of severe hail [221].

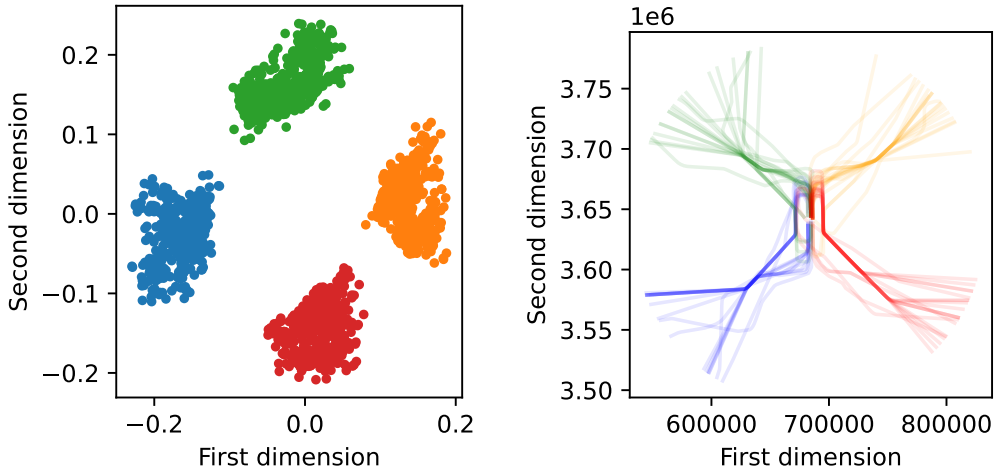
9.3.3 MDS Results

Scikit-learn [222] was used to compute the 2D embeddings of all aircraft trajectories during thunderstorm activities. The data structure is readily apparent from looking at the MDS embeddings in Fig. 9.1(b). For instance, we should expect there to be four main clusters and consequently, a clustering algorithm should have the parameters tuned to obtain 4 clusters as a result.

Obviously, if the representation is accurate, it should make hardly any difference between performing clustering directly on the trajectory data and on the trajectory embeddings. Fig. 9.3 confirms this similarity. This feature is unique to MDS, as it is not guaranteed for other dimensionality reduction techniques such as PCA [204]. Fig. 9.1(b) also provides insight into the recognition of uncommon, irregular arrival patterns. Trajectories represented by embeddings lying in sparse areas are less commonly observed. Dense areas are usually found around the Standard Instrument Arrival (STAR) patterns.

The embeddings, however, provide an greater understanding of the situation. Fig. 9.4 zooms in on cluster A shown in Fig. 9.1(b). For instance, trajectories 1158, 1885, and 3407 correspond to landings from the south of the airport, thus placed in the bottom half of the cluster. It is also clear that the trajectory 1967, lying in the dense area, seems to be the popular choice and indeed coincides with the standard arrival at DFW. By continue plotting several trajectories from the dense areas of Fig. 9.1(b), we can make the similar remarks:

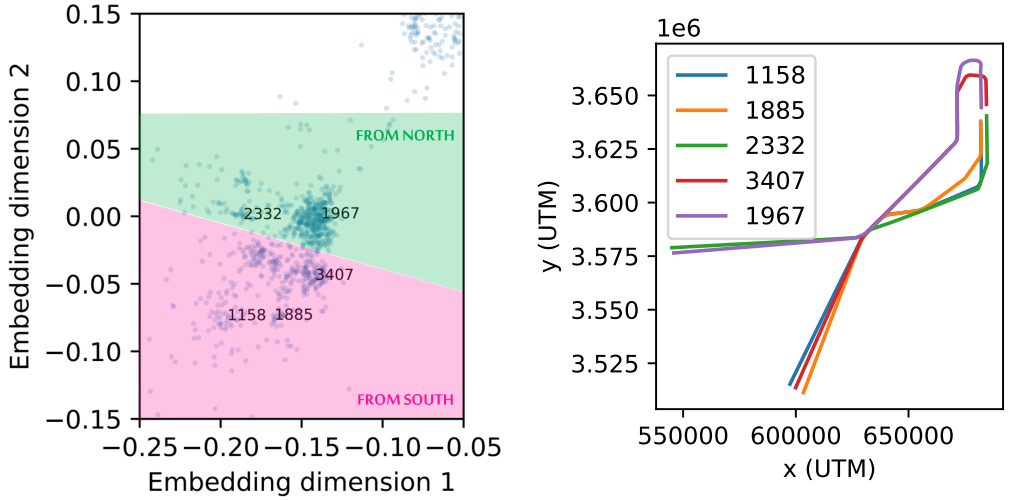
1. Clusters A, B, C, and D are associated with four approaches from the SW, NE, NW and SE directions,
2. The bottom part of cluster A corresponds to approaches from the South of



(a) Clusters of embeddings

(b) Clusters of trajectories

Figure 9.3: Performing clustering on MDS embeddings (left) and directly on the trajectory (right) shows similar results. Each color represents one cluster. Obtained with DBSCAN.



(a) Embeddings of cluster A.

(b) Trajectories corresponding to cluster A

Figure 9.4: Zoom in on cluster A, there exist several sub-clusters whose centroids' corresponding trajectories are plotted on the right. Denser areas denote more popular patterns.

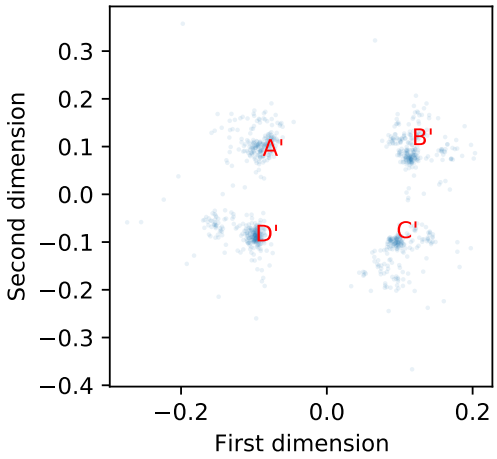


Figure 9.5: 25% set of trajectories

the airport,

3. The top part of cluster B corresponds to approaches from the South of the airport.

A visual inspection of MDS embeddings reveals these subtle features that are not immediately clear from looking at the original trajectory dataset (Fig. 9.1). Uncommon arrival patterns are also easy to identify since their embeddings lie in the sparse areas of the figure.

9.4 TECHNICAL APPROACH

9.4.1 *Two-step MDS*

The problem (9.4) is nonlinear and often contains a large number of local minima. In other words, for the same MDS problem, there could be multiple solutions that yield the same *stress*. As a result, embeddings resulting from different MDS runs cannot be compared against each other.

As an example, instead of the whole dataset, we perform MDS on just 25% of the data points. The result is shown in Fig. 9.5. Despite these 25% data points being extracted from the same original dataset, not only were the clusters placed at different coordinates compared to Fig. 9.1(b), but the ordering also changed as well. As a result, remarks made about clusters such as 1), 2), 3) in Section 9.3.3 cannot be transferred to this new scenario.

To circumvent this problem, we propose a two-step MDS as illustrated in Fig. 9.6. From the *training set* of aircraft trajectories, we perform traditional MDS

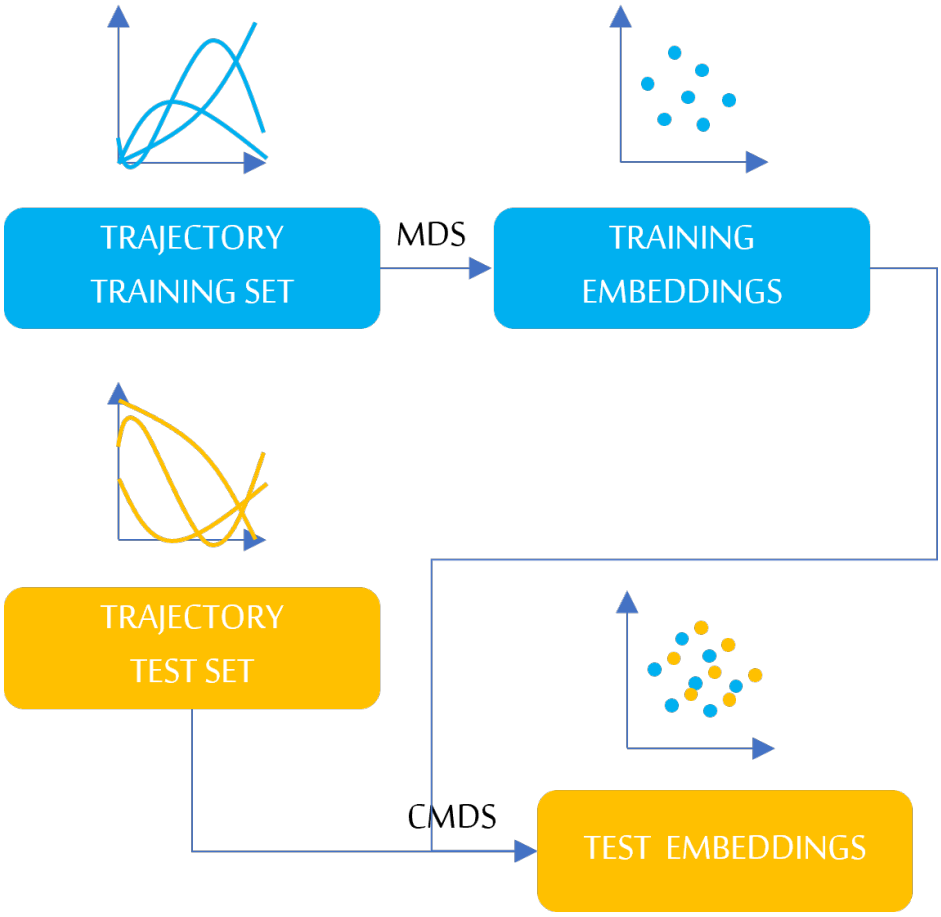


Figure 9.6: Two-step MDS algorithm.

presented in Section 9.3.3 to obtain the *training embeddings*. Then, for the *test set* of trajectories, not only are the distances between themselves computed but also the distances between test and training trajectories. Then, by treating the training embeddings as fixed, embeddings of the test trajectories are computed by optimizing an *augmented stress* function through an algorithm called Constrained MDS (CMDS). This algorithm is presented in Section 9.7.2.

Due to the additional constraints imposed by training embeddings, the placement of test embeddings becomes a determined or even overdetermined problem where a unique solution can be expected. This allows the reuse of remarks formerly made about training embeddings, similar to Section 9.3.3.

9.4.2 Derivation of CMDS

We define two additional sets of trajectories, $\mathcal{T} = \{c_j, \quad 1 \leq j \leq m\}$ and $\mathcal{S} = \{x_i, \quad 1 \leq i \leq n\}$. The set $\mathcal{T}$ includes "training trajectories", which is the training set for computation of the "training embeddings". The other set $\mathcal{S}$ includes trajectories which we wish to find their corresponding embeddings, not only with regard to the distances between themselves δ_{sym}, but also to the distances to the anchor embeddings δ_{a}. Intuitively, the training embeddings serve as "landmark" to constrain the test point embedding placement thus yielding a robust solution against different initial solutions. Suppose the embeddings of $\mathcal{A}$ are given by A, which is the solution to the traditional MDS problem in Section 9.3.3. We define the *posterior stress function* as:

$$\begin{aligned}\sigma_p(Y) = &\sum_{1\leq i<j\leq n} w_{ij}(d_{ij}(Y) - \delta_{\text{sym},ij})^2 \\ &+ \sum_{i=1}^{n}\sum_{j=1}^{m} w_{\text{a},ij}(d_{\text{a,ij}}(Y,A) - \delta_{\text{a},ij})^2\end{aligned} \tag{9.5}$$

The first sum is similar to that of traditional MDS, while the second sum expresses the constraints in the relative positions of the test embeddings compared to the training embeddings. For convenience, we will call the first term *within-set stress* and the second term *between-sets stress*. Suppose $p = \dim(c_j) = \dim(x_i), \forall c_j \in \mathcal{A}, x_i \in \mathcal{T}$, we can extend the distance function (9.2):

$$d_{\text{a}}(x_i, c_j) = \sqrt{\sum_{l=1}^{f}\sum_{k=1}^{N}\left(x_i[k,l] - c_j[k,l]\right)^2} \tag{9.6}$$

and thus,

$$\begin{aligned}\Delta_{\text{sym}} &= [\delta_{\text{sym},ij}]_{i,j} \\ \delta_{\text{sym},ij} &= d(x_i, x_j), \quad x_i, x_j \in \mathcal{T}\end{aligned} \tag{9.7}$$

and,

$$\begin{aligned}\Delta_{\text{a}} &= [\delta_{\text{a},ij}]_{i,j} \\ \delta_{\text{a},ij} &= d_a(x_i, c_j), \quad x_i \in \mathcal{T}, c_j \in \mathcal{A}\end{aligned} \tag{9.8}$$

Because the training embeddings and the test embeddings lie in the same Euclidean space $\mathbb{R}^q$, we can rewrite (9.3) as:

$$d_{\text{a}}(Y, A) = [d(Y_{\cdot,i}, A_{\cdot,j})]_{i,j} \tag{9.9}$$

Like before, we seek for the optimal embeddings of test trajectories such that the posterior stress function is minimized. The CMDS problem is about finding

$$Y^* = \arg\min_{Y \in \mathbb{R}^q} \sigma_p(Y) \tag{9.10}$$

Note that the training embeddings do not participate as decision variables and therefore remain invariant across different attempts to optimize (9.10). This is the key difference between MDS and CMDS.

