



Degree Project in Technology

Second cycle, 30 credits

Machine Learning and Motion Coordination for Battery Electric Vehicles

Safe, Efficient and Smart Framework for Online Learning

JACOPO DALLAFIOR

Machine Learning and Motion Coordination for Battery Electric Vehicles

**Safe, Efficient and Smart Framework for Online
Learning**

JACOPO DALLAFIOR

Master's Programme, ICT Innovation, 120 credits

Date: November 14, 2025

Supervisors: Esteban Gelso, Maliheh Sadeghi Kati, Amir Daghestani

Examiner: Mikael Johansson

School of Electrical Engineering and Computer Science

Host company: Volvo Group

Swedish title: Maskininlärning och rörelse koordinering för batterielektriska fordon

Swedish subtitle: Säkert och smart inlärningsramverk för inlärning online

Abstract

The integration of electric trailers with battery electric trucks offers a promising solution for extending driving range and improving energy efficiency in long-haul transport. A central challenge is optimizing the torque split between truck and trailer when the trailer's internal power-loss characteristics are unknown and proprietary. This thesis addresses this challenge by developing a safe online learning and control framework that enables the truck to identify the trailer's loss behavior during operation and leverage this knowledge for energy-efficient coordination.

The framework is modular, consisting of four interconnected blocks: learning, exploration, safety simulation, and control allocation. Learning is performed using Recursive Least Squares (RLS) as a lightweight parametric baseline and Gaussian Process (GP) regression as a non-parametric, uncertainty-aware alternative. Exploration is guided by heuristic coverage and Bayesian Optimization-inspired strategies, while safety is enforced via short-horizon dynamic simulations and Control Barrier Functions (CBFs). The learned trailer power-loss model is then embedded in optimization-based control allocation, solved via Quadratic Programming for RLS and Sequential Quadratic Programming for GP-based models.

Simulation studies on drive cycles show that GP models achieve up to 97% learning accuracy, reduce energy consumption by 2-4% compared to baseline strategies, and operate within 1% of an oracle with full trailer knowledge, all while preserving safety against jackknifing and trailer swing. These findings highlight the feasibility and benefits of safe online learning for articulated electric vehicles and provide insight into future deployment of data-driven control in heavy-duty transport.

Keywords

electric trucks, e-trailer, torque allocation, online learning, Gaussian processes, system identification, exploration, energy efficiency, safety-critical control, control barrier functions

Sammanfattning

Integrationen av elektriska släpvagnar med batterielektriska lastbilar erbjuder en lovande lösning för att förlänga körsträckan och förbättra energieffektiviteten inom fjärrtransporter. En central utmaning är att optimera vridmomentfördelningen mellan lastbil och släp när släpets interna effektförluster är okända och proprietära. Denna avhandling behandlar denna utmaning genom att utveckla ett säkert ramverk för onlineinlärning och styrning som gör det möjligt för lastbilen att identifiera släpets förlustbeteende under drift och utnyttja denna kunskap för energieffektiv samordning.

Ramverket är modulärt och består av fyra sammankopplade block: inlärning, utforskning, säkerhetssimulering och styrallokering. Inlärningen genomförs med Recursive Least Squares (RLS) som en lättviktig parametrisk baslinje och Gaussian Process (GP)-regression som ett icke-parametriskt, osäkerhetsmedvetet alternativ. Utforskningen styrs av heuristisk täckning och strategier inspirerade av Bayesisk optimering, medan säkerheten garanteras genom korttidsdynamiska simuleringar och Control Barrier Functions (CBFs). Den inlärd modellen av släpets effektförluster används sedan i en optimeringsbaserad styrallokering, löst med kvadratisk programmering för RLS och sekventiell kvadratisk programmering för GP-baserade modeller.

Simuleringsstudier på körcykler visar att GP-modeller uppnår upp till 97% inlärningsnoggrannhet, minskar energiförbrukningen med 2–4% jämfört med baslinjestrategier och fungerar inom 1% av ett orakel med full kännedom om släpet, samtidigt som säkerheten mot fällknivs- och slängrorelser bevaras. Dessa resultat visar på genomförbarheten och fördelarna med säker onlineinlärning för ledbundna elfordon och ger insikter för framtida implementering av databaserad styrning inom tung transport.

Nyckelord

elektriska lastbilar, e-släp, vridmomentfördelning, onlineinlärning, Gaussiska processer, systemidentifiering, utforskning, energieffektivitet, säkerhetskritisk styrning, kontrollbarriärfunktioner

Acknowledgments

This master's thesis, conducted as the final component of the EIT Digital double-degree programme in Autonomous Systems in collaboration with Volvo and KTH Royal Institute of Technology.

Stockholm, November 2025

Jacopo Dallafior

Contents

1	Introduction	1
1.1	Background	2
1.2	Problem Statement	3
1.3	Purpose	5
1.4	Research Methods	5
1.5	Related Work and Positioning of the Thesis	7
1.6	Structure of the thesis	8
1.7	Sustainability, ethics, and social impact	9
2	Background	11
2.1	System overview	11
2.1.1	Torque Mapping and Loss Modeling	12
2.1.2	Powertrain Losses and Available Signals	13
2.2	Least Squares Regression	14
2.3	Gaussian Processes for Regression	15
2.3.1	Kernel functions	15
2.3.2	Posterior prediction	16
2.3.3	Hyperparameter optimization	17
2.3.4	Sparse Solution	18
2.3.5	Heteroscedastic observation noise	18
2.3.6	Why Gaussian Processes	19
2.4	Exploration via Bayesian Optimization	19
2.5	Optimal Control Theory	20
2.5.1	Quadratic Programming	22
2.5.2	Sequential Quadratic Programming	22
2.5.3	Control Barrier Functions: Theory and Safety Enforcement	24
2.5.4	Control Barrier Functions for Non Affine Systems	25
2.5.5	Control Barrier Functions over a finite Horizon	26
3	Methodology	27
3.1	Learning of the Power Loss Surface	29

