UNIVERSITY OF CALIFORNIA,
IRVINE

Smoothing and Imputation of Longitudinal Vehicle Trajectory Data

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Civil Engineering

by

Ximeng Fan

Dissertation Committee:
Professor Wen-Long Jin, Chair
Professor Stephen G. Ritchie
Professor R. Jayakrishnan (Jay)

2023

DEDICATION

To my family

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express my sincerest gratitude to Professor Wenlong Jin for his advice during my doctoral study for the past five years. I am deeply appreciative of Professor Jin for not only providing me with invaluable professional guidance but also for serving as a role model in terms of dedication to studies and work. We have navigated through the challenges of a global pandemic, an event of extraordinary rarity in a century. Throughout this challenging period, I am sincerely thankful for the understanding and support that Professor Jin extended to me, not only in my current research but also in finding my true interest in future development.

I would like to thank my dissertation committee members, Professor Stephen Ritchie and Professor R. Jayakrishnan and, for their time and valuable suggestions. I also want to express my gratitude to Professor Wilfred Recker, Professor Michael McNally, Professor Jean-Daniel Saphores, and Professor Michael Hyland for their teaching and encouragement.

I also want to express my gratitude to all my ITS friends, including Xuting Wang, Lu Xu, Yiqiao Li, Dingtong Yang, Jared Sun, Arash Ghaffar, Chenying Qin, Brian Casebolt, Guoliang Feng, Koti Reddy Allu, Negin Shariat, Naila Sharmeen, Montana Reinoehl, Siwei Hu, Boyuan Jiang, Rony Gracious, Younghun Bahk, Jooneui Hong, and Llorenç Miquel I Solé. This journey is more fulfilling because of your company.

Taking this chance, I want to thank Real Madrid Football Club (RMCF). Since elementary school, this has consistently remained one of my most significant sources of joy and inspiration. Defeat wins, comebacks, etc., all inspire me. As Madridistas often say, football is a 90-minute affair, and the outcome can remain uncertain right up to the 89th minute. In a broader context, life itself is an extensive journey. It's essential not to yield or surrender prematurely, but it's equally vital to adapt and change formation if we find ourselves heading in the wrong direction.

In addition, I want to express my gratitude to some of my lifelong friends. Bocheng Wan, Lianghui Li, Menyyuan Wan, Ziya Zhong, and Zhou Zhou, with whom I have gone through more than 15 years of highs and lows. I would also like to express my gratitude to Tong Lu, Chenyu Sun, and Xiaolan Yu. Our paths crossed through a series of fortuitous encounters, and they have been witnesses to my growth, both academically and in love. AS flame of fire we gather, as skyful of stars we scatter.

I wish to extend a heartfelt, special thank you to my family, my parents Hong Fan and Xiaoling Guo; and my grandparents Jiuduo Guo and Yanlan Liao. Their unwavering love and support are priceless gifts that I can never fully repay. They have consistently expressed their sole desire to see me happy. I aspire to bring them pride, even if it's just a small measure of it. I also want to thank my close relatives, my aunt Xiaoan Guo; and my cousins Ziyue Zhang and Yiyang Zhang. Every Spring Festival we spend together forms one of the most beautiful parts of my childhood memories. During my growing up, this lovely big family has brought me a lot of happiness.

Moreover, I want to express my gratitude to my husband, Albert Lee. His unwavering support, patience, and unshakable belief in me have been the bedrock of my journey to complete this journey.

Your presence, with a listening ear, a comforting embrace, and unwavering respect, has made all the difference. I cannot put into words how fortunate I feel to have clicked that button and accepted the conversation from you. You are the best sweetie in the world for me.

In closing, I wish to extend my deepest love towards the heavens, where my beloved grandmother, Donglan Li, resides. Thank you for being the steadfast safe harbor that sustained me through my challenging and painful childhood, a time marred by the discriminatory tradition of favoring males over females. Despite your slender and fragile shoulders, you provided me with unwavering support and strength. I hope you know the significance of everything you've done for me. It is also my earnest wish that you can perceive my diligent efforts in fulfilling all the promises I made to you. Next time, I will place a lily on your gravestone, wishing that in your next life, you may find happiness, freedom, love, and never be taken for granted by undeserving men.

VITA

Ximeng Fan

EDUCATION

Ph.D. in Transportation and System Engineering **2023**
University of California, Irvine *Irvine, CA*

M.S. in Transportation and System Engineering **2020**
University of California, Irvine *Irvine, CA*

B.S. in Civil Engineering **2018**
Central South University *Irvine, CA*

RESEARCH EXPERIENCE

Graduate Research Assistant **2019–2023**
University of California, Irvine *Irvine, California*

TEACHING EXPERIENCE

Teaching Assistant **2022–2023**
University of California, Irvine *Irvine, California*

REFEREED CONFERENCE PUBLICATIONS

Quadratic Programming Method for Vehicle Trajectory Imputation Using Fixed and Mobile Sensor Data **2024-01**
Transportation Research Board Annual Meeting (No. 24-03230)

Two-step quadratic programming for physically meaningful smoothing of longitudinal vehicle trajectories **2023-03**
The 25th International Symposium on Transportation and Traffic Theory (ISTTT25) (Under review)

Novel Approach for Correcting and Smoothing Longitudinal Trajectories of Individual Vehicles **2022-01**
Transportation Research Board Annual Meeting (No. 22-02118)

Periodicity Detection of Simulated Traffic Data and Calibration of Network Fundamental Diagrams in Signalized Road Networks **2021-01**
Transportation Research Board Annual Meeting (No. 21-02673)

Impact of Advisory Speed Limit on the Overall Performance of Signalized Networks: A Network Fundamental Diagram Approach **2020-01**

Transportation Research Board Annual Meeting (No. 20-02733)

ABSTRACT OF THE DISSERTATION

Smoothing and Imputation of Longitudinal Vehicle Trajectory Data

By

Ximeng Fan

Doctor of Philosophy in Civil Engineering

University of California, Irvine, 2023

Professor Wen-Long Jin, Chair

The purpose of this study is to develop a methodology for processing vehicle trajectory data which are presented as a series of discrete positions of vehicles recorded over consecutive time intervals. The framework combines vehicle trajectory smoothing and imputation, ensuring that speeds and higher-order derivatives of positions are consistently defined as symplectic differences in positions, while adhering to physically meaningful bounds determined by traffic laws, drivers' behaviors, and vehicle characteristics.

To remove the outliers and high-frequency noises in speeds and higher-order derivatives, we incorporate some basic principles, including internal consistency, bounded speeds and higher-order derivatives, and minimum MAE between the raw and smoothed positions, based on physical properties and empirical observations. We propose an iterative method. One iteration comprises four types of calculations: differentiation, correction, smoothing, and integration. We adopt the adaptive average method for correction, the Gaussian filter for smoothing, and minimizing the MAEs as the objective in integration. The efficacy of the method is numerically shown with the NGSIM data. However, it is mathematically challenging to demonstrate when the iterations converge or even that the iterations can converge, leading us to develop more mathematically tractable techniques that can either be proved to converge or get rid of iterations.

We then propose a simplified iterative moving average method that makes the ranges of the smoothed

speeds, acceleration rates, and jerks align with physical meaning, while preserving the average speeds or total travel distance for a specified time duration segment of a vehicle's trajectory. Theoretically, we prove that without termination, the speed converges to a constant value after an infinite number of iterations, ensuring the termination of our method and physically meaningful ranges in speeds and their derivatives. Numerically, we demonstrate the advantages of the method in achieving physically and behaviorally meaningful ranges by applying it to the NGSIM dataset and comparing the results with manually re-extracted data and traditional filtering methods.

As another extension of the first smoothing method, We propose a two-step quadratic programming method that incorporates insights into human behavior, particularly the tendency to minimize jerks during motion, and integrates prior position errors derived from pixel length in video images. This method operates without the need for iterative processes, facilitating a single-round solution. Mathematically, we establish the existence and uniqueness of solutions to the quadratic programming problems, thus ensuring the well-defined nature of the method. Numerically, using NGSIM data, we compare the method with an existing approach with respect to the manually re-extracted ones and show the robustness of the method upon the highD data.

In addition, we investigate the scenarios involving missing portions of trajectories. In the last part of this dissertation, we consider segment scenarios where leading and trailing vehicles' trajectories are obtainable through mobile sensors, while those of intermediate vehicles require imputation based on detected entering and exiting times from loop detectors, and propose a three-step quadratic programming method for longitudinal trajectory imputation of fully sampled vehicles. The method ensures maintaining safe inter-vehicle spacing and adheres to physically meaningful speed, acceleration, and jerk ranges. Using NGSIM and highD data, we demonstrate the great performance of the method in imputing trajectories for three-, four-, five-, and six-vehicle platoons and illustrate its successful application in capturing the true conditions of a mixed-traffic system including 10% connected vehicles (CVs) and 10% CAVs.

Chapter 1

Introduction

1.1 Research background

Vehicle trajectory data provide a rich source of information for investigating traffic dynamics in both spatial and temporal domains, and accurate and complete vehicle trajectory data is crucial for numerous applications. Vehicle trajectory data can be collected through fixed-location sensors, such as loop detectors (Chen et al., 2001), cameras installed on high buildings (FHWA, 2007), stationary drones (Krajewski et al., 2018), or helicopters (Zheng et al., 2022), which can record all vehicle trajectories at the study sites with high sampling frequencies. On the other hand, mobile sensors include GPS devices (Yuan et al., 2010) and vehicle-equipped LiDARs (Sun et al., 2018); they have been widely applied to connected and autonomous vehicles (CAVs) to record their own trajectories and detect the trajectories of surrounding vehicles within their detection ranges (Yu et al., 2021), and there are numerous private and publicly available datasets for both naturalistic driving and autonomous vehicles (Masello et al., 2022).

However, limitations in sensor systems, image quality, extraction methods, and post-processing smoothing techniques can compromise the accuracy of such data, leading to errors and high-

frequency noises in positions, speeds, and higher-order derivatives. For example, in the NGSIM datasets, there can be an error of 1.2 m in positions and the sampling frequency is 10 Hz, the maximum error in speeds can be as high as 24 m/s, and those in acceleration rates and jerks as high as 480 m/s^2 and 9600 m/s^3, respectively. In addition, technical, infrastructure, and regulatory challenges impede the rapid advancements of CAVs. Even by the year 2050, it is projected that only approximately 50% of passenger vehicles sold will be highly autonomous (Litman, 2020), resulting in sampling limitations for collected trajectories and an incomprehensive picture of traffic conditions.

These problems pose many challenges to various applications of the data. While these problems have been the subject of study for many years, existing methods may yield smoothed or imputed trajectories lacking physically and behaviorally meaningful ranges of speed and higher-order derivatives, without providing mathematical guarantees of existence and uniqueness. This dissertation addresses the aforementioned challenges by introducing a framework that leverages physical properties, vehicle characteristics, and human driving behaviors to smooth and impute longitudinal vehicle trajectory data.

1.2 Research objective

This study centers on the processing of vehicle trajectory data, with the primary goal of formulating a methodology to smooth and impute longitudinal vehicle trajectory data. The objective is to align speeds and higher-order derivatives of positions with physical characteristics, vehicle attributes, and driver behaviors.

- **To propose an principle-based method for smoothing longitudinal vehicle trajectory data**

  Accurate vehicle trajectory data is crucial for numerous applications. However, limitations in sensor systems, image quality, extraction methods, and post-processing smoothing techniques can compromise the accuracy of such data, leading to errors in positions, speeds, accelerations, and jerks. To bridge these gaps, there is a need to introduce innovative approaches for smoothing vehicle trajectory data. In this dissertation, we incorporate some first principles for smoothing vehicle trajectories based on physical properties and empirical observations. We propose an iterative method based on these first principles.

- **To propose a simplified iterative moving average method for smoothing longitudinal vehicle trajectory data**

  Due to the difficulties in demonstrating when the iterations of the aforementioned method converge or even that the iterations can converge, we adopt the idea of the aforementioned method and improve it. We propose a simplified and straightforward iterative moving average method that ensures termination when the ranges of the smoothed speeds, acceleration rates, and jerks align with physical meaning, while preserving the average speeds or total travel distance for a specified time duration segment of a vehicle's trajectory.

- **To propose a quadratic programming method for physically meaningful smoothing of longitudinal vehicle trajectory data**

  With the same objective as the first part, we propose a two-step quadratic programming

method that ensures the smoothed speeds and higher order derivatives of positions are consistently defined as symplectic differences in positions, while adhering to physically meaningful bounds determined by traffic laws, drivers' behaviors, and vehicle characteristics. This method allows for the selection of the highest-order derivatives, which is as high as that in the definition of roughness. Additionally, we provide proof of the existence and uniqueness of solutions.

- **To propose a longitudinal vehicle trajectory data imputation method of fully sampled vehicles**

  Incomplete vehicle trajectory data results in an incomplete understanding of traffic conditions and poses challenges for their practical applications. We investigate scenarios involving missing portions of trajectories, requiring the imputation of missing values with the surrounding vehicles. We propose a three-step programming method for imputing longitudinal trajectory data of fully sampled vehicles using fixed and mobile sensor data. The method ensures maintaining safe inter-vehicle spacing and adheres to physically meaningful speed, acceleration, and jerk ranges.

- **To test the efficacy of our methods with empirical vehicle trajectory data**

  To assess the effectiveness of the proposed methods, we apply them to the NGSIM camera 6 datasets (FHWA, 2007). Specifically, for the smoothing methods, we compare their performance with an existing method (Montanino and Punzo, 2015) with respect to manually re-extracted data (Coifman and Li, 2017a). Additionally, we employ the methods with the highD datasets (Krajewski et al., 2018) to evaluate their robustness.

1.3 Research outline

The overall framework of the dissertation can be illustrated via Figure 1.1, and the outline of this dissertation are introduced as follows.

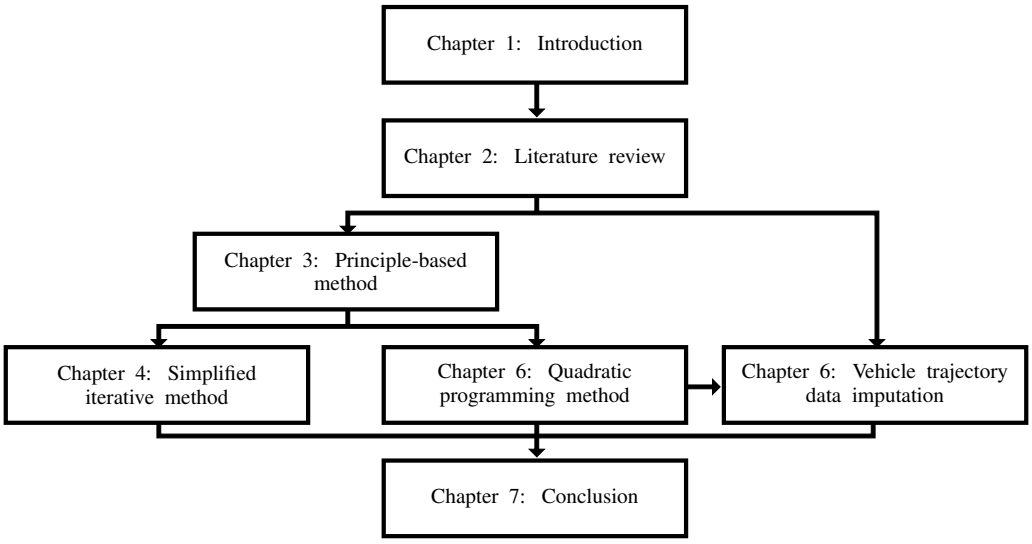


Figure 1.1: Framework of the dissertation

Chapter 1 introduces the research background and objective of this dissertation.

Chapter 2 provides a detailed literature review regarding the four important components of the proposed framework: the collection of vehicle trajectory data, problems associated with vehicle trajectory data, methods for smoothing vehicle trajectory data, and techniques for imputing vehicle trajectory data.

Chapter 3 incorporates some first principles for smoothing vehicle trajectories, including the internal consistency among the positions, speeds, and higher-order derivatives, bounded speeds and higher-order derivatives, and minimum MAE between the raw and smoothed positions, and proposes an iterative method based on these first principles. One iteration comprises four types of calculations: differentiation, correction, smoothing, and integration. In differentiation, we compute speeds, accelerations, and jerks from trajectory data; in correction, we eliminate outliers in speeds, especially negative values; in smoothing, we reduce noises in accelerations and jerks; and

in integration, we recalculate accelerations, speeds, and positions from jerks, and find the initial values via optimization problems. We adopt the adaptive average method for correction, the Gaussian filter for smoothing, and minimizing the MAEs as the objective in integration. We then numerically show the efficacy of our method upon the vehicle trajectories in the Next-Generation Simulation (NGSIM) dataset. However, it is mathematically challenging to demonstrate when the iterations converge or even that the iterations can converge, leading us to develop more mathematically tractable techniques that can either be proved to converge or get rid of iterations.

Chapter 4 presents a simplified and straightforward iterative moving average method for smoothing vehicle trajectory data. The method guarantees physically meaningful ranges for the smoothed speeds, acceleration rates, and jerks of a vehicle's trajectory while preserving its average speeds or total travel distance over a specified time duration. In each iteration, our method consists of two steps. In the first step, we pad the speed profile with the average speed. We then apply a moving average method to speeds using different kernel shapes. In the third step, we normalize the filtered speeds to preserve the mean value. The iterative process terminates when speeds, acceleration rates, and jerks fall within reasonable bounds. We then theoretically prove that without termination, the speed converges to the average speed after an infinite number of iterations. Finally, we apply our method to smooth vehicle trajectories in the NGSIM dataset and compare the results with ground truth data and traditional filtering methods.

Chapter 5 proposes a two-step quadratic programming method to smooth vehicle trajectory data. The method ensures that smoothed speeds and higher order derivatives of positions are consistently defined as symplectic differences in positions, and present linear inequality constraints based on bounded derivatives of positions. We leverage our prior knowledge of position error, determined by pixel length in video images (FHWA, 2007; Krajewski et al., 2018), and formulate two sequential quadratic programming problems. In the first step, we minimize the discrepancy between half-smoothed and raw positions, subject to physically meaningful bounds on speeds and higher order derivatives of half-smoothed positions. In the second step, we minimize the roughness of

the smoothed positions, maintaining the same bounds on speeds and higher order derivatives of smoothed positions, along with additional bounds on the smoothed positions themselves. This step allows the smoothed positions to deviate from the raw data by at most those of the half-smoothed positions and the prior position error. In both steps, the bounds are applied to derivatives, whose order is as high as that in the definition of roughness; i.e., the second step dictates the highest order of derivatives. This method gets rid of the iterative process, allowing the entire process to be completed in a single round. We then analytically prove the existence and uniqueness of solutions to both quadratic programming problems. Finally, using both NGSIM and highD data, we calibrate the highest order of derivatives and compare our method with an existing approach with respect to manually re-extracted data.

Chapter 6 investigates the scenarios involving missing portions of trajectories, proposing a three-step quadratic programming method for longitudinal trajectory imputation of fully sampled vehicles using fixed and mobile sensor data. We consider segment scenarios where leading and trailing vehicles' trajectories are obtainable through mobile sensors, while those of intermediate vehicles require imputation based on detected entering and exiting times from fixed sensors. Throughout all steps, we incorporate physically meaningful ranges of speeds, accelerations, and jerks as constraints. In step 1, we calculate the fastest possible trajectory that maintains a safe distance from the leading vehicle. In step 2, we determine the slowest possible trajectory that maintains safe distance from the following vehicles. Finally, we compute the trajectory lying between the slowest and fastest possible trajectories, optimizing for the minimal sum of squared jerks. Using NGSIM and highD data, we numerically show the efficacy of our method for three-, four-, five-, and six-vehicle platoons, as well as a traffic system comprising 10% CVs and 10% CAVs.

Chapter 7 summarizes the results and findings of this dissertation, and proposes the potential extensions for future research.

A list of notation is given in **Table** 1.1. The superscript k denotes the "layer" of the method in Chapter 3 (referred to Figure 3.1), and in Chapters 4 to 6, it denotes the order of the derivatives of

positions. $\hat{x}_m$ and $\tilde{x}_m$ are featured in both Chapter 4 and Chapter 5, referred to as filtered positions and normalized-filtered positions, and half-smoothed and smoothed positions, respectively.

Table 1.1: List of notations

Variables	Definitions	Variables	Definitions
M	Total number of time instants	$E(\cdot)$	Mean value
K	The highest order of derivatives of positions		
$x_-^{(k)}$	Lower bound of the kth order derivatives of positions, $k = 1, \cdots, K$, $k = 1,2,3$ correspond to v_-, a_-, and j_-		
$x_+^{(k)}$	Upper bound of the kth order derivatives of positions, $k = 1, \cdots, K$, $k = 1,2,3$ correspond to v_+, a_+, and j_+		
Notations for one-vehicle scenarios			
L	Order of the polynomial function for speed interpolation		
t	Time	m	Time index
t_m	mth time instant	Δt	Time-step size
ε	Prior position error	ϕ_m	Kernel of the filter
ψ_m	Truncated kernel	w	Kernel size of the filter (it is an odd number)
σ	Standard derivation of the Gaussian filter		
x_m	Raw position at t_m	$\hat{x}_m$	Filtered / half-smoothed position at t_m
$\tilde{x}_m$	Normalized-filtered / smoothed position at t_m	$\bar{x}_m$	Manually re-extracted position at t_m
$\mathbf{x}_-$	Vector of the lower bounds of positions	$\mathbf{x}_+$	Vector of the upper bounds of positions
$x_m^{(k)}$	kth order derivative of x at t_m, $k = 1, \cdots, K$, $k = 1,2,3$ correspond to v, a, and j		
$\hat{x}_m^{(k)}$	kth order derivative of $\hat{x}$ at t_m, $k = 1, \cdots, K$	$\tilde{x}_m^{(k)}$	kth order derivative of $\tilde{x}$ at t_m, $k = 1, \cdots, K$
$\bar{x}_m^{(k)}$	kth order derivative of $\bar{x}$ at t_m, $k = 1, \cdots, K$		
Notations for multiple-vehicle scenarios			
N	Number of vehicles in a platoon	t_m^n	mth time instant of vehicle n's trajectory
τ_1^n	Time gap between vehicle n and its leading vehicle		
τ_2^n	Time gap between vehicle n and its following vehicle		
ζ_1^n	Jam spacing between vehicle n and its leading vehicle		
ζ_2^n	Jam spacing between vehicle n and its following vehicle		
$x^n(t)$	Detected position of vehicle n at time t	$\hat{x}_+^n(t)$	Fastest possible position of vehicle n at time t
$\hat{x}_-^n(t)$	Slowest possible position of vehicle n at time t	$\tilde{x}^n(t)$	Imputed position of vehicle n at time t
$\bar{x}^n(t)$	Real position of vehicle n at time t		

Chapter 2

Literature review

2.1 Collection and application of vehicle trajectory data

Continuously growing sensing and extraction technology increases the availability of high-fidelity vehicle trajectory data, which have further contributed to much in-depth transportation research and widened new horizons at both vehicle-level and network-level studies. The data collection process typically involves several stages, including system setup, trajectory dection, trajectory extraction, and post-processing.

Trajectory data can be recorded using two types of sensors, some of which are shown in Figure 2.1. On the other hand, fixed-location sensors, such as loop detectors (Chen et al., 2001), cameras mounted on high buildings (FHWA, 2007), drones (Krajewski et al., 2018) or helicopters (Zheng et al., 2022), and roadside lidars (Sun et al., 2018), offers the capability to record all vehicle trajectories within a limited detection range while requiring relatively high installation and maintenance costs. They have contributed to many in-depth studies including driving behaviors analysis (Chiabaut et al., 2010; Chen et al., 2014), car-following models calibration and validation (Kesting and Treiber, 2008; Punzo and Montanino, 2016), vehicle emission estimation (Sun et al., 2015;

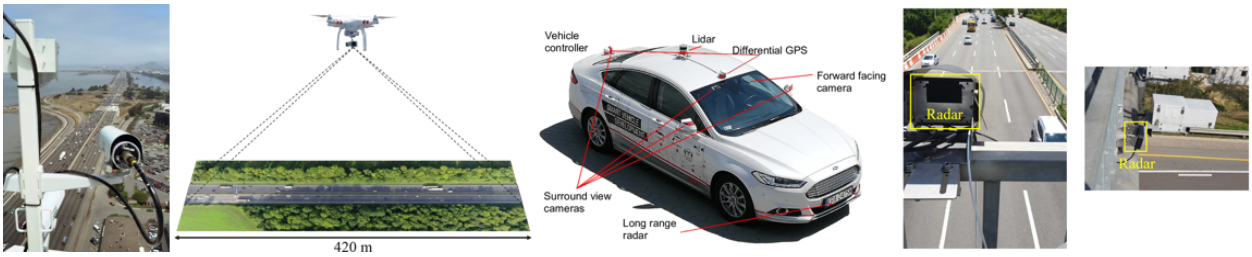


Figure 2.1: Sensors for detecting vehicle trajectory data. Sources: (FHWA, 2007; Krajewski et al., 2018; Belcarz et al., 2018; Lim et al., 2020)

da Rocha et al., 2015), and eco-driving strategy development (Yang and Jin, 2014; Gong et al., 2016).

The other type is called mobile sensors, such as GPS devices (Yuan et al., 2010) and vehicle-equipped lidars (Sun et al., 2018), which enable the recording of complete vehicle trajectories and have been widely applied in connected and autonomous vehicles (CAVs). CAVs are usually equipped with multiple sensors, such as Lidars, cameras, and GPS, empowering them to record their own trajectories and detect the trajectories of surrounding vehicles within their detection ranges (Yu et al., 2021). These data are frequently used for studies in large networks including network design and improvement (Zheng et al., 2011) and route planning (Yuan et al., 2010, 2011a,b).

2.2 Problems lying in vehicle trajectory data

Vehicle trajectory data often suffer from errors and incompleteness, making it challenging to obtain an accurate picture of traffic conditions. Errors and noises in vehicle positions can occur at each stage in the aforementioned data collection process, caused by detector vibrations, blockage of vehicles by road signs, projection angles of detectors, shadows, image resolutions, and so on. With high sampling frequencies, errors in speeds and higher order derivatives of positions can be further amplified (Coifman et al., 1998; Coifman and Li, 2017a). For example, with $25Hz$ sampling frequency, a 0.1 m position error can translate to errors up to 5 m/s in speeds, 250 m/s^2 in accelerations, and 12500 m/s^3 in jerks. Furthermore, when collecting data from mobile sensors,

only a limited number of vehicles equipped with such sensors are observable, while others remain transparent to data collection efforts. This limitation persists, especially due to technical, infrastructure, and regulatory challenges. Remarkably, even in the year 2050, it is anticipated that only approximately 50% of passenger vehicles sold will be highly autonomous (Litman, 2020). Consequently, a prolonged period of mixed traffic, comprising both human-driven vehicles (HDVs) and CAVs, is foreseen, leading to sampling limitations in the trajectories collected. This scenario imposes constraints on the trajectories that can be collected. Additionally, despite the existence of some high-resolution GPS datasets (Punzo and Simonelli, 2005), GPS data, particularly those derived from mobile phones, frequently exhibit relatively high position errors. GPS devices, for example, have an average user range error of 7.8 m (25.6 ft) with a 95% probability (Wang et al., 2021).

The aforementioned challenges result in an incomprehensive picture of traffic conditions and give rise to several difficulties, including significantly reducing the sensitivity of the objective function in model calibration and further reducing the reliability of the calibration results (Ossen and Hoogendoorn, 2008, 2009), reducing the accuracy of trajectory prediction (Altché and de La Fortelle, 2017), and inaccurate estimation of traffic emission (Tsanakas et al., 2022).

2.3 Typical ranges of speeds and higher-order derivatives

Typically, vehicles on a road are expected to either halt or progress, indicating that their speed should not fall below zero. (Martinez and Canudas-de Wit, 2007) found that jerks are typically bounded within $[-8,8]$ m/s^3 based on the experiments conducted with LOLA test vehicles. The International Organization for Standardization (ISO) has also established requirements for adaptive cruise control (ACC) systems, stipulating that the acceleration of ACC systems must not exceed 2 m/s^2 and that the average rate of change of automatic deceleration (negative jerk) over one second must not surpass -2.5 m/s^3 (ISO, 2010). (Elert, 2012) observed that the average accelerations

of ordinary vehicles are usually between 3 m/s^2 to 4 m/s^2. (Bokare and Maurya, 2017) conducted tests examining accelerating and decelerating behaviors for various vehicle types on a highway, revealing that maximum acceleration and deceleration values for petrol vehicles were approximately 3 m/s^2 and 4 m/s^2, respectively. Based on a freeway experiment involving sixteen Honda Accords and 108 randomly sampled drivers, (Feng et al., 2017) demonstrated that the 99th percentile acceleration and jerk values were 2.85 m/s^2 and 2.6 m/s^3. Moreover, some studies on adaptive cruise control have also utilized a jerk range of $[-5,5]$ m/s^3 (Al-Gabalawy et al., 2021).

2.4 Methods for improving the quality of the vehicle trajectory data

To address these challenges and enhance the accuracy and comprehensiveness of collected vehicle trajectory data, various strategies can be employed. Correcting errors and reducing noise in trajectory data involves improvements at each stage of data collection. This includes selecting road sections devoid of large road signs, making the detectors directly above the recording vehicles, utilizing high-resolution cameras and low-vibration carriers, and manual re-extraction. In addition, post-processing smoothing methods can further help to estimate the most probable values of the ground truth data from observed raw data (Whittaker, 1922). In cases where trajectory data represent only sampled traffic flow, trajectory imputation methods leveraging multi-source detected trajectory data can aid in reconstructing fully sampled data. These methods contribute to a more comprehensive and accurate representation of traffic patterns, despite the inherent challenges in data collection.

2.4.1 Existing methods for smoothing vehicle trajectories

improving post-processing smoothing methods can be a cost-effective solution to correcting observed vehicle trajectories[1], especially when the sensor system, image quality, and extraction methods are already optimal under labor, financial, and other constraints. Smoothing methods can be considered low-pass filters, which can be in frequency or time domains (Smith, 1997). Many frequency-domain filters, such as the Butterworth filter (Pollock et al., 1999) and wavelet-based filters (Young, 1992), have been applied to filter out high frequency oscillations in speeds (Fard et al., 2017; Dong et al., 2021). However, there is no guarantee that the smoothed speeds and higher order derivatives of positions adhere to physically meaningful bounds determined by traffic laws, drivers' behaviors, and vehicle characteristics (Pendrill and Eager, 2020; Jin, 2021). In the time domain, positions, speeds, accelerations, and higher order derivatives of positions can be smoothed by various convolution or recursive filters (Ma and Andréasson, 2005; Thiemann et al., 2008; Coifman and Li, 2017a; Krajewski et al., 2018), in which the smoothed position at time t_m, $\tilde{x}_m$, is calculated by the weighted average of raw or smoothed positions in a surrounding time window. These methods are straightforward to implement, but separately smoothed positions, speeds, and higher order derivatives may not be consistent with each other or guaranteed to satisfy physically meaningful bounds (Thiemann et al., 2008). In addition, these methods do not ensure physically meaningful smoothed trajectories, as they may not adequately account for the bounded nature of speeds, acceleration rates, and jerks observed in daily driving experiences, vehicle mechanics, traffic laws, and observations (Jin, 2021).

The primary challenge in smoothing vehicle trajectories and general time series data is striking the right balance between fidelity and smoothness. Smoothing data with lower roughness may result in greater deviations from the raw data, whereas smoothing data closer to the raw data tends to be rougher. In the literature, another category of smoothing methods, based on splines,

[1] Smoothing methods can help to correct vehicle trajectories, since, in general, smoothing methods can help to find the most probable estimations of the true values (Whittaker, 1923).

addresses this challenge by formulating an optimization problem. These methods aim to obtain reconstructed data that achieves the optimal combination of fidelity and smoothness. In the well-known smoothing splines method (Whittaker, 1922; Eubank, 1999), which minimizes a weighted sum of the discrepancy, measured by the sum of the squared differences between the raw and smoothed data, and the roughness, measured by the sum of squared third-order derivative of the smoothed data. To the best of our knowledge, the smoothing splines method has not been applied to smooth vehicle trajectories; even if it is applied, the weights lack physical meaning for smoothing vehicle trajectories. In contrast, regression splines methods fit raw data with piecewise polynomial functions, which are smoothly connected at the separation points (known as knots) (Sheppard, 1914; Whittaker and Robinson, 1924); (Toledo et al., 2007; Venthuruthiyil and Chunchu, 2018) adopted these methods to divide the whole interval of a vehicle's trajectory into several smaller sub-intervals and approximate such sub-trajectories with low-degree polynomial functions. Here the piecewise polynomial functions and conditions at knots guarantee the smoothness of the fitted data, and the least square regression process ensures the fidelity of the smoothed trajectory by minimizing the discrepancy between the polynomials and raw data. Furthermore, the penalty splines method was proposed in (Eilers and Marx, 1996), by fitting raw data with piecewise B-splines and minimizing a weighted sum of the discrepancy and the roughness. Thus, it can be considered a combination of smoothing and regression splines methods. This method was applied to smooth vehicle trajectories in (Marczak and Buisson, 2012). However, there lacks a systematic method to choose knots in regression and penalty methods; furthermore, some of the existing methods fail to include physically meaningful bounds on various derivatives of positions (Marczak and Buisson, 2012; Venthuruthiyil and Chunchu, 2018), and others struggle to ensure the existence and uniqueness of solutions of the constrained optimization problem (Toledo et al., 2007).

In addition, a combination of different filters has been employed, which involves the smoothing of speeds using a Butterworth filter, followed by locally approximating the speeds as fifth order polynomials and eliminating the outliers via solving optimization problems (Montanino and Punzo, 2013, 2015). This method also faces difficulties in finding unique optimal solutions, and spline

filters may lack boundedness in speeds and higher order derivatives.

2.4.2 Existing methods for imputing vehicle trajectories

Numerous methods have been proposed for imputing vehicle trajectory data. One category of methods focuses on the imputation of sparse single-vehicle trajectories. Periodical idle - acceleration - cruise - deceleration - idle activity pattern was assumed, based on which probabilistic models are proposed to impute sparse data into high-resolution vehicle trajectories (Hao et al., 2014; Shan et al., 2016; Hao et al., 2016; Shan et al., 2018; Wang et al., 2019). In addition, some studies have explored the utilization of filters, such as the particle filter and the unscented Kalman filter, for the imputation of vehicle trajectories (Wei et al., 2020; Mu et al., 2021). Another significant area of research focuses on imputing fully sampled trajectories. One approach involves utilizing detected data from double-loop detectors to estimate travel time and vehicle trajectories based on traffic flow models (Coifman, 2002). With the advancement of CAVs, mobile sensor data have been increasingly incorporated into vehicle trajectory imputation, where the trajectories of CAVs are given and the trajectories of HDVs are imputed with the assumption that human driving behaviors can be described by some car-following models (Wang et al., 2020; Zhou et al., 2022; Yao et al., 2022), such as Wiedemann model (Wiedemann, 1974), Gipps model (Gipps, 1981), intelligent driver model (IDM)(Treiber et al., 2000), and simplified Newell's model (Newell, 2002). Furthermore, to enhance the accuracy of trajectory imputation, data fusion algorithms have been introduced to combine information from different sources, resulting in improved trajectory estimation (Chen et al., 2022).