9.4.3 SMACOF for Solving CMDS

Algorithm 9.1 solves the CMDS problem. It involves iterating the Constrained Guttman Transform until convergence. The complexity of the algorithm was not demonstrated due to the scope of this chapter, but it was demonstrated experimentally that it was fast. For a detailed derivation of the expressions, we refer readers to Section 9.7.2.

Algorithm 9.1: SMACOF algorithm for Constrained MDS

1: Set initial solution $Y^{[0]}$ and compute the Constrained MDS stress function $\sigma_p(Y^{[0]}) = \sigma_p^{[0]}$.
2: **while** $|\sigma_p^{[k]} - \sigma_p^{[k-1]}| > \epsilon$ and $k < k_{\max}$ **do**
3: $k \leftarrow k + 1$.
4: $Y^{[k]}$ = RHS of Equation (9.28).
5: $Z \leftarrow Y^{[k]}$.
6: Recompute $\sigma_p^{[k]}$.
7: **end while**

9.5 NUMERICAL RESULTS: VISUALIZATION OF AIR TRAFFIC FOR COMPARISON BETWEEN TWO DATES WITH SIMILAR THUNDERSTORM DEVELOPMENT

9.5.1 Interpretation of Results

Following the method introduced in Section 9.7.1, the two dates of 12th April 2022 18:00 and 17th March 2021 04:45 were found to have similar thunderstorm development: the storms were formed at the airport center, then gradually moved towards the SE (Fig. 9.9). They also lasted for a similar period of time, with a similar maximum VIL density. The CMDS embeddings allow quick visualization of air traffic during the two time periods, detailing not only whether there are perturbations to common approach patterns, but also where the perturbations were seen. Some remarks can be made out of Fig. 9.7 including:

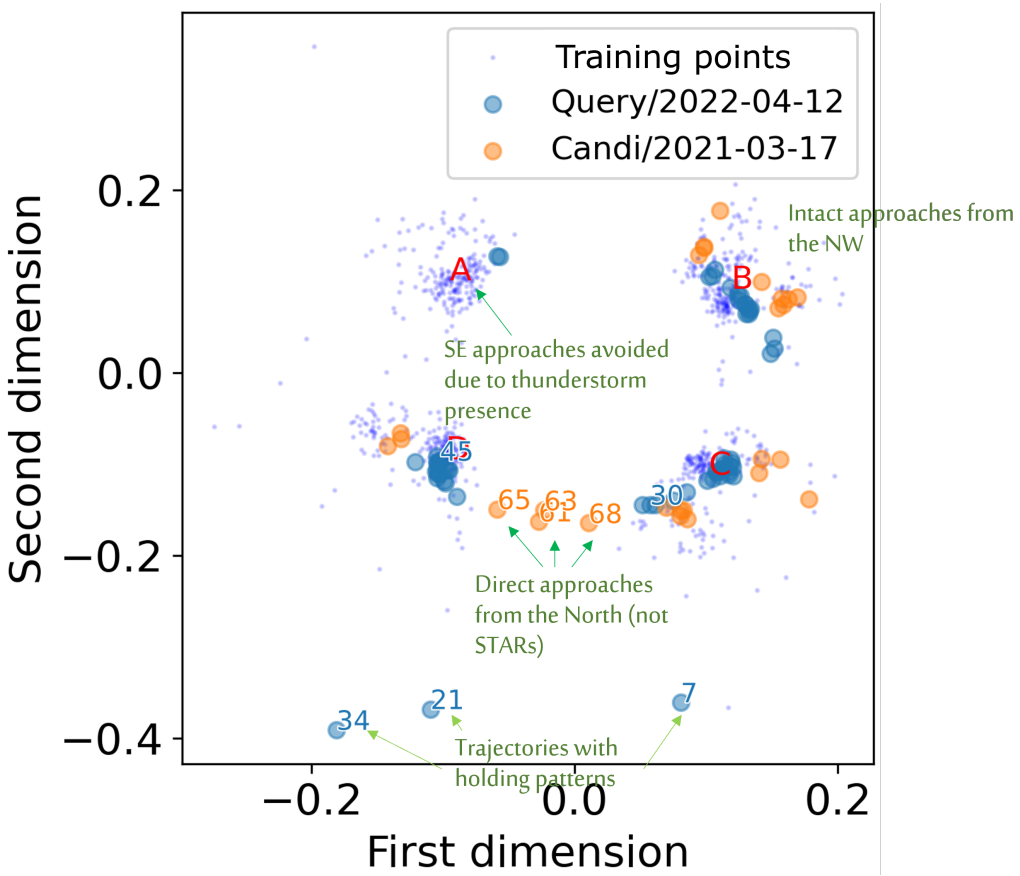


Figure 9.7: Embeddings of trajectories in two similar thunderstorm dates (query sequence and cadidate sequence). The further away from dense areas of the anchor points, the more anomalous the trajectory is. The plot allows visualizing both anomalies and similarities at the same time.

1. Cluster A in Fig. 9.7 corresponds to approaches from the SE of the airport and was virtually avoided by both dates. Only two aircraft of the querying date chose to take this approach. This could be due to the presence of thunderstorm activities in the same area, as illustrated in Fig. 9.9.
2. Cluster C in Fig. 9.7 corresponds to approaches from the SW and were virtually intact. However, the querying date includes embeddings located at the bottom of the cluster, indicating that approaches were from the North of the airport (i.e., the standard arrival), while on the candidate date, pilots were cleared for a direct approach from the South. We think this was due to the time of the day when on the candidate date, the thunderstorm happened early in the morning and therefore high demand did not exist.
3. Cluster B in Fig. 9.7 corresponds to approaches from the NW and was operational as well. However, the left part of the cluster includes trajectories approaching from the South that were pushed further away from the centroid on the candidate date. This indicates that these trajectories were more on the abnormal side.
4. Finally, it is obvious that the querying date includes several flights that lie far away from the four main clusters e.g., 34, 21 and 7. These were flights with holding patterns. There were also trajectories 65, 63, and 68 that also seem to lie on the uncommon part of the anchors too. They formed an arc that connects the two main clusters, indicating the trajectories that lie somewhere between the standard arrivals from the NE and NW of DFW. These trajectories are shown in Fig. 9.8. It was conjectured that they were cleared for a direct flight to the final approach fix due to low traffic demand.

9.5.2 Discussions

With the trajectories now represented in a low-dimensional embedding space, visualization, inference, classification, and prediction tasks become substantially simpler than in the original high-dimensional domain of trajectories due to the curse of dimensionality. For instance, anomaly detection can be performed almost visually in Fig. 9.7 as embeddings located in sparse areas indicate anomalies. It is important to note that MDS is not a bijective linear transformation, and therefore, there is no inverse transformation from the embedding back to the original domain.

In unsupervised learning, a distance function must typically be defined. For example, the embeddings in Fig. 9.1 are characteristic of the Euclidean distance. Therefore, MDS can also preview how different distance functions work. This provides insight into the choice of distance function for the application and data

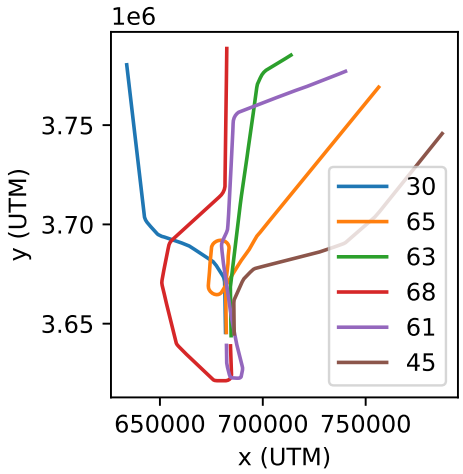


Figure 9.8: Trajectories 65, 63, 68, among the two cluster centroids: trajectories 30 and 45 correspond to standard arrivals from the NW and NE respectively.

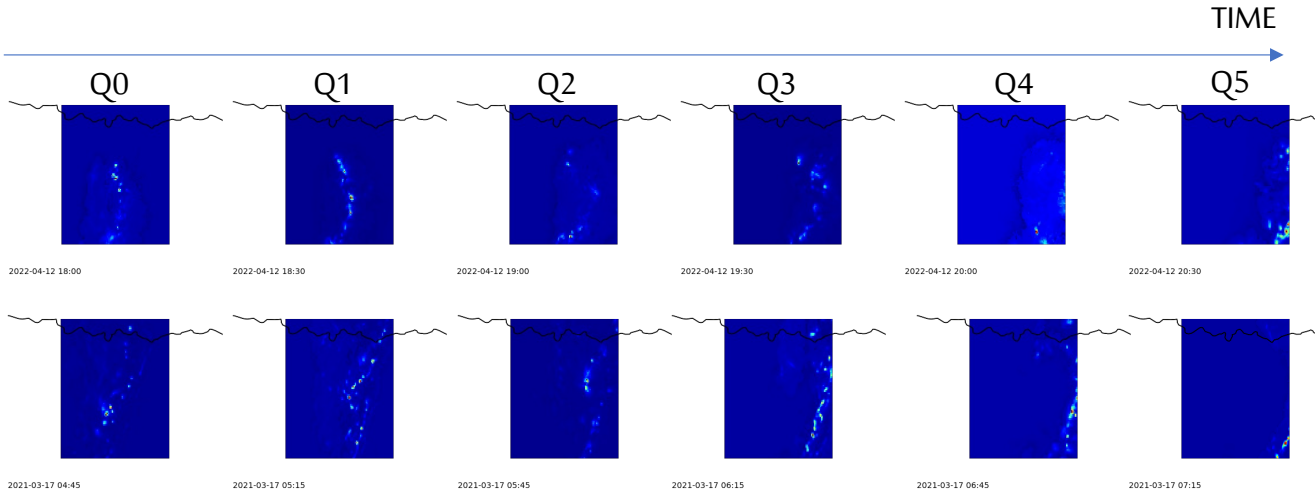


Figure 9.9: The best matching between two dates: 12th April 2022 18:00 (query sequence - above) and 17th March 2021 04:45 (candidate sequence - below).

of interest.

Finally, it is important to note that MDS does not attempt to derive the relationship between aircraft trajectories and weather factors. While the problem goes beyond the scope of this chapter, the fact is that in many cases, machine learning tasks performed on these low-dimensional embeddings are equivalent to performing on the original trajectory dataset and can greatly help the intractability associated with the latter. Hence, the value of MDS can be seen in this regard.

9.6 CONCLUSION

In this chapter, we present the application of MDS to aircraft trajectories. We introduced a two-step MDS method to visualize and make trajectories from different dates comparable. It was shown that flight tracks of aircraft arriving at DFW can indeed be represented on a two-dimensional plane. Many interesting properties were revealed by looking at the embeddings. Interpretation and machine learning tasks such as clustering can be performed on MDS embeddings instead of the original high-dimensional space, and this aspect is unique to MDS (contrary to e.g., PCA). An application for comparing trajectories between two thunderstorm dates was illustrative of the method.

9.7 APPENDICES

9.7.1 Finding Similar Dates with Thunderstorm Developments

The similarity in VIL Density lays the foundation for the distance function between any two radar images. In particular, let $X \in \mathbb{R}^{m \times n}$ and $Y \in \mathbb{R}^{m \times n}$ be the query and target image respectively. We define the Gaussian kernel as $G \in \mathbb{R}^{k \times k}$ of size k. The Gaussian blurring kernel introduces some "leniency" into the position, and allows matching the pixels in the target image at roughly the same position as in the query image. The distance between the two images is then calculated with the following formula [223]:

$$R(X,Y) = \frac{\sum_{x',y'} \left(X(x',y') - Y(x',y') * G(x',y')\right)^2}{\sqrt{\sum_{x',y'} (X(x',y') * G(x','y))^2}} \frac{1}{\sqrt{\sum_{x',y'} (Y(x',y') * G(x',y'))^2}} \tag{9.11}$$

where $*$ denotes the convolution operator. This idea can be compared to the CWAP introduced in [219].

A thunderstorm's development can span a period of time. To find the most similar thunderstorm development in the past, we may adopt a strategy similar to a pruned tree search. The keyframe is defined as the characteristic moment of the thunderstorm and serves as the starting point of the search. This usually coincides with the moment when a thunderstorm reaches its maximum intensity. The algorithm begins with the calculation of all distances between the current keyframe and all historical VIL Density data in the HRRR database. Then only the top v results were retained, and we backtracked to the previous moment in the query sequence. The same process is repeated but this time for v candidates. Each sequence fitness is graded by:

$$F(T_Q, T_C) = \sum_n R(T_Q[n], T_C[n]) \tag{9.12}$$

where $T_Q[n]$ is the n^{th} VIL radar image of the query thunderstorm T_Q. The results are then presented in descending order of candidate sequence's grades, with the lowest score indicating the most similar thunderstorm.