3.1.1	System Identification Approach via Recursive Least Squares	30
3.1.2	Gaussian Process Learning	31
3.1.3	Prior Knowledge and Normalization Strategy	34
3.2	Exploration Strategies	36
3.2.1	Grid-Based Heuristic Exploration	37
3.2.2	Uncertainty-Based Exploration	37
3.3	Motion Prediction	38
3.3.1	Dynamical Model Description	38
3.3.2	Prediction Method	39
3.3.3	Unsafe Behavior Detection Criteria	41
3.4	Motion Coordination	41
3.4.1	Optimal control strategies	43
3.4.2	Control Barrier Function	45
3.4.3	Final Formulation	48
4	Results	51
4.1	Data and Environment Setup	51
4.1.1	Measurement Setup	52
4.1.2	Driving System and Drive Cycles	53
4.2	Evaluation Metrics and Benchmarks	54
4.3	Gaussian Process Learning Analysis	56
4.3.1	Computational Cost and Sparse Approximation	57
4.3.2	Hyperparameter Optimization and Model Adaptation	58
4.3.3	Surface Evolution over Time	60
4.4	Recursive Least Squares Learning Analysis	61
4.4.1	Impact of Noise on RLS	63
4.5	Exploration Analysis	64
4.5.1	Grid-Based Exploration	64
4.5.2	UCB Parameter Sensitivity	64
4.6	Energy Efficiency Analysis	66
4.6.1	Energy Consumption	66
4.6.2	Energy Losses	68
4.6.3	Learning and Exploration Cost	70
4.7	Safety Evaluation	71
4.7.1	Detection of Unsafe Behavior	71
4.7.2	Safety Mechanisms: Rollout Simulation and Control Barrier Function	73
4.8	Optimal Control Performance	74
4.8.1	Uncertainty-Aware Optimization	75

5	Conclusion and Future work	79
5.1	Contributions to Literature	80
5.2	Future Work	81
A	Force Request Computation	83
B	Heteroscedastic noise	85
B.1	Composite Kernel Formulation	85
B.2	Modeling the Noise Variance	86
C	Induction point selection	88
D	Dynamic Vehicle Model	89
D.1	Equations of the vehicle	89
D.1.1	Equations of Motion of the Truck	89
D.1.2	Equations of Motion of the Trailer	90
D.1.3	Coupling Equations	90
D.1.4	Model Summary	91
E	Gaussian Process Derivatives	92
E.1	Joint distribution with derivatives	92
E.2	Squared Exponential kernel	92
E.3	Use in control	93
F	Solver Implementation	94
	References	95

List of acronyms and abbreviations

AEV	Articulated Electric Vehicle
ARD	Automatic Relevance Determination
BFGS	Broyden–Fletcher–Goldfarb–Shanno
BO	Bayesian Optimization
CAN	Controller Area Network
CBF	Control Barrier Function
EV	Electric Vehicle
FITC	Fully Independent Training Conditional
FMU	Functional Mock-up Unit
GP	Gaussian Process
GPR	Gaussian Process Regression
IM	Induction Motor
ISO	Isotropic
KKT	Karush Kuhn Tucker
LCB	Lower Confidence Bound
MPC	Model Predictive Control
OLS	Ordinary Least Squares
PMSM	Permanent Magnet Synchronous Motor
QP	Quadratic Programming
RLS	Recursive Least Squares
RMSE	Root Mean Squared Error

SE	Squared Exponential
SQP	Sequential Quadratic Programming
UCB	Upperr Confidence Bound
WHVC	World Harmonized Vehicle Cycle

Chapter 1

Introduction

In the last years, the electric revolution have interested nearly all types of transportation. Electric vehicles (EVs) are becoming increasingly common, particularly in urban and short-range applications. However, one of the most critical limitations with this vehicle is their restricted driving range. Although this is not a major concern for vehicles that operate short distances, it becomes a significant obstacle in the context of long-haul transportation. Trucks that need to travel hundreds of kilometers per day cannot rely fully on electric solutions alone due to the combination of limited range and long charging times.

To overcome this limitation, several truck manufacturers have begun exploring the idea of using electric trailers, those equipped with their own electric motors and battery packs [1]. These trailers support the propulsion system of the truck, effectively increasing the total driving range. Having both the truck and the trailer that contribute to the motion result in an improvement of the overall performance and efficiency of the system. However, this set-up also introduces a new layer of complexity, particularly in coordinating the distribution of ueue between the two units in various driving scenarios. This concept of distribution is defined as torque split, that can be properly defined as the total required torque divided between the truck and the trailer.

Achieving optimal torque split typically requires full access to motor characteristics. Truck manufacturers typically produce electric trucks but do not manufacture electric trailers. As a result, vehicle combinations often consist of the truck from one manufacturer and a trailer from another manufacturer. It may be challenging to determine the trailers characteristics, as sharing or obtaining the necessary data for integration could be difficult. This creates major challenges when the truck attempts to optimize energy usage throughout the entire system, as it cannot accurately estimate the contribution of the trailer to propulsion.

This thesis tackles this challenge by addressing two key objectives. It explores methods for safely learning useful information about the trailer motor, and

it proposes strategies to exploit this learned knowledge to optimize energy consumption in the combined truck–trailer system.

1.1 Background

Articulated Electric Vehicles (AEVs) used for transportation consist of a truck unit with an electric driveline that powers the vehicle and normal unpowered trailer. Examples of electric trucks developed by Volvo Trucks include FH-Areo electric, FM electric, FE electric, with a maximum driving range estimated at approximately 300km for the first two and around 450km for the FE electric under typical operating conditions [2]. Due to this limited range, electric trucks are currently used primarily for urban and short-distance logistics, long-haul trucking, where daily distances often exceed 681.31 km [3], remains dominated by diesel vehicles, as current battery charging times and driving range are not sufficient for this type of travelling.

To address this limitation, manufacturers have begun exploring the concept of electric trailers, which are equipped with their own electric motors and battery packs. By contributing additional propulsion and energy storage capacity, these trailers can extend the total range travelled. For example the article [1] shows that the addition of an electric semi-trailer can reduce fuel consumption up to 14% with an average savings of 12% while the article proposed from [4] shows how the use of the e-trailer can achieve 36.3% efficiency improvements.

The general structure of such an AEV system is illustrated in Figure 3.10.

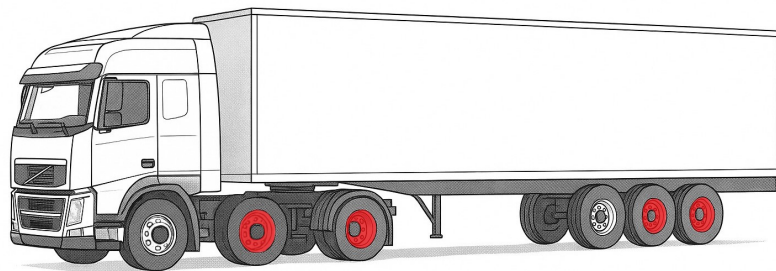


Figure 1.1: The image shows a truck and trailer combination, the wheel with propulsion are here shown in red color.