Despite significant advancements in trajectory imputation methods, there are still some limitations. The studies using probabilistic models to impute single vehicle trajectories usually assume some format of accelerations/decelerations, both accelerations/decelerations being linear functions of time (Hao et al., 2014, 2016) or piecewise-constant (Wang et al., 2019), which contradicts the em-

pirical experiments that indicate the boundedness of speeds, accelerations, and jerks (Pendrill and Eager, 2020; Jin, 2021). For the studies adopting particle filters (Wei et al., 2020; Mu et al., 2021), ensuring the accuracy of particle weights necessitates employing very large sample sizes (Straka and Ŝimandl, 2009). Besides, the imputation of single vehicle trajectories overlooks the influence of vehicle-to-vehicle interactions, while the reliance on historical data for parameter calibration, such as accelerations and decelerations, fails to adapt to real-time changes in the external environment. As a result, there is a growing need for the imputation of fully sampled trajectories to provide a more comprehensive and accurate description of traffic flow. Earlier studies in this area primarily utilized loop detector data and did not exploit the potential of mobile sensor data. Consequently, these studies had to make assumptions that all vehicles travel at the same speed passing through a given band between two vehicle passages (Coifman, 2002). Additionally, studies assuming vehicles moving according to some models involving multiple parameters, such as IDM and Wiedemann models, encounter challenges when it comes to parameter determination. Furthermore, other car-following models, such as Gipps and Newell's model, may lead to unbounded higher-order derivatives of positions, conflicting with empirical observations. Moreover, it is worth noting that certain methods have primarily undergone validation under simulated conditions, lacking experiments with real-world data sources that would offer greater fidelity (Wei et al., 2019, 2020).

Chapter 3

An iterative method for smoothing based on first principles

3.1 Introduction

In light of the physical properties and empirical observations, we proceed from the basic principles. Intuitively, a good method should satisfy the following principles: (P1) The internal consistency among the derivatives at different orders, i.e. positions, speeds, accelerations, and jerks, should be guaranteed. The derivatives of positions should be consistent with speeds, the integrals of speeds should be consistent with positions, and likewise for accelerations and jerks. (P2) According to the first principles of road traffic, as stated in (Jin, 2021), speeds, accelerations, and jerks should be bounded. (P3) The mean absolute errors (MAEs) in the raw positions should be kept to a minimum, which implies that the modification of the raw positions should be kept to a minimum. This will help us preserve the main information in the raw data, including cruising, accelerating, decelerating, and stopping behaviors.

In this chapter, we propose an iterative method to smooth and correct the longitudinal vehicle

trajectory data which are detected at fixed study sites and extracted at high sampling frequencies. These data usually include small errors in positions, but such errors are greatly amplified in speeds, accelerations, and jerks after differentiation due to high sampling frequencies. One iteration comprises four types of calculations: differentiation, correction, smoothing, and integration. In differentiation, we compute speeds, accelerations, and jerks from trajectory data; in correction, we eliminate outliers in speeds, especially negative values; in smoothing, we reduce noises in accelerations and jerks; and in integration, we recalculate accelerations, speeds, and positions from jerks, and find the initial values via optimization problems. We analyze different strategies and adopt the adaptive average method for correction, the Gaussian filter for smoothing, and minimizing the MAEs as the objective in integration. While it is ideal for all the derivatives to be physically meaningful after one iteration, if there are still outliers after one iteration, we will begin another to fine-tune the results. We then show the efficacy of our method upon a sample trajectory from the NGSIM dataset, followed by comparing our smoothed trajectories with the raw trajectories and with those obtained from the state-of-the-art method proposed in (Montanino and Punzo, 2015), and validate the smoothed trajectories with those manually re-extracted by (Coifman and Li, 2017a).

All of the aforementioned principles are maintained in our approach. P1 is maintained by differentiation and integration, which guarantees internal consistency among the different-order derivatives of positions. Following P2, the physically meaningful bounds of speeds, accelerations, and jerks are maintained in correction and smoothing and serve as the criterion to determine when the iterations terminate; our method will end only if all speeds, accelerations, and jerks are physically meaningful. P3 helps us determine the initial values in integration, where the initial values are calculated in such a way that the MAEs between derivatives before and after smoothing are minimized.

The remainder of the chapter is organized as follows. In section 3.2, we introduce our method for smoothing and correcting vehicle trajectories. In section 3.3, We discuss the effect of different

smoothing methods and justify our selection, as well as develop an algorithm for choosing parameters in the Gaussian filter. In section 3.4, we apply our proposed method to the NGSIM dataset, and numerically show the efficacy of our method. We also compare our method with a state-of-the-art approach and validate it upon the manually re-extracted freeway trajectories. Finally, we conclude our study in section 3.5 and propose some future extensions.

3.2 A principle-based iterative method for smoothing

In this section, we present an iterative method for smoothing and correcting longitudinal vehicle trajectories. We first present the flow chart of our method and the framework of the calculations, and we then introduce each calculation in detail.

The framework of our proposed method is shown as Figure 3.1. The method takes into account positions, speeds, accelerations, and jerks. The input is the raw position profile and the output is the smoothed position profile. One loop in the dashed black box represents one iteration. One iteration includes four different colored arrows that represent four types of calculations in the method: differentiation, correction, smoothing, and integration. The red arrow represents differentiation, the green arrow represents correction, the yellow arrow represents smoothing, and the blue arrow represents integration. The entire framework contains four layers. All the derivatives in one layer should be internally consistent, satisfying both differentiation and integration relations, and once a correction or smoothing procedure is completed, we proceed to the next layer. Each time after we integrate accelerations into speeds, we judge whether all speeds, accelerations, and jerks are physically meaningful. All derivatives are ideally expected to be within physically meaningful ranges after one iteration. However, if there are still outliers after one iteration, we will start another iteration with smaller parameters to fine-tune the results. Finally, when all speeds, accelerations, and jerks are physically meaningful, we consider that smoothing and correction are finished and calculate the smoothed positions by integrating the speed profile. A detailed description of each

calculation will follow.

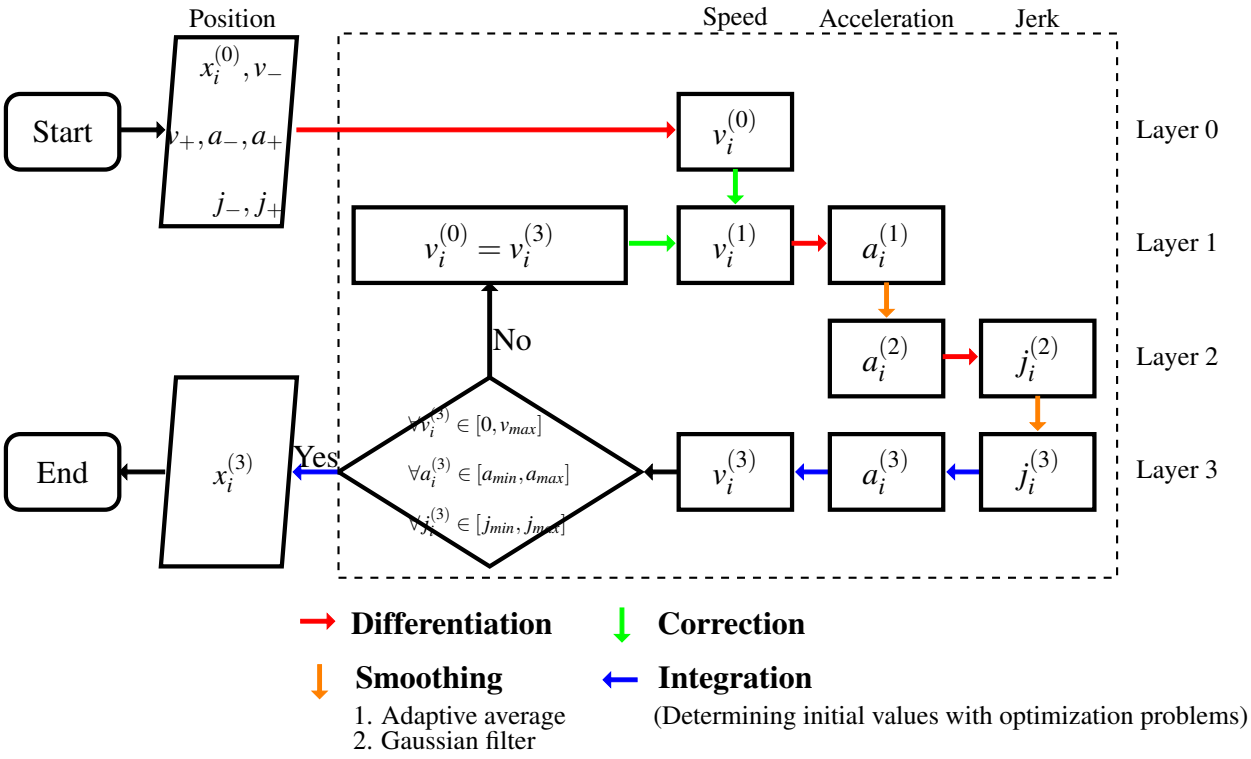


Figure 3.1: The framework of the calculations

3.2.1 Differentiation of positions

In this study, we consider the longitudinal kinematic behavior of vehicles. As pointed out by P1, positions, speeds, accelerations, and jerks should be internally consistent. We calculate speeds by differentiating positions. We adopt the symplectic discretization (mixed implicit-explicit Euler discretization) method, which was analytically proved to be the only physically meaningful discretization method that can always lead to collision-free and forward-traveling solutions (Jin, 2019). The general discretized longitudinal vehicle motion dynamic can be illustrated via Figure 3.2. Each black node represents a position, each green node represents a speed, each blue node represents an acceleration, and each yellow node represents a jerk. The arrows' directions indicate the directions of calculations. For example, the speed at time instant t_m can be calculated with the positions at t_{m-1} and t_m. There are t_M discrete positions in the position profile. The lengths

of corresponding speed, acceleration, and jerk profiles are t_M, t_{M-1}, and t_{M-2}, respectively. The speed profile covers t_2 to t_M. The acceleration profile covers t_2 to t_{M-1}, while the jerk profile starts at t_3 and ends at t_{M-1}.

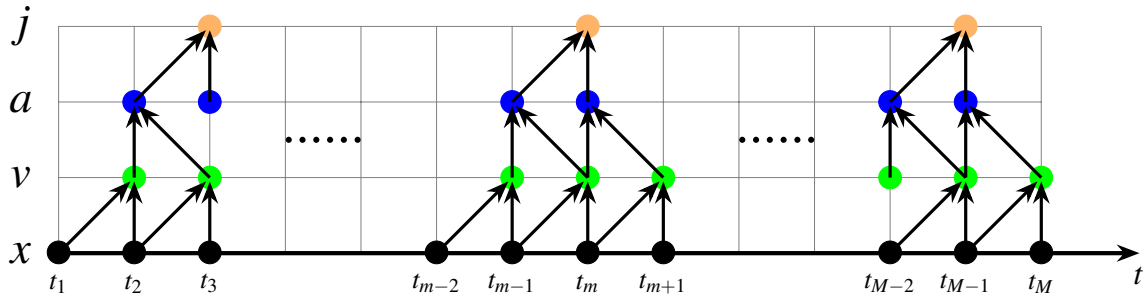


Figure 3.2: Illustration of discretized longitudinal vehicle dynamic

The speed, acceleration, and jerk at each time instant can be calculated as follows:

$$v_m^{(k)} = \frac{x_m^{(k)} - x_{m-1}^{(k)}}{\Delta t}, \quad m = 2,3,.....,M, \tag{3.1a}$$

$$a_m^{(k)} = \frac{v_{m+1}^{(k)} - v_m^{(k)}}{\Delta t} = \frac{x_{m+1}^{(k)} - 2x_m^{(k)} + x_{m-1}^{(k)}}{(\Delta t)^2}, \quad m = 2,3,.....,M-1 \tag{3.1b}$$

$$j_m^{(k)} = \frac{a_m^{(k)} - a_{m-1}^{(k)}}{\Delta t} = \frac{x_{m+1}^{(k)} - 3x_m^{(k)} + 3x_{m-1}^{(k)} - x_{m-2}^{(k)}}{(\Delta t)^3}, \quad m = 3,4,.....,M-1 \tag{3.1c}$$

where $x_m^{(k)}$, $v_m^{(k)}$, $a_m^{(k)}$ and $j_m^{(k)}$ are the position, speed, acceleration, and jerk at the k th layer ($k = 0,1,2,3$) at time t_m. Speeds, accelerations, and jerks are the first, second, and third-order derivatives of positions. Once the four derivatives are at the same layer, they should satisfy (3.1a) to (3.1c). Higher-order derivatives can be obtained from lower-order derivatives.

3.2.2 Speed correction

As pointed out by P2, speeds should be in a reasonable range. Values outside this range are considered outliers. The objective of this calculation is to remove outliers in speeds that are generated because of errors in position detection.

We develop the adaptive average method to correct the speed profile, in which we average speeds starting at the time when the outlier occurs until the average speed is within the range $[0, v_+]$. The raw speeds are then replaced by the average speed. We take Figure 3.3 as examples to illustrate this process, where $x_m^{(0)}$ and $v_m^{(0)}$ denote the raw position at t_m and the speed calculated from the raw positions at t_{m-1} and t_m. Each black node represents a raw position, and the slope of the line between two adjacent nodes represents a speed. The average speed between two non-adjacent positions is equal to the slope of the single line connecting these two position nodes. In Figure 3.3a, $v_{m+1}^{(0)}$ is negative and needs correction. We add up the speeds starting from $v_{m+1}^{(0)}$ and find that the nearest time point that can lead to a physically meaningful average speed is $m+3$. Therefore, we replace the raw speeds in such a period with the average speed between m and $m+3$, as illustrated by the red line, and the new positions are shown as the red nodes. Similarly, when a speed exceeds the upper bound, as Figure 3.3b shows, we also replace the raw speeds in such period with the average speed. In both examples, the average speed can be calculated by (3.2):

$$v_{m+1}^{(1)} = v_{m+2}^{(1)} = v_{m+3}^{(1)} = \frac{v_{m+1}^{(0)} + v_{m+2}^{(0)} + v_{m+3}^{(0)}}{3}, \tag{3.2}$$

where $v_m^{(1)}$ is the corrected speed at time t_m. Additionally, when a speed outlier appears at the end of the recording duration, we replace it with v_+ if it is larger than v_+, and with zero if it is negative.

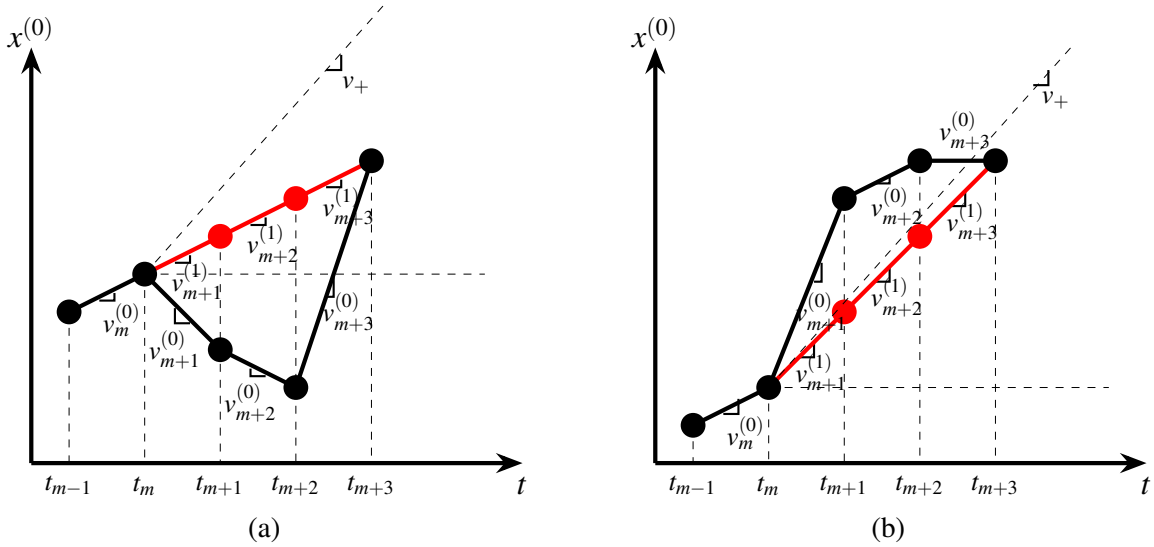


Figure 3.3: Illustration of the adaptive average method

3.2.3 Acceleration and jerk smoothing

After eliminating all outliers in the speed profile, we calculate the acceleration profile $a^{(1)}$ by differentiating the speed profile according to (5.2b). According to P2, accelerations are also bounded, and values outside the physically meaningful range $[a_-, a_+]$ are regarded as outliers. If outliers exist in $a^{(1)}$, we should smooth $a^{(1)}$. Here we consider two alternatives.

Here we consider the Gaussian filter, which averages the input signal with a Gaussian kernel in the following:

$$\phi(l) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{l^2}{2\sigma^2}}, \tag{3.3}$$

where l is the variable and it represents the distance to the center of the kernel, σ is the standard deviation. With (3.3) the weights of each value within the kernel can be calculated. A true Gaussian response would have infinite impulse responses and they decay rapidly, thus we truncate the

Gaussian kernel at discrete time points as follows:

$$\psi(l) = \frac{\phi(l)}{\sum_{\alpha=-p\sigma}^{p\sigma} \phi(\alpha)}, \tag{3.4}$$

where p represents that we smooth the value at m with the values from $m - p\sigma$ to $m + p\sigma$, $2\sigma + 1$ values in total. We choose $p = 4$ where the amplitude of the Gaussian function is about 3×10^{-4} of that at m, thus the kernel size of the Gaussian filter depends on the standard derivation. We handle the edge of the signal via zero padding.

An example of calculating the smoothed acceleration at time i using the truncated Gaussian filter with the standard derivation $\sigma(a) = 1$ time-step is presented, which means that $a_m^{(2)}$ will be smoothed with the values from $m-4$ to $m+4$, 9 values in total. The smoothed acceleration ($a_m^{(2)}$) will be the sum of itself and the 8 adjacent values multiplied by weights corresponding to their distances to m, which can be written as follows:

$$a_m^{(2)} = \frac{a_m^{(1)} + e^{-\frac{1}{2}}(a_{m-1}^{(1)} + a_{m+1}^{(1)}) + e^{-2}(a_{m-2}^{(1)} + a_{m+2}^{(1)}) + e^{-\frac{9}{2}}(a_{m-3}^{(1)} + a_{m+3}^{(1)}) + e^{-8}(a_{m-4}^{(1)} + a_{m+4}^{(1)})}{\sum_{\alpha=-4}^{4} e^{-\frac{\alpha^2}{2}}}. \tag{3.5}$$

By applying the Gaussian filter throughout the entire $a^{(1)}$, we can smooth accelerations and calculate $a^{(2)}$.

With $a^{(2)}$, we can then obtain the jerk profile $j^{(2)}$ with (3.1c). Similarly, jerks should be bounded, and if $j^{(2)}$ also contains outliers that are outside $[j_-, j_+]$, we smooth the jerk profile with the aforementioned procedure and calculate $j^{(3)}$.

3.2.4 Integration to lower-order derivatives

According to the 'internal consistency' mentioned in P1, accelerations should be equal to integrals of jerks, speeds should be equal to integrals of accelerations, and positions should be equal to integrals of speeds. Accelerations, speeds, and positions can be calculated as follows:

$$a_m^{(k)} = a_2^{(k)} + \Delta t \sum_{l=3}^{m} j_l^{(k)}, \quad m = 3,4,.....,I-1, \tag{3.6a}$$

$$v_{m+1}^{(k)} = v_2^{(k)} + \Delta t \sum_{l=2}^{m} a_l^{(k)}, \quad m = 3,4,.....,I-1, \tag{3.6b}$$

$$x_m^{(k)} = x_1^{(k)} + \Delta t \sum_{l=2}^{m} v_l^{(k)}, \quad m = 2,3,.....,I, \tag{3.6c}$$

where k represents the layer in Figure 3.1, here $k = 3$, and $a_2^{(k)}$, $v_2^{(k)}$, $x_1^{(k)}$ are the initial values of the acceleration, speed, and position profiles, respectively. The calculation is illustrated in Figure 3.4, with each black node representing a position, each green node representing a speed, each blue node representing an acceleration, and each yellow node representing a jerk. The arrows' directions indicate the directions of calculations. For example, the acceleration at time m can be calculated with the jerk at m and the acceleration at $m-1$. The calculation can be regarded as an inversed procedure of Figure 3.2. However, unlike in Figure 3.2 where the entire speed, acceleration, and jerk profiles can be calculated once the position profile is known, the initial values of the acceleration, speed, and position profiles are still unknown when calculating them by integrating the jerk profile, as shown by the hollow nodes in Figure 3.4. Since errors may exist in the current initial values, we need to modify the initial values of the acceleration, speed, and position profiles.

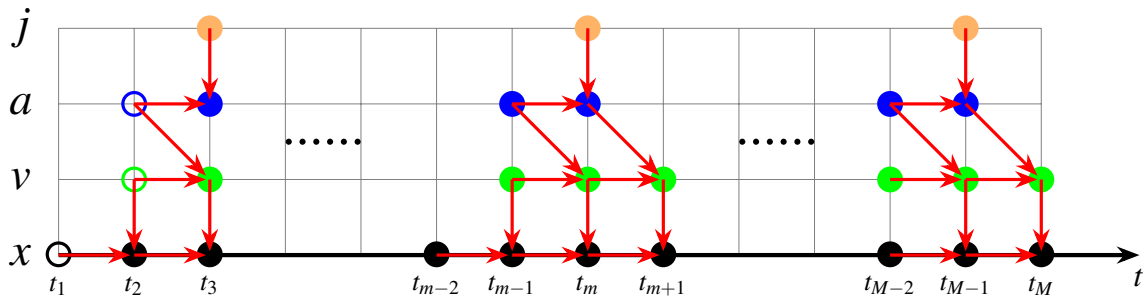


Figure 3.4: Illustration of initial values in the integration

As pointed out by P3, we should minimize the MAE between the smoothed and the raw positions and preserve the main information in the raw data. Correspondingly, the MAEs between the acceleration and speed profiles before and after integration should also be modest. To obtain the best initial acceleration, we formulate an optimization problem to minimize the MAE between the integral of the jerk profile and the acceleration profile at layer 2 ($a^{(2)}$) with the objective function as follows:

$$E(|a^{(3)} - a^{(2)}|) = \frac{1}{M-2}\sum_{m=2}^{M-1}|a_m^{(3)} - a_m^{(2)}| = \frac{1}{M-2}\sum_{m=2}^{M-1}|a_2^{(3)} + \Delta t\sum_{l=3}^{m} j_l^{(3)} - a_m^{(2)}|, \tag{3.7}$$

where $a^{(2)}$ and $j^{(3)}$ are the outputs of 'acceleration and jerk smoothing'. Thus, the only unknown variable is $a_2^{(3)}$. We choose the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm (Fletcher, 2013) for solving the optimization problem, and $a_2^{(2)}$ as the initial guess of $a_2^{(3)}$.

With $a_2^{(3)}$, as well as the entire jerk profile ($j^{(3)}$), we can calculate the entire acceleration profile $a^{(3)}$. After obtaining the entire acceleration profile, the same procedure can be applied to calculate the speed profile. From Figure 3.1 we can see that the objective is to find out $v_2^{(3)}$ which leads to the minimum MAE between the integral of $a^{(3)}$ and $v^{(1)}$. At this point, we have finished one iteration of the method.

3.3 Method iteration and choice of smoothing method and parameters

In this section, we first introduce how our method iterates. We then determine the best method for smoothing the acceleration and jerk profiles, as well as the best parameter settings in the Gaussian filter.

3.3.1 Method iteration

After we integrate accelerations to speeds, as introduced in section 2.4, we check whether all speeds, accelerations, and jerks are physically meaningful. The physically meaningful ranges of speeds, accelerations, and jerks are set as follows: 1. speeds should be greater than zero and not significantly exceed the speed limit. 2. Taking references of previous experiment, the range for accelerations is set as $-5\ m/s^2$ to $4\ m/s^2$ (Elert, 2012; Bokare and Maurya, 2017). 3. The range for jerks is set as $-8\ m/s^3$ to $8\ m/s^3$. If there are still outliers, as Figure 3.1 shows, we take the current speed profile as the new $v^{(0)}$ and conduct another iteration to fine-tune the results.

When all speeds, accelerations, and jerks are physically meaningful, we consider that we have completed all the calculations in the dashed black box in Figure 3.1, and we can calculate the ultimate output of our method. We integrate the speed profile back to the position profile according to (3.6c), and calculate the initial position that leads to the smallest $E(|x^{(3)} - x^{(0)}|)$. So far, the smoothed vehicle trajectory is obtained. Therefore, with the choice of the Gaussian filter for smoothing the accelerations and jerks, the final flow chart of our method can be plotted as Figure 3.5.

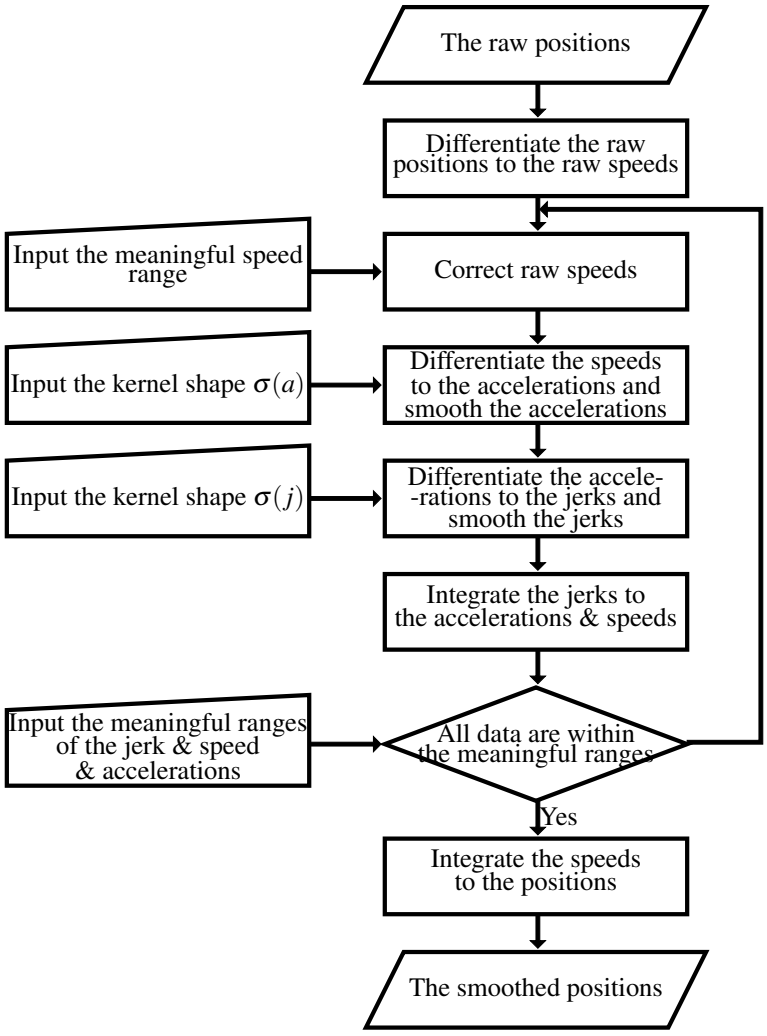


Figure 3.5: The final flow chart of the proposed method

3.3.2 Choice of parameters in the Gaussian filter

The parameters that need careful tuning are the standard derivations in the Gaussian filter for smoothing accelerations and jerks ($\sigma(a)$ and $\sigma(j)$). If we choose too small $\sigma(a)$ and $\sigma(j)$, the method will go through a large number of iterations, which significantly increases the computation cost and may also result in an increase in position errors following numerous integral calculations; on the other hand, if we choose too large standard derivations, the method can be completed in a single iteration, but the local information in the raw data may be overlooked. Except for the first

iteration, the subsequent iterations are mainly used to fine-tune the values if the speed, acceleration, or jerk profiles still contain outliers after the first iteration. Therefore, we set both $\sigma(a)$ and $\sigma(j)$ to be 1 time-step in the subsequent iterations, and we will further discuss the parameter settings in the first iteration.

As pointed out by P3, the MAE in the raw positions should be kept to a minimum. To determine the best combination of $\sigma(a)$ and $\sigma(j)$, we choose the MAE between the raw and the smoothed positions as the evaluation metric, and it can be calculated as follows:

$$E(|x^{(3)} - x^{(0)}|; \sigma(a), \sigma(j)) = \frac{\sum_{m=1}^{M} |x_m^{(3)} - x_m^{(0)}|}{M}, \tag{3.8}$$

where $I = T/\Delta t - 1$ and T is the recording duration. Based on this, we develop a grid-search algorithm to determine $\sigma(a)$ and $\sigma(j)$. The flow chart of the algorithm is shown as Figure 3.6, and detailed processes will follow.

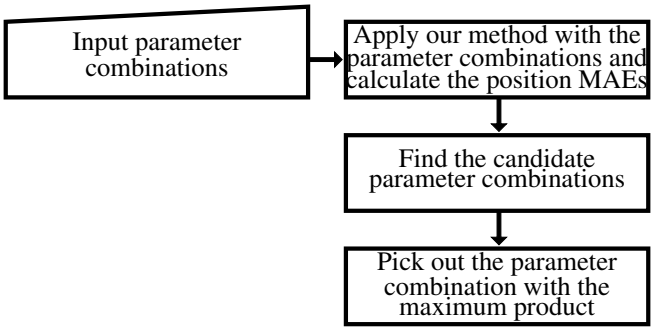


Figure 3.6: Process for determining standard derivations in the Gaussian filter

1. Investigate $\sigma(a)$ and $\sigma(j)$ values ranging from 1 to 10 time-steps at 1 time-step intervals (100 combinations), apply our method, and calculate the corresponding MAEs between the raw and the smoothed positions. This step returns 100 MAEs.

2. Pick out all the combinations of $\sigma(a)$ and $\sigma(j)$ which can satisfy $\varphi(x; \sigma(a), \sigma(j)) < \varepsilon$ as candidate parameter combinations, where ε is a small number representing the acceptable threshold, $\varphi(x; \sigma(a), \sigma(j)) = |E(|x^{(3)} - x^{(0)}|; \sigma(a), \sigma(j)) - \min E(|x^{(3)} - x^{(0)}|)|$ and

$\min E(|x^{(3)} - x^{(0)}|)$ is the minimum MAE among the 100 values.

3. Among the candidate parameter combinations, choose the one with the largest $\sigma(a) \cdot \sigma(j)$. If two combinations lead to the same $\sigma(a) \cdot \sigma(j)$, choose the one with the larger $\sigma(a)$.

At this point, we have completed the choice of $\sigma(a)$ and $\sigma(j)$. In summary, step 1 calculates MAEs for all possible parameter combinations, step 2 ensures that the modification to the raw positions retains a modest MAE between the raw and the smoothed positions, and step 3 ensures the quickest convergence.

3.4 Calibration and validation with the NGSIM data

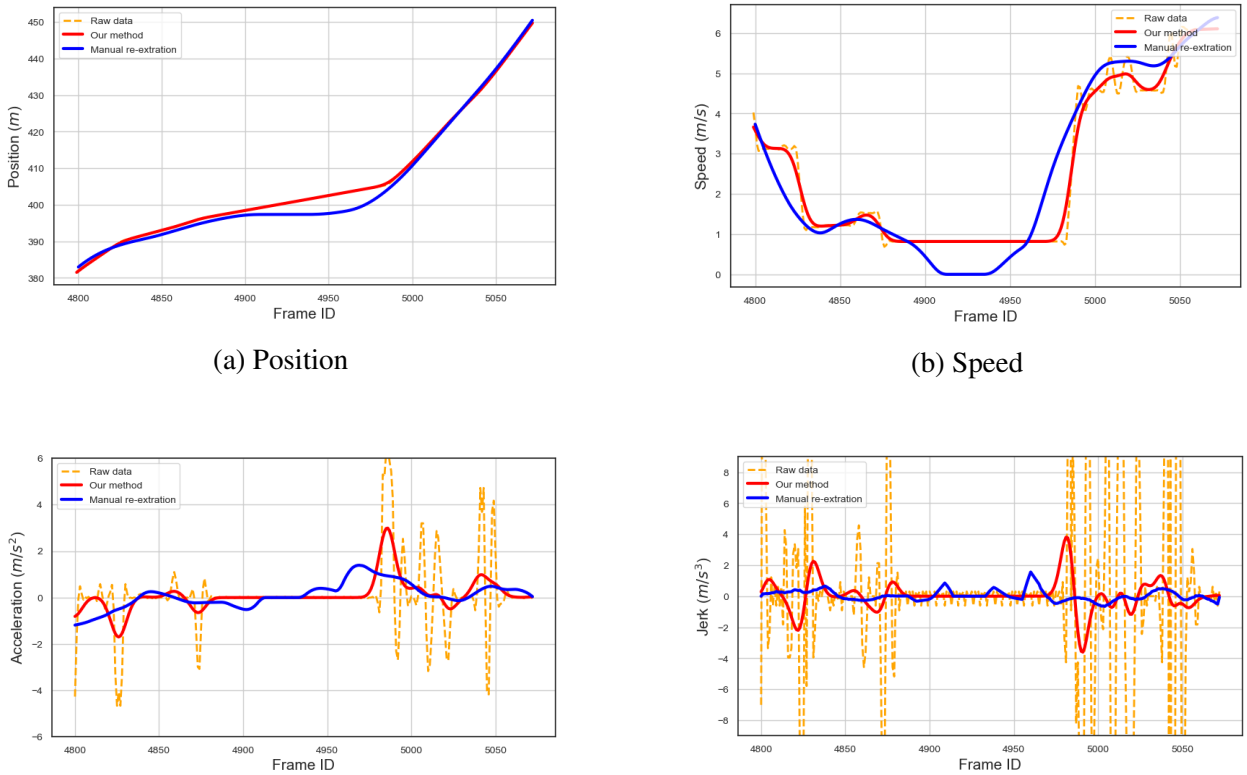
In this section, we apply our method to the NGSIM dataset. We first demonstrate the efficacy of our method upon a sample trajectory. We then compare our method with an existing method and validate our method upon the manually re-extracted freeway trajectories.

3.4.1 Calibration with a sample trajectory

We first choose vehicle 1486 from the NGSIM I80 dataset as a representation of vehicles. In this example, we can find many outliers and high-frequency noises in acceleration and jerk profiles. Because the speed limit on the I80 freeway is 65 *mph* (28.9 m/s), we consider the feasible speed ranges to be 0 to 30 m/s. The ranges of accelerations and jerks are set as $[-5,4]$ m/s^2 and $[-8,8]$ m/s^3. According our parameter choice algorithm, $\sigma(a) = 3$ time-steps (0.3 seconds) and $\sigma(j) = 2$ time-steps (0.2 seconds) are adopted, which can maintain $E(|x^{(3)} - x^{(0)}|)$ at 0.122 meters.

With the above settings, we show how our method can smooth and correct the trajectories of the two vehicles in Figure 3.7. The pre- and post-processed position profiles are displayed in Figure 3.7a, in

which the final positions are nearly identical to the raw positions. The speed profiles are displayed as Figure 3.7b, with the orange dashed, red, and blue curves representing the raw, smoothed, and manually re-extracted speed profiles. The smoothed and manually re-extracted speed profiles retain the majority of the original characteristics while greatly smoothing out the rapid oscillations in the raw data. However, the stop-and-go behavior shown in the manually re-extracted speed is not shown in the smoothed data. The acceleration profiles are shown as Figure 3.7c. As indicated by the orange dashed curves, the raw acceleration profiles contain a large number of outliers and noises and thus require smoothing. Smoothing can significantly reduce outliers and noises in the acceleration profiles, making all accelerations physically meaningful. Figure 3.7d displays the jerk profiles. As can be seen, the raw jerk profiles contain many extreme values, all of which are removed after smoothing. The final jerk profiles are free of quick oscillations while retaining the main information in the raw data.