9.7.2 Derivation of SMACOF for CMDS Problem

In the following, we will adapt the Scaling by Majorizing a Complicated Function (SMACOF) algorithm specifically for Euclidean distances, both between trajectories and between embeddings. While the idea can be compared to [224, 225], much of the latter works focus on finding the exact location of tags with cellular towers' position known in advance. As such, the original and embedding domains share the same number of dimensions. Nevertheless, we provide an alternate solution to the problem as follows. Equation (9.5) can be rewritten as:

$$\begin{aligned} \sigma_p(Y) &= \sum_{i<j} w_{ij} \left(||y_i - y_j|| - \delta_{\text{sym},ij} \right)^2 \\ &+ \sum_{i=1}^{n} \sum_{j=1}^{m} w_{\text{a},ij} (||y_i - a_j|| - \delta_{\text{a},ij})^2 \\ &= \sigma_{ws}(Y) + \sigma_{bs}(Y) \end{aligned} \tag{9.13}$$

We focus on the between-sets stress term σ_{bs} first, since the other term is similar to the raw stress function.

$$\begin{aligned}\sigma_{bs}(Y) &= \sum_{i=1}^{n}\sum_{j=1}^{m} w_{\mathrm{a},ij}(||y_i - a_j|| - \delta_{\mathrm{a},ij})^2 = \sum_{i=1}^{n}\sum_{j=1}^{m} w_{\mathrm{a},ij}\delta_{\mathrm{a},ij}^2 + \\ &\sum_{i=1}^{n}\sum_{j=1}^{m} w_{\mathrm{a},ij}||y_i - a_j||^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{m} w_{\mathrm{a},ij}\delta_{\mathrm{a},ij}||y_i - a_j|| \\ &= \sum_{i=1}^{n}\sum_{j=1}^{m} w_{\mathrm{a},ij}\delta_{\mathrm{a},ij}^2 + \eta(Y) - 2\rho(Y),\end{aligned} \tag{9.14}$$

where we have defined:

$$\begin{aligned}\eta(Y) &= \sum_{i=1}^{n}\sum_{j=1}^{m} w_{\mathrm{a},ij}||y_i - a_j||^2 \\ \rho(Y) &= \sum_{i=1}^{n}\sum_{j=1}^{m} w_{\mathrm{a},ij}\delta_{\mathrm{a},ij}||y_i - a_j||\end{aligned}$$

Let the weight matrix be $W_{\mathrm{a}} = [w_{\mathrm{a},ij}]_{i,j}, \quad 1 \le i \le n, 1 \le j \le m$. To vectorize the second term $\eta(Y)$, we define two additional following linear operators:

$$R_{W_{\mathrm{a}}}Y = \begin{bmatrix} \sqrt{w_{\mathrm{a},11}} & 0 & 0 & \dots & 0 & 0 & 0 \\ \sqrt{w_{\mathrm{a},12}} & 0 & 0 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \sqrt{w_{\mathrm{a},1m}} & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & \sqrt{w_{\mathrm{a},21}} & 0 & \dots & 0 & 0 & 0 \\ 0 & \sqrt{w_{\mathrm{a},22}} & 0 & \dots & 0 & 0 & 0 \\ 0 & \vdots & 0 & \dots & 0 & 0 & 0 \\ 0 & \sqrt{w_{\mathrm{a},2m}} & 0 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \sqrt{w_{\mathrm{a},nm}} \end{bmatrix} Y \tag{9.15}$$

and

$$M_{W_a}A = \begin{bmatrix} \sqrt{w_{a,11}} & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & \sqrt{w_{a,12}} & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \dots & \sqrt{w_{a,1m}} \\ \sqrt{w_{a,21}} & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & \sqrt{w_{a,22}} & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \dots & \sqrt{w_{a,2m}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \sqrt{w_{a,n1}} & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & \sqrt{w_{a,n2}} & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & \sqrt{w_{a,nm}} \end{bmatrix} A \tag{9.16}$$

In essence, the $R_{(\cdot)}$ operator repeats each row of Y for m times and multiplies each row with the corresponding weight with respect to the m^{th} training embedding. The $M_{(\cdot)}$ operator, on the other hand, repeats the whole matrix A vertically for n times and multiplies each row with the corresponding weight with respect to the n^{th} test point. Then:

$$\begin{aligned} \eta(Y) &= \sum_{i=1}^{n}\sum_{j=1}^{m} w_{a,ij}||y_i - a_j||^2 \\ &= Tr((R_{W_a}Y - M_{W_a}A)^\top (R_{W_a}Y - M_{W_a}A)), \end{aligned} \tag{9.17}$$

where Tr denotes the trace of a matrix. It should be obvious that while $\eta(Y)$ is linear and convex, the first-order term $\rho(Y)$ adds the nonlinearity and removes convexity from the problem. Similar to SMACOF, we will majorize $\rho(Y)$ and adopt an interative scheme that yields a monotonic converging sequence of σ_{bs} to the local minimum.

Using Cauchy-Schwarz:

$$\begin{aligned} \langle z_i - a_j, y_i - a_j \rangle &\leq ||y_i - a_j||||z_i - a_j|| \\ ||y_i - a_j|| &\geq \frac{\langle z_i - a_j, y_i - a_j \rangle}{||z_i - a_j||} \\ -||y_i - a_j|| &\leq \frac{-\langle z_i - a_j, y_i - a_j \rangle}{||z_i - a_j||} \end{aligned}$$

Equality is achieved when $z_i = x_i, \forall i$. Since

$$\begin{aligned} -\rho(Y) &= -\sum_{i=1}^{n}\sum_{j=1}^{m} w_{\mathrm{a},ij}\delta_{\mathrm{a},ij}||y_i - a_j|| \\ &\leq -\sum_{i=1}^{n}\sum_{j=1}^{m} \frac{w_{\mathrm{a},ij}\delta_{\mathrm{a},ij}}{||z_i - a_j||}\langle z_i - a_j, y_i - a_j\rangle, \end{aligned}$$

this suggests defining the following weight matrix:

$$\begin{aligned} W_{\mathrm{z}} &= [W_{\mathrm{z},ij}]_{i,j} \\ W_{\mathrm{z},ij} &= \frac{w_{\mathrm{a},ij}\delta_{\mathrm{a},ij}}{||z_i - a_j||} \end{aligned} \tag{9.18}$$

Then, with some manipulations, we obtain:

$$-\rho(Y) \leq -Tr((R_{W_{\mathrm{z}}}Y - M_{W_{\mathrm{z}}}A)^{\top}(R_{W_{\mathrm{z}}}Z - M_{W_{\mathrm{z}}}A)) \tag{9.19}$$

and the equality follows if and only if $Y = Z$. Finally, we can rewrite (9.14) as

$$\begin{aligned} \sigma_{bs} \leq &\sum_{i=1}^{n}\sum_{j=1}^{m} w_{\mathrm{a},ij}\delta_{\mathrm{a},ij}^2 + \\ &+ Tr((R_{W_{\mathrm{a}}}Y - M_{W_{\mathrm{a}}}A)^{\top}(R_{W_{\mathrm{a}}}Y - M_{W_{\mathrm{a}}}A)) \\ &- 2Tr((R_{W_{\mathrm{z}}}Y - M_{W_{\mathrm{z}}}A)^{\top}(R_{W_{\mathrm{z}}}Z - M_{W_{\mathrm{z}}}A)) \end{aligned} \tag{9.20}$$

For convenience, let us call the right-hand side Σ_{bs}. Note that Σ_{bs} majorizes σ_{bs} and gives equality if and only if $Y = Z$. It should be immediately clear that deriving the gradient for Σ_{bs} is now straightforward.

The raw-stress σ_{ws} term is similar to the classical MDS, thus admits the same majorization function [212]:

$$\sigma_{ws} \leq \Sigma_{ws} = \sum_{1\leq i<j\leq n} w_{ij}\delta_{ij}^2 + Tr(Y^{\top}VY) - 2Tr(Y^{\top}B(Z)Z) \tag{9.21}$$

with V given by:

$$V = \sum_{i<j}(e_i - e_j)(e_i - e_j)^{\top} \tag{9.22}$$

where $e_i = [\mathbf{1}_{i=j}]_j$, i.e., a vector of zeros, except at i^{th} entry, where it is one. The $B(Z)$ matrix is given by:

$$\begin{aligned} B(Z) &= [b_{ij}]_{i,j} \\ b_{ij} &= \begin{cases} -\frac{w_{ij}\delta_{\mathrm{sym},ij}}{||z_i - z_j||}, & \text{for } i \neq j \text{ and } ||z_i - z_j|| > 0 \\ 0, & \text{for } i \neq j \text{ and } ||z_i - z_j|| = 0. \end{cases} \\ b_{ii} &= -\sum_{j\neq i} b_{ij} \end{aligned} \tag{9.23}$$

Denote $\Sigma_p(Y)$ the majorization function of $\sigma_p(Y)$, we can write:

$$\begin{aligned}\Sigma_p(Y) &= Tr((R_{W_a}Y - M_{W_a}A)^\top(R_{W_a}Y - M_{W_a}A)) \\ &- 2Tr((R_{W_z}Y - M_{W_z}A)^\top(R_{W_z}Z - M_{W_z}A)) \\ &+ Tr(Y^\top VY) - 2Tr(Y^\top B(Z)Z) + C\end{aligned} \tag{9.24}$$

where

$$C = \sum_{1\le i<j\le n} w_{ij}\delta_{ij}^2 + \sum_{i=1}^{n}\sum_{j=1}^{m} w_{a,ij}\delta_{a,ij}^2 \tag{9.25}$$

and this term is not significant to finding the solution to the optimization problem since it does not contain any decision variable.

Since $\Sigma_p(Y)$ is convex and differentiable, one may find the minimum by setting its first derivative to zero:

$$\begin{aligned}0 &= \frac{\partial}{\partial Y}\Sigma_p(Y) = 2R_{W_a}R_{W_a}^\top Y - 2R_{W_a}^\top M_{W_a}A \\ &- 2R_{W_z}^\top(R_{W_z}Z - M_{W_z}A) + \frac{\partial}{\partial Y}\Sigma_{bs}(Y) \\ &= 2R_{W_a}R_{W_a}^\top Y - 2R_{W_a}^\top M_{W_a}A \\ &- 2R_{W_z}^\top(R_{W_z}Z - M_{W_z}A) - 2VY - 2B(Z)Z\end{aligned} \tag{9.26}$$

After some simplifications, we obtain the algebraic equation that allows finding the minimizer of the majorization function $\Sigma_p(Y)$ at Z.

$$(R_{W_a}^\top R_{W_a} + V)Y = R_{W_a}^\top M_{W_a}A + R_{W_z}^\top(R_{W_z}Z - M_{W_z}A) + BZ \tag{9.27}$$

Usually, $R_{W_a}^\top R_{W_a} + V$ is not invertible, but a substitution of Moore-Penrose inverse gives the minimizer:

$$\begin{aligned}Y^* = (R_{W_a}^\top R_{W_a} + V)^+(R_{W_a}^\top M_{W_a}A \\ +R_{W_z}^\top(R_{W_z}Z - M_{W_z}A) + BZ)\end{aligned} \tag{9.28}$$

To follow the convention, we call (9.28) the *Constrained Guttman Transform*. The result is obtained from iterating (9.28) and setting $Z = Y^{[k]}$ until convergence.

10

Real-time Localized Anomaly Detection for Identifying Non-compliant Approaches

Abstract

In this chapter, we extend the anomaly detection discussion from Chapter 4 by considering a nuanced variation of the problem. Instead of using all accumulated observations to test a binary hypothesis (i.e., whether observations are *nominal* or *abnormal*), we focus on assessing whether a time series significantly deviates from the norm at a specific moment. Typically, such assessments are made using a nearest-neighbor algorithm, which requires the involvement of numerous data points from the training set to test the hypothesis. However, through the use of Functional Principal Component Analysis (FPCA), we will show that a more refined and efficient approach is possible. We apply the algorithm to the problem of identying non-compliant approaches at Toulouse Blagnac Airport.

Contents

10.1 INTRODUCTION

During the final approach, pilots are typically required to follow a set of procedures, as detailed in flight path safety management during nominal operations, to align with the runway's centerline and follow a 3° glideslope until the runway threshold. However, there are various factors that may cause the aircraft to be too high, too low, too fast or too slow during the final approach:

1. Pilot's lack of familiarity with the airport or misinterpretation of landing instrument indications.
2. Weather conditions, such as low visibility, might prompt pilots to initiate early or late descents. Tailwinds and headwinds can cause the aircraft to move faster or slower than usual, which could lead to overshooting or undershooting the runway threshold.
3. Air traffic controller's instructions, such as those requiring an aircraft to maintain a higher altitude due to traffic congestion, may also contribute to deviations from the standard glideslope.