The propulsion is distributed between the truck and the trailer, this requires torque coordination between the two units, torque split. Optimal torque split strategies are well-studied in electric vehicles with dual motors in the same drivetrain [5] [6], where full system information is available and can be used in centralized optimization algorithms. However, in the case of a truck-trailer combination where the trailer is manufactured by a third party, the truck's control unit often lacks

detailed knowledge about the trailer’s motor characteristics. This creates a novel research challenge on how to optimize torque split between a known and an unknown subsystem as is schematized in Figure 1.2. At the heart of this problem is the need to access critical information from the trailer while using a minimal number of CAN bus ports. The current CAN bus, originally designed for non-motorized trailers, is not sufficient for electric trailers. Although it is now being adapted to support the electric revolution, another challenge is that the interface is standardized, which limits design flexibility and complicates the integration of new functionalities.



Figure 1.2: System overview, the e-truck (known, controllable) interfaces with an e-trailer treated as an unknown subsystem; only limited signals are available across the truck–trailer link, motivating online identification and safety-aware control.

From a control and optimization standpoint, two types of information are typically required, such as efficiency maps or power loss models [6] [7]. Efficiency maps are useful for performance analysis and decision-making across operating zones, while power loss surfaces are directly applicable to optimization. Due to their mathematical properties, such as similar-quadratic behavior over specific input domains, this thesis chooses to focus on learning the power loss surface (3D function of torque and angular velocity) of the trailer drivetrain as the basis for torque optimization. The limitations of using efficiency maps in this context, and the advantages of the power loss formulation, are discussed in detail in **Section 2.2**. To enable optimal torque split through control allocation, the truck must first estimate the trailer’s power loss characteristics. This thesis addresses this by applying two learning strategies: (1) a system identification approach, building upon and enhancing a method previously developed at Volvo [8], and (2) a machine learning-based method to evaluate whether the estimation can be improved. Both approaches rely on actively exploring the trailer’s power loss map through observed system behavior. A key challenge not fully addressed in previous work is how to ensure this exploration and learning process, remains both safe and efficient.

1.2 Problem Statement

Learning the power loss characteristics of the drivetrain of the trailer is a central goal of this thesis, however, it is too wide to be defined as a clear objective for this

thesis, since this objective is divided into more complex underlying problems. The real challenges related to the learning is to have an online automated learning, this means that the learning must occur in real-time while the truck is driving and must not interfere with the safety or stability of the vehicle.

Automated learning means the algorithm should actively guide the vehicle's behavior to explore the torque and velocity space on which the power loss depends. This exploration must be conducted efficiently and safely, to avoid causing unstable or dangerous behavior.

A critical risk during such learning is the occurrence of unsafe driving modes, such as jackknifing and trailer swing, those are shown in Figure 1.3.

To avoid these risks, the learning process must include safety mechanisms that

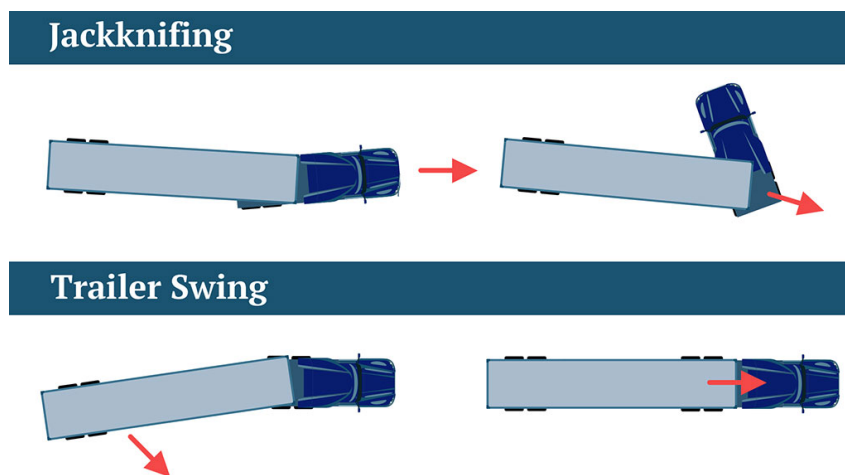


Figure 1.3: (a) Jackknifing occurs when the angle between the truck and trailer becomes too sharp, especially during braking or abrupt maneuvers, potentially leading to loss of control.(b) Trailer swing refers to uncontrolled lateral motion of the trailer, which can happen during poorly coordinated torque transitions or high-speed driving [9].

prevent control actions likely to lead to such instabilities.

Another core problem lies in accurately estimating the power loss from the limited information available on the CAN bus. Since direct access to the trailer's internal parameters is unavailable, the system must identify which variables are sufficient to estimate power loss reliably, and how to construct a model from them.

The research question and the core objective for this thesis are as follows.

How can a vehicle intelligently and safely learn the power loss characteristics of an unknown trailer drivetrain, using only limited external data, and use this knowledge to enable optimal torque split?

1.3 Purpose

The purpose of this thesis is to evaluate whether real-time learning of an electric trailer's behavior can improve torque coordination and overall driving efficiency in AEVs. Currently, if the trailer unit is unknown due to proprietary data restrictions or the absence of an interface for sharing knowledge. A common industrial approach is to implement a rule-based control strategy to allocate power between the vehicle units throughout the entire trip. This approach may lead to suboptimal performance in varying driving conditions.

This thesis proposes a framework to investigate whether online learning using either system identification or machine learning can offer a more adaptive and efficient alternative. The intended contribution is not a final product ready for deployment, but a proof-of-concept research framework that explores the feasibility and potential benefits of real-time learning and data-driven torque optimization. This thesis offers industrial insights into whether such learning-based strategies are worth exploring further. From an academic perspective, the thesis aims to demonstrate how complex engineering challenges can be decomposed into manageable sub-problems such as safety-aware learning, exploration, and estimation and integrated into a coherent, working solution. It also highlights how modern machine learning techniques can be safely applied to explore unknown systems within strict operational constraints.

1.4 Research Methods

This thesis is developed utilizing a modular methodology, with an approach inspired by the increasing use of reinforcement learning (RL) in robotics and industrial applications. The framework mimics the concept in RL of alternating between exploration and exploitation: the system first learns the trailer's power loss behavior during operation, and then uses the acquired knowledge to optimize performance. The modular methods chosen consist in the division of four main tasks that will be carried individually and then connected, to avoid a monolithic pipeline that would have resulted in a more complex and difficult to understand approach. The Figure 1.4 show all the different blocks combined together in what is the main combined framework for this project.