(a) Position

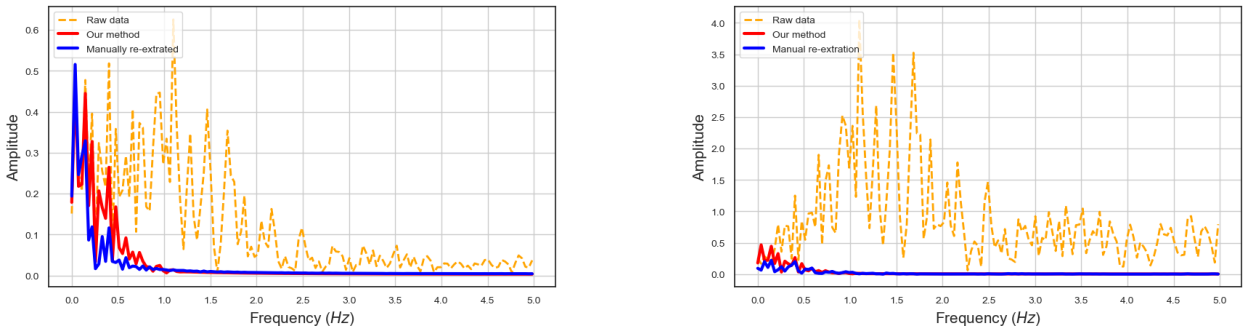
(b) Speed

(c) Acceleration

(d) Jerk

Figure 3.7: Smoothed positions, speeds, accelerations, and jerks of vehicle 1486

The frequency spectrums of the acceleration and jerk profiles before and after processing are displayed as Figure 3.8. It can be observed that for both vehicles, high-frequency values which are out of the human operational range (up to 2 Hz) (Punzo et al., 2011) are removed. This is in line with the aforementioned finding that the final acceleration and jerk profiles are free of quick oscillations.



(a) Acceleration (b) Jerk

Figure 3.8: Frequency spectrums of accelerations and jerks of vehicle 1486

3.4.2 Comparison with an existing method and validation with manually re-extracted freeway trajectories

Some people have smoothed the NGSIM I80-1 dataset, and made the smoothed data publicly accessible, providing a good benchmark for data smoothing and correction. We also apply our method to this dataset for further comparison. The speed range is considered to be $[0,30]$ m/s, and other ranges are the same as those in section 3.4.1. Because all trajectories in this dataset were obtained using the same sensors and recognition technology, their error patterns should be comparable. Considering that applying the parameter choice algorithm to each individual requires high computation cost, we apply the method to 30% randomly-selected vehicles at the site and calculate the average MAE between the raw and the smoothed positions. We choose $\sigma(a) = 4$ time-steps (0.4 seconds) and $\sigma(j) = 2$ time-step (0.2 seconds) according to the results of our parameter choice algorithm, and the average MAE between the raw and the smoothed positions is

0.1 meters.

As a pioneer in vehicle trajectory data smoothing, (Montanino and Punzo, 2013, 2015) proposed a well-known multistep optimization method and revealed smoothed trajectory data. More recently, (Coifman and Li, 2017a) manually re-extracted the vehicle trajectories from the record of camera 6 at the I80 site, and recalculated speeds with the re-extracted data taking a separate instrumented probe vehicle study under similar traffic conditions as the reference (Coifman et al., 2016). From the presented video (Coifman and Li, 2017b) we can find that many erroneous recognitions are revised. The re-extracted trajectories are closer to the ground truth, and their released dataset has been used as the 'ground truth' NGSIM data (Dong et al., 2021). We follow the suggestion in (Coifman and Li, 2017a) and recalculate the positions, accelerations, and jerks with the new speeds, and obtain the control group. For the raw trajectories, as well as the trajectories smoothed by us and (Montanino and Punzo, 2013, 2015), we then pick out the trajectories in the region recorded by camera 6, as the pink area in Figure 3.9 shows. There are around 150,000 vehicle·frame in total.

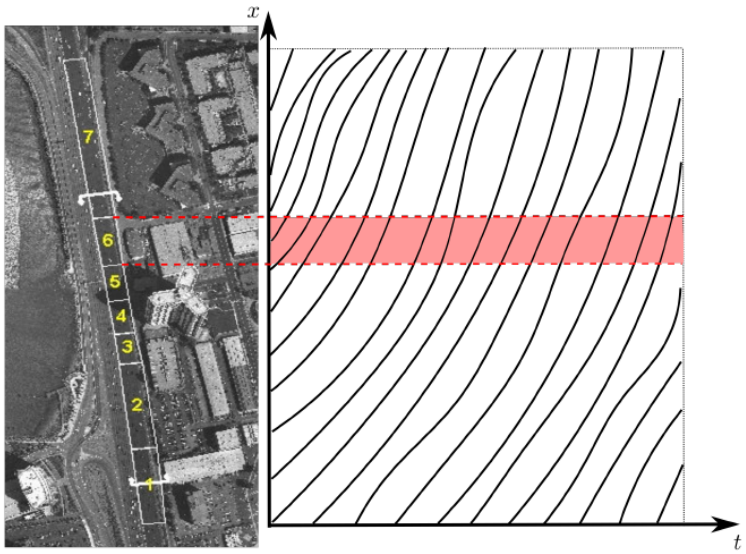
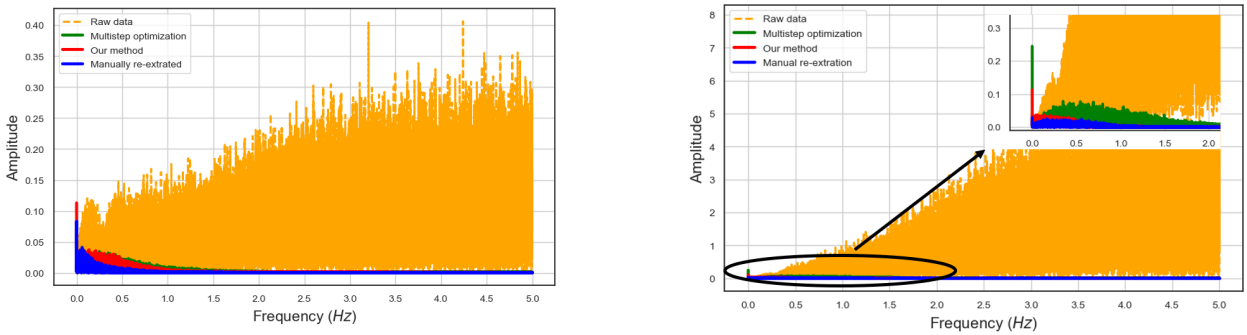


Figure 3.9: Illustration of capturing the trajectories on the I80 freeway recorded by camera 6

Figure 3.10a and Figure 3.10b display the comparison of the frequency spectrums of the acceleration and jerk profiles. As shown by cyan dashed lines, noises contained in the original signals spans the entire frequency range. All methods can eliminate a large number of noises and make

frequency spectrums (purple, red and black curves) within the range of the human-vehicle response frequency (up to 2 Hz).



(a) Acceleration

(b) Jerk

Figure 3.10: Frequency spectrums of accelerations (i) and jerks (j) obtained with different approaches

We compute the mean, standard deviation, range, and proportion of outliers in the speed, acceleration, and jerk profiles of the aforementioned four datasets. We also compare the raw data, as well as the smoothed data obtained with the multistep optimization method and our method, with the data obtained with the manual re-extraction. The mean squared error (MSE) and the mean absolute error (MAE) are chosen as the indicators. For speed profiles, the three indicators can be calculated through (3.9) to (3.10), and likewise for the acceleration and jerk profiles:

$$E((\bar{v}-v^{(3)})^2) = \frac{1}{M-1}\sum_{i=2}^{M}(\bar{v}_m - v_m^{(3)})^2, \tag{3.9}$$

$$E(|\bar{v}-v^{(3)}|) = \frac{1}{M-1}\sum_{i=2}^{M}|\bar{v}_m - v_m^{(3)}|, \tag{3.10}$$

where $\bar{v}$ is the manually re-extracted speed profile and $v^{(3)}$ denotes the speed profile to be compared with $v^{(3)}$, which in this case are the raw speed profile, the speed profile smoothed via (Montanino

and Punzo, 2015), and the speed profiles smoothed using our method.

Results are presented in **Table** 3.1. Means of speeds, accelerations, and jerks are preserved by all methods. Outliers in the raw acceleration and jerk profiles account for 15% and 42.3% of total data. Both our method and the manual re-extraction eliminate these outliers. The multistep optimization method reduces the outliers in accelerations and jerks to less than 0.05% and 0.6%, respectively. As illustrated by the MSE and MAE, the difference between the raw speeds and manually re-extracted speeds is not significant. However, the discrepancy is significantly amplified in the acceleration and jerk profiles, with the MSE between the raw and manually re-extracted accelerations and jerks being $44.56 m^2/s^4$ and $9960.09 m^2/s^6$. Both the multistep optimization method and our method can greatly reduce such discrepancies. From the perspectives of all three indicators, the speeds, accelerations, and jerks smoothed by our method are closer to manually re-extracted ones.

Table 3.1: Comparison of different approaches

variables		Raw data	Multistep optimization	Our method	Manual re-extraction
Speed (m/s)	Mean	8.07	8.05	8.07	7.88
	Std	4.07	4.01	4.0	3.89
	Range	[0,36.0]	[0,27.0]	[0,27.1]	[0,26.4]
	Outliers (%)	0	0	0	0
	Comparison to $E((\bar{v}-v^{(3)})^2)$	0.88	0.47	0.41	/
	re-extracted data $E(\lvert\bar{v}-v^{(3)}\rvert)$	0.62	0.48	0.46	/
Acceleration (m/s^2)	Mean	-0.04	-0.04	-0.06	0.03
	Std	6.69	0.92	0.79	0.58
	Range	[-176.5,292.2]	[-14.1,4.5]	[-5,4]	[-4.2,3.5]
	Outliers (%)	15	0	0	0
	Comparison to $E((\bar{a}-a^{(3)})^2)$	44.56	0.68	0.49	/
	re-extracted data $E(\lvert\bar{a}-a^{(3)}\rvert)$	2.64	0.6	0.49	/
Jerk (m/s^3)	Mean	-0.17	-0.13	-0.06	-0.02
	Std	99.80	2.41	1.34	0.62
	Range	[-4171.5,2954.4]	[-141.0,39.4]	[-8,8]	[-5.3,7.0]

Outliers (%)		42.3	0.6	0	0
Comparison to	$E((\bar{j}-j^{(3)})^2)$	9960.09	5.9	1.98	/
re-extracted data	$E(\lvert\bar{j}-j^{(3)}\rvert)$	32.79	1.65	0.98	/

3.5 Conclusion

In this paper, we introduce three first principles for smoothing and correcting longitudinal vehicle trajectories: (P1) The internal consistency among derivatives at different orders should be guaranteed, satisfying both differentiation and integration relations. (P2) All speeds, accelerations, and jerks should be bounded within physically meaningful ranges. (P3) The MAE in the raw positions should be kept to a minimum, thus the modifications of the raw positions should be kept to a minimum. Based on these principles, we propose an iterative method to smooth longitudinal vehicle trajectories which are detected at fixed study sites and extracted at high sampling frequencies. One iteration comprises four types of calculations: differentiation, correction, smoothing, and integration. In differentiation, we compute speeds, accelerations, and jerks from trajectory data; in correction, we eliminate outliers in speeds, especially negative values; in smoothing, we reduce noises in accelerations and jerks; and in integration, we recalculate accelerations, speeds, and positions from jerks, and find the initial values via optimization problems. Our design uses the adaptive average method for correction, the Gaussian filter for smoothing, and minimizing the MAEs as the optimization objective in integration. We then numerically demonstrate the efficacy of our method upon the NGSIM data. We select a sample trajectory from the NGSIM I80 dataset. The comparison of the speed, acceleration, and jerk profiles before and after smoothing demonstrates that all outliers and rapid oscillations are eliminated, while the main information in the raw data, including cruising, accelerating, decelerating, and stopping behaviors, are retained. The frequency spectrums of the acceleration and jerk profiles also show that high-frequency noises are removed. We finally compare our method with an existing method and validate our method

upon the manually re-extracted trajectories. The raw accelerations and jerks include a significant number of outliers, accounting for 15% and 42.3% of total data, respectively. Both the multistep optimization method and our method have good performance in removing outliers and reducing high-frequency noises. According to the histograms and statistic summary, the speed, acceleration, and jerk profiles smoothed using our method are closer to those manually re-extracted and can be deemed more accurate.

Starting with the first principles, we propose this method to smooth vehicle trajectories. It should also be noted that for smoothing the acceleration and jerk profiles, apart from the alternatives we presented in this paper, our method is also open to other options. In this research, we propose a general framework for smoothing longitudinal vehicle trajectories, and more advanced strategies can be developed based on this.

The limitations and potential extensions of our method are as follows. Our method may require multiple iterations, with P2 serving as the terminating criteria for iterations. However, it is mathematically challenging to demonstrate when the iterations converge or even that the iterations can converge. This study can serve as a starting point for the development of more mathematically tractable techniques that can either be proved to converge or get rid of iterations. In addition, with the accessibility of data collected in different ways, we also plan to test the proposed method with data from other fixed-location sensors, including highD, inD, and rounD data which are collected with drones (Krajewski et al., 2018; Bock et al., 2020; Krajewski et al., 2020) and Lidar data (Chang et al., 2019; Schwall et al., 2020). Furthermore, the ideas used in this method can be extended to investigate the trajectories including missing values, and impute for the missing values.

Chapter 4

Simplified iterative moving average method for smoothing

4.1 Introduction

The aim of this study is to propose a simplified and straightforward iterative moving average method that guarantees physically meaningful ranges for the smoothed speeds, acceleration rates, and jerks of a vehicle's trajectory while preserving its average speeds or total travel distance over a specified time duration. In each iteration, our method consists of three steps. Firstly, we pad the speed profile with the average speed and calculate the speed deviations. Secondly, we apply a moving average method with different kernel shapes, representing different weights, to speed deviations. Finally, we normalize the filtered speed deviations and add the average speed back to obtain the output of one iteration. The iterative method terminates when the speeds, acceleration rates, and jerks are within reasonable bounds. Our method is still straightforward to implement. The differentiation of speeds, acceleration rates, and jerks from the positions is done consistently, thus avoiding any inconsistency issues. In addition, we mathematically prove that the speed be-

comes constant after an infinite number of iterations without termination; this property ensures that the bounds in speeds and their derivatives are physically meaningful with the corresponding termination criterion. We empirically demonstrate the efficacy of our proposed method by applying it to smooth all 1714 trajectories recorded by camera 6 in the NGSIM I80-1 dataset. We compare our smoothed trajectories with the raw trajectories, those obtained from the multistep optimization method proposed in (Montanino and Punzo, 2015), and manually re-extracted trajectories (Coifman and Li, 2017a).

Traditionally, when it comes to handling padding or boundaries in convolutional filtering methods, there are various approaches. These include introducing constant (including zero) or symmetric values outside the two boundaries of a time series (Smith, 1997; Getreuer, 2013). For our study, we have opted for padding with the average speed. This approach guarantees that our method can terminate within a finite number of iterations. During each iteration, after applying the padding and filtering the corresponding speeds, we proceed to normalize the resulting speeds. This normalization ensures that the average speeds remain consistent, thereby maintaining the same total travel distance throughout the study period. We will numerically demonstrate the impacts of these two features on the accuracy and efficiency of the proposed method.

To the best of our knowledge, the iterative moving average method has not been applied to smooth vehicle trajectories. However, there are a few related studies. First, the method was applied to smooth images iteratively in (Wu et al., 2015), using the Savitzky-Golay (SG) filter. The iterations continued until the relative difference between the original and smoothed signals reached a termination criterion, which is arbitrarily chosen without physical interpretations. While the method's convergence was empirically demonstrated, it lacked theoretical proof. In addition, there are no discussions on normalization in each iteration. In contrast, our approach employs a more physically meaningful termination criterion based on the bounds of speeds and their derivatives; and normalization is incorporated in each iteration. We also provide a mathematical proof of convergence to ensure physically meaningful derivatives. Second, the iterative moving average

method was used in empirical mode decomposition (EMD) for nonstationary nonlinear signals, as discussed in (Huang et al., 1998). This involved iteratively sifting individual Intrinsic Mode Functions (IMFs) to obtain an invariant remaining signal. Lin et al. (2009) improved the traditional sifting algorithm by introducing convolutional Toeplitz filters and mathematically proved its convergence. However, these EMD methods iteratively process the remaining signal, sifting out higher frequency modes first. This approach is less efficient compared to our iterative moving average method, which directly smooths the original signal (vehicle speeds). Although Dong et al. (2021) used EMD to smooth vehicle trajectories by discarding high-frequency modes, it is not guaranteed that the resulting trajectory is sufficiently smooth. Therefore, they resorted to further employing a frequency-domain Butterworth filter. However, there is still no guarantee that speeds and their derivatives remain within physically meaningful bounds in the smoothed trajectories. In contrast, our simple iterative moving average method does not suffer from these limitations and ensures both smoothness and physically meaningful bounds in the smoothed vehicle trajectories.

The rest of this chapter is organized as follows. In section 3.2, we introduce our method for correcting and smoothing vehicle trajectories, as well as prove that our method can terminate within a finite number of iterations. In section 3.3, we introduce the algorithm for choosing the parameters of the filter, apply our method to smooth the vehicle trajectory data recorded in the NGSIM dataset, and compare our smoothed data with an existing method with respect to the manually re-extracted data. Finally, we conclude our study in section 3.4 and propose some future extensions.

4.2 An iterative moving average method for smoothing

In this section, we propose an iterative moving average method for correcting and smoothing longitudinal vehicle trajectories. We first present the framework of the calculations in our methods and introduce each calculation in detail.

4.2.1 Flow chart of the method

We illustrate the iterative process as Figure 4.1. We first differentiate the raw positions to raw speeds, calculate the speed deviations, and launch the iterations. In one iteration, we smooth the speeds, and the smoothing process involves three steps: zero padding the speed deviation profile, applying moving average filters, and normalization. After smoothing the speeds, we differentiate the smoothed speeds into accelerations and jerks and check whether all speed deviations, accelerations, and jerks are within the physically meaningful ranges. If there are still outliers, we take the current speed profile as the new input and conduct another iteration for tuning the results.

Only when all speeds, accelerations, and jerks are physically meaningful, we consider that we have completed all the calculations for smoothing, and the smoothed data can satisfy the termination condition. We then add the average speed back to the smoothed speed deviations, calculate the ultimate output of our method by integrating the speeds back into the positions, and calculate the initial position by formulating an optimization problem. So far, the smoothed positions have been obtained.

Before launching the iterations, we calculate the raw speed from positions as follows:

$$v_m = \frac{x_m - x_{m-1}}{\Delta t}, \;\; m = 2, 3, ..., M, \tag{4.1}$$

where M is the total number of detected positions, $M > 3$ and M is a finite number in our case. We construct the vector of speed deviation u corresponding to the raw speed v. We place the initial speed $x_2^{(1)}$ at the origin, and the speed deviations can be calculated as follows:

$$u_{m-2} = v_m - E(v), \;\; m = 2, 3, \cdot, M. \tag{4.2}$$

We then launch the iterative process, and a detailed introduction to all steps in each iteration will be given.

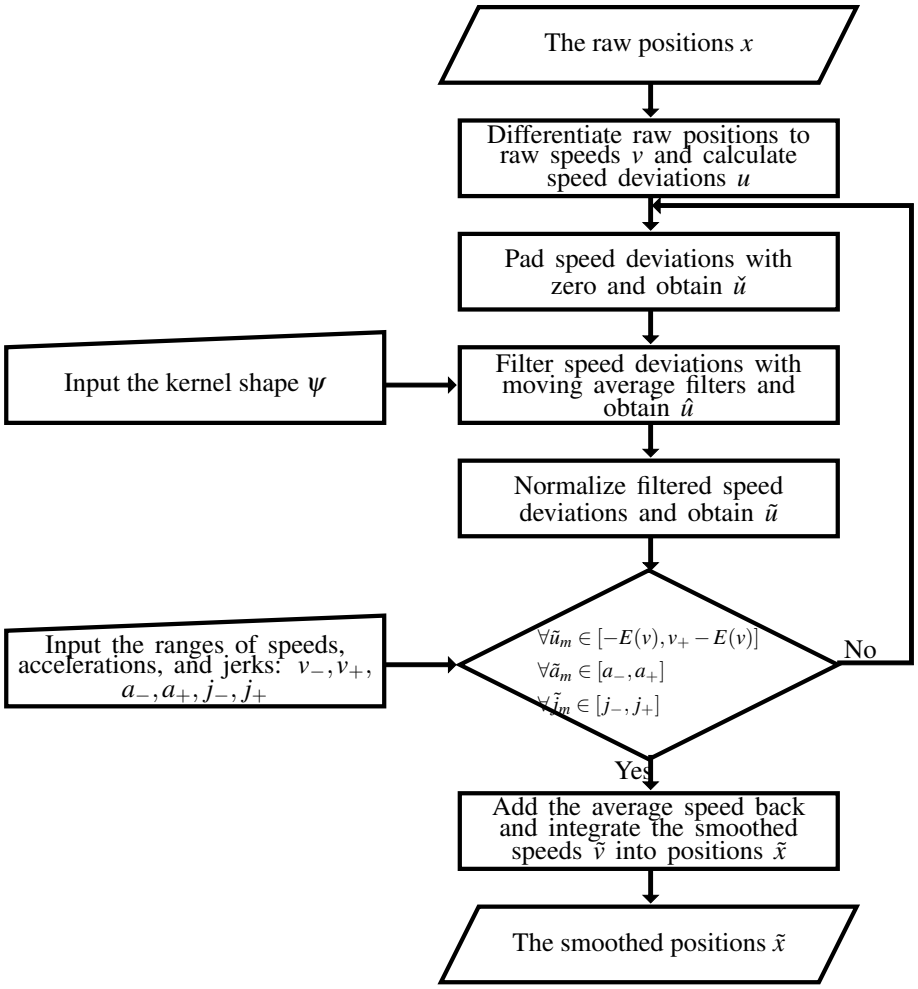


Figure 4.1: Flow chart of the proposed method

4.2.2 Speed smoothing

Following the empirical observations, speeds, accelerations, and jerks should be bounded within physically meaningful ranges (Jin, 2021). After calculating raw speed deviations according to (4.2), we eliminate the outliers, the values outside the physically meaningful ranges, and reduce high-frequency noises in speeds, accelerations, and jerks. In this regard, we propose the utilization of the moving average filters.

Cyclic mean padding

Time domain moving average filters apply a boundary extension to handle the boundary of the input signal, and the filtering process is conducted on the extended signal (Getreuer, 2013). The boundary extension is usually done using a padding technique, which involves adding extra values to the boundaries of the input signal. We pad the speed profile with the average speed and consider that the padded speeds represent one cycle of a loop that repeats forever (Elliott, 2013). This is equivalent to smoothing speed deviations u with zero padding at the boundary and consider the padded speed deviations represent one cycle of a loop that repeats forever. The padded signal of u can be written as follows:

$$\breve{u} = (\cdots, 0, \cdots, 0, u_0, u_1, \cdots, u_{M-2}, 0, \cdots, 0, u_0, u_1, \cdots, u_{M-2}, 0, \cdots 0, \cdots), \tag{4.3}$$

$\breve{u}$ satisfies the following relation,

$$\breve{u}(m+M-2+w) = \breve{u}(m). \tag{4.4}$$

Applying moving average filters

After padding the signal, we smooth the signal with time-domain moving average filters. We use $\psi(l)$ to denote the kernel function, where l denotes the variable and it represents the distance between the calculated point and the center point.

We first consider the simple moving average filter, which is an intuitive and easy-to-implement method for smoothing signals. It is a simple sliding-window spatial filter that replaces the center value in the window with the mean of all the values within the window (Smith, 2013). Considering

we are going to smooth u_m, the kernel can be written as follows:

$$\phi^s(l) = \frac{1}{w}, \;\; l = -\frac{w-1}{2}, 1-\frac{w-1}{2}, \cdots, \frac{w-1}{2}, \tag{4.5}$$

where w is the window size. For the simple moving average filter, truncation isn't necessary, yielding a truncated kernel identical to the original ($\psi^s(l) = \phi^s(l)$).

In addition to the simple moving average filter, we explore weighted moving average kernels. These kernels share the same concept but assign greater weight to center values. They compute target values through weighted averages of all the values within the window. We consider exponential, triangular, and Gaussian kernels. The kernel functions and the corresponding truncated kernels can be written as follows:

- Exponential kernel

$$\phi^e(l) = e^{-|l|/\tau}, \;\; l = \frac{w-1}{2}, 1-\frac{w-1}{2}, \cdots, \frac{w-1}{2}, \tag{4.6}$$

  where τ is the time constant of the function. Because we consider a symmetric kernel for the filters, w should be odd. We normalize the exponential kernel at discrete time points as (4.7):

$$\psi^e(l) = \frac{\phi^e(l)}{\sum_{\alpha=-\frac{w-1}{2}}^{\frac{w-1}{2}} \phi^e(\alpha)}. \tag{4.7}$$

- Triangular kernel

$$\phi^t(l) = 1 - \frac{2}{w+1}|l|, \;\; l = -\frac{w-1}{2}, 1-\frac{w-1}{2}, \cdots, \frac{w-1}{2}. \tag{4.8}$$

We normalize the triangular kernel at discrete time points the same way as (4.7):

$$\psi^t(l) = \frac{\phi^t(l)}{\sum_{\alpha=-\frac{w-1}{2}}^{\frac{w-1}{2}} \phi^t(\alpha)}. \tag{4.9}$$

- Gaussian kernel

$$\phi^g(l) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{l^2}{2\sigma^2}}, \tag{4.10}$$

where σ is the standard deviation (Romeny, 2008). With (4.10) the weights of adjacent values near m can be calculated. A true Gaussian response would have infinite impulse responses and they decay rapidly, thus we truncate the Gaussian kernel at discrete time points as follows:

$$\psi^g(l) = \frac{\phi_l}{\sum_{\alpha=-p\sigma}^{p\sigma} \phi^g(\alpha)}, \tag{4.11}$$

where p represents that we smooth the value at m with the values from $m - p\sigma$ to $m + p\sigma$, $2p\sigma + 1$ values in total. We choose $p = 4$ where the amplitude of the Gaussian function is about 3×10^{-4} of that at m, thus the kernel size of the Gaussian filter can be written as $8\sigma + 1$, that is, $w = 8\sigma + 1$.

We use t_m to denote the time instant of the target value we want to smooth. With (4.5), as well as (4.7) and (4.11), the weights of the neighboring values near m can be calculated, and the smoothed speed deviation can be written as follows:

$$\hat{u}_m = \sum_{l=m-\frac{w-1}{2}}^{m+\frac{w-1}{2}} \psi(l-m)\check{u}_l, \; m = 0, 1, \cdots, M-2, \tag{4.12}$$

where $\psi(*)$ represents the truncated kernel of the filters, here we omit the superscript. The kernels

under consideration in this study must meet the following criteria: firstly, the weights must be non-negative; secondly, the sum of the weights within the window must equal one; thirdly, the kernel should exhibit symmetry around its center, meaning the window size should be odd; and additionally, the weight must be non-increasing as the distance to the center increases.

In this study, we apply the moving average via circular convolution. We consider kernel function span $m-\frac{w-1}{2}$ to $m+\frac{w-1}{2}$, pad the kernel function by zeros and let it spam from $-\frac{w-1}{2}$ to $M-2+\frac{w-1}{2}$, and consider the padded kernel represents one cycle of a loop that repeats forever. That is (Elliott, 2013):

$$\check{\psi}(m+M-2+w) = \check{\psi}(m). \tag{4.13}$$

We then move the kernel function along the positive direction of the x-axis, the circular convolution of the signal $\check{u}$ and the padded kernel function $\check{\psi}$ can be written as follows:

$$\{\check{\psi} * \check{u}\}(m) = \sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \check{\psi}(m-l)\check{u}_l, \;\; m = -\frac{w-1}{2} \cdots M-2+\frac{w-1}{2}. \tag{4.14}$$

In this case, the convolution result will be a $(M-2+w)-$point sequence.

Lemma 4.1. *With symmetric filter kernels, the filtered signal, which was calculated via (4.12), can be written as follows:*

$$\hat{u}_m = \{\check{\psi} * \check{u}\}(m), \;\; m = 0, 1, \cdots, M-2, \tag{4.15}$$

where there are $M-1$ *speed deviation values.*

Proof. The kernel $\check{\psi}$ for smoothing speed deviation $\check{u}_m$ can be conceptualized as a $(M-2+w)-$point sequence spanning from $-\frac{w-1}{2}$ to $M-2+\frac{w-1}{2}$, with all points except those within

the range of $m-\frac{w-1}{2}$ to $m+\frac{w-1}{2}$ equal zero. Therefore, (4.12) can be written as follows:

$$\begin{aligned}\sum_{l=m-\frac{w-1}{2}}^{m+\frac{w-1}{2}} \psi(l-m)\breve{u}_l &= \sum_{l=m-\frac{w-1}{2}}^{m+\frac{w-1}{2}} \breve{\psi}(l-m)\breve{u}_l \\ &= \sum_{l=-\frac{w-1}{2}}^{m-\frac{w-1}{2}-1} \breve{\psi}(l-m)\breve{u}_l + \sum_{l=m-\frac{w-1}{2}}^{m+\frac{w-1}{2}} \breve{\psi}(l-m)\breve{u}_l + \sum_{l=m+\frac{w-1}{2}+1}^{M-2+\frac{w-1}{2}} \breve{\psi}(l-m)\breve{u}_l\end{aligned}$$

As it has been mentioned that the filter kernel should be symmetric around its center, ψ is an even function. Therefore, assuming l is the variable, when $l=-\frac{w-1}{2},-\frac{w-1}{2}+1,\cdots,\frac{w-1}{2}$, we have the following relationship:

$$\breve{\psi}(l)=\breve{\psi}(-l).$$

Thus the equation can be rewritten as follows:

$$\begin{aligned}\sum_{l=m-\frac{w-1}{2}}^{m+\frac{w-1}{2}} \breve{\psi}(l-m)\breve{u}_l &= \sum_{l=2m-M-\frac{w-1}{2}+2}^{m-\frac{w-1}{2}-1} \psi(m-l)\breve{u}_l + \sum_{l=m-\frac{w-1}{2}}^{m+\frac{w-1}{2}} \psi(m-l)\breve{u}_l + \sum_{l=m+\frac{w-1}{2}+1}^{\frac{w-1}{2}+2m} \psi(m-l)\breve{u}_l. \\ &= \sum_{l=2m-M-\frac{w-1}{2}+2}^{2m+\frac{w-1}{2}} \psi(m-l)\breve{u}_l.\end{aligned}$$

In addition, because both $\breve{\psi}$ and $\breve{u}_m$ are periodic with the periodicity $M+w-2$, the equation can be rewritten as:

$$\sum_{l=m-\frac{w-1}{2}}^{m+\frac{w-1}{2}} \breve{\psi}(l-m)\breve{u}_l = \sum_{l=2m-M-\frac{w-1}{2}+2}^{\frac{w-1}{2}+2m} \psi(m-l)\breve{u}_l = \sum_{l=-\frac{w-1}{2}}^{\frac{w-1}{2}+2m} \psi(m-l)\breve{u}_l = \{\breve{\psi}*\breve{u}\}(m).$$

Let $m==0,1,\cdots,M-2$ and (4.15) is proved. □

By applying the filters throughout the entire signal, in our case, the speed deviations, we can smooth out the outliers and high-frequency noises in speeds, accelerations, and jerks.

Normalization

In the smoothing process, it is desirable to preserve the total travel distance. This equates to preserving the average speed or speed deviation.

Lemma 4.2. *With the average speed deviation after filtering being calculated as follows:*

$$E(\hat{u}) = \frac{\sum_{m=0}^{M-2}\left(\sum_{l=m-\frac{w-1}{2}}^{m+\frac{w-1}{2}} \breve{\psi}(l-m)\breve{u}_l\right)}{M-1}, \tag{4.16}$$

average speed deviation after filtering, denoted as $E(\tilde{u})$, may not be equivalent to the original average speed deviation, denoted as $E(u)$. This implies that the summation of speed deviations after filtering may not equal the summation of the original speed deviations.

Proof. The filtered speed deviations should satisfy the following relation:

$$\begin{aligned}
\sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \hat{u}_m &= \sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \left(\sum_{l=m-\frac{w-1}{2}}^{m+\frac{w-1}{2}} \breve{\psi}(l-m)\breve{u}_l\right) \\
&= \sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \left(\sum_{m=l-\frac{w-1}{2}}^{l+\frac{w-1}{2}} \breve{\psi}(l-m)\breve{u}_l\right) \\
&= \sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{u}_l \left(\sum_{m=l-\frac{w-1}{2}}^{l+\frac{w-1}{2}} \breve{\psi}(l-m)\right) = \sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{u}_l = 0.
\end{aligned}$$

With the sum of speed deviations from $t_{-\frac{w-1}{2}}$ to $t_{M-2+\frac{w-1}{2}}$ being 0, the sum of the filtered speed deviations from t_0 to t_{M-2} may not equal to 0. To illustrate this point, we consider a three-digit time series $(2,-1,-1)$. When we apply a 3-point simple moving average filter to smooth the speed deviations, we obtain the sequence $\left(\frac{2}{3},\frac{1}{3},0,-\frac{2}{3},-\frac{1}{3}\right)$. The filtered speed corresponding to the input can be expressed as $\left(\frac{1}{3},0,-\frac{2}{3}\right)$, and its sum is $-\frac{1}{3}$.

Hence, it is evident that the sum of the filtered speed deviations does not align with the sum of the original speed deviations. □

Consequently, directly integrating speeds filtered by moving average filters into positions can lead to an alteration in the total travel distance. Therefore, an additional step, termed 'normalization', is introduced in the smoothing procedure. This step involves adjusting the filtered speed deviations to ensure that the resulting average is zero. The normalized filtered speed deviations can be written as follows:

$$\tilde{u}_m = \hat{u}_m + (E(u) - E(\hat{u})) = \hat{u}_m - E(\hat{u}), \tag{4.17}$$

where $E(\hat{u})$ and $E(u)$ denote the average of the filtered and raw speed deviations, respectively.

Check for termination

We then calculate the accelerations and jerks with the smoothed speed derivations $\tilde{u}$. We still employ the symplectic discretization (mixed implicit-explicit Euler discretization) to calculate accelerations, and jerks (Jin, 2019).

Acceleration and jerk at each time instant can be calculated as follows:

$$\tilde{a}_m = \frac{\tilde{u}_{m-1} - \tilde{u}_{m-2}}{\Delta t}, \quad m = 2, 3, ..., M-2, \tag{4.18a}$$

$$\tilde{j}_m = \frac{\tilde{a}_m - \tilde{a}_{m-1}}{\Delta t} = \frac{\tilde{u}_{m-1} - 2\tilde{u}_{m-2} + \tilde{u}_{m-3}}{(\Delta t)^2}, \quad m = 3, 4, ..., M-1. \tag{4.18b}$$

We check whether all speed deviations, accelerations, and jerks are physically meaningful and within the following ranges:

$$\forall \tilde{u}_m \in [v_- - E(v), v_+ - E(v)] = [-E(v), v_+ - E(v)] \tag{4.19a}$$

$$\forall a_m \in [a_-, a_+] \tag{4.19b}$$

$$\forall j_m \in [j_-, j_+] \tag{4.19c}$$

If there are still outliers, we take $\tilde{u}$ as new u and conduct another iteration for tuning the results. Only when all speeds, accelerations, and jerks are physically meaningful, we consider that we have completed all the iterations and the smoothed data can satisfy the termination condition.