As a result, the detection of abnormal final approaches is critical for the safety and efficiency of aviation operations. The French Aviation Authority risk portfolio categorizes Non-stabilized Approaches (NSA) as those with an "interception angle $> 45°$ (or $> 30°$ on parallel active approaches) and/or an intermediate leg shorter than 30 seconds (or 2Nm for GNSS App) before the Final Approach Point (FAP) and/or glide path interception from above and/or a non-adapted speed (greater than 180 kts)." This definition extends the concept of a non-stabilized approach, typically defined as occurring 1000 ft from the runway threshold, which is a strong precursor to the risk of a go-around.

This chapter extends the methodology presented in [30], where an energy approach was presented. The motivation for anomaly detection with energy is from the view that landing is a process of trading off altitude and speed–or in physical terms, potential energy and kinetic energy. In other words, a smooth landing should follow a gradual decrease in both altitude and speed–not one of the two. The total specific energy E_T can be computed from:

$$E_T = \frac{1}{2}(G_s^2 + V_z^2) + gh, \tag{10.1}$$

where G_s is the ground speed and V_z is the vertical speed. By plotting E_T against distance to runway threshold D, it is possible to identify atypical approach patterns from typical ones in an unsupervised manner. Clearly, this approach is better in the sense that complex approach patterns may exist for individual airport and weather

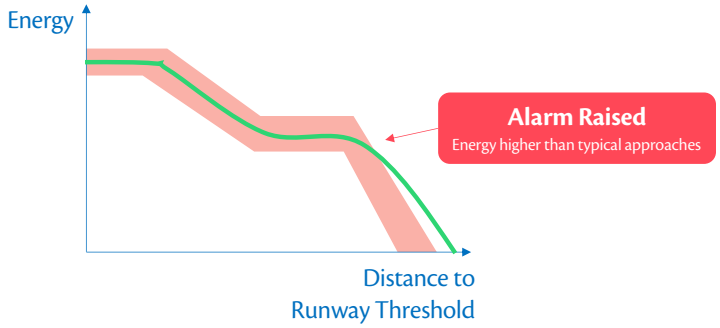


Figure 10.1: Anomaly Localization: raising alarm when energy E_T is substantially higher than what is typicall observed at the same distance to runway threshold D.

condition, and sometimes the "by-the-book" criteria for non-stabilised approach like descent angle or altitude that initiate the final descent may not suit all scenarios.

However, the unsupervised detection approach may only provide pilots or ATCOs with information such as "your approach has higher energy than 95% of the approaches seen at this airport," without specifically explaining why nor whether it is possible to continue landing or a go-around should be initiated (Figure 10.1).

Similar to Chapter 4, real-time detection and early warning issuance to air traffic controllers (ATCOs) and pilots are critical. In an attempt to address this problem, [30] used a sliding window to locally compare the dimensionality-reduced FPCA score vector with neighboring vectors using the Global-Local Outlier Score from Hierarchies (GLOSH). However, this approach introduces some delay into the algorithm, as the first sliding window must be filled before detection can occur. In response, we propose a probabilistic framework to detect anomalies localized at specific distances from the runway threshold. This approach operates similarly to the Local Outlier Factor (LOF) but does not require the entire training dataset to compute the distance to neighbors. Consequently, the algorithm is simpler, faster, and equivalently effective.

10.2 ANOMALY DETECTION FRAMEWORK

Figure 10.2 provides an overview of the anomaly detection framework. Similar to the approach outlined in Chapter 4, this algorithm consists of two phases: the *learning phase* and the *operating phase*. During the learning phase, the energy curves of the training set are decomposed into FPCA scores, effectively reducing the dimensionality of the data. Subsequently, a Gaussian Mixture Model is fitted to

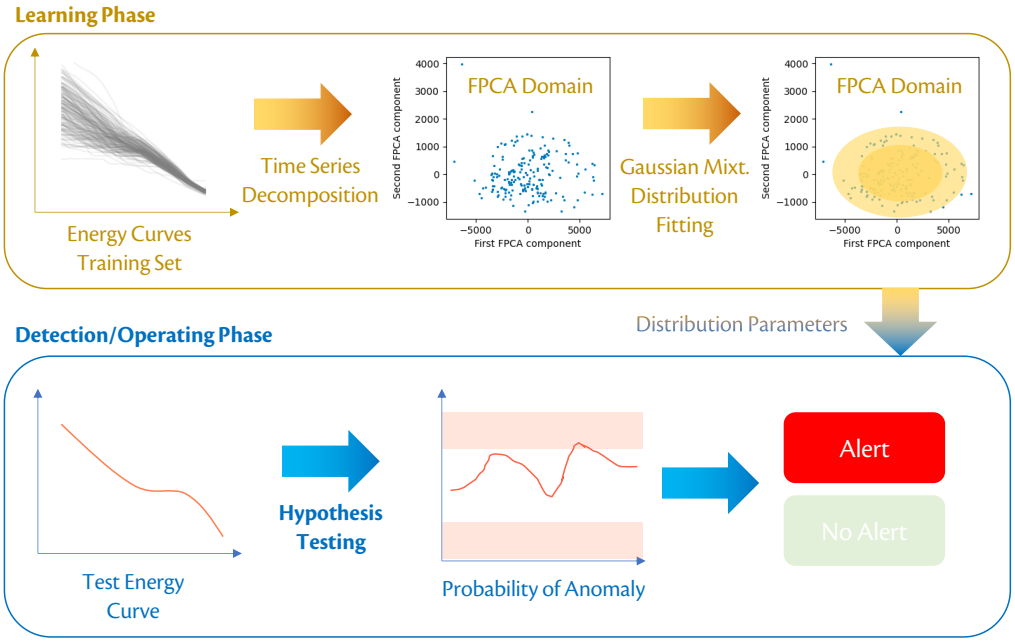


Figure 10.2: Localized anomaly detection framework.

the FPCA score vectors, and the resulting model parameters are then transferred to the operating phase. Unlike Nearest Neighbor methods, this approach does not require the retention of a significant portion or the entirety of the training set for the detection operation, which can be costly in terms of memory usage. The algorithm's reliance on compact distribution parameters makes it a more efficient alternative.

The localized anomaly detection approach discussed in this chapter differs from the framework introduced in Chapter 4 in its focus on identifying non-compliant or non-stabilized approaches. In this context, the primary concern is how the energy value at a specific distance from the runway threshold compares to typical values observed at that same distance. The specific trajectory that leads to this energy value at that particular moment is fully irrelevant. In contrast, the probabilistic framework described in Chapter 4 takes into account the entirety of all observations up to the exact moment of detection to determine whether the observed trajectory deviates from the norm. In essence, the methodology presented in this chapter is concerned with a single moment, while the previous framework considers all observations leading up to that moment.

Figure 10.3 illustrates the 200 energy curves of A320s landing at Toulouse Blagnac Airport. The underlying concept is that a typical approach should pass through the dense areas of the graph, as these represent energy values frequently observed in past landings. It may be careful for an aircraft to avoid sparse or white

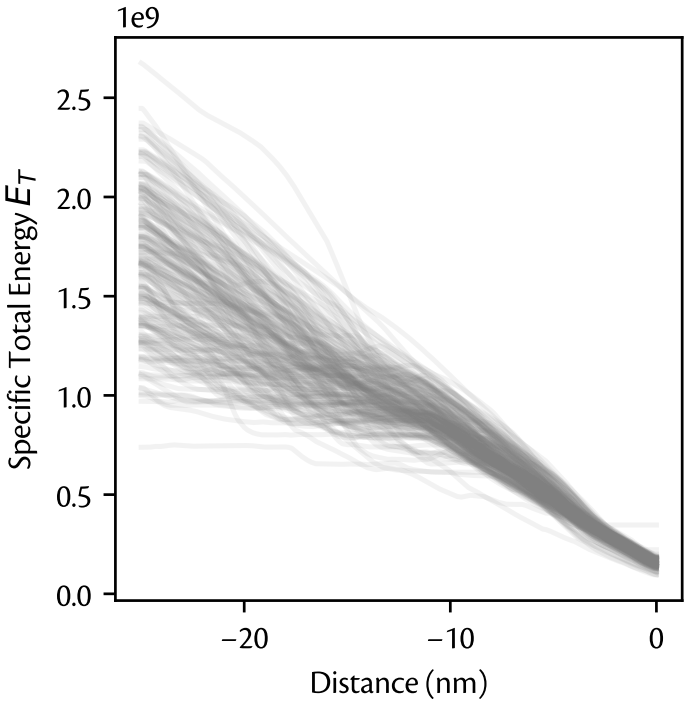


Figure 10.3: Total specific energy of aircraft landing trajectories at Toulouse Blagnac Airport (LFBO).

areas of the graph, as these represent combinations of speed and altitude that have not been previously encountered at similar moments.

The learning phase in this approach is analogous to the one introduced in Chapter 4, with one important difference: a Gaussian Mixture Model is fitted to the FPCA scores (e.g., using the maximum likelihood method) instead of a simple Gaussian distribution. This modification accommodates the possibility that FPCA scores may not follow a normal distribution.

In a real-time detection setting, suppose the FPCA scores' Gaussian Mixture can be represented as:

$$a \sim \sum_{k=1}^{N} \pi_k \mathcal{N}(\mu_{a,k}; \Sigma_{a,k}) \tag{10.2}$$

and the FPCA decomposition is:

$$E_T(t) = \mu_{E_T}(t) + \Phi(t)^\top a, \tag{10.3}$$

where:

$$\Phi(t) = [\phi_1(t), \ldots, \phi_M(t)], \quad M = \dim a$$

Notice that (10.3) is an affine transformation of a, thus $E_T(t)$'s distribution is a Gaussian Mixture as well. The following result is trivial to show:

Lemma 10.1. If $x \in \mathbb{R}^n$ is a random vector that has a Gaussian Mixture distribution parametrized by (π_k, μ_k, Σ_k), then for any affine transform $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^n$, the random vector $Ax + b$'s distribution is also a Gaussian Mixture parametrized by $(\pi_k, A\mu_k + b, A\Sigma_k A^\top)$.

Proof. Recall that if x follows a Gaussian Mixture, then the corresponding characteristic function is:

$$\phi_x(t) = \sum_k \pi_k \phi_{x_k}(t) = \sum_k \pi_k \exp\left[it^\top \mu_k - \frac{1}{2} t^\top \Sigma_k t\right] \tag{10.4}$$

For the random vector $y = Ax + b$, the characteristic function of y is:

$$\begin{aligned}
\phi_y(t) &= e^{it^\top b} \phi_x(A^\top t) \\
&= e^{it^\top b} \left(\sum_k \pi_k \exp\left[i(A^\top t)^\top \mu_k - \frac{1}{2}(A^\top t)^\top \Sigma_k (A^\top t) \right] \right) \\
&= \sum_k \pi_k \exp\left[it^\top (A\mu_k + b) - \frac{1}{2} t^\top A \Sigma_k A^\top t \right]
\end{aligned} \tag{10.5}$$

and notice that this is the exact characteristic function of a Gaussian Mixture distribution parametrized by $(\pi_k, A\mu_k + b, A\Sigma_k A^\top)$. □

Lemma 10.1 suggests that the variable E_T in (10.3) is also a Gaussian Mixture, and that:

$$P(E_T \leq y) = \sum_{k=1}^{N} \pi_k F_N \left(\frac{y - \mu_{E_T}(t) - \Phi(t)^\top \mu_{a,k}}{\sqrt{\Phi(t)^\top \Sigma_{a,k} \Phi(t)}} \right) \tag{10.6}$$

It is now obvious that expression (10.6) can be used to test the hypothesis H_0: the specific energy at distance to threshold D is significantly higher or lower that typical values, with a level of significance α.

10.3 NUMERICAL SIMULATIONS

Figure 10.4 displays the energy curves of five randomly selected flights, derived from radar tracks of A320s landing at Toulouse Blagnac (LFBO) airport. The fifth flight intercepts the glideslope at a slightly lower altitude and airspeed than usual, while flights 3 and 4 are positioned somewhat above the glideslope at approximately 8 nautical miles from the runway threshold.

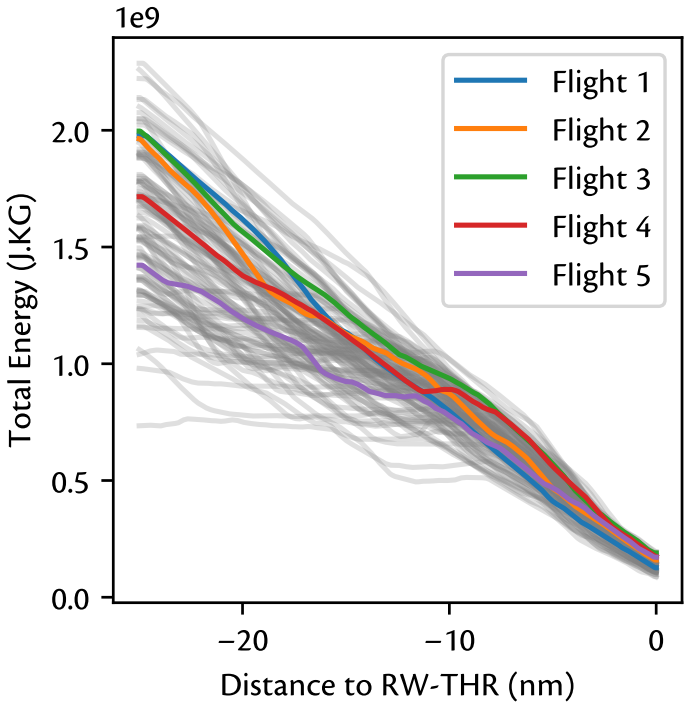


Figure 10.4: 5 random energy curves of flights landing at LFBO, along with the training set curves in gray.