1. Learning. The first block focuses on building an accurate model of the trailer's drivetrain power losses as a function of torque and speed. Previous research [8] used Ordinary Least Squares (OLS) and random probing, which proved inefficient for online adaptation. This thesis improves on that baseline by employing Recursive Least Squares (RLS) [10], for real-time parametric identification, and extending it with Gaussian Process (GP) regression to capture non-linear surfaces [11] and provide calibrated uncertainty estimates. Such uncertainty is key for

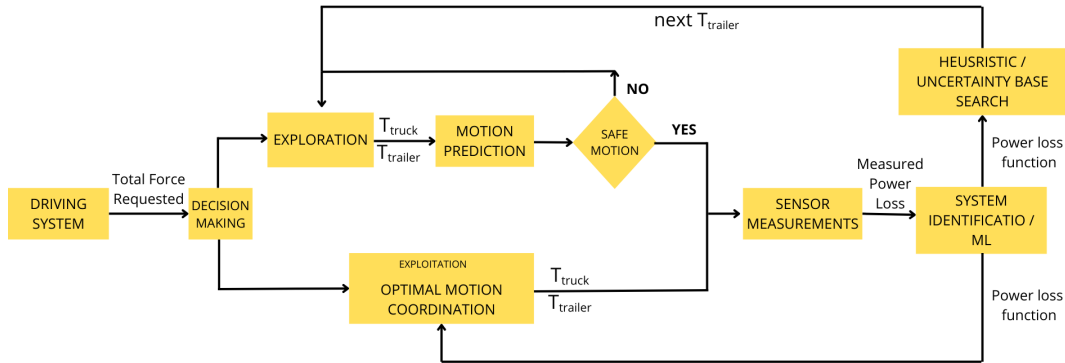


Figure 1.4: Overview of the safe online learning and control framework. A total force request from the driving system is processed by the decision-making module. In exploration, candidate truck–trailer torque values are generated and evaluated by a motion prediction block using short-horizon dynamics. Safe proposals trigger sensor-based power loss estimation via system identification or machine learning, with uncertainty quantification. The learned power loss surface guides both exploration and exploitation, where optimal torque coordination is computed for energy efficiency. Unsafe torque splits are rejected and re-evaluated.

guiding safe learning in low-data regimes [12], where neural networks or other heavy offline methods are unsuitable.

2. Exploration. To learn effectively, the system must sample informative operating points rather than rely on random actions. Exploration strategies are therefore introduced to drive data collection: heuristic grid-based coverage ensures uniform sampling of the torque–velocity map, while uncertainty-driven exploration [13], inspired by Bayesian Optimization, prioritizes regions where the GP model is least certain [14], [15]. This combination balances exploration efficiency with model fidelity.

3. Safety Simulation. Since exploratory torque splits may lead to unsafe dynamics, each candidate input is first screened through a short-horizon safety simulation. This predictive check uses a dynamic truck–trailer model to anticipate behaviors such as jackknifing or trailer swing, rejecting unsafe candidates before they are applied. In this way, the framework guarantees that learning proceeds without compromising vehicle stability.

4. Control Allocation. Once the trailer loss surface has been sufficiently learned, the framework transitions to exploitation, where torque is allocated between the truck and trailer motors to minimize total energy consumption. The choice of solver depends on the model: Quadratic Programming (QP) is used when the loss surface is represented by RLS, while Sequential Quadratic Programming (SQP) is employed for GP-based models, as it can handle non-convex costs and incorporate uncertainty into a risk-aware optimization.

Together, these four blocks form an online learning and control pipeline that enables both data-driven adaptation and rigorous safety guarantees.

The developed framework is evaluated against two baselines:

- The constant torque split strategy [16] that is a common industrial approach.
- The improved RLS and GP-based approaches.

The metrics used for the evaluation are the learning accuracy, measured using percentage Root Mean Squared Error (RMSE) between the estimated and true power loss values and the driving efficiency, measured by the total energy consumed over a simulated trip.

1.5 Related Work and Positioning of the Thesis

Research on torque allocation and energy optimization in electric vehicles has been active for over a decade, with most approaches assuming full access to motor characteristics. For example, optimal torque split strategies in dual-motor drivetrains rely on centralized optimization where efficiency maps or loss surfaces are fully known [5, 6]. These methods achieve high performance but are not directly applicable to articulated electric vehicles (AEVs), where the trailer may be manufactured by a different company and proprietary information is unavailable. This creates a knowledge gap that prevents the direct transfer of existing centralized strategies to the truck–trailer context.

Early work at Volvo explored system identification techniques to approximate unknown loss surfaces. Erdinc et al. [8] applied Ordinary Least Squares (OLS) with random probing, but this approach was inefficient and not well suited for online learning. Recursive Least Squares (RLS) was later introduced as a lightweight parametric estimator with online update capabilities [10]. While efficient, these methods rely on polynomial assumptions and cannot capture complex nonlinearities. Machine learning methods, in particular Gaussian Processes (GPs), have emerged as promising non-parametric alternatives: they provide smooth function approximations, calibrated predictive uncertainty, and are compatible with optimization-based control [11, 12]. Recent studies demonstrate that GPs enable cautious control in low-data regimes and can be integrated with model predictive control for energy management [17, 18].

Another important line of work relates to how data are collected for system identification. Random or grid-based probing ensures coverage but is inefficient in high-dimensional spaces. Bayesian Optimization (BO) and its extensions offer a principled way to guide exploration using model uncertainty. Acquisition strategies such as Upper and Lower Confidence Bounds (UCB/LCB) balance exploitation and exploration, achieving sublinear regret guarantees [14, 15, 19]. Such methods have

been applied in robotics and adaptive control, but their integration into heavy-duty vehicle energy optimization remains underexplored.

Finally, ensuring safety during learning and exploitation is a critical challenge. Classical torque allocation methods rarely account for vehicle-level stability risks such as jackknifing or trailer swing. Control Barrier Functions (CBFs) provide formal guarantees of forward invariance for safety sets and can be embedded as constraints in optimization-based controllers [20, 21]. Extensions to non-affine dynamics and finite-horizon safety enforcement broaden their applicability to complex vehicle systems [22, 23]. However, prior studies typically assume known system dynamics, whereas articulated electric vehicles involve significant uncertainty in trailer behavior.