4.2.3 Integration of the smoothed speeds

After the iterations terminate, We add the average speed back to the smoothed speed deviation and calculate the smoothed speed:

$$\tilde{v}_{m+2} = \tilde{u}_m + E(v), \;\; m = 0, 1, \cdots, M-2. \tag{4.20}$$

Smoothed positions can be calculated by integrating the smoothed speeds as follows:

$$\tilde{x}_m = \tilde{x}_1 + \Delta t \sum_{l=2}^{m} \tilde{v}_l, \;\; m = 2, 3, 4, ..., M, \tag{4.21}$$

where $\tilde{x}_1$ is the initial smoothed position.

Since errors may exist in the initial detected position, after we obtain the normalized-filtered speeds, we calculated the smoothed positions, the output of our method, by formulating an optimization problem. We minimize the residuals between the integrals of smoothed speeds and raw positions, which can be calculated as follows:

$$E(|\tilde{x} - x|) = \frac{1}{M} \sum_{m=1}^{M} |\tilde{x}_m - x_m| = \frac{1}{M} \sum_{m=1}^{M} |\tilde{x}_1 + \Delta t \sum_{l=2}^{m} \tilde{v}_l - x_m|. \tag{4.22}$$

Thus, the only unknown variable is $\tilde{x}_1$. We choose the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm (Fletcher, 2013) for solving the optimization problem, and x_1 as the initial guess of $\tilde{x}_1$.

4.3 Proof of termination within a finite number of iterations

We use the simple moving average filter as an example, and other weighted moving average filters can be discussed in a similar way. We show that the iterative process in our method is guaranteed to conclude within a finite number of iterations.

4.3.1 Discrete Fourier Transform (DFT) of the kernel functions

Using $\check{U}$ to denote the discrete Fourier transform (DFT) of $\check{u}$, which can be written as follows (Oppenheim et al., 1997):

$$\check{U}_m = \sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \check{u}_l e^{-im\frac{2\pi}{M-2+w}l}, \;\; m = -\frac{w-1}{2}, -\frac{w-1}{2}+1, \cdots, M-2+\frac{w-1}{2}, \tag{4.23}$$

where $M-2+w$ is the length of $\check{U}$.

Correspondingly, the DFT of the padded kernel function can be written as follows:

$$\check{\Psi}(m) = \sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \check{\psi}(l) e^{-im\frac{2\pi}{M-2+w}l}, \;\; m = -\frac{w-1}{2}, \cdots, M-2+\frac{w-1}{2}. \tag{4.24}$$

Because the kernel function is symmetric at the origin, (4.24) can be written as follows:

$$\begin{aligned}
\check{\Psi}(m) &= \sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \check{\psi}(l) e^{-im\frac{2\pi}{M-2+w}l} \\
&= \psi(0) + \sum_{l=1}^{\frac{w-1}{2}} \check{\psi}(l) \left(e^{-im\frac{2\pi}{M-2+w}l} + e^{im\frac{2\pi}{M-2+w}l} \right) + \sum_{l=\frac{w-1}{2}+1}^{M-2+\frac{w-1}{2}} \check{\psi}(l) e^{-im\frac{2\pi}{M-2+w}l} \\
&= \check{\psi}(0) + 2\sum_{l=1}^{\frac{w-1}{2}} \check{\psi}(l) cos(m\frac{2\pi}{M-2+w}l).
\end{aligned} \tag{4.25}$$

When $m = 0$, we can easily find that $\check{\Psi}(0) = \sum_{l=-\frac{w-1}{2}}^{\frac{w-1}{2}} \check{\psi}(l) = 1$.

Lemma 4.3. *Given (4.25), when $m \neq 0$, we have the following inequality:*

$$|\check{\Psi}(m)| \leq 1 - \check{\psi}(1)\left[2 - 2cos(\frac{2\pi}{M-2+w})\right] < 1, \tag{4.26}$$

Proof. From the values that m can pick, we can have the following inequality:

$$\frac{1}{M-2+w} \leq \frac{m}{M-2+w} \leq \frac{M+\frac{w}{2}-\frac{5}{2}}{M-2+w} < 1.$$

Therefore, from (4.25), we can have the following inequality:

$$\begin{aligned}
\check{\Psi}(m) &= \check{\psi}(0) + 2\sum_{l=1}^{\frac{w-1}{2}} \check{\psi}(l)cos(m\frac{2\pi}{M-2+w}l) \\
&\leq 1 - 2\check{\psi}(1) + 2\check{\psi}(1)cos(\frac{m}{M-2+w}2\pi) \\
&\leq 1 - 2\check{\psi}(1) + 2\check{\psi}(1)cos(\frac{1}{M-2+w}2\pi) \\
&= 1 - \check{\psi}(1)\left[2 - 2cos(\frac{1}{M-2+w}2\pi)\right].
\end{aligned}$$

Because both $w \geq 3$ and $M > 3$ in our case, $\frac{1}{M-2+w} \leq \frac{1}{5}$ and $0 < cos(\frac{2\pi}{M-2+w}) < 1$. Therefore, $1 - \check{\psi}(1)\left[2 - 2cos(\frac{1}{M-2+w}2\pi)\right] < 1$ and (4.26) is proved.

□

4.3.2 Proof of the termination

Theorem 4.4. *Convolution in the time domain corresponds to pointwise multiplication in the frequency domain. Given the padded speed deviations $\check{u}$ and kernel function ψ, we have the following*

relation:

$$\{\breve{\psi} * \breve{u}\}(m) \leftrightarrow \breve{U}_m \breve{\Psi}(m), \tag{4.27}$$

where $\{\breve{\psi} * \breve{u}\}(m)$ *denotes the convolution of* $\breve{\psi}(m)$ *and* $\breve{u}_m$

Proof. With $\{\breve{\psi} * \breve{u}\}(m)$ calculated by (4.14). The DFT of $\{\breve{\psi} * \breve{u}\}(m)$ can can be calculated as follows:

$$\begin{aligned}\sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \left(\sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{\psi}(m-l)\breve{u}_l \right) e^{-im\frac{2\pi}{M-2+w}m} &= \sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{u}_l \sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{\psi}(m-l) e^{-im\frac{2\pi}{M-2+w}m} \\ &= \sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{u}_l \sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{\psi}(m-l) e^{-im\frac{2\pi}{M-2+W}(m-l)} e^{-im\frac{2\pi}{M-2+W}(l)}.\end{aligned}$$

We replace $m-l$ by y, then the equation can be written as follows:

$$\begin{aligned}\sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \left(\sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{\psi}(m-l)\breve{u}_l \right) e^{-im\frac{2\pi}{M-2+w}m} &= \sum_{l=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{u}_l e^{-im\frac{2\pi}{M-2+W}(l)} \sum_{y=-\frac{w-1}{2}-l}^{M-2+\frac{w-1}{2}-l} \breve{\psi}(y) e^{-im\frac{2\pi}{M-2+W}(y)} \\ &= \breve{U}_m \breve{\Psi}(m)\end{aligned}$$

□

Lemma 4.5. *According to Parseval's theorem, the energy of a signal in the time domain equals the energy of the transformed signal in the frequency domain. With the padded speed deviations* $\breve{u}$ *as an example, it can be written as follows (Van Drongelen, 2018):*

$$\sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{u}_m^2 = \frac{1}{M-2+w} \sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{U}_m^2. \tag{4.28}$$

The energy of the speed deviation is guaranteed to decrease after being filtered with the aforementioned filters:

$$\sum_{m=0}^{M-2} \hat{u}_m^2 \leq \left(1 - \breve{\psi}(1)\left[2 - 2cos(\frac{2\pi}{M-2+w})\right]\right)^2 \sum_{m=0}^{M-2} u_m^2. \tag{4.29}$$

Proof. With **Lemma** 4.3 and **Theorem** 4.4, once $\breve{u}$ is filtered by the moving average filter, the energy of the signal after filtering can be calculated as follows:

$$\begin{aligned}\sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} [\{\breve{\psi} * \breve{u}\}(m)]^2 &= \frac{1}{M-2+w} \sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \left[\breve{U}_m \breve{\Psi}(m)\right]^2 \\ &\leq \left(1 - \breve{\psi}(1)\left[2 - 2cos(\frac{2\pi}{M-2+w})\right]\right)^2 \frac{1}{M-2+w} \sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{U}_m^2\end{aligned}$$

Therefore, the following relation can be obtained:

$$\begin{aligned}\sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} [\{\breve{\psi} * \breve{u}\}(m)]^2 &\leq \left(1 - \breve{\psi}(1)\left[2 - 2cos(\frac{2\pi}{M-2+w})\right]\right)^2 \sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{u}_m^2 \\ &< \left(1 - \breve{\psi}(1)\left[2 - 2cos(\frac{2\pi}{M-2+w})\right]\right)^2 \sum_{m=0}^{M-2} u_m^2.\end{aligned}$$

We then remove the padding area and have the following relation:

$$\begin{aligned}\sum_{m=0}^{M-2} \hat{u}_m^2 = \sum_{m=0}^{M-2} [\{\breve{\psi} * \breve{u}\}(m)]^2 &\leq \sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} [\{\breve{\psi} * \breve{u}\}(m)]^2 \\ &\leq \left(1 - \breve{\psi}(1)\left[2 - 2cos(\frac{2\pi}{M-2+w})\right]\right)^2 \sum_{m=0}^{M-2} u_m^2 < \sum_{m=0}^{M-2} u_m^2\end{aligned}$$

□

$\breve{u}$ may have a different mean value from the raw data, and we shift the filtered $\breve{u}$ back to ensure the mean is equal to zero by normalization.

Lemma 4.6. *The normalization process in our method will never increase the energy of the signal. That is:*

$$\sum_{m=0}^{M-2} \tilde{u}_m^2 \leq \sum_{m=0}^{M-2} \hat{u}_m^2. \tag{4.30}$$

Proof. We define a function $g(e)$ of e as follows:

$$g(e) = \sum_{m=0}^{M-2} (\hat{u}_m - e)^2 = \sum_{m=0}^{M-2} \hat{u}_m^2 - 2e \sum_{m=0}^{M-2} \hat{u}_m + (M-1)e^2.$$

$g(e)$ is a convex function, and we obtain its minimum value when the following relation holds:

$$\frac{dg(e)}{de} = -2 \sum_{m=0}^{M-2} \hat{u}_m + 2(M-1)e = 0.$$

In other words, $e = \frac{\sum_{m=0}^{M-2} \hat{u}_m}{M-1}$, indicating that $g(e)$ reaches the minimum value when e equals to the mean value of $\hat{u}_m$. Because normalization makes zero the mean value of $\tilde{u}_m$, we have the following relation:

$$\sum_{m=0}^{M-2} \tilde{u}_m^2 \leq \sum_{m=0}^{M-2} \hat{u}_m^2,$$

where $\tilde{u}$ is a $(M-1)$-point sequence and the equality only holds when the average of $\tilde{u}_m$ is zero. □

The variance of $x^{(1)}$ is equal to the signal energy of u, as follows:

$$(M-1) \cdot var(v) = \sum_{m=0}^{M-2} u_m^2 = \sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \check{u}_m^2. \tag{4.31}$$

Therefore, we can have the following theorem.

Theorem 4.7. *Without termination, the smoothed speed deviations $\tilde{u}$ will finally converge to zeros after an infinite number of iterations after applying our method. That is, if the iterations continue*

without termination,

$$\tilde{v} \to E(v). \tag{4.32}$$

The iterative smoothing process will always terminate within finite iterations once we choose the physically meaningful ranges of speeds, accelerations, and jerks as the termination condition.

Proof. **Lemma** 4.5 and **Lemma** 4.6 proved that the signal energy of $\tilde{u}$ satisfies the following inequality:

$$\begin{aligned}\sum_{m=0}^{M-2} \tilde{u}_m^2 \leq \sum_{m=0}^{M-2} \hat{u}_m^2 &\leq \left(1 - \breve{\psi}(1)\left[2 - 2cos(\frac{2\pi}{M-2+w})\right]\right)^2 \sum_{m=-\frac{w-1}{2}}^{M-2+\frac{w-1}{2}} \breve{u}_m^2 \\ &\leq \left(1 - \breve{\psi}(1)\left[2 - 2cos(\frac{2\pi}{M-2+w})\right]\right)^2 \sum_{m=0}^{M-2} u_m^2\end{aligned}$$

Hence, supposing the method undergoes n iterations, the eventual signal energy of u converges toward $\left(1 - \breve{\psi}(1)\left[2 - 2cos(\frac{2\pi}{M-2+w})\right]\right)^{2n} \sum_{m=0}^{M-2} u_m^2$. In the scenario of iterations continue infinitely, i.e., as $n \to \infty$, $\left(1 - \breve{\psi}(1)\left[2 - 2cos(\frac{2\pi}{M-2+w})\right]\right)^{2n} \to 0$.

Here we can conclude that $\sum_{m=0}^{M-2} \tilde{u}_m^2$ will finally converge to zero if the iteration continues without termination and all values of u_m will approach zero. That is, the speed $\tilde{v} \to E(v)$ and the derivatives of $\tilde{v}$ will be all-zero. Therefore, the iterative smoothing process will always terminate within finite iterations once we choose the physically meaningful ranges of speeds, accelerations, and jerks as the termination condition. □

4.4 Method calibration and comparison with an existing method and manual re-extraction

In this section, we first introduce an algorithm to determine the window size of the filters. We then demonstrate the efficacy of our method when adopting the simple moving average, exponential, triangular, and Gaussian filters upon the NGSIM camera 6 trajectories, and compare our method with a state-of-the-art method with respect to the manually re-extracted data smoothed by SavitzkyGolay (SG) filter.

As a pioneer in vehicle trajectory data smoothing, (Montanino and Punzo, 2013, 2015) proposed a well-known multistep optimization method and revealed smoothed trajectory data. More recently, (Coifman and Li, 2017a) manually re-extracted the vehicle trajectories from the record of camera 6 on the I80 freeway from 4:00 pm to 4:15 pm on April 13, 2005, and smoothed the re-extracted data with the SG filter. From the presented video (Coifman and Li, 2017b) we can find that many erroneous recognitions are revised. The re-extracted trajectories are closer to the ground truth, and their released dataset has been used as the 'ground truth' NGSIM data (Dong et al., 2021). Following the suggestion in (Coifman and Li, 2017a), we calculate the positions, accelerations, and jerks with the new speeds, and construct the benchmark.

Because the speed limit on the I80 freeway is 65 mph (28.9 m/s). We consider the feasible speed ranges to be 0 to 30 m/s. For both conditions, the ranges of accelerations and jerks are set as $[-5,4]$ m/s^2 and $[-8,8]$ m/s^3.

4.4.1 Choice of parameters

Apart from the ranges of derivatives of positions, the window size of the filter kernels (w) is the key parameter in our method. Choosing an overly small window size leads to a high iteration

count and increased computation cost; while if we choose a too-large window size, the method can be completed in a single iteration, but the local information in the raw data may be overlooked. Subsequent iterations primarily fine-tune data with outliers in speed, acceleration, or jerk profiles after the initial iteration. For this, we set w as 5 time-steps for simple moving average and triangular filters, and 9 time-steps for exponential and Gaussian filters in subsequent iterations. Detailed parameter settings for the first iteration will be further discussed.

We choose the MAE between the smoothed and raw positions as the evaluation metric for choosing the proper window size. The MAE should be kept to a modest value, which can be regarded as a way of avoiding over-smoothness. The MAE between the smoothed and raw positions can be calculated as follows:

$$E(|\tilde{x}-x|;w)=\frac{\sum_{m=1}^{M}|\tilde{x}_m-x_m|}{M}. \tag{4.33}$$

Based on this, we propose an algorithm to determine the values of w. In the algorithm, we investigate w ranging from 5 to 41 time-steps at 4 time-step intervals for the simple moving average and triangular filters, and 9 to 81 time-steps at 8 time-step intervals for the exponential and Gaussian filters to apply our method. Therefore, there are 10 optional w for each kernel. The algorithm works as follows:

1. For each w, smooth the raw positions with our method in the way shown as Figure 4.1, and calculate the MAE between the smoothed and raw positions. This step returns 10 MAEs.

2. Pick out all the window sizes w which can satisfy $\varphi(x;w)<\varepsilon$ as candidate parameter combinations, where ε is a small number representing the acceptable threshold, $\varphi(x;w)=E(|\tilde{x}-x|;w)-\min E(|\tilde{x}-x|)$ and $\min E(|\tilde{x}-x|)$ is the minimum MAE among the 10 values.

3. Among the candidate window sizes w, choose the largest one.

At this point, we have completed the choice of the parameters for smoothing. In summary, step 1

calculates MAEs for all tested parameters, step 2 ensures that the modification to the raw positions retains a modest MAE between the smoothed and raw positions, and step 3 ensures the quickest convergence.

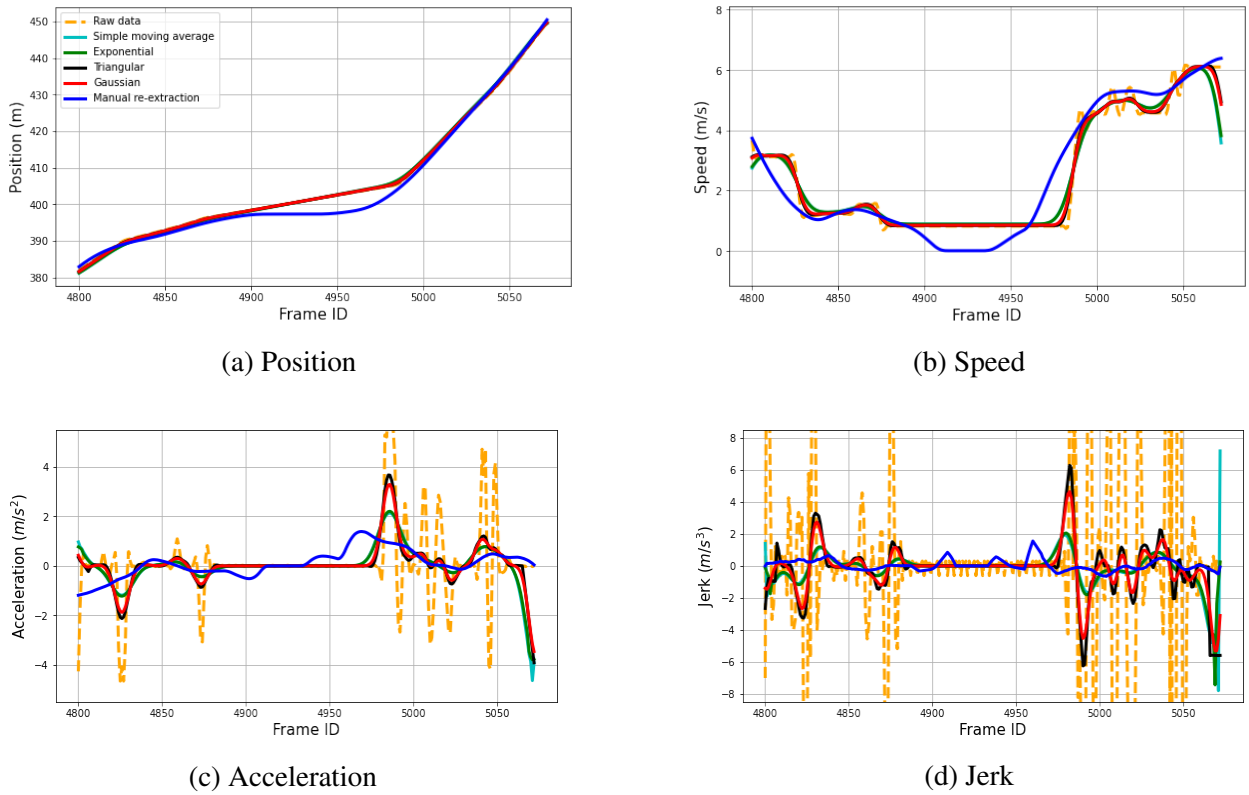
4.4.2 Calibration with sample trajectories

We first show the performance of our method upon a sample trajectory, the trajectory of vehicle 1486, which is also one example in (Coifman and Li, 2017a). The sample trajectory includes all of the cruising, accelerating, and decelerating behaviors, and we can find many outliers in the derivatives of positions, especially in the jerks.

We choose the parameters with our parameter choice algorithm. Setting $\varepsilon = 0.02\ m$, our algorithm returns 13, 25, 17, and 33 time-step window sizes for the simple moving average, exponential, triangular, and Gaussian filters, respectively. The corresponding MAEs between the smoothed and raw positions are 0.49, 0.46, 0.16, and 0.2 meters, respectively.

With the above settings, we smooth the positions, speeds, accelerations, and jerks, and evaluate the performance of our method by comparing the smoothed data with those extracted manually. The results are shown as Figure 4.2. Our methods are robust to different smoothing kernels, and all the tested kernels can make all speeds, accelerations, and jerks physically meaningful with small position modifications. In general, our method yields small differences between the smoothed and raw positions. However, these differences are significantly amplified in derivatives of positions, particularly in accelerations and jerks.

The pre- and post-processed positions are displayed in Figure 4.2a, in which our smoothed positions exhibit a high degree of similarity to the raw positions and different smoothing kernels lead to little difference. Manual re-extracted positions reveal a stop-and-go behavior in the middle of the trajectory, where the vehicle moves at low speeds but does not stop in the raw trajectory. This



(a) Position

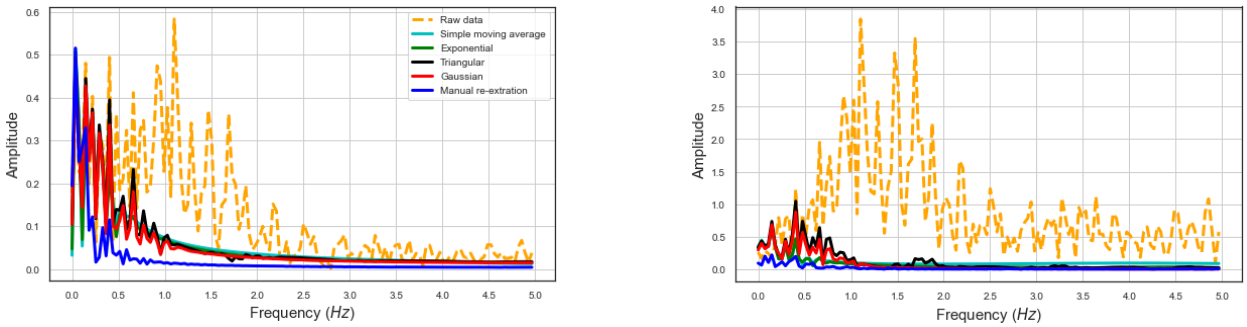
(b) Speed

(c) Acceleration

(d) Jerk

Figure 4.2: Smoothed positions, speeds, accelerations, jerks, and spacing versus relative speed of vehicle 1486

can also be reflected in speeds, which is shown in Figure 4.2b. Our method smooths out the corners and rapid oscillations in speeds while preserving most of the essential information, including decelerating, accelerating, and cruising behaviors. Nevertheless, in the middle of the speed profile, our smoothed speeds do not reduce to zero, and the accelerating behavior lags behind that obtained by manual re-extraction. The differences between the smoothed and raw accelerations and jerks are more pronounced, as shown by Figure 4.2c and Figure 4.2d. Our method is robust to different types of convolution filters. Every filter we tested successfully eliminates all outliers and unrealistic oscillations in the raw accelerations and jerks, and the general trends of the smoothed data obtained from different filters are the same. However, there were minor differences in the smoothed accelerations and jerks produced by different filters. In addition, the smoothed accelerations and jerks differ from the raw data at the endpoint of the raw data, because the speed at the endpoint is greatly different from the average speed.



(a) Acceleration

(b) Jerk

Figure 4.3: Frequency spectrums of accelerations and jerks of vehicle 1486

The frequency spectrums of the acceleration and jerk profiles before and after smoothing are displayed in Figure 4.3. It is evident that the noises in the raw accelerations and jerks occupy the entire frequency range, including high-frequency values that are beyond the human operational range (up to 2 Hz) (Punzo et al., 2011). Both our proposed method and the manual re-extraction can eliminate these high-frequency values. This observation aligns with our earlier finding that our method can effectively remove quick oscillations in acceleration and jerk profiles.

4.4.3 Evaluation of different filters and comparison with an existing method

After showing the performance of our method with a single trajectory, we apply our method adopting the four convolution filters to smooth the "NGSIM I80 camera 6" dataset. Due to the high computational cost of the parameter determination algorithm, we randomly select 30% of the vehicles at the site and calculate the mean absolute error (MAE) between the smoothed and raw positions under different parameters. Our algorithm returns 13, 81, 29, and 49 time-steps as the window sizes for the simple moving average, exponential, triangular, and Gaussian filters, respectively; corresponding MAEs between the smoothed and raw positions are 0.18, 0.17, 0.17, and 0.16 meters. The total computation time for selecting the parameters for one filter and smoothing all the trajectories is approximately 2 minutes on an Asus computer with an Intel i7 2.90 GHz CPU.

To verify the convergence pattern, we choose vehicle 1486, apply our method with different kernel

functions adopting the aforementioned parameters, and plot the natural logarithm of speed variances versus the iteration numbers. The result is shown as Figure 4.4, from which we can see the natural logarithm of speed reduces in a linear pattern, meaning that the speed variances decrease in an exponential pattern. This is in line with our proof in **Theorem** 4.7.

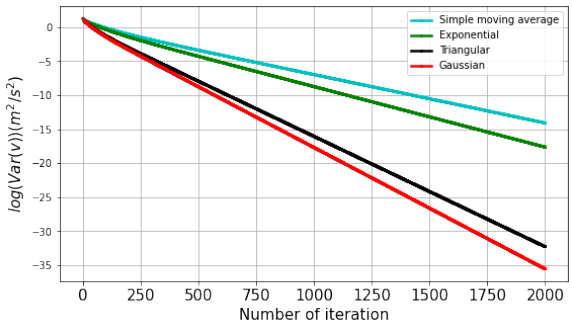


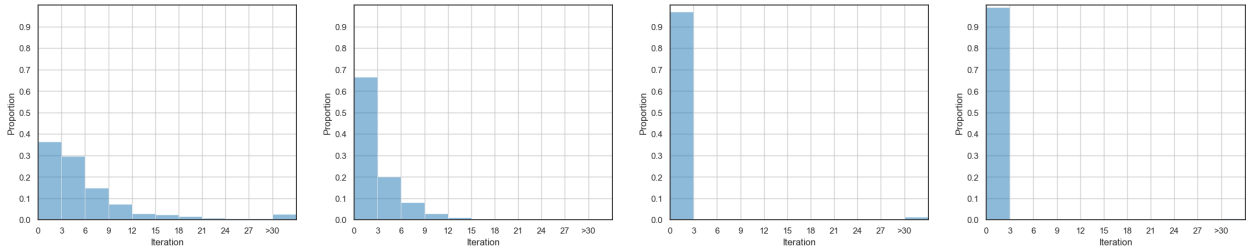
Figure 4.4: Speed variance decrease according to the iteration numbers

We calculate the number of iterations our method needs to go through before termination for smoothing each trajectory. We set the upper limit of iterations to 20000. The average required iterations, maximum required iterations, the proportion of trajectories finished in a single iteration, and the proportion of trajectories finished within 10 iterations of the four filters are shown in **Table** 4.1, and the distributions of the required iterations are shown as Figure 4.5. The averages of the required iterations for different filters range from about 1 to 9. The maximum required iteration of all is 470. That is, all of the 1714 trajectories in the dataset satisfy the termination condition. The average number of required iterations is minimized when utilizing the Gaussian filter, while the exponential filter boasts the lowest maximum number of iterations. In comparison to other filters, the exponential filter demonstrates the highest degree of stability in terms of the required iterations, as indicated by its smallest standard deviation. The distributions of required iterations are different for different filters. For the simple moving average filter, over 85% of trajectories can be smoothed within 10 iterations, while this value rises to 97% for the exponential and triangular filters, and reaches 99% for the Gaussian filter.

We compute the mean, standard deviation, range, and proportion of outliers in the speed, accelera-

Table 4.1: Number of the required iterations using different convolution filters

	Simple moving average filter	Exponential filter	Triangular filter	Gaussian filter
Average	6.4	3.12	4.22	1.9
Maximum	131	53	470	228
Std	9.98	3.74	27.41	11.44
$= 1$ (%)	25	32	96.1	99.1
≤ 10 (%)	86.2	97	97	99.1



(a) Moving average filter (b) Exponential filter (c) Triangular filter (d) Gaussian filter

Figure 4.5: Distribution of the number of required iterations using different convolution filters

tion, and jerk profiles of the raw data and the data obtained by the aforementioned three methods. We also compare the raw data, as well as the smoothed data obtained with the multistep optimization method (Montanino and Punzo, 2015) and our method, with the manually re-extracted data smoothed by the SG filter (simply written as "manual re-extraction" in the following) (Coifman and Li, 2017a). The mean squared error (MSE) and the mean absolute error (MAE) are chosen as the evaluation metrics. For positions, the evaluation metrics can be calculated through (4.34) to (4.35), and likewise for speeds, accelerations, and jerks:

$$E((\bar{\mathbf{x}}-\tilde{\mathbf{x}})^2) = \frac{1}{M}\sum_{m=1}^{M}(\bar{x}_m-\tilde{x}_m)^2, \tag{4.34}$$

$$E(|\bar{\mathbf{x}}-\tilde{\mathbf{x}}|) = \frac{1}{M}\sum_{m=1}^{M}|\bar{x}_m-\tilde{x}_m|, \tag{4.35}$$

where $\bar{\mathbf{x}}$ is the vector of manually re-extracted positions where there are M elements, and $\tilde{\mathbf{x}}$ denotes the vector of the positions to be compared with $\bar{\mathbf{x}}$, which in this case are the raw positions and the

positions smoothed via the multistep optimization method and our method.

The statistical comparison of the performance of different methods is presented in **Table** 5.3. The MSE and MAE between the smoothed positions obtained by our method and the manually re-extracted positions are nearly equivalent to those between the raw and manually re-extracted positions. All methods eliminate the outliers in speeds, and Our methods preserve the means of the speeds. Outliers in the raw accelerations and jerks constitute 15% and 42.3% of the total data, respectively. Both our proposed method and the manual re-extraction eliminate these outliers. The multistep optimization method reduces the proportion of outliers in accelerations to less than 0.05% and the outliers in jerks to 0.6%. The difference between the raw and manually re-extracted speeds is not significant, as indicated by the MSE and MAE. However, this discrepancy is significantly amplified in accelerations and jerks, with the MSEs approaching 45 m^2/s^4 and $10,000\ m^2/s^6$, respectively. Both our proposed method and the multistep optimization method can greatly reduce such discrepancies. From the perspectives of all three indicators (speeds, accelerations, and jerks), the smoothed data obtained by our method are closer to the manually re-extracted ones than those obtained by the multistep optimization method.

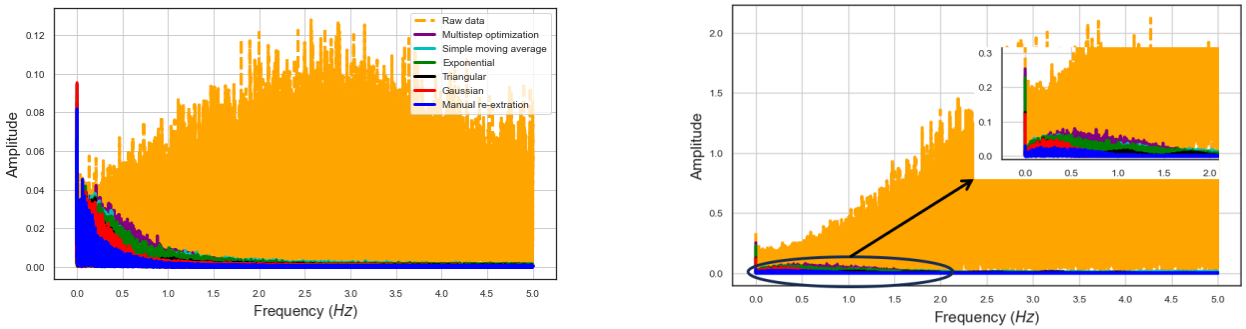
The comparison of the four different filters used in our method indicates that there are little differences in the smoothed speeds and accelerations. However, more significant differences are observed in the smoothed jerks. Notably, the Gaussian filter exhibits the best performance compared to the other filters.

Table 4.2: Comparison of different methods

variables		Raw data	Multistep optimization	Our method				Manual re-extraction
				Filter 1	Filter 2	Filter 3	Filter 4	
Positions (m)	$E((\mathbf{x}^* - \hat{\mathbf{x}})^2)$	1.91	2.18	2.19	2.12	2.08	2.04	/
	$E(\lvert\mathbf{x}^* - \hat{\mathbf{x}}\rvert)$	0.98	0.99	1.05	1.04	1.02	1.04	/
Speed (m/s)	Mean	8.07	8.05	8.06	8.07	8.07	8.07	7.88
	Std	4.07	4.01	3.95	3.96	3.96	3.97	3.89
	Range	[0,36.0]	[0,27.0]	[0,26.9]	[0,27]	[0,26.8]	[0,26.7]	[0,26.4]
	Outliers (%)	0	0	0	0	0	0	0

	$E((\mathbf{v}^* - \hat{\mathbf{v}})^2)$	0.88	0.47	0.43	0.42	0.38	0.37	/
	$E(\lvert\mathbf{v}^* - \hat{\mathbf{v}}\rvert)$	0.62	0.48	0.46	0.46	0.44	0.44	/
	Mean	-0.04	-0.04	-0.03	-0.04	-0.05	-0.05	0.03
	Std	6.69	0.92	0.81	0.8	0.66	0.66	0.58
Acceleration	Range	[-176.5,292.2]	[-14.1,4.5]	[-5.0,4.0]	[-4.4,4.0]	[-5.0,4.0]	[-5.0,4.0]	[-4.2,3.5]
(m/s^2)	Outliers (%)	15	0	0	0	0	0	0
	$E((\mathbf{a}^* - \hat{\mathbf{a}})^2)$	44.56	0.68	0.66	0.64	0.4	0.4	/
	$E(\lvert\mathbf{a}^* - \hat{\mathbf{a}}\rvert)$	2.64	0.60	0.55	0.54	0.45	0.44	/
	Mean	-0.17	-0.13	-0.11	-0.11	-0.06	-0.06	-0.02
	Std	99.80	2.41	1.81	1.86	1.07	0.97	0.62
Jerk	Range	[-4171.5,2954.4]	[-141.0,39.4]	[-8.0,8.0]	[-8.0,8.0]	[-8.0,8.0]	[-7.4,8.0]	[-5.3,7.1]
(m/s^3)	Outliers (%)	42.3	0.6	0	0	0	0	0
	$E((\mathbf{j}^* - \hat{\mathbf{j}})^2)$	9960.09	5.90	3.46	3.65	1.33	1.12	/
	$E(\lvert\mathbf{j}^* - \hat{\mathbf{j}}\rvert)$	32.79	1.65	1.23	1.31	0.82	0.75	/

Furthermore, we analyze the frequency spectrums of the acceleration and jerk profiles of the raw data and data obtained by different methods. As Figure 4.6 shows, the noises contained in the raw data, which can be seen from the orange dashed curves, span the entire frequency range. Both the multistep optimization method and our method effectively eliminate a large portion of the noise, resulting in frequency spectrums that fall within the range of the human-vehicle response frequency (up to 2 Hz). Compared to the multistep optimization method, the frequency spectrum of the smoothed jerk profile obtained by our method is closer to that obtained by the manual re-extraction. The Gaussian filter shows the best performance among the four tested filters.