Figure 10.5 displays the real-time p-values for each flight. Ideally, these curves should hover around 0.5. With a significance level of $\alpha = 0.05$, the red shaded rectangles indicate the regions where alarms will be triggered, pointing to atypical energy values at the corresponding distances. As discussed in Chapter 4, we exclude the last 3 nautical miles of data from the analysis due to the divergence in approximation power, which is suspected to result from the window effect.

It is obvious from Figure 10.5 that Flights 3 and 4 are positioned at the very top, but not to the extent that would trigger an alarm. These trajectories are closed to be considered non-compliant approaches. This observation corresponds with the fact that these two curves also appear near the upper boundary of the traversed learning set in Figure 10.4. As previously mentioned, Flight 5 approaches the runway slightly shallow, but it remains within the typical range throughout its descent.

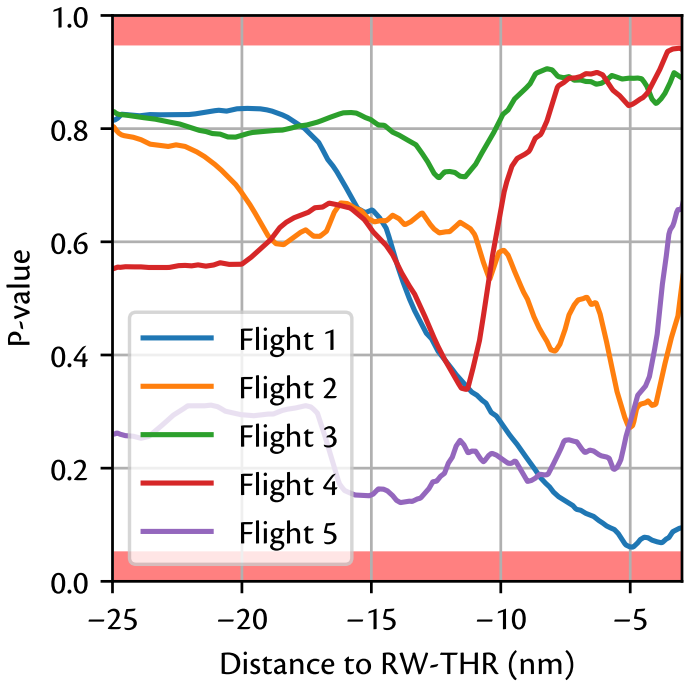


Figure 10.5: P-values curves corresponding to the 5 chosen flights. The red shaded rectangles indicate $p = 0.05$ and $p = 0.95$. Curves hitting these rectangles should trigger alarm as the specific energy is either significantly high or low.

10.4 CONCLUSION

In this chapter, we have extended the probabilistic framework presented in Chapter 4 to address the problem of identifying localized anomalies, specifically for the detection of non-compliant approaches in an unsupervised manner. The methodology requires only a training sample of energy curves and does not require any specific detection instructions for its implementation. Consequently, the algorithm is versatile and adaptable to various airports, even those with complex landing patterns. Compared to the approach presented in [30], our method does not involve any sliding window, making it faster and more efficient. We believe that this approach can be further extended to multivariate signals and has the potential for application in a wide range of other disciplines.

11

Conclusion

SUMMARY

In this dissertation, we have overviewed a multitude of signal processing and machine learning issues associated with vehicle trajectories. Our exploration encompasses modeling, data compression, anomaly detection, dimensionality reduction, prediction, and the unsupervised segmentation of time series that exhibit repetitive and localized patterns. It is widely acknowledged that these challenges span virtually across all Intelligent Transportation System (ITS) applications. Consequently, the advancements explored in this work could significantly influence the design and implementation of ITS in various cities and urban environments, thereby aiding the transition towards more efficient, comfortable, and environmentally friendly transportation systems and urban centers.

Random Impulses Models. When considering the increasing advancement of AI and data-driven methodologies, some of which have been demonstrated to surpass the expectations of researchers, it is imperative to exercise caution, particularly in the context of road safety for all participants. This caution underscores the continued relevance and importance of signal modeling techniques. This value is further emphasized in situations where data is scarce and cost-effectiveness is a priority. Just as Fourier Transforms have been and continue to be the cornerstone of stationary signal processing, Random Impulse models offer a balanced approach between the adaptability of learning models and the rigor of mathematical models. These models are transparent, free from any ambiguity that would impede mathematical analysis, and thus provide insights that can inform the design of crucial applications that would otherwise be unfeasible. This was successfully achieved in Chapter 3, leading to two important applications—data compression and anomaly

detection.

V2X Channels Congestion Control. The core issue of congestion control in Vehicle-to-Everything (V2X) channels can be addressed through the silent inference scheme, in which only the necessary information should be transmitted, while the receiver should invest significant effort to minimize the need for frequent retransmission attempts from the transmitter. In Chapter 5, we investigated how linear transformative coding could help lower the message rate by broadcasting the dimensionality-reduced representation vector of the trajectory, which is estimated in real-time using a Kalman or Particle Filter. Chapter 6 extended this investigation by exploring the optimal solution for the silent inference problem, revealing that a V2X message format should incorporate both the mean trajectory and residual dynamics information to optimize the expected time between retransmissions. This approach resulted in a 2.5 to 4-fold improvement in message rate while maintaining an acceptable error tolerance of approximately 3 feet.

Trajectory Filtering and Smoothing. The Random Impulses Models also propose a dynamic model for the filtering, smoothing, and prediction of future trajectories by using a Gaussian Mixture Kalman Filter (GMKF). The details of the reference implementation were discussed in Chapter 7, which demonstrated an enhancement in the estimation confidence interval of the velocity by a factor of two.

Anomaly Detection. The topic of Anomaly Detection was explored in Chapters 4 and 10, where a probabilistic framework was introduced for real-time detection of anomalies within time series data. Chapter 4 outlined the likelihood and Bayesian methodologies for one-shot anomaly detection. Meanwhile, Chapter 10 presented a technique to locally compare the features of a time series with others in the training set within a compact representation domain, a method that proved useful in the detection of unstabilized landings.

The field of unsupervised time series segmentation, closely related to the problem of changepoint detection, was also a subject of exploratory research discussed in Chapter 8. Moreover, the aircraft landing trajectory representation and visualization problem, presented in Chapter 9, offers a potent tool for visualizing and verifying clustering results. It allows controllers and analysts to rapidly comprehend and assess situations, as well as facilitate comparisons of airport conditions during events such as thunderstorms. This research also lays the groundwork for discovering more efficient representation methods that can potentially help transfer learning and expedite reinforcement learning algorithms in the future.

FUTURE WORKS

For the Random Impulses Models, enhancements such as incorporating feedback effects could be considered to prevent the lateral component from diverging to infinity. Additional applications stemming from these models could also be explored, and it is anticipated that they may find applicability in areas such as antenna beamforming and radar tracking.

In the realm of Anomaly Detection, the established framework could be expanded to accommodate multivariate time series and even extend to non-continuous variables. Lastly, the chapter on Multi-Dimensional Scaling (MDS) can be further investigated to develop compact representations that could facilitate the learning of air traffic situations at an airport, thereby helping transfer learning and expediting the training process for reinforcement learning.

Publications

JOURNAL PAPERS

1. D. T. Hoang*, V. Martinez, P. Maréchal, D. Delahaye, "Exploring the Random Impulses Vehicle Trajectory Model for Dimensionality Reduction and Motion Extraction from Aerial Videos," IEEE Transactions on Intelligent Transportation Systems, 2023, **in submission**.
2. D. T. Hoang*, V. Martinez, P. Maréchal, D. Delahaye, "Probabilistic Methods for Real-time Unsupervised Anomalous Trajectory Detection," IEEE Transactions on Intelligent Transporation Systems, 2021, **under revision**.

CONFERENCE PAPERS

1. D. T. Hoang*, B. Wang, Y. Li, D. Delahaye, P. Wei, "Anchored Multidimensional Scaling: Hidden Gems in Trajectory Clustering and Traffic Comparison in Similar Thunderstorm Developments," **in preparation**.
2. D. T. Hoang*, D. Delahaye, "Online Detection and Localization of Anomalous Aircraft Trajectories during Landing: A Bayesian Hypothesis Testing Approach," **in preparation**.
3. D. T. Hoang*, V. Martinez and D. Delahaye, "Dual Dirichlet Processes for Unsupervised Segmentation of Functional Data with Application to Lane-changing Behavior Characterization," **in preparation**.
4. D. T. Hoang*, V. Martinez and D. Delahaye, "Spherical Codec for V2X Cooperative Awareness Trajectory Compression: A Preliminary Study," 97th IEEE Vehicular Technology Conference (VTC), 2023.
5. D. T. Hoang*, V. Martinez and D. Delahaye, "Recognition of Outlying Driving Behaviors: A Data-Driven Perspective with Applications to V2X Collective Perception," 2021 IEEE Vehicular Networking Conference (VNC), 2021, pp. 52-59, doi: 10.1109/VNC52810.2021.9644627.

PATENTS

1. D. T. Hoang, D. Delahaye, V. Martinez, ”Optimization of message generation frequency and compression of V2X data,” European Patent No EP2230-6821.4, **pending**.

TECHNICAL POSTERS

1. D. T. Hoang, H. Yang, L. Lapasset, D. Delahaye, ”StratoEye: An Open Platform for Language Model Integration in Air Traffic Surveillance and Optimization,” 2023 SESAR Innovation Days (SID).

References

[1] CAR 2 CAR Communication Consortium. C2C PowerPoint Toolkit.

[2] ETSI. Intelligent Transport System (ITS); Vehicular Communications; Basic Set of Applications; Collective Perception Service; Release 2.

[3] ETSI. Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Analysis of the Collective Perception Service (CPS); Release 2.

[4] American Automobile Association. Think You're In Your Car More? You're Right. Americans Spend 70 Billion Hours Behind the Wheel.

[5] V. Mavrin, K. Magdin, V. Shepelev, and I. Danilov, "Reduction of environmental impact from road transport using analysis and simulation methods," *Transportation Research Procedia*, vol. 50, pp. 451–457, 2020.

[6] European Commission (EC). Sustainable and Smart Mobility Strategy – putting European transport on track for the future.

[7] SAE, *SAE Levels of Driving Automation Refined for Clarity and International Audience*.

[8] McKinsey&Company. Advanced driver-assistance systems: Challenges and opportunities ahead.

[9] B. Mahaur, N. Singh, and K. Mishra, "Road object detection: a comparative study of deep learning-based algorithms," *Multimedia Tools and Applications*, vol. 81, no. 10, pp. 14 247–14 282, 2022.

[10] V. Vukadinovic, K. Bakowski, P. Marsch, I. D. Garcia, H. Xu, M. Sybis, P. Sroka, K. Wesolowski, D. Lister, and I. Thibault, "3GPP C-V2X and IEEE 802.11 p for Vehicle-to-Vehicle communications in highway platooning scenarios," *Ad Hoc Networks*, vol. 74, pp. 17–29, 2018.

[11] European Telecommunications Standards Institute (ETSI), "ETSI TS 103 324 V0.0.35 (2022-06), Intelligent Transport System (ITS); Vehicular Communications; Basic Set of Applications; Specification of the Collective Perception Service; Release 2," 2022.

[12] European Commission. Study on the Deployment of C-ITS in Europe: Final Report; Framework Contract on Impact Assessment and Evaluation Studies in the Field of Transport MOVE/A3/119-2013-Lot No 5 "Horizontal". [Online]. Available: https://transport.ec.europa.eu/system/files/2016-10/2016-c-its-deployment-study-final-report.pdf

[13] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE international conference on image processing (ICIP)*. IEEE, 2016, pp. 3464–3468.

[14] J. Liu, X. Mao, Y. Fang, D. Zhu, and M. Q.-H. Meng, "A survey on deep-learning approaches for vehicle trajectory prediction in autonomous driving," in *2021 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2021, pp. 978–985.

[15] S. Boonmee and P. Tangamchit, "Portable reckless driving detection system," in *2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, vol. 1. IEEE, 2009, pp. 412–415.

[16] T. H. Dinh, V. Martinez, and D. Delahaye, "Recognition of Outlying Driving Behaviors: A Data-Driven Perspective with Applications to V2X Collective Perception," in *2021 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2021, pp. 52–59.