In summary, existing research provides strong foundations in torque allocation, system identification, machine learning, exploration strategies, and safety-critical control. However, no prior work has combined these elements into a unified framework for online learning of trailer characteristics under limited data availability, while simultaneously enforcing safety. This thesis contributes by bridging these research directions: it integrates parametric and non-parametric online learning, Bayesian optimization-inspired exploration, rollout-based and barrier-function safety enforcement, and optimization-based torque allocation. In doing so, it addresses the research question of how an AEV can intelligently and safely learn an unknown trailer's power loss characteristics and exploit this knowledge for energy-efficient coordination.

1.6 Structure of the thesis

The remainder of this thesis is organized in four main parts that follow a natural progression from theory to implementation and evaluation. Chapter 2 introduces the necessary background, starting with the structure of the system and the modeling of trailer power losses, before presenting the theoretical tools that underpin the proposed framework. These include parametric methods such as Least Squares, non-parametric approaches such as Gaussian Processes with uncertainty quantification, exploration strategies based on Bayesian Optimization, optimization techniques including Quadratic and Sequential Quadratic Programming, and finally safety concepts grounded in Control Barrier Functions. Chapter 3 describes the methodology, where these elements are combined into a modular framework composed of four functional blocks: learning, exploration, safety simulation, and control allocation. The chapter also outlines the simulation environment, assumptions, and data setup. Chapter 4 reports the results obtained from the implementation of the framework, evaluating model learning accuracy, exploration efficiency, safety performance, and the effectiveness of the control allocation

when compared to baseline strategies. Finally, Chapter 5 concludes the work by summarizing the main findings, discussing the limitations of the current approach, and proposing directions for future research in safe online learning and control of articulated electric vehicles.

1.7 Sustainability, ethics, and social impact

This work addresses sustainability at three coupled levels: energy, materials, and operations. By learning the trailer's power-loss surface online and allocating torque to minimize conversion losses under safety constraints, the framework reduces electrical energy drawn for a given duty cycle, which in turn lowers upstream emissions for grids that are not yet fully decarbonized. Because the optimization acts at the axle and not only at the mission level, it can also reduce thermal stress on converters and machines, indirectly extending component lifetime and delaying material replacement. At the same time, ethical deployment requires that efficiency gains never trade against safety. The project therefore treats safety as a hard constraint: candidate torque splits are screened with short-horizon vehicle dynamics and filtered through barrier-function constraints so that exploration never induces jackknifing or trailer swing, and exploitation cannot erode stability margins. Data collected from the trailer are minimized to what is strictly necessary for estimation, processed on-board, and used solely for control; this respects the proprietary nature of third-party subsystems and reduces the privacy and cybersecurity surface exposed by inter-unit communication. The technical goal, learning-based torque coordination under uncertainty, is aligned with environmental objectives, constrained by formal safety guarantees.

Chapter 2

Background

This section collects the theory required for a reader to follow the thesis and fully understand it.

Starting with the system structure and why modeling trailer losses is central to energy efficiency. Then, after presenting Least Squares as a classical parametric baseline, Gaussian Processes are introduced as a non-parametric alternative to approximate the unknown loss surface. Unlike RLS, they not only provide flexible function approximation but also deliver calibrated uncertainty estimates. Since learning requires active data collection, exploration principles are introduced from Bayesian Optimization, which formalize how to trade off exploration with informative sampling.

Once a model is available, torque allocation becomes an optimization problem, requiring methods such as Quadratic Programming and Sequential Quadratic Programming. Finally, because even optimal allocations can be unsafe, Control Barrier Functions are introduced as a mechanism to ensure vehicle stability and safety during both learning and exploitation.

This progression provides the theoretical foundation needed for the methodology and results: starting from system modeling, moving through data-driven learning and exploration, and concluding with optimization and safety enforcement.

2.1 System overview

The considered system is a tractor-trailer combination in which the trailer is an electric unit with its own traction axle. The tractor must split a requested longitudinal effort between its own drive units and the trailer axle, despite having only a limited set of trailer-side signals. The objective is to minimize total energy consumption during driving while staying within safety envelopes. To this end, the tractor learns online the trailer power-loss surface $P_{\text{loss}}(T, \omega)$ from measurements, exploring several operating points (T, ω) safely, and computes a torque split that accounts for

model uncertainty and safety constraints.

The symbols used throughout all the thesis will be now presented in Table 2.1.

Table 2.1: Symbols used

Symbol	Meaning (units)
$T \in \mathbb{R}$	Axle torque at a given unit (N m)
$\omega \in \mathbb{R}_{\geq 0}$	wheel angular speed (rad/s)
$P_{\text{loss}}(T, \omega)$	Electrical–mechanical conversion losses (W) as a function of axle torque and wheel angular speed
$F_{x,\text{req}} \in \mathbb{R}$	Requested total longitudinal force (N)
$\mathbf{T} = [T_{\text{truck}}, T_{\text{trailer}}]^\top$	Torque split decision (N m)
$\mu(x), \sigma(x)$	GP predictive mean and std. at $x = (T, \omega)$
$\mathcal{D}_k = \{(x_i, y_i)\}_{i=1}^{n_k}$	Data up to step k , with $y_i = P_{\text{loss}}(x_i)$
$\Delta t = 1/f_c$	Sampling time; f_c controller rate (s, Hz)
\mathcal{T}, Ω	Admissible torque/speed domains
T^{\min}, T^{\max}	Torque bounds (per unit) (N m)
ΔT^{\max}	Torque rate limit (N m per step)
r_w	Wheel effective radius (m)
g	Overall gear ratio (dimensionless)
x	Vehicle state used for safety checks (various)
$h(x)$	Safety function; safe set $\mathcal{C} := \{x : h(x) \geq 0\}$
N_s	Horizon length for safety screening (steps)

2.1.1 Torque Mapping and Loss Modeling

Having introduced the system and notation, it is important to define how the actual requested longitudinal effort is mapped into axle torques. This connection is crucial, since torque allocation directly determines the operating points (T, ω) at which losses are observed and learned by the tractor.

Given a requested longitudinal force $F_{x,\text{req}}$, the tractor distributes nominal axle forces $F_{x,i}$ such that $\sum_i F_{x,i} \approx F_{x,\text{req}}$. The corresponding torques follow

$$T_i \approx \frac{r_{w,i}}{g_i} F_{x,i},$$

where $r_{w,i}$ and g_i denote wheel radius and gear ratio, respectively. The allocation problem then seeks a torque vector \mathbf{T} that lies within bounds and rate limits, while minimizing overall conversion losses.