(a) Acceleration

(b) Jerk

Figure 4.6: Frequency spectrums of accelerations and jerks of the dataset

4.5 Conclusion

In this paper, we propose a straightforward, iterative moving average method for smoothing longitudinal vehicle trajectory data. In each iteration, we pad both ends of the speed profile with the average speed and apply a moving average method with different kernel shapes, representing different weights, to speeds. We then normalize the filtered speeds to preserve the average speed and maintain the total travel distance, and differentiate the normalized filtered speeds into accelerations and jerks to check whether the termination conditions are satisfied. If they are satisfied, the iterative process concludes and we integrate these smoothed speeds back into the positions. Otherwise, we take these smoothed speeds as the new input and initiate another iteration to further smooth the speeds. Mathematically, we prove that our method can terminate within a finite number of iterations, in accordance with predefined termination criteria. We then numerically demonstrate the efficacy of our method upon the trajectories in the NGSIM I80 camera 6 dataset. The speed, acceleration, and jerk profiles before and after smoothing indicate that our method eliminates all outliers and rapid oscillations while retaining most of the information in the raw data, including accelerating, decelerating, and cruising behaviors. However, because the speed at the endpoint is greatly greater than the average speed, the smoothed data at the endpoint shows differences from the raw data. In addition, this study reports on the smoothing of 1714 trajectories in the dataset. Our method for smoothing all the trajectories satisfies the termination condition, although the requisite iterations varied based on different kernels. Statistical comparisons among different methods reveal the efficacy of our proposed method in eliminating outliers, and that the speed, acceleration, and jerk profiles smoothed using our method can better resemble those manually re-extracted, underscoring their enhanced accuracy. Among the kernels tested, the Gaussian kernel emerged as the frontrunner, exhibiting superior performance compared to other tested kernels.

The contributions of this study are two fold. On one hand, we introduce a novel iterative moving average method to effectively smooth longitudinal vehicle trajectory data. This method effectively eliminates outliers and high-frequency noise from the raw data, resulting in smoothed speeds, ac-

celerations, and jerks that align with physical interpretations. On the other hand, we provide a rigorous mathematical proof demonstrating that, without termination, the speeds finally converge to a constant value after an infinite number of iterations, ensuring that the bounds in speeds and their derivatives are physically meaningful within a finite number of iterations with the corresponding termination criterion.

It should be noted that our proposed method is open to other options for smoothing speeds such as the weighted kNN regression algorithm and the Savitzky-Golay filter, in addition to the alternatives presented in this study. Furthermore, although our method cannot correct all detection errors, it can eliminate outliers in higher-order derivatives of positions and enhance the accuracy of smoothed trajectories at a low computational cost. The proposed method thus represents a general iterative framework for smoothing longitudinal vehicle trajectories, with the potential for the development of more advanced strategies based on our approach.

In the future, with the accessibility of data from various sources, we plan to test the proposed method using data obtained from other sources. In addition, our method may necessitate multiple iterations, with the physically meaningful ranges of speeds, accelerations, and jerks serving as the termination criterion. This research provides a foundation for the development of more mathematically tractable techniques that incorporate additional driving behavior principles, which could potentially eliminate the need for the iterative process. Furthermore, this study focuses solely on the smoothing of longitudinal vehicle trajectories, and we aim to extend the method to smooth lateral vehicle trajectories with varying termination conditions.

Chapter 5

Two-step quadratic programming for physically meaningful smoothing

5.1 Introduction

In this study, we propose a two-step quadratic programming method to address the gap. Our approach ensures that smoothed speeds and higher order derivatives of positions are consistently defined as symplectic differences in positions, while adhering to physically meaningful bounds. We carefully discuss the properties of resulted difference matrices, and present linear inequality constraints based on bounded derivatives of positions. Our method is closely related to the aforementioned smoothing splines method, which addresses the trade-off between fidelity and smoothness by solving an optimization problem with the weighted sum of the discrepancy and roughness as the objective function. However, instead of formulating a single quadratic programming problem, we leverage our prior knowledge of position error, determined by pixel length in video images (FHWA, 2007; Krajewski et al., 2018), and formulate two sequential quadratic programming problems. In the first step, we minimize the discrepancy between half-smoothed and

raw positions, subject to physically meaningful bounds on speeds and higher order derivatives of half-smoothed positions. In the second step, we minimize the roughness of the smoothed positions, maintaining the same bounds on speeds and higher order derivatives of smoothed positions, along with additional bounds on the smoothed positions themselves such that the smoothed positions are allowed to deviate from the raw data by at most those of the half-smoothed positions and the prior position error. Therefore, our method is fundamentally different from the smoothing splines method, as well as other splines methods. In **Table** 5.1, we provide a conceptual comparison of our method with different splines methods.

Table 5.1: Comparisons of our method and existing splines methods

Method		Discrepancy	Roughness	Splines basis	Bounded derivatives of positions	Prior position error
Smoothing splines		√	√			
Regression splines		√		√		
Penalty splines		√	√	√		
(Toledo et al., 2007)		√		√	√	
(Marczak and Buisson, 2012)		√	√	√		
(Venthuruthiyil and Chunchu, 2018)		√		√		
Our method	Step 1	√			√	
	Step 2		√		√	√

In both steps, the bounds are applied to derivatives, whose order is as high as that in the definition of roughness; i.e., the second step dictates the highest order of derivatives. Thus, the second step is equivalent to minimizing the acceleration, jerk, or snap, if the highest order of derivatives is two, three, or four, respectively. The highest order of derivatives is a central parameter in our method and existing splines methods. In the literature, it is generally preassumed. For example, (Whittaker, 1922) proposed the use of the sum of squared third-order derivatives to measure the roughness of the smoothed data. If we study a vehicle trajectory as a time series of positions, the third-order derivatives are jerks, and this choice is consistent with the minimum jerk principle in human body movements (Flash and Hogan, 1985) and driving experiences (Othman et al., 2008). However, the principles of both minimum jerks and minimum snaps were used to design drones'

and robot manipulators' trajectories (Gasparetto and Zanotto, 2010; Mellinger and Kumar, 2011).[1] In this study, we will calibrate the highest order with empirical data, considering both the quality of smoothed trajectory and computational complexity. However, we only consider up to snaps with the highest order of four, as planning physical quantities higher than snaps is considered impractical due to the large increase in complexity (Lambrechts et al., 2005).

Based on the properties of difference matrices, we observe that the coefficient matrix in the sum of squared highest order derivatives in the second step is not positive definite, and thus require that the first few smoothed positions match the corresponding half-smoothed ones, where the number of such equality constraints equals the highest order of derivatives.

Furthermore, mathematically, we establish the existence and uniqueness of solutions to both quadratic programming problems, thus ensuring the well-defined nature of our method. This is achieved by demonstrating that, in both steps, the domains defined by linear inequality constraints are non-empty and that the objective functions are strictly convex. Numerically, we employ interior point methods to solve the quadratic programming problems and discuss the computational complexity. Empirically, using both NGSIM and highD data, we calibrate the highest order of derivatives and compare our method with an existing approach with respect to manually re-extracted data.

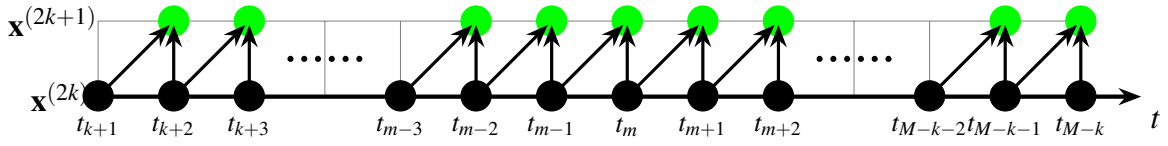
The rest of this chapter is organized as follows. In Section 4.2, we define derivatives of positions with symplectic differences and formulate physically meaningful bounds as linear inequalities. In Section 4.3, we introduce the two-step quadratic programming method. In Section 4.4, we prove the existence and uniqueness of solutions to both quadratic programming problems and discuss the computational complexity for solving the problem with interior point methods. In Section 4.5, we calibrate the highest order of derivatives and compare our method with an existing approach with respect to manually re-extracted NGSIM data and apply our method to highD data. In Section 4.6, we discuss future extensions of our method.

[1] The trajectory design problem in the x dimension is related to but different from the trajectory smoothing problem in that one needs to find physically meaningful times $\tilde{t}_m$ passing given waypoints x_m.

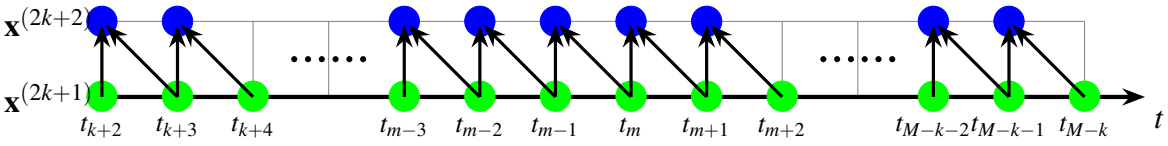
5.2 Derivatives of positions and their physically meaningful bounds

In this section, we first define the derivatives of positions with symplectic differences and then introduce linear inequality constraints based on physically meaningful bounds on the derivatives. These constraints are imposed on both half-smoothed and smoothed trajectories, such that the resulting speeds, accelerations, jerks, and higher order derivatives are consistently defined and physically meaningful.

5.2.1 Derivatives of positions



(a) Implicit differences



(b) Explicit differences

Figure 5.1: Symplectic discretization for derivatives of positions

For a time-continuous trajectory, $x(t)$, the kth order derivative is denoted by $x^{(k)}(t)$ for $k = 1, \cdots, K$, where K is the highest order of derivatives. Thus we have

$$x^{(k)}(t) = \frac{dx^{(k-1)}(t)}{dt} = \frac{d^{(k)}x(t)}{dt^{(k)}}. \tag{5.1}$$

We employ the symplectic discretization (mixed implicit-explicit Euler discretization) method to define the kth order derivatives, $x_m^{(k)}$, at t_m ($m = \lfloor \frac{k+1}{2} \rfloor + 1, \cdots, M - \lfloor \frac{k}{2} \rfloor$), where $\lfloor \cdot \rfloor$ is the floor function. For car-following models, this method leads to collision-free and forward-traveling solutions

(Jin, 2019). In particular,

$$x_m^{(2k+1)} = \frac{x_m^{(2k)} - x_{m-1}^{(2k)}}{\Delta t}, \quad k = 0, 1, \cdots \tag{5.2a}$$

$$x_m^{(2k+2)} = \frac{x_{m+1}^{(2k+1)} - x_m^{(2k+1)}}{\Delta t}, \quad k = 0, 1, \cdots \tag{5.2b}$$

That is, speed, jerk, and odd-order derivatives are calculated with backward differences; and acceleration, snap, and even-order derivatives with forward differences. The discretization scheme is depicted in Figure 5.1, where the nodes represent the derivatives of positions and arrows represent the calculation direction.

We denote the vector of $x_m^{(k)}$ for $m = \lfloor \frac{k+1}{2} \rfloor + 1, \cdots, M - \lfloor \frac{k}{2} \rfloor$ and $k = 0, \cdots, K$ by $\mathbf{x}^{(k)}$. Then (5.2a) and (5.2b) can be rewritten with matrices as follows:

$$\mathbf{x}^{(k)} = \frac{((\mathbf{0}|I_{M-k}) - (I_{M-k}|\mathbf{0}))\mathbf{x}^{(k-1)}}{\Delta t}, \quad k = 1, \cdots, K, \tag{5.3}$$

where I_{M-k} is a $(M-k) \times (M-k)$ identity matrix, and $\mathbf{0}$ is a $(M-k) \times 1$ vector of zeros. We then write $\mathbf{x}^{(k)}$ in positions as follows:

$$\mathbf{x}^{(k)} = \frac{W^{(k)}\mathbf{x}}{\Delta t^k}, \tag{5.4}$$

fleqnfalsewhere $W^{(k)}$ is the kth difference matrix which has a staircase structure, and its dimension is $(M-k) \times M$ (Strang, 1993).In particular, $W^{(0)} = I_M$. Following (5.3), other difference matrices can be written as follows:

$$W^{(k)} = ((\mathbf{0}|I_{M-k}) - (I_{M-k}|\mathbf{0}))W^{(k-1)}, \quad k = 1, \cdots, K. \tag{5.5}$$

Lemma 5.1. *For $k = 1, \cdots, K$, the sum of each row in $W^{(k)}$ is zero; i.e.,*

$$W^{(k)} \cdot \mathbf{1} = \mathbf{0}, \tag{5.6}$$

where $\mathbf{1}$ is a $(M-k) \times 1$ vector of ones.

Proof. For $k = 1$, from (5.5) we have $W^{(1)} = ((\mathbf{0}|I_{M-1}) - (I_{M-1}|\mathbf{0}))$. Thus, $W^{(1)} \cdot \mathbf{1} = \mathbf{0}$, since each row has exactly one 1 and one -1, with all other elements being zeros. Further from (5.5) we have $W^{(2)} \cdot \mathbf{1} = ((\mathbf{0}|I_{M-2}) - (I_{M-2}|\mathbf{0}))W^{(1)} \cdot \mathbf{1} = \mathbf{0}$ and $W^{(k)} \cdot \mathbf{1} = \mathbf{0}$ for $k \geq 2$. □

Lemma 5.2. *With D defined as*

$$D = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ -1 & 1 & \ddots & & \vdots \\ 0 & -1 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -1 & 1 \end{pmatrix}_{M \times M},$$

$W^{(k)}$ can be written as

$$W^{(k)} = (\underset{(M-k)\times k}{\mathbf{0}} | I_{M-k}) D^k, \tag{5.7}$$

where $\mathbf{0}$ is a zero matrix with the dimension written below it.

Proof. In the case of $k = 0$, $W^{(0)} = I_M = I_M D^0$, and the lemma is correct for this base case.

We assume that the lemma holds for $k-1$, that is, $W^{(k-1)} = (\underset{(M-k+1)\times(k-1)}{\mathbf{0}} | I_{M-k+1}) D^{k-1}$.

Now we consider the case of k. Together with (5.5), $W^{(k)}$ can be written as follows:

$$\begin{aligned} W^{(k)} &= ((\mathbf{0}|I_{M-k}) - (I_{M-k}|\mathbf{0}))W^{(k-1)} \\ &= ((\mathbf{0}|I_{M-k}) - (I_{M-k}|\mathbf{0}))(\underset{(M-k+1)\times(k-1)}{\mathbf{0}}|I_{M-k+1})D^{-1}D^k, \end{aligned}$$

where

$$D^{-1} = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ 1 & 1 & \ddots & & \vdots \\ 1 & 1 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 1 & 1 & \cdots & 1 & 1 \end{pmatrix}_{M\times M}.$$

Therefore, we can calculate $W^{(k)}$ as follows:

$$W^{(k)} = \left[(\underset{(M-k)\times k}{\mathbf{0}}|I_{M-k}) - (\underset{(M-k)\times(k-1)}{\mathbf{0}}|I_{M-k}|\underset{(M-k)\times 1}{\mathbf{0}})\right] D^{-1}D^k = (\underset{(M-k)\times k}{\mathbf{0}}|I_{M-k})D^k,$$

which completes the inductive step and hence the proof of the lemma.

□

Corollary 5.3. *We define two submatrices of $W^{(k)}$, $W_1^{(k)}$ and $W_2^{(k)}$, as follows:*

$$W_1^{(k)} \equiv (W_{i,q}^{(k)})_{1\le i\le M-k, 1\le q\le k}, \tag{5.8a}$$

$$W_2^{(k)} \equiv (W_{i,q}^{(k)})_{1\le i\le M-k, k+1\le q\le M}. \tag{5.8b}$$

Then $W_1^{(k)}$ *and* $W_2^{(k)}$ *are submatrices of* D^k*:*

$$W_1^{(k)} = (D_{i,q}^k)_{k+1\le i\le M,1\le j\le k}, \tag{5.9a}$$

$$W_2^{(k)} = (D_{i,q}^k)_{k+1\le i\le M,k+1\le j\le M}. \tag{5.9b}$$

Proof. D^k is a $M\times M$ square matrix, and it can be partitioned into four blocks as follows:

$$D^k = \left(\begin{array}{c|c} \underset{k\times k}{\mathbf{D}_{11}^k} & \underset{K\times(M-k)}{\mathbf{D}_{12}^k} \\ \hline \underset{(M-k)\times k}{\mathbf{D}_{21}^k} & \underset{(M-k)\times(M-k)}{\mathbf{D}_{22}^k} \end{array}\right).$$

Together with **Lemma** 5.2, we can write $W^{(k)}$ as follows:

$$W^{(k)} = (\underset{(M-k)\times k}{\mathbf{0}}|I_{M-k}) \left(\begin{array}{c|c} \underset{k\times k}{\mathbf{D}_{11}^k} & \underset{K\times(M-k)}{\mathbf{D}_{12}^k} \\ \hline \underset{(M-k)\times k}{\mathbf{D}_{21}^k} & \underset{(M-k)\times(M-k)}{\mathbf{D}_{22}^k} \end{array}\right) = (\underset{(M-k)\times k}{\mathbf{D}_{21}^k} \mid \underset{(M-k)\times(M-k)}{\mathbf{D}_{22}^k}).$$

From the definition of $W_1^{(k)}$ and $W_2^{(k)}$, we know that $W^{(k)} = (W_1^{(k)}|W_2^{(k)})$, and the dimensions of $W_1^{(k)}$ and $W_2^{(k)}$ are $(M-k)\times k$ and $(M-k)\times(M-k)$, respectively. Therefore, $W_1^{(k)}$ and $W_2^{(k)}$ satisfy the following equations:

$$W_1^{(k)} = \underset{(M-k)\times k}{\mathbf{D}_{21}^k} = (D_{i,q}^k)_{k+1\le i\le M,1\le j\le k},$$

$$W_2^{(k)} = \underset{(M-k)\times(M-k)}{\mathbf{D}_{22}^k} = (D_{i,q}^k)_{k+1\le i\le M,k+1\le j\le M}.$$

□

In addition, the sum of squared kth order derivatives can also be written with $\mathbf{x}$ and $W^{(k)}$ as

$(\mathbf{x}^{(k)})^\top(\mathbf{x}^{(k)}) = \frac{\mathbf{x}^\top(W^{(k)})^\top W^{(k)}\mathbf{x}}{\Delta t^{2k}}$, where $\top$ is the transpose operator, and the dimension of $(W^{(k)})^\top W^{(k)}$ is $M \times M$.

Corollary 5.4. *The rank of* $(W^{(k)})^\top W^{(k)}$ *is* $M-k$.

Proof. From **Lemma** 5.2, it is evident that the rank of $W^{(k)}$ is equal to $M-k$. According to the Rank-Nullity theorem (Strang, 2014), for any matrix W of dimensions $n_1 \times n_2$ (where $n_1 < n_2$), the sum of the rank and nullity of W is equal to n_2. Here, the nullity of W refers to the dimension of its null space.

We define the null space of $W^{(k)}$ as $N(W^{(k)}) = \{\mathbf{x} \in R^M : W^{(k)}\mathbf{x} = \mathbf{0}\}$. For any $\mathbf{x} \in N(W^{(k)})$, the following equation always holds:

$$(W^{(k)})^\top W^{(k)}\mathbf{x} = \mathbf{0},$$

which implies that $\mathbf{x} \in N((W^{(k)})^\top W^{(k)})$. Therefore, we can draw the following conclusion:

$$N(W^{(k)}) \subseteq N((W^{(k)})^\top W^{(k)}).$$

Similarly, the null space of $(W^{(k)})^\top W^{(k)}$ can be written as $N((W^{(k)})^\top W^{(k)}) = \{\mathbf{x} \in R^M : (W^{(k)})^\top W^{(k)}\mathbf{x} = \mathbf{0}\}$. For any $\mathbf{x} \in N((W^{(k)})^\top W^{(k)})$, the following equation always holds:

$$\mathbf{x}^\top(W^{(k)})^\top W^{(k)}\mathbf{x} = (W^{(k)}\mathbf{x})^\top W^{(k)}\mathbf{x} = \mathbf{0},$$

from which we can know that $W^{(k)}\mathbf{x} = \mathbf{0}$ and $\mathbf{x} \in N(W^{(k)})$, and we can conclude:

$$N((W^{(k)})^\top W^{(k)}) \subseteq N(W^{(k)}).$$

Therefore, we can conclude that $N((W^{(k)})^\top W^{(k)}) = N(W^{(k)})$, and the dimensions of $N(W^{(k)})$ and

$N((W^{(k)})^\top W^{(k)})$ are the same. Therefore, according to the Rank-Nullity theorem, we have

$$Rank((W^{(k)})^\top W^{(k)}) = Rank(W^{(k)}) = M - k.$$

□

5.2.2 Linear inequality constraints based on bounded derivatives of positions

Following the empirical observations, all higher order derivatives of positions should be bounded (Jin, 2021; Pendrill and Eager, 2020). Using k to denote the order of the derivative, the boundedness of higher order derivatives can be written as follows:

$$x_-^{(k)} \leq \mathbf{x}^{(k)} \leq x_+^{(k)}, \;\; k = 1, \cdots, K \tag{5.10}$$

where $x_-^{(k)}$ and $x_+^{(k)}$ represent the lower and upper bounds of $\mathbf{x}^{(k)}$.

Replacing $\mathbf{x}^{(k)}$ by positions, we can rewrite (5.10) as follows:

$$\Delta t^k x_-^{(k)} \leq W^{(k)}\mathbf{x} \leq \Delta t^k x_+^{(k)}, \;\; k = 1, \cdots, K. \tag{5.11}$$

Therefore, with (5.11), the boundedness of all the derivatives of positions can be expressed using linear inequalities in positions.

5.3 Two-step quadratic programming method

In this section, we first introduce the framework of our two-step method for smoothing longitudinal vehicle trajectories, followed by a comprehensive explanation of each step.

We illustrate the flow chart of our method with Figure 5.2. The initial inputs consist of raw positions. The final outputs consist of smoothed positions, from which speeds and higher order derivatives can be calculated. The two rectangles represent the two steps in our method, with each step involving a quadratic programming problem. A comprehensive explanation of the method is provided in the subsequent subsections. As both step 1 and step 2 produce smoother positions, we differentiate between their outputs by referring to the results of step 1 and step 2 as "half-smoothed positions" and "smoothed positions", respectively.

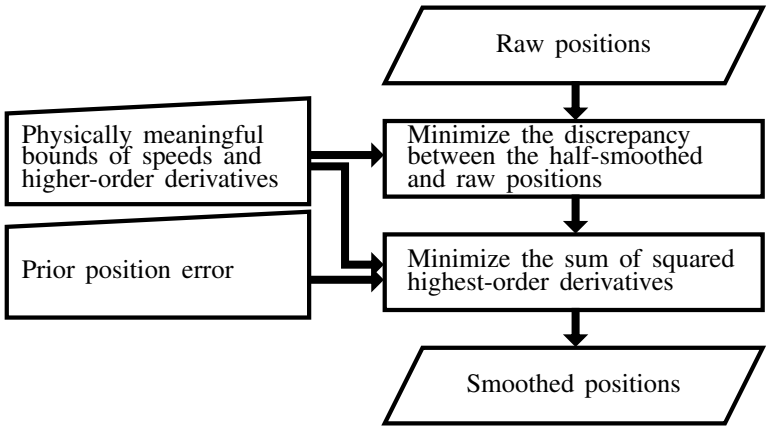


Figure 5.2: The flow chart of the two-step quardratic programming method

5.3.1 First step: minimization of the discrepancy between the half-smoothed and raw positions

We formulate a quadratic programming problem to address the concern of preserving fidelity. The objective of this quadratic programming problem is to minimally modify the raw positions while concurrently guaranteeing that all higher order derivatives fall within the physically meaningful

ranges. The quadratic programming problem can be written as follows:

$$\min_{\hat{\mathbf{x}}} (\hat{\mathbf{x}} - \mathbf{x})^\top (\hat{\mathbf{x}} - \mathbf{x}), \tag{5.12a}$$

$$s.t.$$

$$x_-^{(k)} \leq \hat{\mathbf{x}}^{(k)} \leq x_+^{(k)}, \ \ k = 1, \cdots, K, \tag{5.12b}$$

where $\hat{\mathbf{x}}$ denotes the vector of the variables in the quadratic programming problem and $\hat{\mathbf{x}}^{(k)}$ denotes the vector of the kth order derivatives of $\hat{\mathbf{x}}$.

Together with the discretization scheme we introduced in section 2, (5.12a) to (5.12b) can be rewritten as a standard quadratic programming problem, as follows:

$$\min_{\hat{\mathbf{x}}} (\hat{\mathbf{x}} - \mathbf{x})^\top (\hat{\mathbf{x}} - \mathbf{x}), \tag{5.13a}$$

$$s.t.$$

$$-W^{(k)} \hat{\mathbf{x}} \leq -\Delta t^k x_-^{(k)}, \ \ k = 1, \cdots, K, \tag{5.13b}$$

$$W^{(k)} \hat{\mathbf{x}} \leq \Delta t^k x_+^{(k)} \ \ k = 1, \cdots, K. \tag{5.13c}$$

By solving this quadratic programming problem, the resulting half-smoothed positions ensure that all speeds and higher order derivatives are within physically meaningful ranges.

5.3.2 Second step: minimization of the sum of squared highest order derivatives

According to the features of trajectory detection, the errors of raw positions are typically confined within the prior position error, which is determined by the length represented by a pixel in video images (FHWA, 2007; Krajewski et al., 2018). However, certain raw positions may exhibit detec-

tion errors that surpass this anticipated bound. Therefore, the second step of our method allows the smoothed positions to deviate from the raw data by at most those of the half-smoothed data and the prior position error. To be more specific, considering the mth position in the trajectory, the lower bound is set as $\hat{x}_m$ when the difference $x_m - \hat{x}_m$ exceeds the prior position error. In cases where the difference is within the prior position error, the lower bound is adjusted to the raw position minus the prior position error. Similarly, the upper bound for the mth position in the trajectory is determined as $\hat{x}_m$ if the difference $\hat{x}_m - x_m$ exceeds the prior position error. In situations where the difference falls within the prior position error, the upper bound is set to the raw position plus the prior position error. Thus the ranges of the mth position in the trajectory can be written as follows:

$$\min\{x_m - \varepsilon, \hat{x}_m\} \leq \tilde{x}_m \leq \max\{x_m + \varepsilon, \hat{x}_m\}, \tag{5.14}$$

where $\tilde{x}_m$ is the mth position in the smoothed trajectory, and ε is the prior position error. We use $\mathbf{x}_-$ and $\mathbf{x}_+$ to denote the vectors of the lower and upper bounds of positions hereafter, both of which are $(M+1) \times 1$ vectors.

Apart from the position ranges, we apply the same linear inequality constraints with physically meaningful bounds as those used in the first step.

The objective function in the second step is to minimize the roughness, which can be indicated via the sum of squared highest order derivatives, as represented by the following objective function:

$$\min_{\tilde{\mathbf{x}}} (\tilde{\mathbf{x}}^{(K)})^\top \tilde{\mathbf{x}}^{(K)}. \tag{5.15}$$

Given that $\tilde{\mathbf{x}}^{(K)} = \frac{W^{(K)}\tilde{\mathbf{x}}}{\Delta t^K}$, the objective function can be reformulated as:

$$\min_{\tilde{\mathbf{x}}} \frac{1}{\Delta t^{2K}} \tilde{\mathbf{x}}^\top (W^{(K)})^\top W^{(K)} \tilde{\mathbf{x}}. \tag{5.16}$$

For any real vector $\tilde{\mathbf{x}}$, the term $\tilde{\mathbf{x}}^\top (W^{(K)})^\top W^{(K)} \tilde{\mathbf{x}}$ is always non-negative, and $(W^{(K)})^\top W^{(K)}$ is a positive semi-definite matrix. For the objective function to be strictly convex, it is necessary for $(W^{(K)})^\top W^{(K)}$ to be positive definite (Nocedal and Wright, 2006). However, a positive semi-definite matrix is positive definite if and only if it is invertible (Bhatia, 2009). This requirement conflicts with the findings of **Corollary** 5.4, which states that the dimension of $(W^{(K)})^\top W^{(K)}$ is $M \times M$, while $rank((W^{(K)})^\top W^{(K)}) = M - K$. Therefore, $(W^{(K)})^\top W^{(K)}$ is not invertible or positive definite. Based on **Lemma** 5.2, we require that the initial K elements in $\tilde{\mathbf{x}}^{(K)}$ equal to the half-smoothed positions, which is equivalent to specifying the initial value of the position and the first $(K-1)$ orders of derivatives; that is, we require $\tilde{x}_m = \hat{x}_m$ for $m = 1, \cdots, K$.

Furthermore, since the constant coefficient $\frac{1}{\Delta t^{2K}}$ in the quadratic programming problem does not affect the optimal solution, we can eliminate this coefficient without altering the optimal outcome and re-write the second quadratic programming problem as follows:

$$\min_{\tilde{\mathbf{x}}} (W^{(K)} \tilde{\mathbf{x}})^\top (W^{(K)} \tilde{\mathbf{x}}), \tag{5.17a}$$

$$s.t.$$

$$-\tilde{\mathbf{x}} \leq -\mathbf{x}_-, \tag{5.17b}$$

$$\tilde{\mathbf{x}} \leq \mathbf{x}_+, \tag{5.17c}$$

$$-W^{(k)} \tilde{\mathbf{x}} \leq -\Delta t^k x_-^{(k)}, \;\; k = 1, \cdots, K, \tag{5.17d}$$

$$W^{(k)} \tilde{\mathbf{x}} \leq \Delta t^k x_+^{(k)}, \;\; k = 1, \cdots, K, \tag{5.17e}$$

$$\tilde{\mathbf{x}}_{1:K} = \hat{\mathbf{x}}_{1:K}, \tag{5.17f}$$

where $\hat{\mathbf{x}}_{1:K} = (\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_K)^\top$. (5.17b) and (5.17c) bound the positions, and (5.17d) and (5.17e) bound the speeds and higher order derivatives. (5.17f) is important for obtaining the unique optimal solution, and detailed explanation will be given in section 4.2.

The possible values of K are 1, 2, 3, or 4, and the coefficient matrices differ based on the chosen value of K. For instance, if we take $K = 2$, we only need to consider $W^{(1)}$ and $W^{(2)}$. Once the

problem is formulated, we can solve it to obtain the position vector that yields the smallest sum of squared highest order derivatives. Up to now, we have obtained the smoothed positions, with which all higher order derivatives can be calculated.

It can be observed that the outputs of step 2 (smoothed positions) are equal to those of step 1 (half-smoothed positions) when the prior position error is zero ($\varepsilon = 0\ m$). This suggests that in the absence of any detection errors, the two steps yield identical results. In contrast, a near-infinity prior position error ($\varepsilon \to \infty$) will approximately result in a $(K-1)$th order polynomial, and the highest order derivatives will be identically zero except for the initial values. While neither a zero nor an infinitely large prior position error is practically possible, the two extreme examples indicate that the prior position error helps to balance fidelity and smoothness. In this regard, the prior position error plays a similar role to the weight in the objective function of the smoothing splines method (Whittaker, 1922). However, it is important to note that the prior position error holds significant physical meaning within our specific context and can be derived directly from observations.

5.4 Theoretical properties and computational complexity

In this section, we analyze the properties of the quadratic programming problems in our method. We prove that both quadratic programming problems are not only solvable but also possess unique optimal solutions. Subsequently, we proceed to assess the computational complexity associated with the interior point method employed for solving the quadratic programming problems, particularly with regard to the highest order derivatives.

5.4.1 Existence of solutions

We first demonstrate that the quadratic programming problems in both steps of our methods are solvable under all conditions.

Theorem 5.5. *The quadratic programming problem in step 1, (5.13a) to (5.13c), always has feasible solutions, since the domain is non-empty.*

Proof. From Lemma 5.1, when all elements of vector $\hat{\mathbf{x}}$ are equal (i.e., $\hat{\mathbf{x}}$ is a constant vector), the following equation always holds:

$$W^{(k)} \cdot \hat{\mathbf{x}} = 0, \;\; k = 1, \cdots, K.$$

This equation demonstrates that all the constraints are satisfied for a stationary trajectory, thus leading to the attainment of a feasible solution. □

Theorem 5.6. *The quadratic programming problem in step 2, (5.17a) to (5.17f), always has feasible solutions, since the domain is non-empty.*

Proof. Based on the formulation of the quadratic programming problem in step 2, it is apparent that any solution obtained in step 1 is inherently feasible for step 2. That is, the half-smoothed trajectories are in the domain. Consequently, the feasible regions for the quadratic programming problem in step 2 are guaranteed to be non-empty, ensuring the existence of at least one feasible solution for $\tilde{\mathbf{x}}$ such that $\tilde{\mathbf{x}} = \hat{\mathbf{x}}$.

□

5.4.2 Uniqueness of solutions

We then prove that both quadratic programming problems always have unique optimal solutions.

Theorem 5.7. *The quadratic programming problem in step 1, (5.13a) to (5.13c), always has a unique optimal solution, since the objective function is strictly convex.*

Proof. The objective function in (5.13a) can be re-written as $\hat{\mathbf{x}}^\top\hat{\mathbf{x}} - 2\hat{\mathbf{x}}^\top\mathbf{x} + \mathbf{x}^\top\mathbf{x}$, which is strictly convex in the half-smoothed positions, $\hat{\mathbf{x}}$. Thus, together with **Theorem** 5.5, The quadratic programming problem in step 1 always has a unique optimal solution. □

Theorem 5.8. *The quadratic programming problem in step 2, (5.17a) to (5.17f), always has a unique optimal solution, since the objective function is strictly convex.*

Proof. By incorporating (5.17f) into $\tilde{\mathbf{x}}$, the objective function (5.17a) can be rewritten as follows:

$$\begin{aligned}\min[W^{(K)}(\tilde{\mathbf{x}}^{'} + \tilde{\mathbf{x}}^{''})]^\top[W^{(K)}(\tilde{\mathbf{x}}^{'} + \tilde{\mathbf{x}}^{''})] = (W^{(K)}\tilde{\mathbf{x}}^{''})^\top(W^{(K)}\tilde{\mathbf{x}}^{''}) + 2(W^{(K)}\tilde{\mathbf{x}}^{'})^\top(W^{(K)}\tilde{\mathbf{x}}^{''}) \\ + (W^{(K)}\tilde{\mathbf{x}}^{'})^\top(W^{(K)}\tilde{\mathbf{x}}^{'}),\end{aligned}$$

where $\tilde{\mathbf{x}}^{'} = (\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_K, 0, 0, \cdots, 0)^\top$ and $\tilde{\mathbf{x}}^{''} = (0, \cdots, 0, \tilde{x}_{K+1}, \tilde{x}_{K+2}, \cdots, \tilde{x}_M)^\top$, both of which are $(M) \times 1$ vectors and $\tilde{\mathbf{x}} = \tilde{\mathbf{x}}^{'} + \tilde{\mathbf{x}}^{''}$. $(W^{(K)}\tilde{\mathbf{x}}^{'})^\top(W^{(K)}\tilde{\mathbf{x}}^{'})$ is a constant, and its removal does not affect the optimal solution. With $W_1^{(k)}$ and $W_2^{(k)}$ defined in **Corollary** 5.3, the quadratic programming problem can thus be reformulated as follows by defining $\tilde{\mathbf{x}}^{'}_{1:K} = (\hat{x}_1, \cdots, \hat{x}_K)^\top$ and $\tilde{\mathbf{x}}^{''}_{K+1:M} = (\tilde{x}_{K+1}, \tilde{x}_{K+2}, \cdots, \tilde{x}_M)^\top$:

$$\min(W_2^{(K)}\tilde{\mathbf{x}}^{''}_{K+1:M})^\top W_2^{(K)}\tilde{\mathbf{x}}^{''}_{K+1:M} + (W_1^{(K)}\tilde{\mathbf{x}}^{'}_{1:K})^\top(W_2^{(K)}\tilde{\mathbf{x}}^{''}_{K+1:M}).$$

$W_2^{(K)}$ is a triangular matrix with all the entries on its main diagonal being one and it is a full-rank matrix. Thus $(W_2^{(K)})^\top W_2^{(K)}$ is invertible. Consequently, $(W_2^{(K)})^\top W_2^{(K)}$ is positive definite, ensuring the strict convexity of the objective function (Nocedal and Wright, 2006). Thus, together with **Theorem** 5.6, (5.17a) to (5.17f) always possess a unique optimal solution. □

5.4.3 Computational complexity regarding the highest order of derivatives

Both quadratic programming problems in our method can be written in the following format:

$$\min_{\mathbf{x}} \mathbf{x}^\top G\mathbf{x} + \mathbf{c}^\top \mathbf{x}, \tag{5.18a}$$

$$\text{s.t. } Q\mathbf{x} \leq \mathbf{b}, \tag{5.18b}$$

where $\mathbf{x}$ is the vector of variables, G is an $n_1 \times n_1$ positive definite matrix and Q is an $n_2 \times n_1$ matrix with $\text{rank}(Q) = n_1$. In our scenario, n_1 is a constant and n_2 changes according to the highest order of derivatives (value of K). Interior point methods are quite reliable and work well in solving convex optimization problems (Boyd et al., 2004). The barrier algorithm is a type of interior point method that has been widely used in embedded optimization including IBM ILOG CPLEX (Lima and Seminar, 2010), GUROBI (Gurobi Optimization, LLC, 2023). We discuss the time complexity based on the barrier algorithm.