[17] M. Brännström, E. Coelingh, and J. Sjöberg, "Model-based threat assessment for avoiding arbitrary vehicle collisions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 658–669, 2010.

[18] C.-F. Lin, A. G. Ulsoy, and D. J. LeBlanc, "Vehicle dynamics and external disturbance estimation for vehicle path prediction," *IEEE Transactions on Control Systems Technology*, vol. 8, no. 3, pp. 508–518, 2000.

[19] J. Huang and H.-S. Tan, "Vehicle future trajectory prediction with a DGPS/INS-based positioning system," in *2006 American Control Conference*. IEEE, 2006, pp. 6–pp.

[20] J. J. Olstam and A. Tapani, *Comparison of Car-following models*. Swedish National Road and Transport Research Institute Linköping, Sweden, 2004, vol. 960.

[21] S. Moridpour, M. Sarvi, and G. Rose, "Lane changing models: a critical review," *Transportation letters*, vol. 2, no. 3, pp. 157–173, 2010.

[22] B. I. Sighencea, R. I. Stanciu, and C. D. Căleanu, "A review of deep learning-based methods for pedestrian trajectory prediction," *Sensors*, vol. 21, no. 22, p. 7543, 2021.

[23] S. Capobianco, L. M. Millefiori, N. Forti, P. Braca, and P. Willett, "Deep learning methods for vessel trajectory prediction based on recurrent neural networks," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 6, pp. 4329–4346, 2021.

[24] U.S. Department of Transportation. Next Generation Simulation (NGSIM) Vehicle Trajectories and Supporting Data. [Online]. Available: https://catalog.data.gov/dataset/next-generation-simulation-ngsim-vehicle-trajectories-and-supporting-dataurl

[25] Hanshin Expressway Co. Vehicle trajectory data. [Online]. Available: https://zen-traffic-data.net/english/outline/dataset.html

[26] Y. He, B. Cao, and C.-Y. Chan, "WUT-NGSIM: A High-Precision and Trustworthy Vehicle Trajectory Dataset," 2022. [Online]. Available: https://dx.doi.org/10.21227/hmsb-ka76

[27] E. Martínez-Vera, P. Bañuelos-Sánchez, and G. Etcheverry, "Lane changing model from NGSIM dataset," in *Pattern Recognition: 14th Mexican Conference, MCPR 2022, Ciudad Juárez, Mexico, June 22–25, 2022, Proceedings*. Springer, 2022, pp. 25–34.

[28] X. Zheng, P. Yang, D. Duan, X. Cheng, and L. Yang, "Real-time driving style classification based on short-term observations," *IET Communications*, vol. 16, no. 12, pp. 1393–1402, 2022.

[29] L. Hou, S. E. Li, B. Yang, Z. Wang, and K. Nakano, "Integrated Graphical Representation of Highway Scenarios to Improve Trajectory Prediction of Surrounding Vehicles," *IEEE Transactions on Intelligent Vehicles*, 2022.

[30] G. Jarry, D. Delahaye, F. Nicol, and E. Feron, "Aircraft atypical approach detection using functional principal component analysis," *Journal of Air Transport Management*, vol. 84, p. 101787, 2020.

[31] M. Gariel, A. N. Srivastava, and E. Feron, "Trajectory clustering and an application to airspace monitoring," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1511–1524, 2011.

[32] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH journal*, vol. 1, no. 1, pp. 1–14, 2014.

[33] A. Eidehall and L. Petersson, "Statistical threat assessment for general road scenes using Monte Carlo sampling," *IEEE Transactions on intelligent transportation systems*, vol. 9, no. 1, pp. 137–147, 2008.

[34] G. Genta, *Motor vehicle dynamics: modeling and simulation*. World Scientific, 1997, vol. 43.

[35] G. Rill, *Road vehicle dynamics: fundamentals and modeling*. Crc Press, 2011.

[36] G. Rill and A. A. Castro, *Road vehicle dynamics: fundamentals and modeling with MATLAB®*. CRC Press, 2020.

[37] N. Kaempchen, K. Weiss, M. Schaefer, and K. C. Dietmayer, "IMM object tracking for high dynamic driving maneuvers," in *IEEE Intelligent Vehicles Symposium, 2004*. IEEE, 2004, pp. 825–830.

[38] P. Lytrivis, G. Thomaidis, and A. Amditis, "Cooperative path prediction in vehicular environments," in *2008 11th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2008, pp. 803–808.

[39] A. Barth and U. Franke, "Where will the oncoming vehicle be the next second?" in *2008 IEEE Intelligent Vehicles Symposium*. IEEE, 2008, pp. 1068–1073.

[40] A. Broadhurst, S. Baker, and T. Kanade, "Monte Carlo road safety reasoning," in *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.* IEEE, 2005, pp. 319–324.

[41] N. Kaempchen, B. Schiele, and K. Dietmayer, "Situation assessment of an autonomous emergency brake for arbitrary vehicle-to-vehicle collision scenarios," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 4, pp. 678–687, 2009.

[42] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank, "A system for learning statistical motion patterns," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 9, pp. 1450–1464, 2006.

[43] S. Atev, G. Miller, and N. P. Papanikolopoulos, "Clustering of vehicle trajectories," *IEEE transactions on intelligent transportation systems*, vol. 11, no. 3, pp. 647–657, 2010.

[44] T. Górecki and M. Łuczak, "Non-isometric transforms in time series classification using DTW," *Knowledge-based systems*, vol. 61, pp. 98–108, 2014.

[45] J. Joseph, F. Doshi-Velez, A. S. Huang, and N. Roy, "A Bayesian nonparametric approach to modeling motion patterns," *Autonomous Robots*, vol. 31, no. 4, pp. 383–400, 2011.

[46] Q. Tran and J. Firl, "Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression," in *2014 ieee intelligent vehicles symposium proceedings*. IEEE, 2014, pp. 918–923.

[47] Y. Guo, V. V. Kalidindi, M. Arief, W. Wang, J. Zhu, H. Peng, and D. Zhao, "Modeling multi-vehicle interaction scenarios using gaussian random field," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3974–3980.

[48] H. M. Mandalia and M. D. D. Salvucci, "Using support vector machines for lane-change detection," in *Proceedings of the human factors and ergonomics society annual meeting*, vol. 49, no. 22. SAGE Publications Sage CA: Los Angeles, CA, 2005, pp. 1965–1969.

[49] T. Streubel and K. H. Hoffmann, "Prediction of driver intended path at intersections," in *2014 ieee intelligent vehicles symposium proceedings*. IEEE, 2014, pp. 134–139.

[50] S. Yalamanchi, T.-K. Huang, G. C. Haynes, and N. Djuric, "Long-term prediction of vehicle behavior using short-term uncertainty-aware trajectories and high-definition maps," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.

[51] P. Kumar, M. Perrollaz, S. Lefevre, and C. Laugier, "Learning-based approach for online lane change intention prediction," in *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2013, pp. 797–802.

[52] W. Ding and S. Shen, "Online vehicle trajectory prediction using policy anticipation network and optimization-based context reasoning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 9610–9616.

[53] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi, "Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1672–1678.

[54] A. Kawasaki and A. Seki, "Multimodal trajectory predictions for autonomous driving without a detailed prior map," in *Proceedings of the*

IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 3723–3732.

[55] H. N. Koutsopoulos and H. Farah, "Latent class model for car following behavior," *Transportation research part B: methodological*, vol. 46, no. 5, pp. 563–578, 2012.

[56] P. G. Gipps, "A behavioural car-following model for computer simulation," *Transportation Research Part B: Methodological*, vol. 15, no. 2, pp. 105–111, 1981.

[57] K. I. Ahmed, "Modeling drivers' acceleration and lane changing behavior," Ph.D. dissertation, Massachusetts Institute of Technology, 1999.

[58] R. Chandra, U. Bhattacharya, A. Bera, and D. Manocha, "Traphic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8483–8492.

[59] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[60] K. Messaoud, I. Yahiaoui, A. Verroust-Blondet, and F. Nashashibi, "Attention based vehicle trajectory prediction," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 1, pp. 175–185, 2020.

[61] H. Jeon, J. Choi, and D. Kum, "Scale-net: Scalable vehicle trajectory prediction network under random number of interacting vehicles via edge-enhanced graph convolutional neural network," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 2095–2102.

[62] L. Gong and Q. Cheng, "Exploiting edge features for graph neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 9211–9219.

[63] L. Sun, W. Zhan, and M. Tomizuka, "Probabilistic prediction of interactive driving behavior via hierarchical inverse reinforcement learning," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 2111–2117.

[64] D. Xu, Z. Ding, X. He, H. Zhao, M. Moze, F. Aioun, and F. Guillemard, "Learning from naturalistic driving data for human-like autonomous

highway driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 12, pp. 7341–7354, 2020.

[65] Y. Huang, J. Du, Z. Yang, Z. Zhou, L. Zhang, and H. Chen, "A Survey on Trajectory-Prediction Methods for Autonomous Driving," *IEEE Transactions on Intelligent Vehicles*, 2022.

[66] R. G. Jaimez and M. J. V. Bonnet, "On the Karhunen-Loeve expansion for transformed processes," *Trabajos de estadística*, vol. 2, no. 2, pp. 81–90, 1987.

[67] R. B. Ash and M. F. Gardner, *Topics in Stochastic Processes: Probability and Mathematical Statistics: A Series of Monographs and Textbooks*. Academic press, 2014, vol. 27.

[68] M. Unser, "On the approximation of the discrete Karhunen-Loeve transform for stationary processes," *Signal Processing*, vol. 7, no. 3, pp. 231–249, 1984.

[69] S. Ammoun and F. Nashashibi, "Real time trajectory prediction for collision risk estimation between vehicles," in *2009 IEEE 5th International Conference on Intelligent Computer Communication and Processing*. IEEE, 2009, pp. 417–422.

[70] A. Polychronopoulos, M. Tsogas, A. J. Amditis, and L. Andreone, "Sensor fusion for predicting vehicles' path for collision avoidance systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 3, pp. 549–562, 2007.

[71] A. Kuczura, "Piecewise Markov processes," *SIAM Journal on Applied Mathematics*, vol. 24, no. 2, pp. 169–181, 1973.

[72] P. Deheuvels and G. Martynov, "Karhunen-Loeve expansions for weighted Wiener processes and Brownian bridges via Bessel functions," in *High Dimensional Probability III*. Springer, 2003, pp. 57–93.

[73] R. Durrett, *Probability: theory and examples*. Cambridge university press, 2019, vol. 49.

[74] H. L. Shang, "A survey of functional principal component analysis," *AStA Advances in Statistical Analysis*, vol. 98, no. 2, pp. 121–142, 2014.

[75] J. Brown, Jr, "Mean square truncation error in series expansions of random functions," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 1, pp. 28–32, 1960.

[76] T. H. Dinh, V. Martinez, and D. Delahaye, "Spherical Codec for V2X Cooperative Awareness Trajectory Compression: A Preliminary Study," in *2023 IEEE Vehicular Technology Conference (VTC)*. IEEE, 2023.

[77] G. Jarry, D. Delahaye, and E. Feron, "Trajectory approach analysis: A post-operational aircraft approach analysis tool," in *SID 2019, 9th SESAR Innovation Days*, 2019.

[78] C. Englund, E. E. Aksoy, F. Alonso-Fernandez, M. D. Cooney, S. Pashami, and B. Åstrand, "AI perspectives in Smart Cities and Communities to enable road vehicle automation and smart traffic control," *Smart Cities*, vol. 4, no. 2, pp. 783–802, 2021.

[79] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, "A review on outlier/anomaly detection in time series data," *ACM Computing Surveys (CSUR)*, vol. 54, no. 3, pp. 1–33, 2021.

[80] Y. Li, W. Liu, and Q. Huang, "Traffic anomaly detection based on image descriptor in videos," *Multimedia tools and applications*, vol. 75, no. 5, pp. 2487–2505, 2016.

[81] R. Hinami, T. Mei, and S. Satoh, "Joint detection and recounting of abnormal events by learning deep generic knowledge," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3619–3627.

[82] S. Lee, H. G. Kim, and Y. M. Ro, "STAN: Spatio-temporal adversarial networks for abnormal event detection," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2018, pp. 1323–1327.

[83] C. Lu, J. Shi, and J. Jia, "Abnormal event detection at 150 fps in matlab," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 2720–2727.

[84] H. Y. Ngan, N. H. Yung, and A. G. Yeh, "Outlier detection in traffic data based on the Dirichlet process mixture model," *IET Intelligent Transport Systems*, vol. 9, no. 7, pp. 773–781, 2015.

[85] D. Xu, E. Ricci, Y. Yan, J. Song, and N. Sebe, "Learning deep representation of appearance and motion for anomalous event detection. arXiv," *arXiv preprint arXiv:1510.01553*, 2015.

[86] Z. Rahman, A. M. Ami, and M. A. Ullah, "A real-time wrong-way vehicle detection based on YOLO and centroid tracking," in *2020 IEEE Region 10 Symposium (TENSYMP)*. IEEE, 2020, pp. 916–920.