To evaluate efficiency, a learned trailer power loss model is employed. At any operating point $x = (T, \omega)$, the model provides a predictive distribution with mean $\mu(x)$ and standard deviation $\sigma(x)$. The mean captures the expected efficiency behavior, whereas the variance quantifies model and sensors uncertainty. Both

quantities can be exploited: $\mu(x)$ enters directly in cost minimization, while $\sigma(x)$ supports risk-aware optimization and can guide exploration by encouraging the controller to visit uncertain regions of the map.

2.1.2 Powertrain Losses and Available Signals

For a multi-source drivetrain such as the truck with an e-trailer case, minimizing energy consumption relies on the maximal reduction of power losses per-unit. Previous studies have adopted this principle to optimize torque allocation in multi-motor drive systems [5, 6], typically under the assumption of full model knowledge. However, its application to scenarios involving units with unknown or uncertain characteristics has not yet been explored. The trailer's electrical-mechanical conversion losses are described by the surface

$$P_{\text{loss}}(T, \omega) = P_{\text{in}}(T, \omega) - P_{\text{mech}}(T, \omega),$$

where P_{in} denotes the electrical input power and P_{mech} the mechanical output power. When the machine works as a generator, P_{elec} corresponds to P_{out} .

It is widely used in the literature to approximate the internal power losses as a locally quadratic polynomial, dependent on torque with speed-related coefficients ($a(\omega), b(\omega), c(\omega)$)[6, 7]. A common surrogate is

$$P_{\text{loss}}(T, \omega) \approx a(\omega)T^2 + b(\omega)T + c(\omega), \quad (2.1)$$

This form preserves convexity in T for fixed ω , enabling efficient allocators. When replaced by a Gaussian Process model, the loss surface remains smooth with a proper kernel but may exhibit non-convexities.

The tractor only has access to a limited set of trailer-side signals. These measurements are used to reconstruct or approximate $P_{\text{loss}}(T, \omega)$. Table 2.2 summarizes the available signals and their role within the framework.

Measurement noise is not uniform across the operating map but depends on the input conditions. To capture these effects, observations are modeled as

$$y = P_{\text{loss}}(x) + \varepsilon(x), \quad \varepsilon(x) \sim \mathcal{N}(0, \sigma_m^2(x)), \quad (2.2)$$

with $x = (T, \omega)$ and input-dependent variance $\sigma_m^2(\cdot)$.

Over a control step of duration Δt , the incremental energy penalty can be approximated as $\Delta E \approx P_{\text{loss}}(T, \omega) \Delta t$. For the full vehicle combination, losses add across units,

$$\Delta E_{\text{tot}} = \sum_{u \in \{\text{truck}, \text{trailer}\}} P_{\text{loss}, u} \Delta t.$$

With the system structure and signals defined, the next step is to choose a model that

Table 2.2: Trailer-side signals used by the tractor and typical noise characteristics.

Signal	Purpose in framework	Dominant noise
Wheel speed ω	Input to $P_{\text{loss}}(T, \omega)$; operating-point indexing	Quantization at low ω
Torque estimate \hat{T}	Input to $P_{\text{loss}}(T, \omega)$; allocation variable	Estimation bias; sensor drift
DC link current I_{dc}	Loss/efficiency surrogate via $P_{\text{el}} = V_{\text{dc}} I_{\text{dc}}$	Ripple; temperature drift
DC link voltage V_{dc}	As above; detect supply transients	Converter dynamics

can represent the trailer's unknown power losses while adapting online.

2.2 Least Squares Regression

A classical approach to modeling unknown relationships from data is *Least Squares* regression. Given a set of observations $\{(x_i, y_i)\}_{i=1}^n$ with input $x_i \in \mathbb{R}^d$ and output $y_i \in \mathbb{R}$, the model assumes a parametric structure $y_i \approx \phi(x_i)^\top \theta$, where $\phi(x_i)$ is a feature vector (e.g., polynomial terms) and θ the coefficient vector to be estimated. The OLS estimate minimizes the sum of squared errors:

$$\theta^* = \arg \min_{\theta} \sum_{i=1}^n (y_i - \phi(x_i)^\top \theta)^2, \quad (2.3)$$

which admits the closed-form solution

$$\theta^* = (\Phi^\top \Phi)^{-1} \Phi^\top y, \quad (2.4)$$

where $\Phi = [\phi(x_1)^\top; \dots; \phi(x_n)^\top]$ is the regression matrix and $y = [y_1, \dots, y_n]^\top$ the measured output.

While OLS provides a batch solution, it becomes computationally heavy for online learning, as every new measurement would require recomputing the inverse. For real-time applications, as proposed by [10], the RLS algorithm is widely adopted. RLS updates the parameter vector θ_k efficiently as new data arrive, by minimizing an exponentially weighted squared error:

$$\theta_k = \arg \min_{\theta} \sum_{i=1}^k \lambda^{k-i} (y_i - \phi(x_i)^\top \theta)^2, \quad (2.5)$$

where $0 < \lambda \leq 1$ is the *forgetting factor*, which determines how fast past data are

discounted. The update equations are recursive:

$$K_k = \frac{P_{k-1}\phi(x_k)}{\lambda + \phi(x_k)^\top P_{k-1}\phi(x_k)}, \quad (2.6)$$

$$\theta_k = \theta_{k-1} + K_k (y_k - \phi(x_k)^\top \theta_{k-1}), \quad (2.7)$$

$$P_k = \lambda^{-1} (P_{k-1} - K_k \phi(x_k)^\top P_{k-1}), \quad (2.8)$$

where P_k is the covariance matrix and K_k the gain vector. This formulation allows incremental coefficient updates at each iteration with a computational cost that scales quadratically with the number of adaptive coefficients, making RLS suitable for online estimation.

In the context of trailer power-loss modeling, RLS corresponds to fitting a polynomial function of torque and speed, with coefficients updated as new samples arrive. Its advantages are simplicity, interpretability, and fast updates; however, it cannot quantify predictive uncertainty and its accuracy degrades when the true loss surface deviates from the chosen polynomial structure. For this reason, RLS serves as a baseline in this work, while Gaussian Processes provide a more flexible non-parametric alternative with calibrated uncertainty estimates.

2.3 Gaussian Processes for Regression

Gaussian Process Regression (GPR) provides a nonparametric Bayesian approach for learning unknown functions from data, with explicit quantification of predictive uncertainty. A GP is formally defined as a collection of random variables, any finite subset of which follows a joint Gaussian distribution:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')), \quad (2.9)$$

where $m(x)$ is the mean function and $k(x, x')$ is the covariance (kernel) function. The mean function is often taken as zero, $m(x) = 0$, without loss of generality, since the data can always be normalized. The kernel function encodes prior assumptions on smoothness, scale, and correlations of the latent function [24].