For a quadratic programming problem with linear constraints, with properly chosen parameters, the number of iterations required to solve the problem is proportional to the number of inequality constraints. In the context of (5.18a) and (5.18b), the iteration complexity for converging to a point with a desired accuracy grows like $O(\sqrt{n_2})$ (Boyd et al., 2004). Consequently, in the context of our method where the number of detected positions in one trajectory (value of M) is a constant and $M \gg K$, the theoretical iteration complexity for both steps 1 and 2 grows like $O(\sqrt{2KM})$. It is worth noting that the aforementioned iteration complexity generally represents an upper bound, and the actual performance may be faster in practice, depending on the problem's structure (Floudas and Visweswaran, 1995).

In this study, achieving a balance between the precision of the smoothed data and computational complexity is important. Therefore, we should include the minimum number of inequality constraints when the discrepancy in the smoothed data is negligible.

5.5 Calibration, comparison, and application with NGSIM and highD data

In this section, we first choose the highest order of derivatives (the value of K) to be included in our method. We determine the highest order of derivatives to include in our method by comparing the NGSIM I-80 trajectories (FHWA, 2007) smoothed by our method adopting different highest order of derivatives with the trajectories obtained from manual re-extraction. We then smooth the NGSIM I80 camera 6 trajectories with the proposed method adopting the chosen K and compare our method with a state-of-the-art method. In addition, we apply our method to the new-released highD data (Krajewski et al., 2018) to show its robustness.

The bounds of speeds are set according to the speed limits of the study sites. Taking reference of previous studies, we choose $[-4,5]$ m/s^2 as the feasible acceleration range (Elert, 2012; Bokare and Maurya, 2017) and $[-8,8]$ m/s^3 as the feasible jerk range (Martinez and Canudas-de Wit, 2007). To the best of our knowledge, there have been no experimental investigations on vehicle snap bounds. Thus we make the ratio of the maximum absolute snap to the maximum absolute jerk approximately equal to the ratio of the maximum absolute jerk to the maximum absolute acceleration in this study, and choose $[-12,12]$ m/s^4 as the feasible snap range.

5.5.1 Calibration of the highest order of derivatives

(Coifman and Li, 2017a,b) manually re-extracted the positions recorded by camera 6 on the I80 freeway from 4:00 pm to 4:15 pm on April 13, 2005. The released video (Coifman and Li, 2017b) shows that many errors in raw data are addressed, and their re-extracted data have been used as the ground truth data (Dong et al., 2021). As (Coifman and Li, 2017a) suggested, we calculate the positions, accelerations, and jerks with the new speeds and construct the ground truth dataset.

Because the speed limit on the I80 freeway is 65 mph (28.9 m/s), we set the speed range to be $[0, 30]$ m/s. We also determine the bound of position errors prior to applying our method. According to the features of the NGSIM I80 data, the bound of the absolute prior position error is 1.2 m, while the errors in vehicular tracking are notably higher at the beginning (captured by camera 1) and the end (captured by camera 7) of the section, in comparison to the road segments captured by the other cameras (Montanino and Punzo, 2015), and (Coifman and Li, 2017a) showed that the detection errors in trajectories captured by camera 6 mainly fall within a range of 2 feet (0.6 meters). Thus we choose $\varepsilon = 0.6$ m.

Performance of the method with different highest orders of derivatives and without the second step

We pick out all vehicle trajectories from 5 seconds before entering camera 6's area until leaving camera 6's area from the raw NGSIM I80 data, and compare the smoothed trajectories in camera 6 with the manually re-extracted data. We adopt K values from 1 to 4, apply our method, and solve the quadratic programming problems with GUROBI (Gurobi Optimization, LLC, 2023) via CVXPY (Diamond and Boyd, 2016). We determine the appropriate K by comparing the mean squared errors (MSEs) between the smoothed data and the manually re-extracted data, as well as the computational cost.

The MSEs between the half-smoothed and manually re-extracted data, and the smoothed and manually re-extracted data are displayed in **Table** 5.2. The MSEs between the raw and manually re-extracted positions and speeds are 1.91 m^2 and 0.88 m^2/s^2, respectively, which are not significant. However, such MSEs significantly increase in higher order derivatives, as the MSE between the raw and manually re-extracted jerks reaches 9960 m^2/s^6. The MSEs between the half-smoothed and manually re-extracted data continuously decrease as K increases. $K = 1$ has little effect on the MSEs, however, when $K \geq 2$, the MSEs between the half-smoothed and the manually re-extracted data are greatly reduced, especially in higher order derivatives. The MSE between the

half-smoothed and manually re-extracted jerks is reduced to 7.86 m^2/s^6 when $K = 4$, representing a reduction of greater than 99.9%.

Table 5.2: MSEs between half-smoothed and manually re-extracted data, and smoothed and manually re-extracted data adopting different K

variables		Raw data	$K = 1$	$K = 2$	$K = 3$	$K = 4$
Step 1	$E((\tilde{\mathbf{x}} - \bar{\mathbf{x}})^2)$ (m^2)	1.91	1.91	1.91	1.90	1.90
	$E((\tilde{\mathbf{x}}^{(1)} - \bar{\mathbf{x}}^{(1)})^2)$ (m^2/s^2)	0.88	0.87	0.60	0.54	0.50
	$E((\tilde{\mathbf{x}}^{(2)} - \bar{\mathbf{x}}^{(2)})^2)$ (m^2/s^4)	44.56	44.09	5.09	1.79	1.05
	$E((\tilde{\mathbf{x}}^{(3)} - \bar{\mathbf{x}}^{(3)})^2)$ (m^2/s^6)	9960.09	9826.03	499.38	40.98	7.86
Step 1 + Step 2	$E((\tilde{\mathbf{x}} - \bar{\mathbf{x}})^2)$ (m^2)	1.91	2.12	**1.80**	1.87	1.95
	$E((\tilde{\mathbf{x}}^{(1)} - \bar{\mathbf{x}}^{(1)})^2)$ (m^2/s^2)	0.88	0.40	**0.29**	0.32	0.40
	$E((\tilde{\mathbf{x}}^{(2)} - \bar{\mathbf{x}}^{(2)})^2)$ (m^2/s^4)	44.56	4.59	0.26	**0.25**	0.34
	$E((\tilde{\mathbf{x}}^{(3)} - \bar{\mathbf{x}}^{(3)})^2)$ (m^2/s^6)	9960.09	693.29	1.50	**0.63**	**0.63**

As shown in **Table** 5.2, the incorporation of step 2 in addition to step 1 results in a substantially greater reduction in MSEs compared to the application of step 1 alone, which suggests that the incorporation of step 2 makes the smoothed data more correct. The minimum MSEs between our smoothed and manually re-extracted positions, speeds, accelerations, and jerks, corresponding to the K values, are highlighted in bold. Upon including step 2, $K = 2$ yields the smallest MSEs in positions and speeds, $K = 3$ produces the lowest MSEs in accelerations, and both $K = 3$ and $K = 4$ lead to the lowest MSE in jerks. The MSEs between our smoothed and manually re-extracted positions, speeds, and accelerations exhibit little difference when adopting either $K = 2$ or $K = 3$, while the MSE between our smoothed and manually re-extracted jerks decreases by more than half when adopting $K = 3$ in comparison to $K = 2$. Therefore, we will choose $K = 3$ if the computation complexity is acceptable.

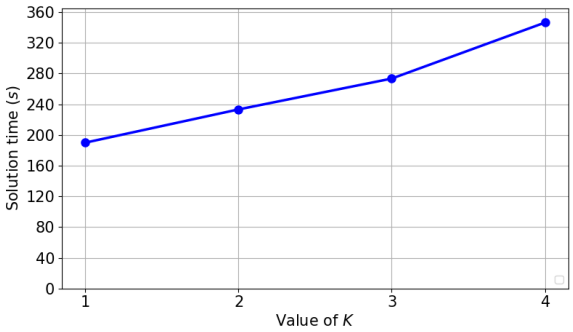


Figure 5.3: The computation time for smoothing the NGSIM I80 camera 6 dataset

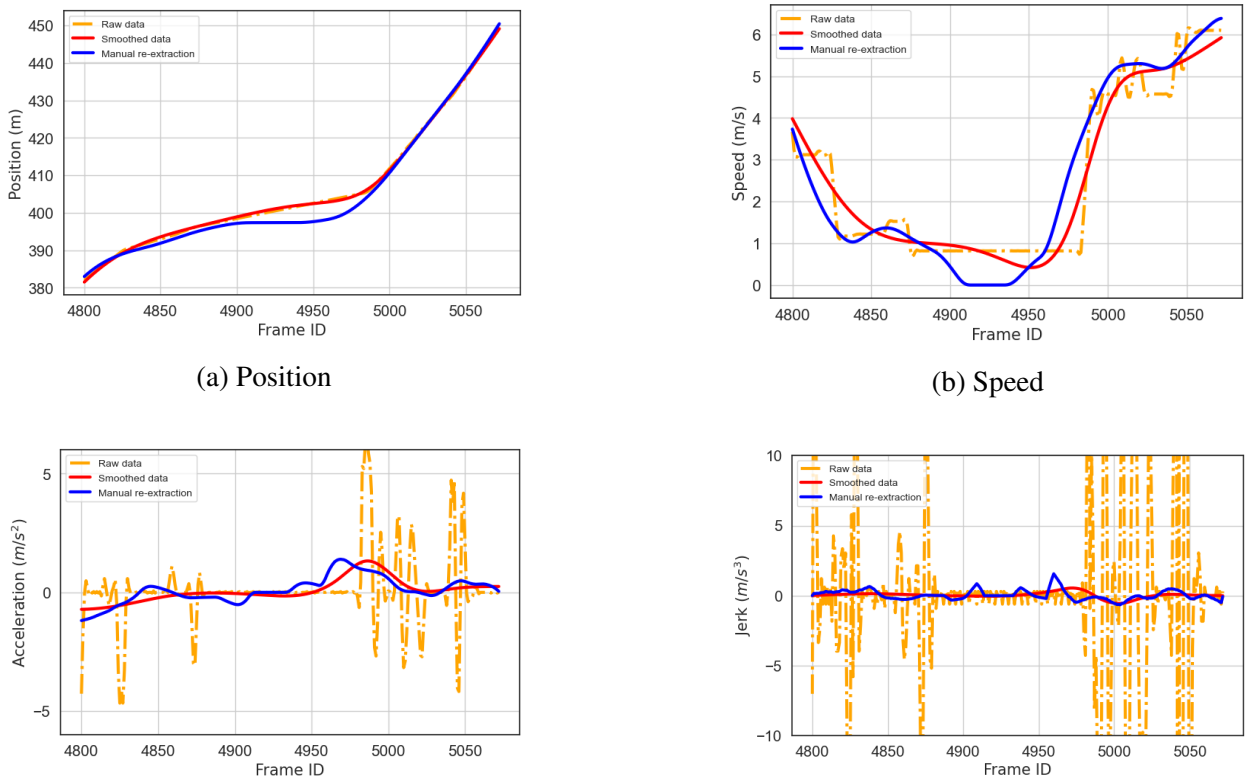
On the other hand, Figure 5.3 presents the duration required for smoothing the NGSIM I80 camera 6 dataset with the two-step method, utilizing an Asus desktop equipped with an Intel Core i7 2.9 GHz CPU. The time required to solve the quadratic programming problem grows as K increases. The smoothing process for the dataset only takes approximately 340 seconds (5 minutes and 40 seconds) even when employing $K = 4$. Collaboratively considering the MSEs between the smoothed data and the manually re-extracted data, as well as the computation cost, we choose $K = 3$ for the subsequent applications. It means that we will minimize the sum of squared jerks with bounded positions, speeds, accelerations, and jerks in step 2.

An example with the two-step method and selected highest order of derivatives

We selected vehicle 1486 from the NGSIM I80 camera 6 dataset, which was previously utilized as an example in (Coifman and Li, 2017a), to implement our method with $K = 3$. The results are depicted in Figure 5.4, where the raw data are shown in yellow, the smoothed data are shown in red, and the manually re-extracted data are shown in blue.

As shown in Figure 5.4a, the disparities between the raw and smoothed positions are relatively minor. However, these differences become more obvious in the speeds, accelerations, and jerks, as illustrated by Figure 5.4b to Figure 5.4d. Notably, many outliers and noises appear in raw accelerations and jerks, particularly in raw jerks. Upon applying our method, all outliers and high-frequency noises are effectively eliminated. The main features in the raw trajectories, including the accelerating and decelerating behaviors, are preserved. In addition, our method effectively removes low constant speeds, which are considered unrealistic based on experiments with probe vehicles (Coifman and Li, 2017a), in the raw data, and the smoothed data obtained from our method closely resemble the manually re-extracted data.

The frequency spectrums of accelerations and jerks of vehicle 1486 before and after applying our method are shown as Figure 5.5. As indicated by the yellow dashed curves, the raw accelerations



(a) Position

(b) Speed

(c) Acceleration

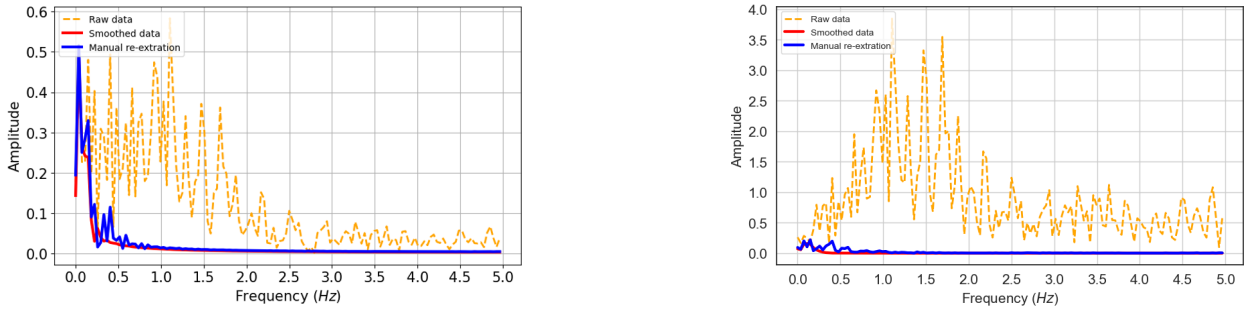
(d) Jerk

Figure 5.4: Positions, speeds, accelerations, and jerks of vehicle 1486

and jerks encompass numerous high-frequency noises that can span the entire frequency range. The red curves show that our method can effectively eliminate these high-frequency noises, resulting in the frequency spectrums of our smoothed data being identical to those of the manually re-extracted data, which are shown by blue curves. The frequency spectrums of our smoothed data are in line with the reasonable human operational range (up to about 2 Hz), as suggested by (Punzo et al., 2011; Zhou et al., 2020).

5.5.2 Comparison with an existing method with respect to manually re-extracted data

We still adopt the manually re-extracted data from (Coifman and Li, 2017a) for comparison (referred to as "manual re-extraction" hereafter). The trajectories smoothed by the multistep opti-



(a) Acceleration

(b) Jerk

Figure 5.5: Frequency spectrums of the accelerations and jerks of vehicle 1486

mization method in (Montanino and Punzo, 2015) have also been released, which provide a great source of comparative data. We adopt them (referred to as "multistep optimization" hereafter) for comparing with the trajectories smoothed by our method. Therefore, we have four sources of data for comparison: (1) raw data, (2) multistep optimization, (3) our method, and (4) manual re-extraction.

The statistic summary of the aforementioned methods is presented in **Table** 5.3. We compute the means and standard deviations for position shifts, as well as the means, standard deviations, ranges, and proportions of outliers for the speeds, accelerations, and jerks in the aforementioned data sources. We also calculate the MSEs between the raw data, as well as the data smoothed by different methods, and the manually re-extracted data as the evaluation metrics.

The average shifts between the raw positions and the positions smoothed by both methods are about zero, while the average position error of the raw positions, which represents the difference between the raw positions and the manually re-extracted positions, amounts to 0.12 m. Compared to raw positions, the multistep optimization method can lead to a larger MSE between the smoothed positions and the manually re-extracted positions. Conversely, the MSE between the positions smoothed by our method and the manually re-extracted positions exhibits a slight reduction compared to the raw positions.

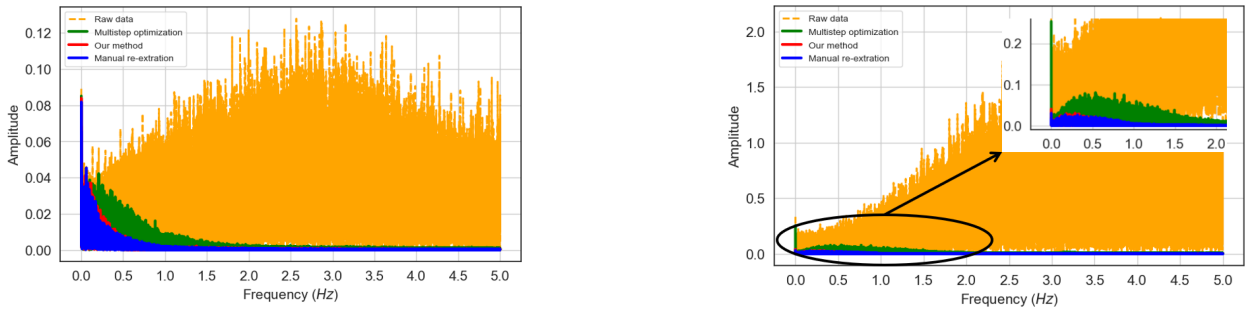
The means of speeds, accelerations, and jerks are preserved by all the aforementioned methods. All

raw speeds are physically meaningful. However, 15.0% and 42.3% of the raw accelerations and jerks are outliers, respectively. The absolute values of the raw accelerations and jerks can almost reach 300 m/s^2 and 4200 m/s^3, respectively. The MSEs between the raw data and the manually re-extracted data reach 44.56 m^2/s^4 and 9960.09 m^2/s^6 in the accelerations and jerks, respectively. The multistep optimization method can reduce the proportion of outliers in accelerations to less than 0.05% and reduce the proportion of outliers in jerks to 0.6%. All outliers in accelerations and jerks are eliminated by our method. The standard derivations of the accelerations and jerks smoothed by our method are the closest to those of the manually re-extracted accelerations and jerks. The positions, speeds, accelerations, and jerks smoothed by our method can better resemble those manually re-extracted data, as the MSEs between our smoothed data and the manually re-extracted data are smaller.

Table 5.3: Comparison of different methods

Variables		Raw data	Multistep optimization	Iterative moving average	Our method	Manual re-extraction
Position (m)	$E((\tilde{\mathbf{x}}-\bar{\mathbf{x}})^2)$ (m^2)	1.91	2.18	2.04	1.87	/
Speed (m/s)	Mean	8.07	8.05	8.07	8.04	7.88
	Std	4.07	4.01	3.97	3.99	3.89
	Range	[0,36.0]	[0,27.0]	[0,26.7]	[0,27.2]	[0,26.4]
	Outliers (%)	0	0	0	0	0
	$E((\tilde{\mathbf{x}}^{(1)}-\bar{\mathbf{x}}^{(1)})^2)$	0.88	0.47	0.37	0.32	/
Acceleration (m/s^2)	Mean (m/s^2)	-0.04	-0.04	-0.05	-0.03	0.03
	Std	6.69	0.92	0.66	0.6	0.58
	Range	[-176.5,292.2]	[-14.1,4.5]	[-5.0,4.0]	[-4.8,4.0]	[-4.2,3.5]
	Outliers (%)	15.0	0	0	0	0
	$E((\tilde{\mathbf{x}}^{(2)}-\bar{\mathbf{x}}^{(2)})^2)$	44.56	0.68	0.4	0.25	/
Jerk (m/s^3)	Mean (m/s^3)	-0.17	-0.13	-0.06	-0.02	-0.02
	Std	99.80	2.41	0.97	0.59	0.62
	Range	[-4171.5,2954.4]	[-141.0,39.4]	[-7.8,8.0]	[-8.0,8.0]	[-5.3,7.1]
	Outliers (%)	42.3	0.6	0	0	0
	$E((\tilde{\mathbf{x}}^{(3)}-\bar{\mathbf{x}}^{(3)})^2)$	9960.09	5.9	1.12	0.63	/

The frequency spectrums of accelerations and jerks smoothed or re-extracted by different methods are shown in Figure 5.6. The yellow dashed curves show that the raw accelerations and jerks include the noises that span the entire frequency range. All these methods can eliminate high-frequency noises, and the frequency of the smoothed accelerations and jerks are within the human-vehicle response frequency (about 2 Hz) (Punzo et al., 2011). Compared to the multistep optimization method, the frequency spectrums of the data smoothed by our method are closer to those of the manually re-extracted data.

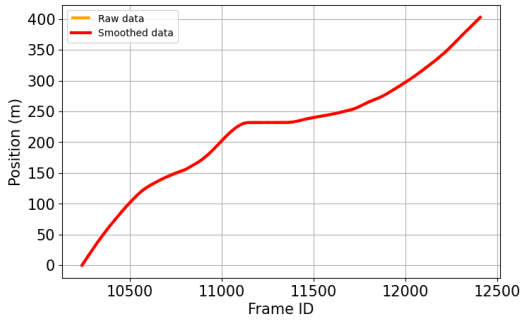


(a) Acceleration (b) Jerk

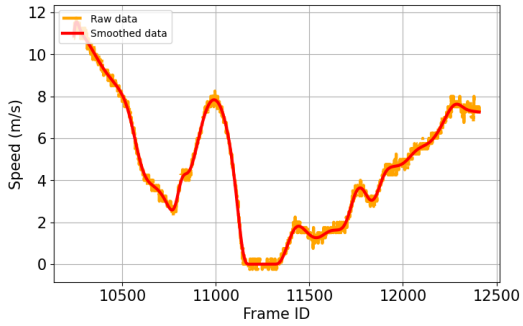
Figure 5.6: Frequency spectrums of the accelerations and jerks of NGSIM I80 camera 6 data

5.5.3 Application to the highD data

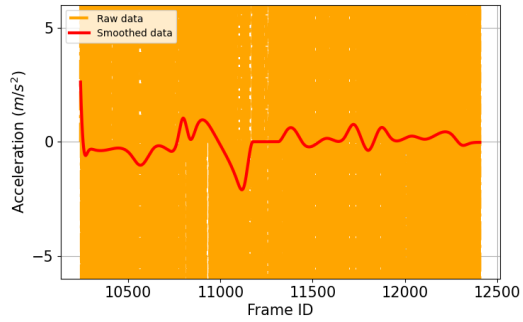
We also apply our method to the highD (Krajewski et al., 2018) data, newly-released vehicle trajectory data collected by drone-equipped high-resolution cameras at a segment of more than 400 meters with a prior position error of 0.1 meters. There is no speed limit at the study site, so we choose $[0,50]$ m/s, a typical constraint of vehicle mechanics, as the feasible speed range. We adopt the same acceleration and jerk ranges as before. There are 60 subsets in total, and we use the "25-tracks" subset as an example. This dataset include the trajectories of 2850 vehicles on a three-lane freeway.



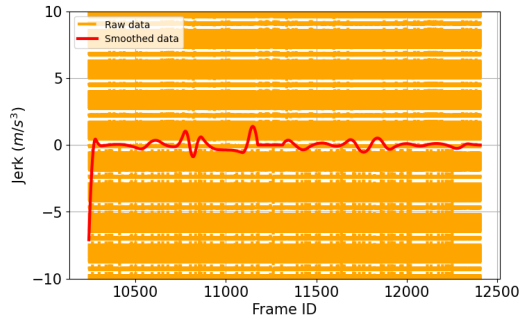
(a) Position



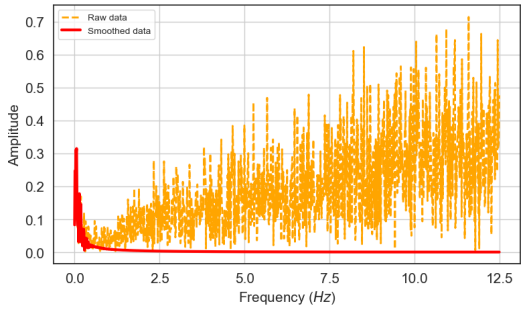
(b) Speed



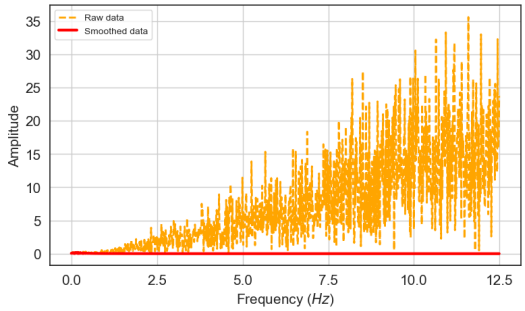
(c) Acceleration



(d) Jerk



(e) Frequency spectrum of accelerations



(f) Frequency spectrum of jerks

Figure 5.7: Positions, speeds, accelerations, jerks, and frequency spectrums of the accelerations and jerks of vehicle 1011

Using vehicle 1011 which experiences accelerating, decelerating, and stopping processes during the recording as an illustrative example, we demonstrate the effects of our method. Figure 5.7a to Figure 5.7d present the positions, speeds, accelerations, and jerks before and after smoothing, with raw data shown as yellow dashed curves and smoothed data represented by red curves. It is observed that the raw speeds exhibit numerous noises, which are translated into outliers and larger

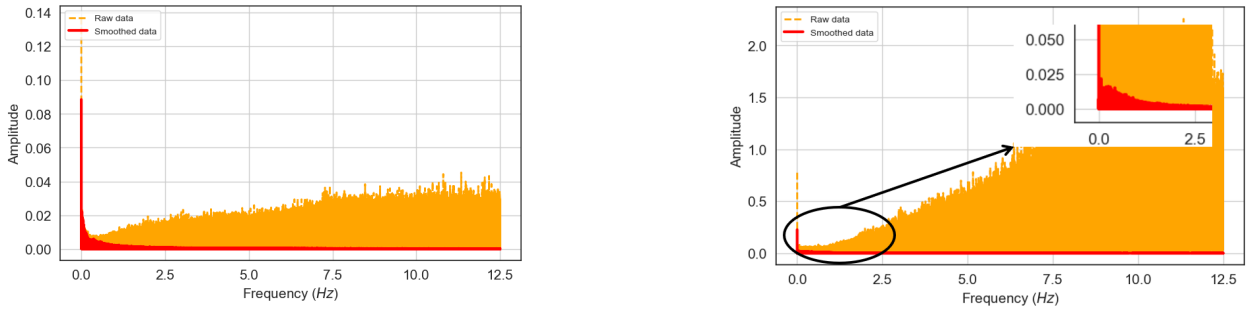
noises in the higher order derivatives. The impacts of our method on the highD data are identical to those on the NGSIM data. Our method effectively eliminates these outliers and high-frequency noises in the higher order derivatives while ensuring bounded modifications of positions, thereby preserving the general trend and main features of the raw speeds. The removal of high-frequency noises can also be reflected by Figure 5.7e and Figure 5.7f, where any noise beyond the reasonable range of human operation (approximately up to 2 Hz) is eliminated.

The statistic summary of the raw and smoothed data is presented in **Table** 5.4. The mean absolute shift between the smoothed and raw positions is 0.05 meters, with a maximum absolute position shift being 0.9 meters. This aligns with the claim that the prior position error of the highD data is 0.1 meters. The range of raw speeds is $[-6.0, 35.3]$ m/s, with about 0.1% of speeds being negative. Our method successfully eliminates all outliers while maintaining the average of the raw speeds. Outliers constitute 57.9% and 74.2% of the raw accelerations and jerks, respectively, with their ranges extending up to $[-200.0, 231.2]$ m/s^2 and $[-7656.2, 9062.5]$ m/s^3. Our method effectively removes these outliers, resulting in the revised ranges of the accelerations and jerks as $[-4.6, 4.0]$ m/s^2 and $[-8.0, 8.0]$ m/s^3, respectively.

Table 5.4: Statistic summary of the raw and smoothed highD 25-tracks dataset

Dataset		Variables	Position shift (m)	Speed (m/s)	Acceleration (m/s^2)	Jerk (m/s^3)
HighD	Raw data	Mean	/	13.87	0.07	-0.39
		Std	/	7.70	6.45	267.76
		Range	/	[-6.0,35.3]	[-200.0, 231.2]	[-7656.2,9062.5]
		Outliers (%)	/	0.1	57.9	74.2
	smoothed data	Mean	0	13.87	0.04	-0.11
		Std	0.06	7.70	0.54	0.92
		Range	[-0.9,0.8]	[0,34.7]	[-4.6,4.0]	[-8.0,8.0]
		Outliers (%)	/	0	0	0
		$E(\lvert\tilde{\mathbf{x}}-\mathbf{x}\rvert)$	0.05	/	/	/

The frequency spectrums of the accelerations and jerks before and after smoothing are shown as Figure 5.8. Similar to the application to the NGSIM data, our method makes the accelerations and jerks in the highD dataset free of high-frequency noises. The frequencies of all smoothed accelerations and jerks fall within the reasonable human operational range (up to 2 Hz).



(a) Acceleration

(b) Jerk

Figure 5.8: Frequency spectrums of the accelerations and jerks of highD "25-tracks" data

5.6 Conclusion

In this study, we proposed a two-step quadratic programming method for smoothing longitudinal vehicle trajectory data. In the first step, we minimize the discrepancy between the half-smoothed and raw positions, while adhering to physically meaningful bounds on the speeds and higher order derivatives of the half-smoothed positions. Subsequently, in the second step, our objective is to minimize the roughness of the smoothed positions, while incorporating the position ranges as additional constraints alongside those imposed in the first step. This step allows the smoothed positions to deviate from the raw data by at most those of the half-smoothed positions and the prior position error, and make the initial few positions in the trajectory the half-smoothed positions to ensure the strict convexity of the objective function. We analytically proved that both quadratic programming problems always yield unique optimal solutions, guaranteeing the well-defined nature of our proposed method. This is achieved by demonstrating that, in both steps, the domains defined by linear inequality constraints are non-empty and that the objective functions are strictly convex.

Numerically, we employed interior point methods to solve the quadratic programming problems, with the computational complexity increasing with the number of positions and the highest order of derivatives. Using NGSIM data, we demonstrated that a highest order of three yields an efficient method and smoothed trajectories comparable to manually re-extracted ones. The results of applying our method to a sample trajectory demonstrate its effectiveness in removing outliers and high-frequency noises in higher order derivatives, with only modest adjustments to the positions. Additionally, we compared our method with an existing approach and show its superior performance in terms of leading to smaller MSEs between the smoothed positions, speeds, accelerations, and jerks and those manually re-extracted. Furthermore, we applied our method to the smoothing of the recently released highD dataset, showcasing the robustness of our approach in handling trajectory data from diverse sources. Our method effectively aligns the smoothed trajectories with the vehicle characteristics and drivers' behaviors inherent in the dataset.

The contributions of this study are four fold. Firstly, we define speeds and higher order derivatives as symplectic differences in positions, carefully discuss the properties of resulted difference matrices, and present linear inequality constraints to ensure bounded derivatives of positions. These definitions, formulations, and properties form the foundation of the ensuing formulation of a well-defined two-step quadratic programming method. Secondly, we introduce a novel method that integrates multiple factors, including discrepancy, roughness, bounded speed and higher order derivatives, and prior position errors, to effectively smooth longitudinal vehicle trajectory data. We provide numerical evidence that demonstrates the practical applicability and effectiveness of our proposed method. Thirdly, we offer theoretical justification for the existence and uniqueness of the optimal solutions in our method. This theoretical analysis establishes a solid foundation for the reliability and robustness of our approach, further supporting its practical implementation. Fourthly, we demonstrate that a highest order of three yields an efficient method and smoothed trajectories comparable to manually re-extracted ones; this finding is consistent with the minimun jerk principle for human body movements (Flash and Hogan, 1985) and the well-established smoothing splines method by (Whittaker, 1922).

In the second step, we incorporate additional equality constraints by ensuring that the initial smoothed positions match the corresponding half-smoothed positions. The number of these constraints is equal to the highest order of derivatives (K), guaranteeing the strict convexity of the objective function. It is worth noting that, in (Whittaker, 1922), additional equality constraints were introduced using $K = 3$ moments. In our future research, we aim to explore the possibility of incorporating these equality constraints into our method, potentially resulting in strictly convex objective functions and comparable smoothing outcomes.

In the future, we intend to apply our method to smooth longitudinal vehicle trajectory data collected from other sources. Additionally, we aim to incorporate different boundedness criteria in the constraints to smooth lateral vehicle trajectory data. Moreover, the concepts introduced in this study open avenues for investigating scenarios involving missing portions of trajectories, requiring the imputation of missing values. We are particularly interested in predicting and planning trajectories to ensure safe, efficient, and environmentally friendly operations for both human-driven and automated vehicles.

In our method, the prior position error functions similarly to the weight in the objective function of the smoothing splines method (Whittaker, 1922). Nevertheless, it is crucial to emphasize that the prior position error carries substantial physical significance within our specific context and can be directly derived from observational data. Consequently, we assert that our two-step quadratic programming method holds applicability for broader smoothing problems, encompassing situations where traditional splines methods are employed and the prior error in the raw data can be reasonably estimated or observed in advance.

Chapter 6

Quadratic programming method for imputation using fixed and mobile sensor data

6.1 Introduction

Empirical investigations have demonstrated the effectiveness of the minimum jerk theory in accurately describing human movements. (Flash and Hogan, 1985) observed that individuals tend to execute arm movements that minimize jerks, and (Othman et al., 2008) discovered a positive correlation between increased stress levels in drivers and higher jerks experienced during motion. (Whittaker, 1922) proposed the use of the sum of squared third-order derivatives, the jerks in the context of vehicle trajectories, to measure the roughness of the smoothed data. The minimum jerk theory has proven effective in smoothing vehicle trajectories by eliminating outliers and high-frequency noises while maintaining consistency with both vehicle characteristics and human driving behaviors. The minimum jerk theory has also found great application in the trajectory

design of drones and robots (Mellinger and Kumar, 2011; Gasparetto and Zanotto, 2007, 2008, 2010).