[87] S. Simpson, D. Bruggeman *et al.*, "Detection and warning systems for wrong-way driving," Arizona. Dept. of Transportation, Tech. Rep., 2015.

[88] M. A. Khan and S. F. Khan, "IoT based framework for Vehicle Over-speed detection," in *2018 1st International Conference on Computer Applications & Information Security (ICCAIS)*. IEEE, 2018, pp. 1–4.

[89] Y. Marumo, H. Tsunashima, T. Kojima, and Y. Hasegawa, "Analysis of braking behavior of train drivers to detect unusual driving," *Journal of Mechanical systems for Transportation and Logistics*, vol. 3, no. 1, pp. 338–348, 2010.

[90] Z. Zhang, X. Mei, and B. Xiao, "Abnormal event detection via compact low-rank sparse learning," *IEEE Intelligent Systems*, vol. 31, no. 2, pp. 29–36, 2015.

[91] Y. Zhang, H. Lu, L. Zhang, X. Ruan, and S. Sakai, "Video anomaly detection based on locality sensitive hashing filters," *Pattern Recognition*, vol. 59, pp. 302–311, 2016.

[92] R. Song, "Driver intention prediction using model-added Bayesian network," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 235, no. 5, pp. 1236–1244, 2021.

[93] J. Colyar and J. Halkias, "US highway 101 dataset. Federal Highway Administration Research and Technology Fact Sheet. Publication number: FHWA-HRT-07-030," *Tech. Rep.*, 2007.

[94] R. Chandra, U. Bhattacharya, T. Mittal, A. Bera, and D. Manocha, "Cmetric: A driving behavior measure using centrality functions," *arXiv preprint arXiv:2003.04424*, 2020.

[95] N. Li, D. W. Oyler, M. Zhang, Y. Yildiz, I. Kolmanovsky, and A. R. Girard, "Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems," *IEEE Transactions on control systems technology*, vol. 26, no. 5, pp. 1782–1797, 2017.

[96] V. Ramanishka, Y.-T. Chen, T. Misu, and K. Saenko, "Toward driving scene understanding: A dataset for learning driver behavior and causal reasoning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7699–7707.

[97] E. Cheung, A. Bera, E. Kubin, K. Gray, and D. Manocha, "Identifying driver behaviors using trajectory features for vehicle navigation," in *2018*

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 3445–3452.

[98] A. Geiger, D. Liu, S. Alnegheimish, A. Cuesta-Infante, and K. Veeramachaneni, "TadGAN: Time series anomaly detection using generative adversarial networks," in *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020, pp. 33–43.

[99] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, "Imitating driver behavior with generative adversarial networks," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 204–211.

[100] B. Zhao, L. Fei-Fei, and E. P. Xing, "Online detection of unusual events in videos via dynamic sparse coding," in *CVPR 2011*. IEEE, 2011, pp. 3313–3320.

[101] V. Nguyen, S. Gupta, S. Rana, C. Li, and S. Venkatesh, "A Bayesian nonparametric approach for multi-label classification," in *Asian conference on machine learning*. PMLR, 2016, pp. 254–269.

[102] W. Yang, Y. Gao, and L. Cao, "TRASMIL: A local anomaly detection framework based on trajectory segmentation and multi-instance learning," *Computer Vision and Image Understanding*, vol. 117, no. 10, pp. 1273–1286, 2013.

[103] M. H. Sharif and C. Djeraba, "An entropy approach for abnormal activities detection in video streams," *Pattern recognition*, vol. 45, no. 7, pp. 2543–2561, 2012.

[104] X. Cui, Q. Liu, M. Gao, and D. N. Metaxas, "Abnormal detection using interaction energy potentials," in *CVPR 2011*. IEEE, 2011, pp. 3161–3167.

[105] C. Li, Z. Han, Q. Ye, and J. Jiao, "Visual abnormal behavior detection based on trajectory sparse reconstruction analysis," *Neurocomputing*, vol. 119, pp. 94–100, 2013.

[106] S. W. Liu, H. Y. Ngan, M. K. Ng, and S. J. Simske, "Accumulated relative density outlier detection for large scale traffic data," *Electronic Imaging*, vol. 2018, no. 9, pp. 239–1, 2018.

[107] X. Jin, Y. Zhang, L. Li, and J. Hu, "Robust PCA-based abnormal traffic flow pattern isolation and loop detector fault detection," *Tsinghua Science & Technology*, vol. 13, no. 6, pp. 829–835, 2008.

[108] J. Owens and A. Hunter, "Application of the self-organising map to trajectory classification," in *Proceedings Third IEEE International Workshop on Visual Surveillance*. IEEE, 2000, pp. 77–83.

[109] X. Wang, K. Tieu, and E. Grimson, "Learning semantic scene models by trajectory analysis," in *European conference on computer vision*. Springer, 2006, pp. 110–123.

[110] E. J. Keogh and M. J. Pazzani, "Scaling up dynamic time warping for data-mining applications," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2000, pp. 285–289.

[111] V. Alexiadis, "Video-Based Vehicle Trajectory Data Collection," in *Transportation Research Board 86th Annual Meeting*. Citeseer, 2006.

[112] CAR 2 CAR Consortium. Survey on ITS-G5 CAM statistics. [Online]. Available: https://www.car-2-car.org/

[113] J. Ramsay and B. Silverman, "Principal components analysis for functional data," *Functional data analysis*, pp. 147–172, 2005.

[114] Y. Hong, "scikit-fda: Principal Component Analysis for Functional Data," B.S. thesis, 2020.

[115] A. Tartakovsky, I. Nikiforov, and M. Basseville, *Sequential analysis: Hypothesis testing and changepoint detection*. CRC Press, 2014.

[116] J. Harding, G. Powell, R. Yoon, J. Fikentscher, C. Doyle, D. Sade, M. Lukuc, J. Simons, J. Wang *et al.*, "Vehicle-to-vehicle communications: readiness of V2V technology for application." United States. National Highway Traffic Safety Administration, Tech. Rep., 2014.

[117] ETSI, "Intelligent Transport Systems (ITS); European profile standard for the physical and medium access control layer of Intelligent Transport Systems operating in the 5 GHz frequency band (ETSI ES 202 663)."

[118] ——, "Intelligent Transport Systems (ITS); Radiocommunications equipment operating in the 5 855 MHz to 5 925 MHz frequency band; Harmonised Standard covering the essential requirements of article 3.2 of Directive 2014/53/EU (ETSI EN 302 571)."

[119] ——. Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service.

[120] TELECOMMUNICATION STANDARDIZATION SECTOR OF ITU, "SERIES X: DATA NETWORKS AND OPEN SYSTEM COMMUNICATIONS, OSI networking and system aspects – Abstract Syntax Notation One (ASN.1), Information technology – ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)."

[121] CAR 2 CAR Communication Consortium. Survey on ITS-G5 CAM statistics.

[122] Continental. Collective Perception in V2X, Communication based on Crash Data in Japan/Germany/US.

[123] ETSI, "Intelligent Transport Systems (ITS); Decentralized Congestion Control Mechanisms for Intelligent Transport Systems operating in the 5 GHz range; Access layer part (ETSI TS 102 687)."

[124] A. Balador, E. Cinque, M. Pratesi, F. Valentini, C. Bai, A. A. Gómez, and M. Mohammadi, "Survey on decentralized congestion control methods for vehicular communication," *Vehicular Communications*, vol. 33, p. 100394, 2022.

[125] F. Marzouk, R. Zagrouba, A. Laouiti, and P. Muhlethaler, "An empirical study of Unfairness and Oscillation in ETSI DCC," *arXiv preprint arXiv:1510.01125*, 2015.

[126] N. Lyamin, A. Vinel, D. Smely, and B. Bellalta, "Etsi dcc: Decentralized congestion control in c-its," *IEEE Communications Magazine*, vol. 56, no. 12, pp. 112–118, 2018.

[127] I. Khan and J. Härri, "Integration challenges of facilities-layer DCC for heterogeneous V2X services," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1131–1136.

[128] S. Kuk, S. Yang, Y. Park, and H. Kim, "On the connectivity problem of ETSI DCC algorithm," *arXiv preprint arXiv:1703.05174*, 2017.

[129] G. Bansal, J. B. Kenney, and C. E. Rohrs, "LIMERIC: A linear adaptive message rate algorithm for DSRC congestion control," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 9, pp. 4182–4197, 2013.

[130] B. Cheng, A. Rostami, M. Gruteser, J. B. Kenney, G. Bansal, and K. Sjoberg, "Performance evaluation of a mixed vehicular network with CAM-DCC and LIMERIC vehicles," in *2015 IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, 2015, pp. 1–6.

[131] T. Tielert, D. Jiang, Q. Chen, L. Delgrossi, and H. Hartenstein, "Design methodology and evaluation of rate adaptation based congestion control for vehicle safety communications," in *2011 IEEE vehicular networking conference (VNC)*. IEEE, 2011, pp. 116–123.

[132] T. Lorenzen and M. T. Garrosi, "Achieving global fairness at urban intersections using cooperative DSRC congestion control," in *Proceedings of the First ACM International Workshop on Smart, Autonomous, and Connected Vehicular Systems and Services*, 2016, pp. 52–59.

[133] T. Lorenzen, "SWeRC: Self-weighted semi-cooperative DSRC congestion control based on LIMERIC," in *2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*. IEEE, 2017, pp. 1–7.

[134] C. B. Math, H. Li, S. H. de Groot, and I. Niemegeers, "Fair decentralized data-rate congestion control for V2V communications," in *2017 24th International Conference on Telecommunications (ICT)*. IEEE, 2017, pp. 1–7.

[135] M. Torrent-Moreno, J. Mittag, P. Santi, and H. Hartenstein, "Vehicle-to-vehicle communication: Fair transmit power control for safety-critical information," *IEEE transactions on vehicular technology*, vol. 58, no. 7, pp. 3684–3703, 2009.

[136] F. Lyu, N. Cheng, H. Zhou, W. Xu, W. Shi, J. Chen, and M. Li, "DBCC: Leveraging link perception for distributed beacon congestion control in VANETs," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4237–4249, 2018.

[137] L. Liang, H. Ye, and G. Y. Li, "Toward intelligent vehicular networks: A machine learning framework," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 124–135, 2018.

[138] U. Jayasankar, V. Thirumal, and D. Ponnurangam, "A survey on data compression techniques: From the perspective of data quality, coding schemes, data type and applications," *Journal of King Saud University-Computer and Information Sciences*, vol. 33, no. 2, pp. 119–140, 2021.

[139] A. Moffat, "Huffman coding," *ACM Computing Surveys (CSUR)*, vol. 52, no. 4, pp. 1–35, 2019.

[140] J. Rissanen and G. G. Langdon, "Arithmetic coding," *IBM Journal of research and development*, vol. 23, no. 2, pp. 149–162, 1979.

[141] L. Gasieniec, M. Karpinski, W. Plandowski, and W. Rytter, "Efficient algorithms for Lempel-Ziv encoding," in *Algorithm Theory—SWAT'96: 5th Scandinavian Workshop on Algorithm Theory Reykjavík, Iceland, July 3–5, 1996 Proceedings 5*. Springer, 1996, pp. 392–403.

[142] G. Manzini, "An analysis of the Burrows—Wheeler transform," *Journal of the ACM (JACM)*, vol. 48, no. 3, pp. 407–430, 2001.

[143] T. Sheltami, M. Musaddiq, and E. Shakshuki, "Data compression techniques in wireless sensor networks," *Future Generation Computer Systems*, vol. 64, pp. 151–162, 2016.

[144] O. M. Nielsen, "Wavelets in scientific computing," Ph.D. dissertation, Technical University of Denmark, 1998.

[145] R. Kumar, "Statistical measures and analysis in electrocardiogram (ECG) signal processing," in *Modelling and Analysis of Active Biopotential Signals in Healthcare, Volume 1*. IOP Publishing Bristol, UK, 2020, pp. 14–1.

[146] R. Bellman, "On the approximation of curves by line segments using dynamic programming," *Communications of the ACM*, vol. 4, no. 6, p. 284, 1961.

[147] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica: the international journal for geographic information and geovisualization*, vol. 10, no. 2, pp. 112–122, 1973.

[148] N. Meratnia and A. Rolf, "Spatiotemporal compression techniques for moving point objects," in *International Conference on Extending Database Technology*. Springer, 2004, pp. 765–782.

[149] G. Trajcevski, H. Cao, P. Scheuermanny, O. Wolfsonz, and D. Vaccaro, "On-line data reduction and the quality of history in moving objects databases," in *Proceedings of the 5th ACM international workshop on Data engineering for wireless and mobile access*, 2006, pp. 19–26.

[150] J. Muckell, J.-H. Hwang, V. Patil, C. T. Lawson, F. Ping, and S. Ravi, "SQUISH: an online approach for GPS trajectory compression," in *Proceedings of the 2nd international conference on computing for geospatial research & applications*, 2011, pp. 1–8.