2.3.1 Kernel functions

A common choice is the squared exponential (SE) kernel, also known as radial basis function (RBF):

$$k_{\text{SE}}(x, x') = \sigma_f^2 \exp\left(-\frac{\|x - x'\|^2}{2\ell^2}\right), \quad (2.10)$$

where σ_f^2 is the signal variance and ℓ is the characteristic lengthscale controlling how rapidly correlations decay. More general is the Automatic Relevance Determination

(ARD) form, where each input dimension has its own lengthscale ℓ_j :

$$k_{\text{SE-ARD}}(x, x') = \sigma_f^2 \exp\left(-\frac{1}{2} \sum_{j=1}^d \frac{(x_j - x'_j)^2}{\ell_j^2}\right). \quad (2.11)$$

A further important property of Gaussian Processes is that, when the kernel is differentiable, the posterior mean and variance are themselves differentiable functions. In fact, derivatives of a GP are again GPs [25], obtained by differentiating the kernel with respect to its arguments. This ensures that gradient information is analytically available and can be consistently incorporated into optimization routines. For our application, this guarantees compatibility with SQP, which requires smooth gradients of the surrogate model, like SE kernel that is infinitely differentiable. Full expressions for derivative covariances are reported in the Appendix E.

2.3.2 Posterior prediction

Suppose the observed data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ with Gaussian measurement noise

$$y_i = f(x_i) + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma_n^2). \quad (2.12)$$

For any finite set of inputs, the GP prior implies that the training outputs $\mathbf{f} = [f(x_1), \dots, f(x_n)]^\top$ and the function values at test inputs $\mathbf{f}^* = f(X^*)$ are jointly Gaussian,

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(X, X) & K(X, X^*) \\ K(X^*, X) & K(X^*, X^*) \end{bmatrix}\right), \quad (2.13)$$

where $K(X, X)$ denotes the kernel matrix between training points, the other blocks are the cross-covariances. Conditioning this joint Gaussian on the observed training outputs yields the posterior distribution for the test values,

$$\mathbf{f}^* | X, X^*, \mathbf{f} \sim \mathcal{N}\left(K(X^*, X)K(X, X)^{-1}\mathbf{f}, K(X^*, X^*) - K(X^*, X)K(X, X)^{-1}K(X, X^*)\right), \quad (2.14)$$

which shows explicitly how the mean prediction is a combination of the training outputs weighted by kernel similarities, while the variance shrinks in regions supported by data.

When noisy observations are considered, with $\mathbf{y} = \mathbf{f} + \varepsilon$ and $\varepsilon \sim \mathcal{N}(0, \sigma_n^2 I)$, the joint prior becomes

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(X, X) + \sigma_n^2 I & K(X, X^*) \\ K(X^*, X) & K(X^*, X^*) \end{bmatrix}\right), \quad (2.15)$$

and conditioning on \mathbf{y} yields the standard predictive equations

$$\mu(x_*) = K(X_*, X)(K(X, X) + \sigma_n^2 I)^{-1} \mathbf{y}, \quad (2.16)$$

$$\sigma^2(X_*) = K(X_*, X_*) - K(X_*, X)(K(X, X) + \sigma_n^2 I)^{-1} K(X, X_*). \quad (2.17)$$

Equation (2.16) provides the predictive mean, which interpolates smoothly between training points under the prior assumptions of the kernel, while Equation (2.17) quantifies epistemic uncertainty. This uncertainty is low near observed data and large in unexplored regions, making GPR a natural tool for exploration–exploitation trade-offs and for risk-aware optimization. As illustrated in Figure 2.1, Gaussian Processes model distributions over functions.

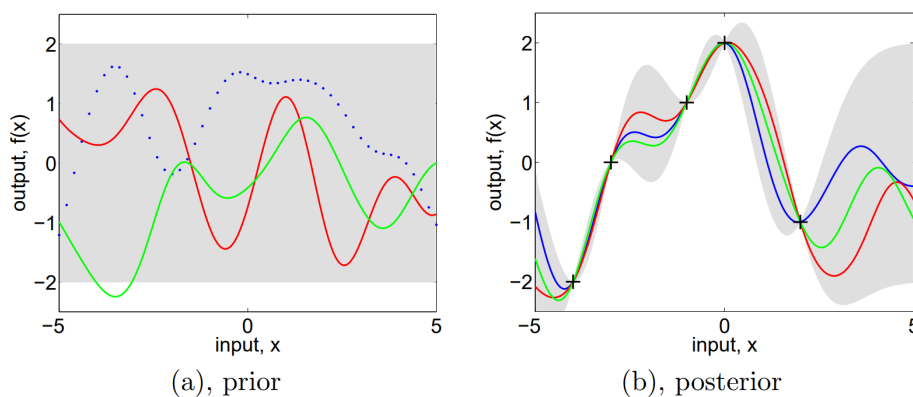


Figure 2.1: Gaussian Process regression. (a) Prior: sample functions (colored lines) drawn from the GP before any data are observed. The shaded region indicates the prior uncertainty (95% confidence interval). (b) Posterior: after conditioning on training points (black crosses), the GP posterior mean (black line) and posterior samples (colored lines) adapt to the data. The shaded region shows the reduced predictive uncertainty near observations and its growth away from them. Adapted from [24].

2.3.3 Hyperparameter optimization

The kernel and likelihood involve hyperparameters $\theta = \{\sigma_f^2, \ell_j, \sigma_n^2\}$. Their values critically affect predictive accuracy. A principled approach is to optimize the log marginal likelihood [24]:

$$\log p(\mathbf{y}|\mathbf{X}, \theta) = -\frac{1}{2} \mathbf{y}^\top (\mathbf{K}_\theta + \sigma_n^2 \mathbf{I})^{-1} \mathbf{y} - \frac{1}{2} \log |\mathbf{K}_\theta + \sigma_n^2 \mathbf{I}| - \frac{n}{2} \log 2\pi. \quad (2.18)$$

The first term enforces data fit, the second penalizes model complexity (Occam’s razor), and the third is a normalization constant. Maximization of this criterion (also

called type-II maximum likelihood or ML-II) automatically balances underfitting and overfitting. Gradient-based optimizers are efficient since analytic derivatives w.r.t. θ exist, but the objective is non-convex and can admit local minima, requiring careful initialization or global optimization strategies.