In this study, we propose a three-step quadratic programming method to impute the fully sampled longitudinal vehicle trajectories from multi-source trajectory data corporately considering the following principles: (P1) Maintaining a safe inter-vehicle spacing. (P2) Ensuring that speeds, accelerations, and jerks remain within physically meaningful ranges. (P3) Considering drivers' tendency to minimize jerks during their movements. The scenario of the problem we discuss is shown in Figure 6.1, where two loop detectors are located at the entrance and the exit of the segment. Mixed traffic comprising of CAVs (marked in red), CVs (marked in green), and HDVs (marked in blue and black) is considered, where CVs can provide their individual trajectories, while CAVs can offer not only their own trajectories but also the trajectories of its preceding and following vehicles. However, for HDVs not recorded by CAVs, we only have the information on their entry and exit times, requiring us to impute their trajectories. The trajectories to be imputed are shown as Figure 6.2, once the detected trajectories of two vehicles (marked in blue) and the entering and exiting times are available, we can impute the trajectories in between. Our method involves solving a quadratic programming problem in each step, with constraints ensuring physically meaningful ranges for speeds, accelerations, and jerks. In step 1, we calculate the fastest possible trajectory that maintains a safe distance from the preceding vehicle. In step 2, we calculate the slowest possible trajectory that maintains safe distance from the following vehicles. Finally, we compute the trajectory that lies between the slowest and fastest possible trajectories, optimizing for the minimal sum of squared jerks. We numerically show the efficacy of our method with the NGSIM (FHWA, 2007) and highD data (Krajewski et al., 2018). We consider scenarios with one, two, three, and four undetected vehicles between a leading and a trailing vehicle, and impute the trajectories of the undetected vehicles. Additionally, we demonstrate the efficacy of our method in a traffic system comprising 10% CVs and 10% CAVs, showcasing its suitability in a mixed-traffic environment.

It should be noted that we focus on longitudinal trajectories and do not consider lane-changing behaviors in this study. If the number of the vehicles between two detected vehicles remains the same when entering and exiting the segment, we consider no lane-changing behavior happens and our method can be applied.

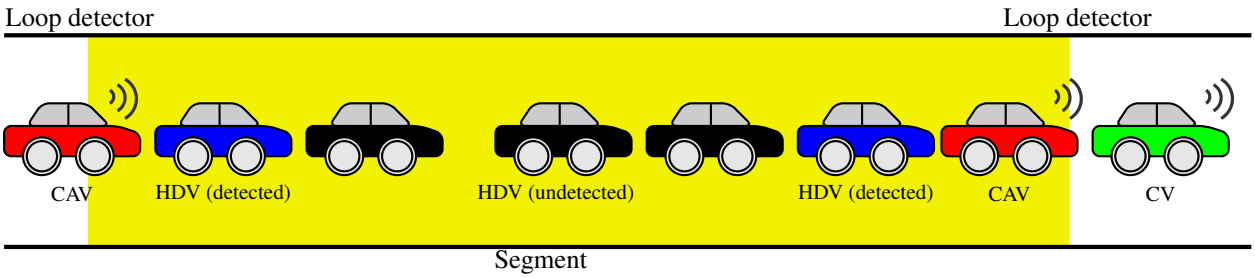


Figure 6.1: Illustration of the mixed traffic scenario

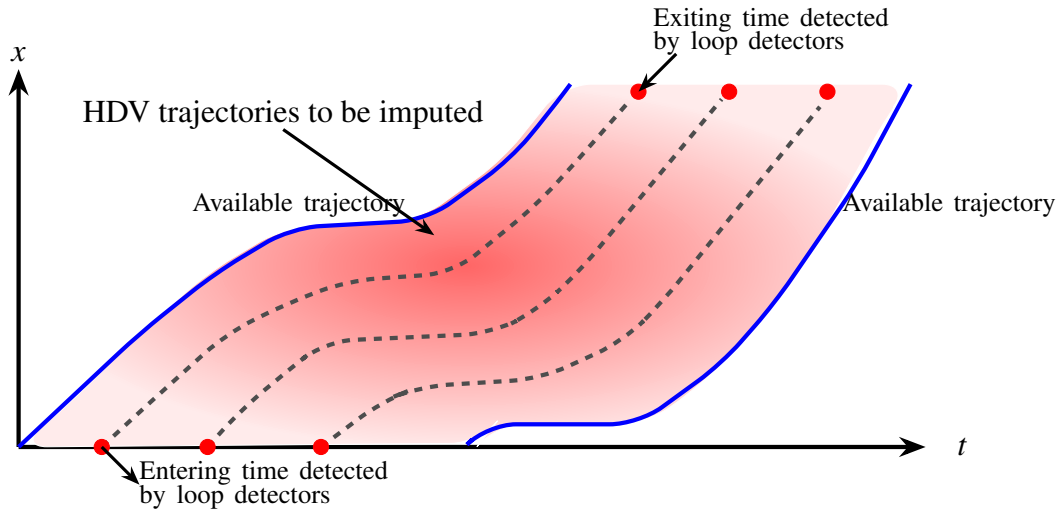


Figure 6.2: Illustration of the trajectories to be imputed

The remaining sections of this chapter are structured as follows. Section 5.2 offers an introduction to internally consistent positions and higher-order derivatives, alongside a presentation of Newell's simplified car-following model. In Section 5.3, we present our quadratic programming trajectory imputation method and outline the procedures for determining the parameters. Section 5.4 presents numerical examples using the NGSIM and highD data. Finally, in Section 5.5, we discuss the contributions and limitations of our study and propose potential extensions for future research.

6.2 Symplectic discretization scheme of positions and Newell's simplified car-following model

In this section, we begin with introducing the symplectic discretization scheme for the positions of vehicles, followed by formulating speeds, accelerations, and jerks as linear combinations of positions and introducing linear inequality constraints based on their physically meaningful bounds. We then introduce Newell's simplified car-following model which is later adopted for calculating safe inter-vehicle spacing.

6.2.1 Symplectic discretization scheme of positions

From a physical standpoint, the derivatives of positions correspond to speeds, while the integrals of speeds yield positions, and likewise for higher-order derivatives. The discretization scheme of position was introduced in section 4.2.1, thus we omit it here.

6.2.2 Newell's simplified car-following model

Newell's simplified car-following model (Newell, 2002) assumes that a following vehicle aims to advance as much as possible while concurrently ensuring a minimum safe inter-vehicle spacing from the leading vehicle, all within the constraints of the speed limit. In our study, we adopt the simplified Newells car-following model to determine the ranges of the positions to be imputed as it offers several advantages, including its minimal parameter requirements, mathematical tractability, and demonstrated consistency with real-world observations (Ahn et al., 2004). Newell's simplified car-following model can be written as follows:

$$x^n(t+\tau_+^n) = \min\{x^{n-1}(t) - \zeta_+^n, x^n(t) + x_+^{(1)}\tau_n^+ \}, \tag{6.1}$$

where τ_+^n and ζ_+^n are the time gap and jam spacing between vehicle n and vehicle $n-1$, $x^n(t)$ is the position of vehicle n at time t, and $x_+^{(1)}$ is the upper bound of the first order derivatives of positions, that is, the speed limit. The model has two phases: the free flow phase and the following phase. In the free flow phase, the vehicle travels at its speed limit: $x^n(t+\tau_n^+) = x^n(t) + x_+^{(1)}\tau_+^n$; while in the following phase, the vehicle experiences constraints imposed by its leading vehicle, necessitating the maintenance of a minimum distance, known as the jam spacing, from the leading vehicle: $x^n(t+\tau_+^n) = x^{n-1}(t) - \zeta_+^n$.

It should be noted that Newell's simplified car-following model does not consider the boundedness of the higher-order derivatives of positions; instead, such boundedness will be incorporated into the constraints of our optimization problems. We will introduce this in detail in the following section.

6.3 Three-step quadratic programming method for imputation

In this section, we first introduce the flowchart illustrating the three-step methodology employed for vehicle trajectory imputation. Subsequently, each step is explained in detail to provide a comprehensive understanding of the proposed approach. We then introduce the process of determining the parameters associated with Newell's simplified car-following model, i.e. jam spacing and time gap.

6.3.1 Introduction of the proposed method

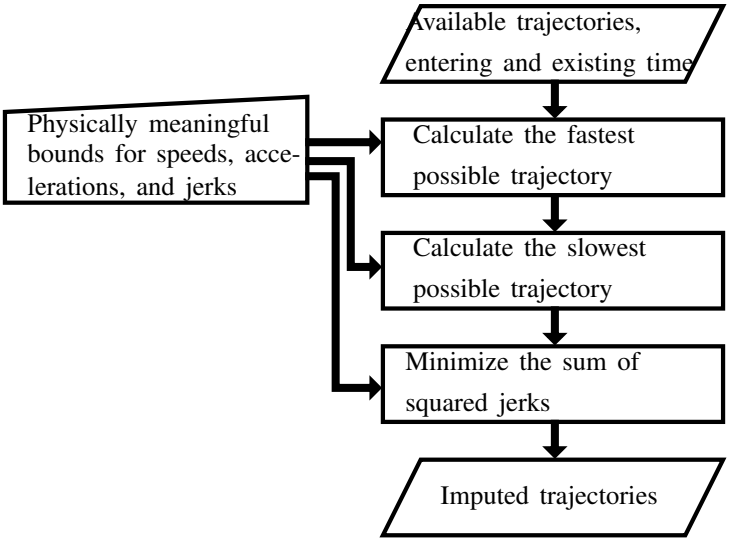


Figure 6.3: The flow chart of the proposed method

The framework and concept of our proposed method are depicted in Figure 6.3 and Figure 6.4, respectively. The scenario under consideration encompasses a heterogeneous mix of vehicles, including CAVs, CVs, and HDVs, as illustrated in Figure 6.1. We focus on a typical scenario involving a platoon of vehicles, where the trajectories of the leading and trailing vehicles are known, while the objective is to impute the trajectories for the intermediate vehicles.

Our method consists of three steps, depicted as rectangles in Figure 6.3. The imputation process for each vehicle within the platoon follows a sequential order, commencing from the foremost vehicle to be imputed and progressing towards the rear. This approach ensures a structured and organized trajectory estimation process. For one trajectory to be imputed, the inputs consist of the nearest available trajectories of its leading and trailing vehicles, along with the entering and exiting times of the intermediate vehicles between the aforementioned leading and trailing vehicles obtained from loop detectors, as indicated by the blue curves and red nodes in Figure 6.4. For simplicity, the illustration in Figure 6.4 omits the available trajectories that will not be utilized for imputation. The red curves illustrate the fastest and slowest possible trajectories, while the gray curve corresponds

to the trajectory that minimizes the sum of squared jerks. This gray curve represents the output of our method, the imputed trajectory.

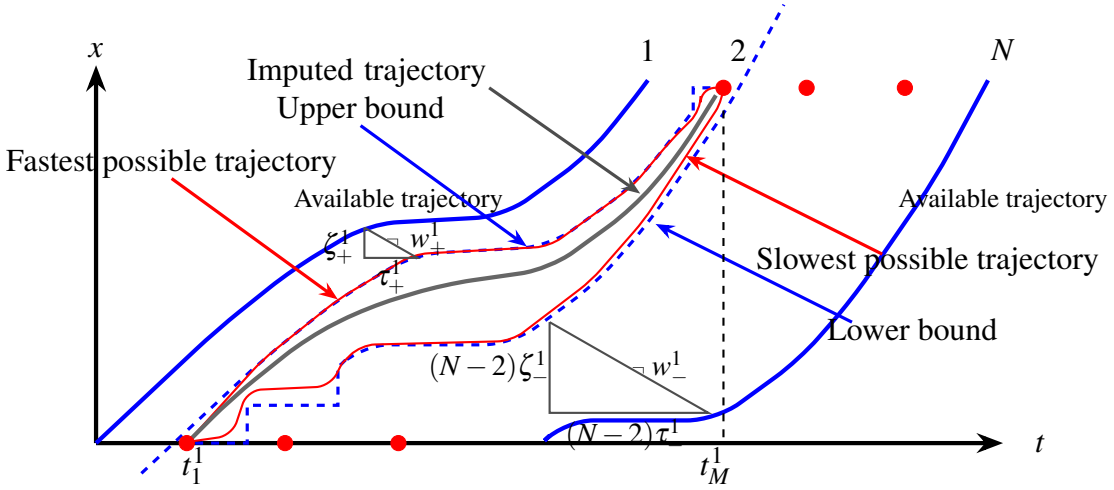


Figure 6.4: Illustration of the safe bounds

Calculating the Fastest Possible Trajectory

In order to satisfy safety regulations, a certain space must be maintained between a vehicle and its leading vehicle. We employ the concept of Newell's simplified car-following model (Newell, 2002) to construct the upper bound of the trajectory to be imputed. This model serves as a foundation for guaranteeing inter-vehicle safety. As shown in Figure 6.4, taking the first HDV that is to be imputed (vehicle 2) as an example, the upper bound is represented by a blue dashed curve, which is a temporal and spatial translation of the trajectory of its leading vehicle (illustrated by the continuous blue curve on the left) by a time gap τ_+^1 and a jam spacing ζ_+^1. By analogy, we can follow the same process and use the trajectory of vehicle 2 to determine the upper bound of the trajectories of subsequent vehicles. Moving forward, we extend this procedure to encompass vehicle n, thereby formulating our method with vehicle n as the target of trajectory imputation.

With the aforementioned procedure, we formulate the upper bound of the trajectory to be imputed. We denote the time vehicle n enters and exits the segment as t_1^n and t_M^n, which can be detected by loop detectors. If the shifted trajectory of the front vehicle ends before the following vehicle

exits the segment ($t_M^n > t_M^{n-1} + \tau_+^n$), we consider the upper bound during the remaining time to be the detected position at t_M^n. Therefore, the upper bound of vehicle n's trajectory can be written as follows:

$$x_+^n(t_m^n) = \begin{cases} x^{n-1}(t_m^n - \tau_+^n) - \zeta_+^n, & \text{when } t_m^n \leq t_M^{n-1} + \tau_+^n \quad (6.2a) \\ x^n(t_M^n), & \text{when } t_m^n > t_M^{n-1} + \tau_+^n, \quad (6.2b) \end{cases}$$

where τ_n^+ and ζ_n^+ are the time gap and jam spacing between vehicle n and vehicle $n-1$, respectively. We use $\mathbf{x}_+^n$ to represent the upper bounds of the trajectory of vehicle n. Then the fastest possible trajectory of vehicle n is the trajectory that is the closest to $\mathbf{x}_+^n$ while making the speeds, accelerations, and jerks with in the physically meaningful ranges, which can be formulated as follows:

$$\min_{\hat{\mathbf{x}}_+^n} (\hat{\mathbf{x}}_+^n - \mathbf{x}_+^n)^T (\hat{\mathbf{x}}_+^n - \mathbf{x}_+^n), \quad (6.3a)$$

$$s.t.$$

$$-\hat{\mathbf{x}}_+^n \leq 0, \quad (6.3b)$$

$$\hat{\mathbf{x}}_+^n \leq \mathbf{x}_+^n, \quad (6.3c)$$

$$-W^{(k)}\hat{\mathbf{x}}_+^n \leq -x_-^{(k)}\Delta t^k, \;\; k = 1,2,3, \quad (6.3d)$$

$$W^{(k)}\hat{\mathbf{x}}_+^n \leq x_+^{(k)}\Delta t^k, \;\; k = 1,2,3, \quad (6.3e)$$

$$\hat{x}_+^n(t_1^n) = x^n(t_1^n), \hat{x}_+^n(t_M^n) = x^n(t_M^n), \quad (6.3f)$$

where $\hat{\mathbf{x}}_+^n$ denote the fastest possible trajectory, and $x_-^{(k)}$ and $x_+^{(k)}$ are the lower and the upper bounds of the kth order derivatives of positions. (6.3b) and (6.3c) ensure that the calculated positions are positive and smaller than the upper bound, (6.3d) and (6.3e) ensure that all speeds, accelerations,

and jerks are bounded, and (6.3f) keeps the detected data by loop detectors.

Calculating the Slowest Possible Trajectory

Vehicle n should also keep safe distance from its followers. This includes two aspects. Firstly, it involves ensuring a safe distance from the available trajectory of the trailing vehicle. Secondly, it necessitates maintaining a safe distance from the positions of its followers when they enter the segment. Similar to the calculation of the upper bound of the trajectory, the lower bound is a translation in time and space of the available trajectory of the trailing vehicle, as shown by the right dashed blue curve in Figure 6.4. The lower bound of vehicle n's trajectory can be written as follows:

$$x_{-}^{n}(t_m^n) = \begin{cases} x^N(t_m^n + (N-n)\tau_{-}^{n}) + (N-n)\zeta_{-}^{n}, & \text{when } t_m^n \geq t_1^N - (N-n)\tau_{-}^{n} \quad (6.4a) \\ \max\{x^n(t_1^n), x^{n+q}(t_1^{n+q}) + q\zeta_{min}\}, & \text{when } t_1^{n+q} - q\tau_{min} \leq t_m^n \leq t_1^{n+q+1} - (q+1)\tau_{min}, \quad (6.4b) \\ & q = 1, \ldots, N-n-1 \end{cases}$$

where τ_{-}^{n} and ζ_{-}^{n} are the time gap and jam spacing between vehicle n its followers, respectively. (6.4a) guarantees the safe distance between the target vehicle and the detected trailing vehicle, and (6.4b) ensures the maintenance of safe distance between the target vehicle and all the vehicles positioned between it and the trailing vehicle upon their entry into the segment.

Similar to step 1, we denote the lower bounds of the trajectory of vehicle n by $\mathbf{x}_{-}^{n}$. Then the slowest possible trajectory of vehicle n is the trajectory that is the closest to $\mathbf{x}_{-}^{n}$ while satisfying the same constraints of speeds, accelerations, and jerks as those in step 1, which can be written as follows:

$$\min_{\hat{\mathbf{x}}_{-}^{n}} (\hat{\mathbf{x}}_{-}^{n} - \mathbf{x}_{-}^{n})^T (\hat{\mathbf{x}}_{-}^{n} - \mathbf{x}_{-}^{n}), \quad (6.5a)$$

$s.t.$

$$-\hat{\mathbf{x}}_{-}^{n} \leq -\mathbf{x}_{-}^{n}, \tag{6.5b}$$

$$\hat{\mathbf{x}}_{-}^{n} \leq \hat{\mathbf{x}}_{+}^{n}, \tag{6.5c}$$

$$-W^{(k)}\hat{\mathbf{x}}_{-}^{n} \leq -x_{-}^{(k)}\Delta t^{k}, \ \ k = 1,2,3, \tag{6.5d}$$

$$W^{(k)}\hat{\mathbf{x}}_{-}^{n} \leq x_{+}^{(k)}\Delta t^{k}. \ \ k = 1,2,3, \tag{6.5e}$$

$$\hat{x}_{-}^{n}(t_1^n) = x^n(t_1^n), \hat{x}_{-}^{n}(t_M^n) = x^n(t_M^n). \tag{6.5f}$$

where $\hat{\mathbf{x}}_{-}^{n}$ is the slowest possible trajectory of vehicle n. (6.5b) ensures that the calculated positions are no less than the lower bounds, and (6.5c) ensures that the calculated trajectory is no more than the fastest possible trajectory. (6.5d) and (6.5e) guarantee the boundedness of speeds, accelerations, and jerks, and (6.5f) helps maintain the detected data.

Minimizing the sum of squared jerks

Finally, we can compute the imputed trajectory for vehicle n, represented by the grey curve in Figure 6.4. Our objective is to minimize the sum of squared jerks. As illustrated by (5.4), we can express all higher-order derivatives as linear combinations of positions using difference matrices. Hence, the objective function in the quadratic programming problem can be formulated as follows:

$$\min_{\tilde{\mathbf{x}}^n} \frac{1}{\Delta t^6}(W^{(3)}\tilde{\mathbf{x}}^n)^T(W^{(3)}\tilde{\mathbf{x}}^n), \tag{6.6}$$

where $\tilde{\mathbf{x}}^n$ denotes the imputed trajectory of vehicle n, the output of our method. For position constraints, we enforce that the positions should not fall below the slowest possible trajectory or exceed the fastest possible trajectory. The constraints for speeds and higher-order derivatives remain the same as those in the previous steps. To ensure the uniqueness of solutions to the quadratic

programming problem, we must ensure that $(W^{(3)})^T W^{(3)}$ forms a positive definite matrix, necessitating the invertibility of matrix $W^{(3)}$ (Nocedal and Wright, 2006). To achieve this, three elements in the vector $\tilde{\mathbf{x}}^n$ must be known. Apart from the initial and last values, we assume the second position is known and it is the average of the slowest and fastest possible trajectory at that time instant. Additionally, the constant coefficient $\frac{1}{\Delta t^6}$ in the objective function does not impact the optimal solution. Consequently, it can be omitted during computation, resulting in the following formulation of the quadratic programming problem:

$$\min_{\tilde{\mathbf{x}}^n} (W^{(3)}\tilde{\mathbf{x}}^n)^T (W^{(3)}\tilde{\mathbf{x}}^n), \tag{6.7a}$$

$$s.t.$$

$$-\tilde{\mathbf{x}}^n \leq -\hat{\mathbf{x}}^n_-, \tag{6.7b}$$

$$\tilde{\mathbf{x}}^n \leq \hat{\mathbf{x}}^n_+, \tag{6.7c}$$

$$-W^{(k)}\tilde{\mathbf{x}}^n \leq -x^{(k)}_- \Delta t^k, \;\; k = 1,2,3, \tag{6.7d}$$

$$W^{(k)}\tilde{\mathbf{x}}^n \leq x^{(k)}_+ \Delta t^k, \;\; k = 1,2,3, \tag{6.7e}$$

$$\tilde{x}^n(t^n_1) = x^n(t^n_1), \tilde{x}^n(t^n_M) = x^n(t^n_M), \tilde{x}^n(t^n_2) = [\hat{x}^n_-(t^n_2) + \hat{x}^n_+(t^n_2)]/2, \tag{6.7f}$$

where $\hat{\mathbf{x}}^n_-$ and $\hat{\mathbf{x}}^n_+$ denote its slowest and the fastest possible trajectories, respectively.

Thus far, we have successfully derived the imputed trajectory for vehicle n. It is noteworthy that in order for our method to maintain its logical coherence, the trailing vehicle should enter the segment prior to the leading vehicle leaving the segment. Failure to satisfy this condition would render the formulation of our method invalid.

6.3.2 Determination of the time gap and jam spacing

In this subsection, we address the methodology for determining the parameters involved in the first two steps, shifting the available trajectories of the leading and trailing vehicles. We begin by presenting the procedure for determining τ_+^n and ζ_+^n based on the front vehicles, followed by the procedures of determining τ_-^n and ζ_-^n based on the following vehicles.

If all the trajectories within the selected segment exhibit parallelism in the time-space diagram, then given a feasible time gap, we can always calculate some jam spacing that makes the shifted trajectory of the leading vehicle precisely coincide with that of the following vehicle. It is important to note that such an extreme condition is highly improbable in practical scenarios, as real-world trajectories of a platoon of vehicles are unlikely to be perfectly parallel. Consequently, in a more realistic setting, we adopt a grid search method to determine the appropriate parameters. This approach allows us to explore a range of feasible parameter values, accounting for variations in vehicle trajectories and enhancing the validity of our findings.

We choose the time gap and jam spacing between the target vehicle and its front vehicle, τ_+^n and ζ_+^n, according to the following procedures.

1. Investigate τ_+^n ranging from τ_{min} to $\min\{t_1^n - t_1^{n-1}, t_M^n - t_M^{n-1}\}$ at 0.2 seconds intervals and ζ_+^n ranging from ζ_{min} to ζ_{max} at 0.5 meters intervals.

2. Pick out the candidate parameter pairs $\langle \tau_+^n, \zeta_+^n \rangle$ which can satisfy the following conditions:

    i. The upper bound of the vehicle's positions corresponding to the time when vehicle n enters the segment should be ahead of the entrance of the segment: $x^{n-1}(t_1^n - \tau_+^n) - \zeta_+^n > x^n(t_1^n)$.

    ii. The upper bound of vehicle n's positions should keep a safe distance from the following vehicles when they enter the segment, meaning that the upper bound should be ahead

of the entrance after shifting by τ_{min} and ζ_{min} when the following vehicles enter the segment: $x^{n-1}(t_1^{n+q} - \tau_+^n - q\tau_{min}) - \zeta_+^n - q\zeta_{min} > x^{n+q}(t_1^{n+q}), q = 1, ..., N-n$.

iii. The upper bound of vehicle n's positions should keep a safe distance from the detected trajectory of the trailing vehicle: $x^{n-1}(t_m^N - \tau_+^n - (N-n)\tau_{min}) - \zeta_+^n > x^N(t_m^N) + (N-n)\zeta_{min}$.

iv. From the last time and position values of the shifted leading vehicle's trajectory, the vehicle should be able to leave the segment at the detected time traveling at the leading vehicles' last speed: $\frac{\zeta_+^n}{t_M^n - t_M^{n-1} - \tau_+^n} \leq x^{(1),n-1}(t_M^{n-1})$, where $x^{(1),n-1}(t_M^{n-1})$ denotes the leading vehicles' last speed and ζ_+^n is the distance between the exit point of the segment and the last value of the shifted leading vehicle's trajectory.

3. For all the candidate parameter pairs $\langle \tau_+^n, \zeta_+^n \rangle$, select the one that minimizes the distance between the shifted leading vehicle's position at vehicle n's entering time and the entrance point of the segment: $\arg\min_{\tau_+^n, \zeta_+^n}[x^{n-1}(t_1^n - \tau_+^n) - \zeta_+^n - x^n(t_1^n)]$

After calculating the fastest possible trajectory, we determine $\langle \tau_-^n, \zeta_-^n \rangle$ which are used to calculate the slowest possible trajectory. The procedures are as follows.

1. Investigate τ_-^n ranging from τ_{min} to $\min\{\frac{t_1^N - t_1^n}{N-n}, \frac{t_M^N - t_M^n}{N-n}\}$ at 0.2 seconds intervals and ζ_-^n ranging from ζ_{min} to ζ_{max} at 0.5 meters intervals.

2. Pick out the candidate parameter pairs $\langle \tau_-^n, \zeta_-^n \rangle$ which satisfy the following conditions:

   i. The lower bound of the vehicle's positions should be no more than the fastest possible positions: $\mathbf{x}_-^n \langle \tau_-^n, \zeta_-^n \rangle \leq \hat{\mathbf{x}}_+^n$.

3. For all the candidate parameter pairs $\langle \tau_-^n, \zeta_-^n \rangle$, select the one that minimizes the distance between the shifted trailing vehicle's position at vehicle n's exiting time and the exit point of the segment: $\arg\min_{\tau_-^n, \zeta_-^n}[x^n(t_M^n) - x^N(t_M^n + (N-n)\tau_-^n) - (N-n)\zeta_-^n]$

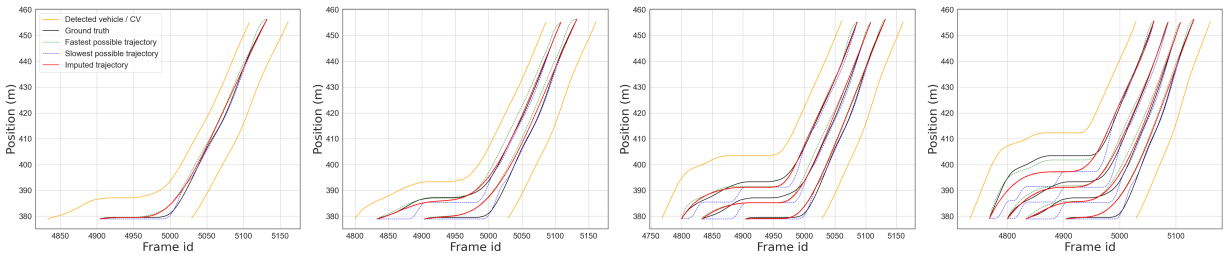
6.4 Numerical experiments

In this section, we numerically show the efficacy of our method upon the manually re-extracted NGSIM I80 data (Coifman and Li, 2017a) and highD "25-tracks" data (Krajewski et al., 2018). The study sites for the two datasets have lengths of approximately 70 meters and 420 meters, respectively. We first consider the scenarios where a leading vehicle and a trailing vehicle whose trajectories are available with one, two, three, and four undetected vehicles in between. The entering and exiting times of these undetected vehicles into and out of the segment are obtained from the loop detectors. We proceed to impute the trajectories of these undetected vehicles using our proposed method. We then consider a traffic system consisting of 40 vehicles. We randomly assign 10% CAVs and 10% CVs and impute the trajectories of all the vehicles in the system.

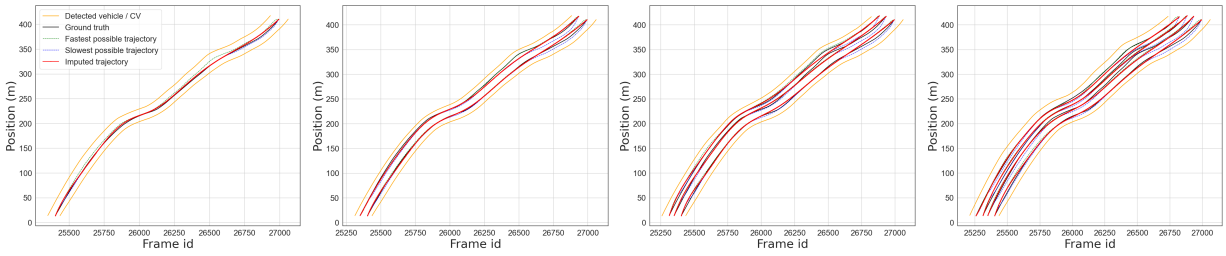
6.4.1 Application to the vehicle platoons of different sizes

Both the NGSIM and highD datasets encompass complete vehicle trajectories within a fixed segment throughout the recording duration. We first select single platoons consisting of different numbers of vehicles from both the NGSIM and highD datasets to show the efficacy of our method. We gradually increase the number of successive vehicles from three to six, while keeping the trajectories of the leading and trailing vehicles known, meaning that the leading and trailing vehicles are either CVs or detected by CAVs. The trajectories of the vehicles in between are to be imputed, while the only information available are their entering and exiting times detected by the loop detectors.

We begin with sample platoons consisting of three vehicles from the aforementioned two datasets: vehicles [1497,1506,1512] in the NGSIM I80 dataset and vehicles [2486,2490,2493] in the highD dataset. Subsequently, we incrementally introduce additional vehicles at the front of the platoons until they reach a size of six vehicles. The two six-vehicle platoons observed in the respective



(a) NGSIM I80 camera 6 dataset



(b) HighD "25-tracks" dataset

Figure 6.5: Imputation of the sample trajectories of three, four, five, and six successive vehicles in the (a) NGSIM I80 camera 6 and (b) highD "25-tracks" datasets

datasets are [1463, 1478, 1486, 1497, 1506, 1512] and [2476, 2482, 2485, 2486, 2490, 2493]. The imputation results are presented in Figure 6.5, with the leftmost column showcasing the imputation results for three-vehicle platoons, followed by four-, five-, and six-vehicle platoons in subsequent columns. In the visualization, the detected trajectories are depicted in orange, while the true trajectories of undetected vehicles are represented in black. Our imputed trajectories are illustrated by continuous red curves, while the fastest and slowest possible trajectories are depicted by dashed green and blue curves, respectively.

Our imputation method consistently exhibits similar effects on platoons of varying sizes across both datasets. Notably, all essential characteristics observed in the real trajectories, including accelerating, decelerating, cruising, and stopping behaviors, are generally preserved in our imputed trajectories. We calculate the mean absolute error (MAE) and the root mean squared error (RMSE) between our imputed trajectories and the true trajectories as follows:

$$E(|\tilde{x}-\bar{x}|) = \frac{\sum_{n=2}^{N-1}\sum_{m=1}^{M}|\tilde{x}^n(t_m^n)-\bar{x}^n(t_m^n)|}{\sum_{n=2}^{N-1}(t_M^n-t_1^n)/\Delta t}, \tag{6.8a}$$

$$\sqrt{E\left((\tilde{x}-\bar{x})^2\right)} = \sqrt{\frac{\sum_{n=2}^{N-1}\sum_{m=1}^{M}\left(\tilde{x}^n(t_m^n)-\bar{x}^n(t_m^n)\right)^2}{\sum_{n=2}^{N-1}(t_M^n-t_1^n)/\Delta t}}, \tag{6.8b}$$

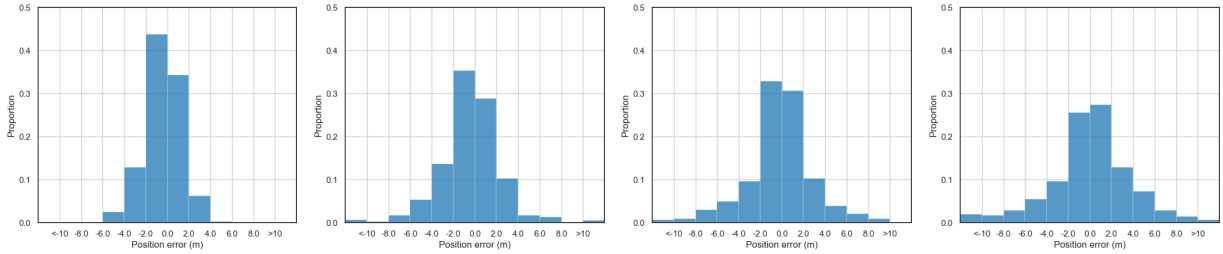
where $\tilde{x}^n(t_m^n)$ and $\bar{x}^n(t_m^n)$ are the imputed and true positions of vehicle n at time t_m^n, respectively. The MAEs and RMSEs between our imputed trajectories and the true trajectories in the sample platoons are listed in **Table** 6.1. In the sample platoons in both datasets, we observe that the MAEs and RMSEs between the minimum-jerk trajectories and the true trajectories are the smallest, albeit with a few exceptions, and the MAEs and RMSEs between the minimum-jerk trajectories and the true trajectories exhibit the highest level of stability across varying platoon sizes. In general, with the increasement of platoon sizes, the errors between the imputed and true trajectories increase accordingly. Nevertheless, our method demonstrates a commendable level of robustness to platoon sizes, as even for six-vehicle platoons, a high degree of accuracy is maintained.

Table 6.1: MAE and RMSE between the imputed and true trajectories in the sample platoons

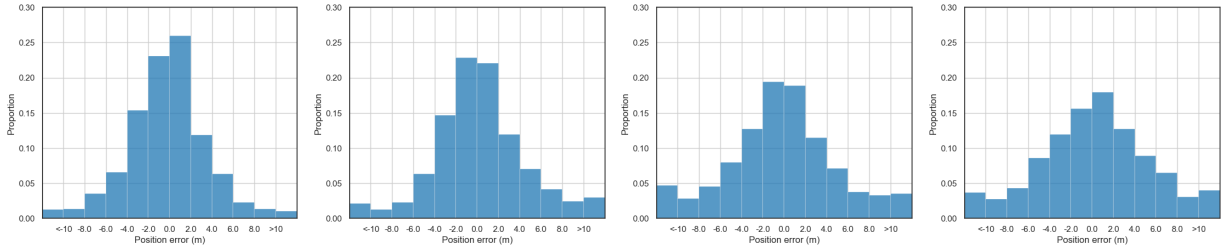
Dataset \ Platoon size			3	4	5	6
NGSIM	Fastest possible trajectories (Step 1)	MAE (m)	1.83	2.52	1.55	1.91
		RMSE (m)	2.26	3.28	2.84	2.42
	Slowest possible trajectory (Step 2)	MAE (m)	0.72	1.35	1.78	2.96
		RMSE (m)	0.87	1.81	2.28	4.25
	Minimum-jerk trajectory (Step 3)	MAE (m)	1.23	1.42	1.18	1.96
		RMSE (m)	1.69	2.14	1.45	2.72
HighD	Fastest possible trajectories (Step 1)	MAE (m)	5.31	1.89	3.16	2.92
		RMSE (m)	5.65	2.63	3.83	3.93
	Slowest possible trajectory (Step 2)	MAE (m)	1.63	3.75	3.28	4.97
		RMSE (m)	1.84	4.59	4.15	5.95
	Minimum-jerk trajectory (Step 3)	MAE (m)	2.23	2.29	2.8	3.01
		RMSE (m)	2.6	3.07	3.58	3.94

We then statistically show the efficacy of our method. We analyze vehicle platoons composed of three, four, five, and six successive vehicles from the NGSIM I80 and highD "25-tracks" datasets. In each dataset, we select 50, 40, 30, and 25 platoons respectively for the three-, four-, five-, and six-vehicle platoons. The numerical experiment encompasses the testing of nearly 300 platoons in total.