[151] J. Muckell, P. W. Olsen, J.-H. Hwang, C. T. Lawson, and S. Ravi, "Compression of trajectory data: a comprehensive evaluation and new approach," *GeoInformatica*, vol. 18, no. 3, pp. 435–460, 2014.

[152] P. Cudre-Mauroux, E. Wu, and S. Madden, "Trajstore: An adaptive storage system for very large trajectory data sets," in *2010 IEEE 26th International Conference on Data Engineering (ICDE 2010)*. IEEE, 2010, pp. 109–120.

[153] A. Nibali and Z. He, "Trajic: An effective compression system for trajectory data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 11, pp. 3138–3151, 2015.

[154] R. H. Bartels, J. C. Beatty, and B. A. Barsky, *An introduction to splines for use in computer graphics and geometric modeling*. Morgan Kaufmann, 1995.

[155] M. Sepulcre, J. Gozalvez, G. Thandavarayan, B. Coll-Perales, J. Schindler, and M. Rondinone, "On the Potential of V2X Message Compression for Vehicular Networks," *IEEE Access*, vol. 8, 2020.

[156] C. De Boor, *A practical guide to splines*. Springer-Verlag New York, 1978, vol. 27.

[157] R. H. Bartels, J. C. Beatty, and B. A. Barsky, "Hermite and cubic spline interpolation," *An Introduction to Splines for Use in Computer Graphics and Geometric Modelling*, pp. 9–17, 1998.

[158] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part V. Multiple-model methods," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1255–1321, 2005.

[159] T. H. Dinh, V. Martinez, and D. Delahaye, "Recognition of Outlying Driving Behaviors: A Data-Driven Perspective with Applications to V2X Collective Perception," in *2021 IEEE Vehicular Networking Conference (VNC)*, 2021, pp. 52–59.

[160] ETSI, "EN 302 637 - V1.3.2 - Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service," 2014.

[161] R. Riebl, H.-J. Günther, C. Facchi, and L. Wolf, "Artery: Extending veins for VANET applications," in *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, 2015, pp. 450–456.

[162] D. Krajzewicz, "Traffic simulation with SUMO–simulation of urban mobility," in *Fundamentals of traffic simulation*. Springer, 2010, pp. 269–293.

[163] M.-S. Alouini and A. J. Goldsmith, "Adaptive modulation over Nakagami fading channels," *Wireless Personal Communications*, vol. 13, pp. 119–143, 2000.

[164] ETSI, "TR 103 562 - V2.1.1 - Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Analysis of the Collective Perception Service (CPS); Release 2," 2019.

[165] I. Karatzas and S. Shreve, *Brownian motion and stochastic calculus*. Springer Science & Business Media, 1991, vol. 113.

[166] V. Kurtc and M. Treiber, "Calibrating the local and platoon dynamics of car-following models on the reconstructed NGSIM data," in *Traffic and Granular flow'15*. Springer, 2016, pp. 515–522.

[167] L. Leclercq, N. Chiabaut, J. Laval, and C. Buisson, "Relaxation phenomenon after lane changing: Experimental validation with NGSIM data set," *Transportation Research Record*, vol. 1999, no. 1, pp. 79–85, 2007.

[168] A. Yilmaz, O. Javed, and M. Shah, "Object tracking," *ACM Computing Surveys*, vol. 38, no. 4, p. 13, 2006.

[169] X. Chen, Z. Li, Y. Yang, L. Qi, and R. Ke, "High-resolution vehicle trajectory extraction and denoising from aerial videos," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 5, pp. 3190–3202, 2020.

[170] C. L. Azevedo, J. L. Cardoso, M. Ben-Akiva, J. P. Costeira, and M. Marques, "Automatic vehicle trajectory extraction by aerial remote sensing," *Procedia-Social and Behavioral Sciences*, vol. 111, pp. 849–858, 2014.

[171] R. Ke, Z. Li, J. Tang, Z. Pan, and Y. Wang, "Real-time traffic flow parameter estimation from UAV video based on ensemble classifier and optical flow," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 54–64, 2018.

[172] D. Zhao and X. Li, "Real-world trajectory extraction from aerial videos-a comprehensive and effective solution," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 2854–2859.

[173] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems," in *2018 21st international conference on intelligent transportation systems (ITSC)*. IEEE, 2018, pp. 2118–2125.

[174] A. G. Wills, J. Hendriks, C. Renton, and B. Ninness, "A Bayesian filtering algorithm for Gaussian mixture models," *arXiv preprint arXiv:1705.05495*, 2017.

[175] M. P. Balenzuela, J. Dahlin, N. Bartlett, A. G. Wills, C. Renton, and B. Ninness, "Accurate Gaussian mixture model smoothing using a two-filter approach," in *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 694–699.

[176] G. Kitagawa, "The two-filter formula for smoothing and an implementation of the Gaussian-sum smoother," *Annals of the Institute of Statistical Mathematics*, vol. 46, pp. 605–623, 1994.

[177] G. Welch, G. Bishop *et al.*, "An introduction to the Kalman filter," 1995.

[178] Ultralytics. Ultralytics YOLOv8. [Online]. Available: https://docs.ultralytics.com/

[179] M. B. Shams, H. Budman, and T. Duever, "Fault detection, identification and diagnosis using CUSUM based PCA," *Chemical Engineering Science*, vol. 66, no. 20, pp. 4488–4498, 2011.

[180] G. Manogaran and D. Lopez, "Spatial cumulative sum algorithm with big data analytics for climate change detection," *Computers & Electrical Engineering*, vol. 65, pp. 207–221, 2018.

[181] S. Thies and P. Molnár, "Bayesian change point analysis of Bitcoin returns," *Finance Research Letters*, vol. 27, pp. 223–227, 2018.

[182] R. P. Adams and D. J. MacKay, "Bayesian online changepoint detection," *arXiv preprint arXiv:0710.3742*, 2007.

[183] M. Lavielle, "Detection of multiple changes in a sequence of dependent variables," *Stochastic Processes and their applications*, vol. 83, no. 1, pp. 79–102, 1999.

[184] S. I. Ko, T. T. Chong, and P. Ghosh, "Dirichlet process hidden Markov multiple change-point model," *Bayesian Analysis*, vol. 10, no. 2, pp. 275–296, 2015.

[185] D. Görür and C. Edward Rasmussen, "Dirichlet process gaussian mixture models: Choice of the base distribution," *Journal of Computer Science and Technology*, vol. 25, no. 4, pp. 653–664, 2010.

[186] E. B. Fox, E. B. Sudderth, M. I. Jordan, and A. S. Willsky, "A sticky HDP-HMM with application to speaker diarization," *The Annals of Applied Statistics*, pp. 1020–1056, 2011.

[187] Federal Highway Administration Research and Technology. US Highway 101 Dataset. [Online]. Available: https://www.fhwa.dot.gov/publications/research/operations/07030/

[188] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Processing*, vol. 167, p. 107299, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0165168419303494

[189] D. Barry and J. A. Hartigan, "Product partition models for change point problems," *The annals of Statistics*, pp. 260–279, 1992.

[190] J. Chen and A. K. Gupta, "Parametric statistical change point analysis: with applications to genetics, medicine, and finance," 2012.

[191] A. Pepelyshev and A. S. Polunchenko, "Real-time financial surveillance via quickest change-point detection methods," *arXiv preprint arXiv:1509.01570*, 2015.

[192] E. Ruggieri, "A Bayesian approach to detecting change points in climatic records," *International Journal of Climatology*, vol. 33, no. 2, pp. 520–528, 2013.

[193] E. Ruggieri and M. Antonellis, "An exact approach to Bayesian sequential change point detection," *Computational Statistics & Data Analysis*, vol. 97, pp. 71–86, 2016.

[194] J.-P. Vert and K. Bleakley, "Fast detection of multiple change-points shared by many signals using group LARS," *Advances in neural information processing systems*, vol. 23, 2010.

[195] J.-J. Jeon, J. H. Sung, and E.-S. Chung, "Abrupt change point detection of annual maximum precipitation using fused lasso," *Journal of Hydrology*, vol. 538, pp. 831–841, 2016.

[196] Y. Li, E. Schofield, and M. Gönen, "A tutorial on Dirichlet process mixture modeling," *Journal of mathematical psychology*, vol. 91, pp. 128–144, 2019.

[197] E. Di Lello, T. De Laet, and H. Bruyninckx, "Hierarchical dirichlet process hidden markov models for abnormality detection in robotic assembly," in

Neural Information Processing Systems, Date: 2012/12/03-2012/12/08, Location: Lake Tahoe, Nevada, 2012.

[198] T. Taniguchi, K. Hamahata, and N. Iwahashi, "Unsupervised segmentation of human motion data using a sticky hierarchical dirichlet process-hidden markov model and minimal description length-based chunking method for imitation learning," *Advanced Robotics*, vol. 25, no. 17, pp. 2143–2172, 2011.

[199] S. M. Kay, *Fundamentals of statistical signal processing: detection theory*. Prentice-Hall, Inc., 1993.

[200] J. Bai and P. Perron, "Computation and analysis of multiple structural change models," *Journal of applied econometrics*, vol. 18, no. 1, pp. 1–22, 2003.

[201] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian data analysis*. Chapman and Hall/CRC, 1995.

[202] ETSI. ETSI EN 302 637 V1.4.1 - Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service. [Online]. Available: https://www.etsi.org/

[203] W. Yang, J. M. Garcia-Rivera, M. A. Petty, G. J. Layne, K. Fenton, J. Cheng, R. Chen, W. Guo, Y. Weng, S. Liu *et al.*, "Quantifying Convective Weather Impacts to Airspace Capacity: Framework and Preliminary Results," *Journal of Air Transportation*, vol. 29, no. 1, pp. 42–55, 2021.

[204] X. He and P. Niyogi, "Locality preserving projections," *Advances in neural information processing systems*, vol. 16, 2003.

[205] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936.

[206] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[207] A. Mukherjee, S. R. Grabbe, and B. Sridhar, "Classification of days using weather impacted traffic in the national airspace system," in *2013 Aviation Technology, Integration, and Operations Conference*, 2013, p. 4403.

[208] S. Wolfe and J. Rios, "A method for using historical Ground Delay Programs to inform day-of-operations programs," in *AIAA Guidance, Navigation, and Control Conference*, 2011, p. 6528.

[209] Y. Wang and S. Grabbe, "Modeling weather impact on airport arrival miles-in-trail restrictions," in *SAE 2013 AeroTech Congress & Exhibition*, no. ARC-E-DAA-TN8460, 2013.

[210] S. R. Grabbe, B. Sridhar, and A. Mukherjee, "Clustering days with similar airport weather conditions," in *14th AIAA Aviation Technology, Integration, and Operations Conference*, 2014, p. 2712.

[211] S. Reitmann, S. Alam, and M. Schultz, "Advanced quantification of weather impact on air traffic management," in *ATM Seminar*, 2019, pp. 20–30.

[212] I. Borg and P. J. Groenen, *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.

[213] J. B. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrika*, vol. 29, no. 1, pp. 1–27, 1964.

[214] J. De Leeuw and P. Mair, "Multidimensional scaling using majorization: SMACOF in R," *Journal of statistical software*, vol. 31, pp. 1–30, 2009.

[215] J. De Leeuw, "Convergence of the majorization method for multidimensional scaling," *Journal of classification*, vol. 5, no. 2, pp. 163–180, 1988.

[216] I. Stoop and J. de Leeuw, "The Stepsize in Multidimensional Scaling Algorithms," in *Third European Meeting of the Psychometric Society, Jouy-en-Josas, France, July 5*, vol. 8, 1983, p. 1983.

[217] J. De Leeuw and W. J. Heiser, "Multidimensional scaling with restrictions on the configuration," *Multivariate analysis*, vol. 5, no. 1, pp. 501–522, 1980.

[218] M. Schäfer, M. Strohmeier, V. Lenders, I. Martinovic, and M. Wilhelm, "Bringing up OpenSky: A large-scale ADS-B sensor network for research," in *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*. IEEE, 2014, pp. 83–94.

[219] M. Matthews and R. DeLaura, "Assessment and interpretation of en route weather avoidance fields from the convective weather avoidance model," in *10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, 2010, p. 9160.

[220] E. A. Kalina, I. Jankov, T. Alcott, J. Olson, J. Beck, J. Berner, D. Dowell, and C. Alexander, "A progress report on the development of the High-Resolution Rapid Refresh ensemble," *Weather and Forecasting*, vol. 36, no. 3, pp. 791–804, 2021.

[221] National Weather Service. VIL Density as a Hail Indicator. [Online]. Available: https://www.weather.gov/lmk/vil_density

[222] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[223] R. Brunelli, *Template matching techniques in computer vision: theory and practice*. John Wiley & Sons, 2009.

[224] C. Di Franco, E. Bini, M. Marinoni, and G. C. Buttazzo, "Multidimensional scaling localization with anchors," in *2017 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE, 2017, pp. 49–54.

[225] M. A. Koledoye, T. Facchinetti, and L. Almeida, "MDS-based localization with known anchor locations and missing tag-to-tag distances," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2017, pp. 1–4.