We plot the distributions of the position errors observed in the imputed positions of platoons consisting of vehicles of varying sizes as Figure 6.6. The eight distributions exhibit a zero-centered pattern, wherein the errors decrease as absolute values increase. Notably, the position errors in the NGSIM I80 dataset tend to cluster around zero more prominently, primarily due to the length of trajectories being shorter. The majority of position errors fall within the range of -10 meters to 10 meters. As the number of vehicles within the platoons increases, the position errors gradually become less concentrated around zero. For the three-vehicle platoons in the NGSIM I80 and highD



(a) NGSIM I80 camera 6 dataset



(b) HighD "25-tracks" dataset

Figure 6.6: Position error distributions of the imputed trajectories of three, four, five, and six successive vehicles in the (a) NGSIM I80 camera 6 and (b) highD "25-tracks" datasets

datasets, about 80% and 50% position errors range from -2 meters to 2 meters, respectively. While for the six-vehicle platoons, the corresponding proportions become 50% and 30%, respectively.

We also choose the MAEs and RMSEs between the imputed and true trajectories in all selected platoons as the indicators. The MAEs and RMSEs between the imputed and true trajectories in the platoons with different sizes are presented in **Table** 6.2. In comparison to the outputs of step 1 and step 2, representing the fastest and slowest possible trajectories respectively, we find that the outputs of step 3, the minimum-jerk trajectories, exhibit the smallest MAEs and RMSEs in relation to the true trajectories. Overall, the RMSEs exhibit a similar trend to the MAEs, indicating an increase in errors with an increasing number of successive vehicles to be imputed. The RMSEs consistently exhibit higher values compared to the MAEs, indicating that the position errors within a trajectory are not uniformly distributed across time. This finding aligns with the characteristics of the considered scenarios, where the loop detectors are positioned at the entrance and exit points of the segment. Therefore, the entering and exiting times are accurately captured, while the imputed trajectories in the middle of the segment exhibit relatively larger errors.

Table 6.2: MAE and RMSE between the imputed and true trajectories in the two datasets

Dataset \ Platoon size			3	4	5	6
NGSIM	Fastest possible trajectories (Step 1)	MAE (m)	1.88	2.48	2.62	3.29
		RMSE (m)	2.56	3.92	3.72	4.46
	Slowest possible trajectory (Step 2)	MAE (m)	1.99	2.79	2.94	3.74
		RMSE (m)	2.69	4.15	4.29	5.61
	Minimum-jerk trajectory (Step 3)	MAE (m)	1.23	1.95	2.09	2.69
		RMSE (m)	1.65	2.81	3.00	3.77
HighD	Fastest possible trajectories (Step 1)	MAE (m)	3.55	4.23	4.52	4.93
		RMSE (m)	4.75	6.36	6.36	6.51
	Slowest possible trajectory (Step 2)	MAE (m)	4.4	4.45	6.19	6.32
		RMSE (m)	6.4	6.08	8.44	8.26
	Minimum-jerk trajectory (Step 3)	MAE (m)	2.78	3.27	3.97	4.19
		RMSE (m)	3.8	4.63	5.42	5.55

6.4.2 Application to a sample mixed-traffic system

Furthermore, we select a specific scenario from the highD dataset, which encompasses a sequence of 40 successive vehicles exhibiting no lane-changing behaviors. In this scenario, we make the assumption that both connected and autonomous vehicles (CAVs) and connected vehicles (CVs) exhibit a market penetration rate (MPR) of 10%, resulting in the presence of four CAVs and four CVs within this particular scenario. We assume the leading and trailing vehicles within the platoon are CVs, whereas the remaining CAVs and CVs are randomly assigned to the other vehicles in the scenario. The minimum number of trajectories to be imputed between two available trajectories is set at one (a three-vehicle platoon), while the maximum number is set at 6 (an eight-vehicle platoon). It is noteworthy that CVs are capable of reporting solely their own trajectories, whereas CAVs possess the ability to report not only their own trajectories but also those of their adjacent vehicles.

The trajectories imputed by our method are shown as Figure 6.7, where the scenario can be regarded as including two four-vehicle platoons, two five-vehicle platoons, two six-vehicle platoons, and one eight-vehicle platoon for imputation. The CAVs are shown in purple and the CVs or the trajectories recorded by CAVs are shown in orange. The true trajectories are shown in black, while

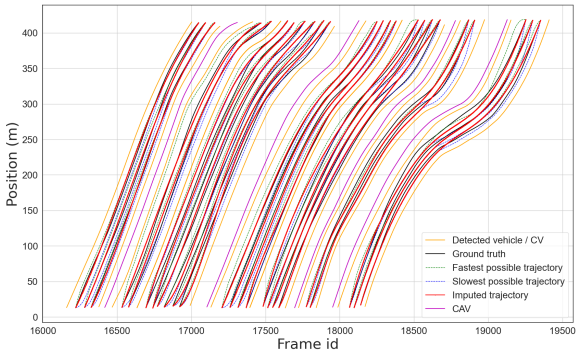
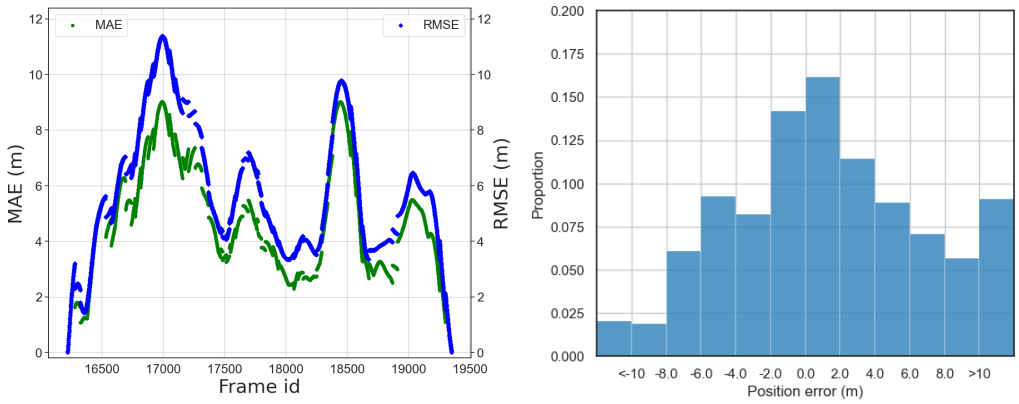


Figure 6.7: Results of our trajectory imputation method in the system scenario

the fastest possible trajectories, the slowest possible trajectories, and the imputed trajectories are shown in green, blue, and red, respectively. From the imputed trajectories, we can see that our method can generally unearth the real trajectories. While some errors exist, and such errors are the most apparent in the eight-vehicle platoon compared to the platoons consisting of fewer vehicles.



(a) (b)

Figure 6.8: Position errors versus time and distribution of position errors

To better illustrate the efficacy of our method, we plot the MAEs and RMSEs of imputed trajectories at each time instant, and the distribution of position errors as Figure 6.8a and Figure 6.8b. It can be found that MAEs and RMSEs show the same trend, both of which fluctuate over time, with the values at the two edges being the smallest. The MAEs and RMSEs range from $[0,9]$ m and $[0,11]$ m, respectively. The distribution of the position errors in this scenario has a trend similar to Figure 6.6, with MAE and RMSE being 4.96 meters and 6.59 meters, respectively.

6.5 Conclusion

In this study, we propose a three-step quadratic programming method for imputing fully sampled vehicle trajectories by jointly considering the following principles: (P1) Maintaining a safe inter-vehicle spacing. (P2) Ensuring that speeds, accelerations, and jerks remain within physically meaningful ranges. (P3) Considering drivers' tendency to minimize jerks during their movements. Our method is applicable to scenarios where the trajectories of the leading and trailing vehicles within a segment are known, while the trajectories of intermediate vehicles need imputation. For the trajectories to be imputed, we only know the time they enter and exit the segment. Each step involves solving a quadratic programming problem, where the physically meaningful bounds on speeds, accelerations, and jerks serve as constraints. In the first step, we calculate the fastest possible trajectory that maintains a safe distance from the leading vehicle. In the second step, we determine the slowest possible trajectory that maintains safe distance from the following vehicles. Finally, in the third step, we compute the trajectory that lies between the slowest and fastest possible trajectories, optimizing for the minimal sum of squared jerks. We apply our method to the scenarios involving a leading vehicle and a trailing vehicle with a varying number of undetected vehicles in between, and impute the trajectories of the undetected vehicles. Our method demonstrates a high level of accuracy in imputing these trajectories. The distributions of position errors observed in the imputed positions exhibit a zero-centered pattern and decrease as the absolute position errors increase. The MAEs and RMSEs between the imputed and true trajectories indicate that position errors increase with the number of undetected successive vehicles. Furthermore, the fact that RMSEs consistently exceed MAEs suggests that position errors are not uniformly distributed. Moreover, we consider a traffic system consisting of 40 successive vehicles including 10% CVs and 10% CAVs. Our method successfully captures the true conditions of the traffic system, correctly imputing the trajectories of all vehicles within the system. The position errors of the imputed trajectories of all vehicles in the system distribute in the same way as the aforementioned platoons.

Our contribution can be summarized from two perspectives. On one hand, we have introduced a

novel and comprehensive method that integrates multiple principles, including safe inter-vehicle spacing, physically meaningful ranges on speeds, accelerations, and jerks, as well as the minimum-jerk theory, to effectively impute fully sampled vehicle trajectories. This approach addresses the challenges posed by missing data and enables us to obtain accurate and realistic vehicle trajectories. On the other hand, through numerical experimentation, we have demonstrated that the imputed trajectories derived from our method closely align with the true trajectories. This empirical evidence further supports the validity of utilizing the minimum-jerk theory as a reliable approximation of human driving behaviors. Notably, this finding is consistent with the arguments presented in (Flash and Hogan, 1985).

For future studies, we plan to categorize the successive vehicles into different phases, including the approaching phase, free-driving phase, following phase, and braking phase, and study the difference in the efficacy of our method in different phases. Additionally, building upon the ideas proposed in this study, we are particularly interested in exploring the application of trajectory prediction and planning techniques. These works aim to ensure safe, efficient, and environmentally friendly operations for both human-driven and automated vehicles.

Chapter 7

Conclusion

7.1 Summary

Vehicle trajectory data provide a rich source of information for investigating traffic dynamics in both spatial and temporal domains, and accurate and complete vehicle trajectory data is crucial for numerous applications. However, longitudinal vehicle trajectories suffer from errors, noises, and incompleteness due to detection and extraction techniques, posing challenges for various applications. This dissertation introduces a framework that leverages physical properties, vehicle characteristics, and human driving behaviors to smooth and impute longitudinal vehicle trajectory data.

To remove the outliers and high-frequency noises in speeds and higher-order derivatives, we incorporate some first principles, including the internal consistency among the positions, speeds, and higher-order derivatives, bounded speeds and higher-order derivatives, and minimum MAE between the raw and smoothed positions. We propose an iterative method based on these first principles. One iteration comprises four types of calculations: differentiation, correction, smoothing, and integration. In differentiation, we compute speeds, accelerations, and jerks from trajectory

data; in correction, we eliminate outliers in speeds, especially negative values, via the adaptive average method; in smoothing, we reduce noises in accelerations and jerks with the Gaussian filters; and in integration, we recalculate accelerations, speeds, and positions from jerks, and find the initial values via optimization problems which minimize the MAEs between the data before and after smoothing. The efficacy of the method is numerically shown with the NGSIM data. However, it is mathematically challenging to demonstrate when the iterations converge or even that the iterations can converge, leading us to develop more mathematically tractable principles-based techniques that can either be proved to converge or get rid of iterations

We then proposed a simplified iterative moving average method that makes the ranges of the smoothed speeds, accelerations, and jerks align with physical meaning, while preserving the average speeds or total travel distance for a specified time duration segment of a vehicles trajectory. In each iteration, we pad both ends of the speed profile with the average speed and apply a moving average method with different window shapes, representing different weights, to speeds, followed by normalizing the filtered speeds to preserve the average speed and maintain the total travel distance. We differentiate the normalized filtered speeds into accelerations and jerks to check whether the termination conditions are satisfied, and the iterative process concludes only when they are satisfied. Otherwise, we initiate another iteration to further smooth the speeds. Mathematically, we prove that the method can terminate within a finite number of iterations, in accordance with predefined termination criteria. Numerically, the method is validated upon the NGSIM data, effectively eliminating outliers and high-frequency noise while preserving the main information, including acceleration, deceleration, and cruising behaviors, from the raw data.

A two-step quadratic programming method for smoothing vehicle trajectory data was then proposed in Chapter 5, which operates without the need for iterations, and concludes in a single round. In the first step, we minimize the discrepancy between the half-smoothed and raw positions, while adhering to physically meaningful bounds on the speeds and higher order derivatives of the half-smoothed positions. Subsequently, in the second step, our objective is to minimize the roughness

of the smoothed positions, while incorporating the position ranges as additional constraints alongside those imposed in the first step. This step allows the smoothed positions to deviate from the raw data by at most those of the half-smoothed positions and the prior position error, and make the initial few positions in the trajectory the half-smoothed positions to ensure the strict convexity of the objective function. The existence and uniqueness of solutions are analytically proved and the efficacy of the method is numerically shown with the NGSIM data. Comparisons with an existing approach with respect to the manually re-extracted data show the superior performance of the proposed method in terms of leading to smaller MSEs between the smoothed positions, speeds, accelerations, and jerks and those manually re-extracted.

Finally, in Chapter 6, the scenarios involving missing portions of trajectories were investigated. We propose a three-step quadratic programming trajectory imputation method that is applicable to scenarios where the trajectories of the leading and trailing vehicles within a segment are known, while the trajectories of intermediate vehicles need imputation. The method ensures maintaining safe inter-vehicle spacing and adheres to physically meaningful speed, acceleration, and jerk ranges. Each step involves solving a quadratic programming problem, where the physically meaningful bounds on speeds, accelerations, and jerks serve as constraints. The method is applied to diverse scenarios involving a leading vehicle and a trailing vehicle with varying numbers of undetected vehicles in between, effectively imputing the trajectories of these unobserved vehicles with a commendable degree of precision. Furthermore, we extended the assessment to a traffic system consisting of 40 successive vehicles including 10% CVs and 10% CAVs. In this context, the method adeptly captured the real conditions of the traffic system, correctly reconstructing the trajectories of all vehicles within it.

7.2 Future research topics

In Chapter 4, we adopted convolution filters for smoothing the speeds. However, the methodology we proposed is open to different options, for example, the k-nearest-neighbors (kNN) regression algorithm and the Savitzky-Golay filter, and this is one topic for our future study.

In Chapter 5, in the second step of the quadratic programming method, we incorporate additional equality constraints by making the initial several values of the smoothed positions equal to the corresponding half-smoothed positions. The number of constraints equals the highest order of derivatives. This serves to manage the degrees of freedom within the quadratic programming problem, ensuring the strict convexity of the objective function. Nevertheless, alternative methods exist for introducing additional equality constraints, such as employing the concept of moments. Exploring these alternative approaches is a topic worthy of our consideration in future research

In Chapter 6, we current approach processes the entire trajectory directly. In the future, we would like to segment successive vehicles into distinct phases: the approaching phase, the free-driving phase, the following phase, and the braking phase. We plan to examine the variations in the effectiveness of our method across these different phases.

In addition, With the accessibility of more and more high-fidelity vehicle trajectory data collected in different countries from different sources, we are interested in testing our method with other datasets. Moreover, our study solely focuses on the longitudinal vehicle trajectory data, we plan to incorporate different boundedness criteria to smooth lateral vehicle trajectory data. Apart from the smoothing and imputation of vehicle trajectory data, we are also interested in predicting future vehicle behaviors using historical trajectory data from the past several seconds.

Bibliography

Ahn, S., Cassidy, M.J., Laval, J., 2004. Verification of a simplified car-following theory. Transportation Research Part B 38, 431–440.

Al-Gabalawy, M., Hosny, N.S., Aborisha, A.h.S., 2021. Model predictive control for a basic adaptive cruise control. International Journal of Dynamics and Control 9, 1132–1143.

Altché, F., de La Fortelle, A., 2017. An LSTM network for highway trajectory prediction, in: 2017 IEEE 20th international conference on intelligent transportation systems (ITSC), IEEE. pp. 353–359.

Belcarz, K., Białek, T., Komorkiewicz, M., Żołnierczyk, P., 2018. Developing autonomous vehicle research platform–a case study, in: IOP Conference Series: Materials Science and Engineering, IOP Publishing. p. 022002.

Bhatia, R., 2009. Positive definite matrices. Princeton University Press. chapter 1. pp. 1–34.

Bock, J., Krajewski, R., Moers, T., Runde, S., Vater, L., Eckstein, L., 2020. The inD Dataset: A Drone Dataset of Naturalistic Road User Trajectories at German Intersections, in: 2020 IEEE Intelligent Vehicles Symposium (IV), IEEE. pp. 1929–1934.

Bokare, P.S., Maurya, A.K., 2017. Acceleration-deceleration behaviour of various vehicle types. Transportation research procedia 25, 4733–4749.

Boyd, S., Boyd, S.P., Vandenberghe, L., 2004. Convex optimization. Cambridge University Press. chapter 11. pp. 561–630.

Chang, M.F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., et al., 2019. Argoverse: 3D tracking and forecasting with rich maps, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8748–8757.

Chen, C., Petty, K., Skabardonis, A., Varaiya, P., Jia, Z., 2001. Freeway performance measurement system: mining loop detector data. Transportation Research Record 1748, 96–102.

Chen, D., Ahn, S., Laval, J., Zheng, Z., 2014. On the periodicity of traffic oscillations and capacity drop: the role of driver characteristics. Transportation research part B 59, 117–136.

Chen, X., Yin, J., Tang, K., Tian, Y., Sun, J., 2022. Vehicle trajectory reconstruction at signalized intersections under connected and automated vehicle environment. IEEE Transactions on Intelligent Transportation Systems 23, 17986–18000.

Chiabaut, N., Leclercq, L., Buisson, C., 2010. From heterogeneous drivers to macroscopic patterns in congestion. Transportation Research Part B 44, 299–308.

Coifman, B., 2002. Estimating travel times and vehicle trajectories on freeways using dual loop detectors. Transportation Research Part A 36, 351–364.

Coifman, B., Beymer, D., McLauchlan, P., Malik, J., 1998. A real-time computer vision system for vehicle tracking and traffic surveillance. Transportation Research Part C 6, 271–288.

Coifman, B., Li, L., 2017a. A critical evaluation of the next generation simulation (NGSIM) vehicle trajectory dataset. Transportation Research Part B 105, 362–377.

Coifman, B., Li, L., 2017b. I-80 NGSIM validation video. `http://www2.ece.ohio-state.edu/~coifman/documents/I80-NGSIM/`.

Coifman, B., Wu, M., Redmill, K., Thornton, D.A., 2016. Collecting ambient vehicle trajectories from an instrumented probe vehicle: High quality data for microscopic traffic flow studies. Transportation Research Part C: Emerging Technologies 72, 254–271.

Diamond, S., Boyd, S., 2016. CVXPY: A Python-embedded modeling language for convex optimization. The Journal of Machine Learning Research 17, 2909–2913.

Dong, S., Zhou, Y., Chen, T., Li, S., Gao, Q., Ran, B., 2021. An integrated empirical mode decomposition and butterworth filter based vehicle trajectory reconstruction method. Physica A: Statistical Mechanics and its Applications 583, 126295.

Eilers, P.H., Marx, B.D., 1996. Flexible smoothing with B-splines and penalties. Statistical Science 11, 89–121.

Elert, G., 2012. The physics factbookan encyclopedia of scientific essays. Volume of Blood in a Human .

Elliott, D.F., 2013. Handbook of digital signal processing: engineering applications. Elsevier.

Eubank, R.L., 1999. Nonparametric regression and spline smoothing. CRC Press.

Fard, M.R., Mohaymany, A.S., Shahri, M., 2017. A new methodology for vehicle trajectory reconstruction based on wavelet analysis. Transportation Research Part C 74, 150–167.

Feng, F., Bao, S., Sayer, J.R., Flannagan, C., Manser, M., Wunderlich, R., 2017. Can vehicle longitudinal jerk be used to identify aggressive drivers? an examination using naturalistic driving data. Accident Analysis & Prevention 104, 125–136.

FHWA, U., 2007. Department of Transportation. NGSIM–Next Generation SIMulation.

Flash, T., Hogan, N., 1985. The coordination of arm movements: an experimentally confirmed mathematical model. Journal of Neuroscience 5, 1688–1703.

Fletcher, R., 2013. Practical methods of optimization. John Wiley & Sons.

Floudas, C.A., Visweswaran, V., 1995. Quadratic optimization. Handbook of global optimization , 217–269.

Gasparetto, A., Zanotto, V., 2007. A new method for smooth trajectory planning of robot manipulators. Mechanism and Machine Theory 42, 455–471.

Gasparetto, A., Zanotto, V., 2008. A technique for time-jerk optimal planning of robot trajectories. Robotics and Computer-Integrated Manufacturing 24, 415–426.

Gasparetto, A., Zanotto, V., 2010. Optimal trajectory planning for industrial robots. Advances in Engineering Software 41, 548–556.

Getreuer, P., 2013. A survey of gaussian convolution algorithms. Image Processing On Line 2013, 286–310.

Gipps, P.G., 1981. A behavioural car-following model for computer simulation. Transportation Research Part B 15, 105–111.

Gong, S., Shen, J., Du, L., 2016. Constrained optimization and distributed computation based car following control of a connected and autonomous vehicle platoon. Transportation Research Part B 94, 314–334.

Gurobi Optimization, LLC, 2023. Gurobi Optimizer Reference Manual. URL: `https://www.gurobi.com`.

Hao, P., Boriboonsomsin, K., Wu, G., Barth, M., 2014. Probabilistic model for estimating vehicle trajectories using sparse mobile sensor data, in: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), IEEE. pp. 1363–1368.

Hao, P., Boriboonsomsin, K., Wu, G., Barth, M.J., 2016. Modal activity-based stochastic model for estimating vehicle trajectories from sparse mobile sensor data. IEEE Transactions on Intelligent Transportation Systems 18, 701–711.

Huang, N.E., Shen, Z., Long, S.R., Wu, M.C., Shih, H.H., Zheng, Q., Yen, N.C., Tung, C.C., Liu, H.H., 1998. The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. Proceedings of the Royal Society of London. Series A: mathematical, physical and engineering sciences 454, 903–995.

ISO, 2010. Intelligent transport systems–adaptive cruise control systems–performance requirements and test procedures.

Jin, W.L., 2019. Nonstandard second-order formulation of the LWR model. Transportmetrica B 7, 1338–1355.

Jin, W.L., 2021. Introduction to Network Traffic Flow Theory: Principles, Concepts, Models, and Methods. Elsevier. chapter 3. pp. 33–56.

Kesting, A., Treiber, M., 2008. Calibrating car-following models by using trajectory data: Methodological study. Transportation Research Record 2088, 148–156.

Krajewski, R., Bock, J., Kloeker, L., Eckstein, L., 2018. 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. pp. 2118–2125.

Krajewski, R., Moers, T., Bock, J., Vater, L., Eckstein, L., 2020. The rounD Dataset: A Drone Dataset of Road User Trajectories at Roundabouts in Germany, in: 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), IEEE. pp. 1–6.

Lambrechts, P., Boerlage, M., Steinbuch, M., 2005. Trajectory planning and feedforward design for electromechanical motion systems. Control Engineering Practice 13, 145–157.

Lim, H.S., Lee, J.E., Park, H.M., Lee, S., 2020. Stationary target identification in a traffic monitoring radar system. Applied Sciences 10. URL: https://www.mdpi.com/2076-3417/10/17/5838, doi:10.3390/app10175838.

Lima, R., Seminar, E., 2010. Ibm ilog cplex-what is inside of the box, in: Proc. 2010 EWO Seminar, pp. 1–72.

Lin, L., Wang, Y., Zhou, H., 2009. Iterative filtering as an alternative algorithm for empirical mode decomposition. Advances in Adaptive Data Analysis 1, 543–560.

Litman, T., 2020. Autonomous vehicle implementation predictions: Implications for transport planning .

Ma, X., Andréasson, I., 2005. Dynamic car following data collection and noise cancellation based on the Kalman smoothing, in: IEEE International Conference on Vehicular Electronics and Safety, 2005., IEEE. pp. 35–41.

Marczak, F., Buisson, C., 2012. New filtering method for trajectory measurement errors and its comparison with existing methods. Transportation research record 2315, 35–46.

Martinez, J.J., Canudas-de Wit, C., 2007. A safe longitudinal control for adaptive cruise control and stop-and-go scenarios. IEEE Transactions on control systems technology 15, 246–258.

Masello, L., Sheehan, B., Murphy, F., Castignani, G., McDonnell, K., Ryan, C., 2022. From traditional to autonomous vehicles: a systematic review of data availability. Transportation Research Record 2676, 161–193.

Mellinger, D., Kumar, V., 2011. Minimum snap trajectory generation and control for quadrotors, in: 2011 IEEE International Conference on Robotics and Automation, IEEE. pp. 2520–2525.

Montanino, M., Punzo, V., 2013. Making NGSIM data usable for studies on traffic flow theory: Multistep method for vehicle trajectory reconstruction. Transportation Research Record 2390, 99–111.

Montanino, M., Punzo, V., 2015. Trajectory data reconstruction and simulation-based validation against macroscopic traffic patterns. Transportation Research Part B 80, 82–106.

Mu, J., Han, Y., Zhang, C., Yao, J., Zhao, J., 2021. An unscented kalman filter-based method for reconstructing vehicle trajectories at signalized intersections. Journal of advanced transportation 2021.

Newell, G.F., 2002. A simplified car-following theory: a lower order model. Transportation Research Part B 36, 195–205.

Nocedal, J., Wright, S.J., 2006. Quadratic programming. Numerical Optimization , 66–98.

Oppenheim, A.V., Willsky, A.S., Nawab, S.H., Ding, J.J., 1997. Signals and systems. Prentice hall Upper Saddle River, NJ. volume 2. chapter 5,6. pp. 358–513.

Ossen, S., Hoogendoorn, S.P., 2008. Validity of trajectory-based calibration approach of car-following models in presence of measurement errors. Transportation Research Record 2088, 117–125.

Ossen, S., Hoogendoorn, S.P., 2009. Reliability of parameter values estimated using trajectory observations. Transportation research record 2124, 36–44.

Othman, M.R., Zhang, Z., Imamura, T., Miyake, T., 2008. A study of analysis method for driver features extraction, in: 2008 IEEE International Conference on Systems, Man and Cybernetics, IEEE. pp. 1501–1505.

Pendrill, A.M., Eager, D., 2020. Velocity, acceleration, jerk, snap and vibration: Forces in our bodies during a roller coaster ride. Physics Education 55, 065012.

Pollock, D.S.G., Green, R.C., Nguyen, T., 1999. Handbook of time series analysis, signal processing, and dynamics. Elsevier.

Punzo, V., Borzacchiello, M.T., Ciuffo, B., 2011. On the assessment of vehicle trajectory data accuracy and application to the next generation simulation (NGSIM) program data. Transportation Research Part C: Emerging Technologies 19, 1243–1262.

Punzo, V., Montanino, M., 2016. Speed or spacing? cumulative variables, and convolution of model errors and time in traffic flow models validation and calibration. Transportation Research Part B 91, 21–33.

Punzo, V., Simonelli, F., 2005. Analysis and comparison of microscopic traffic flow models with real traffic microscopic data. Transportation Research Record 1934, 53 – 63.

da Rocha, T.V., Leclercq, L., Montanino, M., Parzani, C., Punzo, V., Ciuffo, B., Villegas, D., 2015. Does traffic-related calibration of car-following models provide accurate estimations of vehicle emissions? Transportation research part D 34, 267–280.

Romeny, B.M.H., 2008. Front-end vision and multi-scale image analysis: multi-scale computer vision theory and applications, written in mathematica. volume 27. Springer Science & Business Media.

Schwall, M., Daniel, T., Victor, T., Favaro, F., Hohnhold, H., 2020. Waymo public road safety performance data. arXiv preprint arXiv:2011.00038 .

Shan, X., Hao, P., Chen, X., Boriboonsomsin, K., Wu, G., Barth, M.J., 2016. Probabilistic model for vehicle trajectories reconstruction using sparse mobile sensor data on freeways, in: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), IEEE. pp. 689–694.

Shan, X., Hao, P., Chen, X., Boriboonsomsin, K., Wu, G., Barth, M.J., 2018. Vehicle energy/emissions estimation based on vehicle trajectory reconstruction using sparse mobile sensor data. IEEE transactions on intelligent transportation systems 20, 716–726.

Sheppard, W.F., 1914. Graduation by reduction of mean square of error. Journal of the Institute of Actuaries 48, 171–185.

Smith, S., 2013. Digital signal processing: a practical guide for engineers and scientists. Elsevier.

Smith, S.W., 1997. The Scientist and Engineer's Guide to Digital Signal Processing. California Technical Publishing. chapter 15. pp. 277–284.

Straka, O., Ŝimandl, M., 2009. A survey of sample size adaptation techniques for particle filters. IFAC Proceedings Volumes 42, 1358–1363.

Strang, G., 1993. Introduction to linear algebra. Wellesley-Cambridge Press Wellesley, MA. volume 3. chapter 1. pp. 1–30.

Strang, G., 2014. Differential equations and linear algebra. Wellesley-Cambridge Press Wellesley. chapter 4. pp. 246–338.

Sun, Y., Xu, H., Wu, J., Zheng, J., Dietrich, K.M., 2018. 3-D data processing to extract vehicle trajectories from roadside LiDAR data. Transportation Research Rcord 2672, 14–22.

Sun, Z., Hao, P., Ban, X.J., Yang, D., 2015. Trajectory-based vehicle energy/emissions estimation for signalized arterials using mobile sensing data. Transportation Research Part D 34, 27–40.

Thiemann, C., Treiber, M., Kesting, A., 2008. Estimating acceleration and lane-changing dynamics from next generation simulation trajectory data. Transportation Research Record 2088, 90–101.

Toledo, T., Koutsopoulos, H.N., Ahmed, K.I., 2007. Estimation of vehicle trajectories with locally weighted regression. Transportation Research Record 1999, 161–169.

Treiber, M., Hennecke, A., Helbing, D., 2000. Congested traffic states in empirical observations and microscopic simulations. Physical review E 62, 1805.

Tsanakas, N., Ekström, J., Olstam, J., 2022. Generating virtual vehicle trajectories for the estimation of emissions and fuel consumption. Transportation Research Part C 138, 103615.

Van Drongelen, W., 2018. Signal processing for neuroscientists. Academic press. chapter 7. pp. 107–126.

Venthuruthiyil, S.P., Chunchu, M., 2018. Trajectory reconstruction using locally weighted regression: A new methodology to identify the optimum window size and polynomial order. Transportmetrica A: Transport Science 14, 881–900.

Wang, H., Gu, C., Ochieng, W.Y., 2019. Vehicle trajectory reconstruction for signalized intersections with low-frequency floating car data. Journal of Advanced Transportation 2019.

Wang, S., Bao, Z., Culpepper, J.S., Cong, G., 2021. A survey on trajectory data management, analytics, and learning. ACM Computing Surveys (CSUR) 54, 1–36.

Wang, Y., Wei, L., Chen, P., 2020. Trajectory reconstruction for freeway traffic mixed with human-driven vehicles and connected and automated vehicles. Transportation research part C 111, 135–155.

Wei, L., Wang, Y., Chen, P., 2019. Trajectory reconstruction using automated vehicles motion detection data: A hybrid approach integrating wiedemann model and cellular automation, in: 2019 IEEE Intelligent Transportation Systems Conference (ITSC), IEEE. pp. 1379–1384.

Wei, L., Wang, Y., Chen, P., 2020. A particle filter-based approach for vehicle trajectory reconstruction using sparse probe data. IEEE Transactions on Intelligent Transportation Systems 22, 2878–2890.

Whittaker, E., 1923. On a new method of graduation. Proceedings of the Edinburgh Mathematical Society 41, 63–75.

Whittaker, E.T., 1922. On a new method of graduation. Proceedings of the Edinburgh Mathematical Society 41, 63–75.

Whittaker, E.T., Robinson, G., 1924. The calculus of observations: a treatise on numerical mathematics. Blackie and Son Limited. chapter 11. pp. 285–316.

Wiedemann, R., 1974. Simulation des strassenverkehrsflusses. .

Wu, P., Sun, X., Hu, H., Mao, T., Zhao, W., Sheng, K., Cheung, A.A., Niu, T., 2015. Iterative ct shading correction with no prior information. Physics in Medicine & Biology 60, 8437.

Yang, H., Jin, W.L., 2014. A control theoretic formulation of green driving strategies based on inter-vehicle communications. Transportation Research Part C 41, 48–60.

Yao, Z., Liu, M., Jiang, Y., Tang, Y., Ran, B., 2022. Trajectory reconstruction for mixed traffic flow with regular, connected, and connected automated vehicles on freeway. IET Intelligent Transport Systems .

Young, R.K., 1992. Wavelet theory and its applications. volume 189. Springer Science & Business Media.

Yu, H., Jiang, R., He, Z., Zheng, Z., Li, L., Liu, R., Chen, X., 2021. Automated vehicle-involved traffic flow studies: A survey of assumptions, models, speculations, and perspectives. Transportation research part C 127, 103101.

Yuan, J., Zheng, Y., Xie, X., Sun, G., 2011a. Driving with knowledge from the physical world, in: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 316–324.

Yuan, J., Zheng, Y., Zhang, C., Xie, W., Xie, X., Sun, G., Huang, Y., 2010. T-drive: driving directions based on taxi trajectories, in: Proceedings of the 18th SIGSPATIAL International conference on advances in geographic information systems, pp. 99–108.

Yuan, J., Zheng, Y., Zhang, L., Xie, X., Sun, G., 2011b. Where to find my next passenger, in: Proceedings of the 13th international conference on Ubiquitous computing, pp. 109–118.

Zheng, O., Abdel-Aty, M., Yue, L., Abdelraouf, A., Wang, Z., Mahmoud, N., 2022. CitySim: A Drone-Based Vehicle Trajectory Dataset for Safety Oriented Research and Digital Twins. URL: https://arxiv.org/abs/2208.11036, doi:10.48550/ARXIV.2208.11036.

Zheng, Y., Liu, Y., Yuan, J., Xie, X., 2011. Urban computing with taxicabs, in: Proceedings of the 13th international conference on Ubiquitous computing, pp. 89–98.

Zhou, Y., Ahn, S., Wang, M., Hoogendoorn, S., 2020. Stabilizing mixed vehicular platoons with connected automated vehicles: An h-infinity approach. Transportation Research Part B: Methodological 132, 152–170.

Zhou, Y., Lin, Y., Ahn, S., Wang, P., Wang, X., 2022. Platoon trajectory completion in a mixed traffic environment under sparse observation. IEEE Transactions on Intelligent Transportation Systems .