2.3.4 Sparse Solution

A known limitation of Gaussian Processes is their cubic complexity in the number of data points, $\mathcal{O}(n^3)$, arising from the inversion of the $n \times n$ covariance matrix. As the dataset $\mathcal{D}_k = \{(x_i, y_i)\}_{i=1}^n$ grows online, this inversion quickly becomes the computational bottleneck. For real-time control and allocation, such scaling is undesirable.

Sparse GP methods address this challenge by introducing a smaller set of *inducing points* $Z = \{z_1, \dots, z_m\}$, where $m \ll n$. These points act as a compressed representation of the dataset, chosen either by optimization or heuristics [26]. The observed training inputs $X = \{x_1, \dots, x_n\}$ are then related to Z via cross-covariances, reducing complexity to $\mathcal{O}(nm^2)$ instead of $\mathcal{O}(n^3)$.

One of the most widely used sparse approximations is the *Fully Independent Training Conditional* (FITC) model fully explained by [27]. The approximate covariance matrix is defined as

$$K_{\text{FITC}} = Q_{nn} + \text{diag}(K_{nn} - Q_{nn}), \quad Q_{nn} = K_{nm}K_{mm}^{-1}K_{mn}, \quad (2.19)$$

where:

- K_{nn} is the full covariance over the dataset X ;
- K_{mm} is the covariance among inducing points Z ;
- K_{nm} and K_{mn} are the cross-covariances between X and Z .

This construction preserves the exact marginal variances on the diagonal of K_{nn} , while approximating the off-diagonal correlations through the low-rank term Q_{nn} . The resulting posterior predictions are nearly identical to the full GP, but matrix inversion scales with m instead of n .

2.3.5 Heteroscedastic observation noise

Commonly the GP assumes homoscedastic noise, i.e. constant σ_n^2 [24]. However, in practice noise may depend on the input, $\sigma_n^2(x)$ [28]. Standard GPR then yields miscalibrated uncertainties. Kristian Kersting in [29] proposes a two GP step approach that uses a GP for the mean function $f(x)$ and another GP for the log-variance of the noise, trained jointly. These approaches improve the reliability

of predictive intervals, preventing overconfident exploration in low signal-to-noise regions. A detailed derivation of the heteroscedastic GP formulation, including the composite kernel structure and training of the log-variance process, is provided in Appendix B.

2.3.6 Why Gaussian Processes

Gaussian Process Regression offers a data-efficient surrogate with calibrated predictive uncertainty; its posterior adapts online as data arrive, while ML-II training balances fit and complexity [24]. In the literature, GPs support uncertainty-aware MPC and cautious control, and they can be used for energy measurement and efficiency modeling [17, 18]. Input dependent noise can be handled via heteroscedastic GP formulations to improve calibration [28, 29], and sparse/inducing-point methods provide practical speed–accuracy trade-offs for real time use [27]. Compared with neural networks which typically need large offline datasets, add-on machinery for trustworthy uncertainty, and heavier retraining GPs are better aligned with our low-data, online, safety-critical setting and expose smooth analytic derivatives compatible with SQP. Within this context, prior EV torque allocation studies assume full unit knowledge or rely on parametric loss models.

Gaussian Processes not only provide accurate predictions but also quantify uncertainty, making them ideal for guiding exploration. We now turn to Bayesian Optimization, which leverages this uncertainty to decide where the system should sample next.

2.4 Exploration via Bayesian Optimization

Bayesian Optimization (BO) addresses the problem of optimizing an unknown, expensive-to-evaluate function

$$\min_{x \in \mathcal{X}} f(x), \quad x \in \mathbb{R}^d, \quad (2.20)$$

by building a probabilistic surrogate of f from sequential measurements [19]. A common prior is a Gaussian Process, $f \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot))$. After observing data $\mathcal{D}_n = \{(x_i, y_i)\}_{i=1}^n$ with noisy evaluations $y_i = f(x_i) + \varepsilon_i$, the GP posterior provides, for any $x \in \mathcal{X}$, a predictive distribution

$$f(x) | \mathcal{D}_n \sim \mathcal{N}(\mu_n(x), \sigma_n^2(x)), \quad (2.21)$$

where μ_n and σ_n are the posterior mean and standard deviation, respectively. BO chooses the next query x_{n+1} by optimizing an *acquisition function* that trades off *exploitation* of current knowledge (low μ_n) and *exploration* of uncertain regions

(high σ_n) as can be seen in the Figure 3.7.

For minimization, a widely used acquisition is the Lower Confidence Bound (LCB), while for maximization it is commonly used UCB.

$$\text{LCB}_{\kappa,n}(x) = \mu_n(x) - \beta\sigma_n(x), \quad \beta > 0, \quad (2.22)$$

and the next evaluation is selected as

$$x_{n+1} \in \arg \min_{x \in \mathcal{X}} \text{LCB}_{\kappa,n}(x). \quad (2.23)$$

The parameter β controls the exploration–exploitation trade-off. As explained deeply by [15] and [14] larger values expand the confidence band and favor points with larger $\sigma_n(x)$, while smaller values follow the current mean $\mu_n(x)$. Equivalently, one may define a *negated-UCB* index $\mathcal{A}_{\kappa,n}(x) = -\mu_n(x) + \beta\sigma_n(x)$ and maximize it; both rules are identical up to sign. Under suitable regularity conditions and appropriate choices of κ_n , GP-UCB/LCB strategies admit sublinear cumulative regret bounds, formalizing the notion that uncertainty-guided exploration is sample efficient.

In many applications, not every $x \in \mathcal{X}$ is admissible, it is necessary then to introduce the Constrained Bayesian Optimization. Let a *feasible set*

$$\mathcal{F} = \{x \in \mathcal{X} : g_j(x) \leq 0, j = 1, \dots, J\} \quad (2.24)$$

encode hard constraints (e.g., bounds, rate limits, algebraic consistency). Constrained BO applies the acquisition rule on \mathcal{F} :

$$x_{n+1} \in \arg \min_{x \in \mathcal{F}} \text{LCB}_{\kappa,n}(x). \quad (2.25)$$

For acquisition functions such as LCB/UCB, the exploration term should be driven by the uncertainty coming from the model alone, so that the algorithm targets regions where improved knowledge is attainable rather than merely noisy regions.

Exploration identifies promising and informative torque splits, but during the proposed framework it will be necessary to exploit the learned knowledge. For this reason, we next review Quadratic and Sequential Quadratic Programming as the tools for computing feasible allocations.

2.5 Optimal Control Theory

Optimal control provides a framework for computing control actions that minimize a given cost function while satisfying system constraints. In this section, we introduce the theoretical foundations of the main techniques used in this work: Quadratic

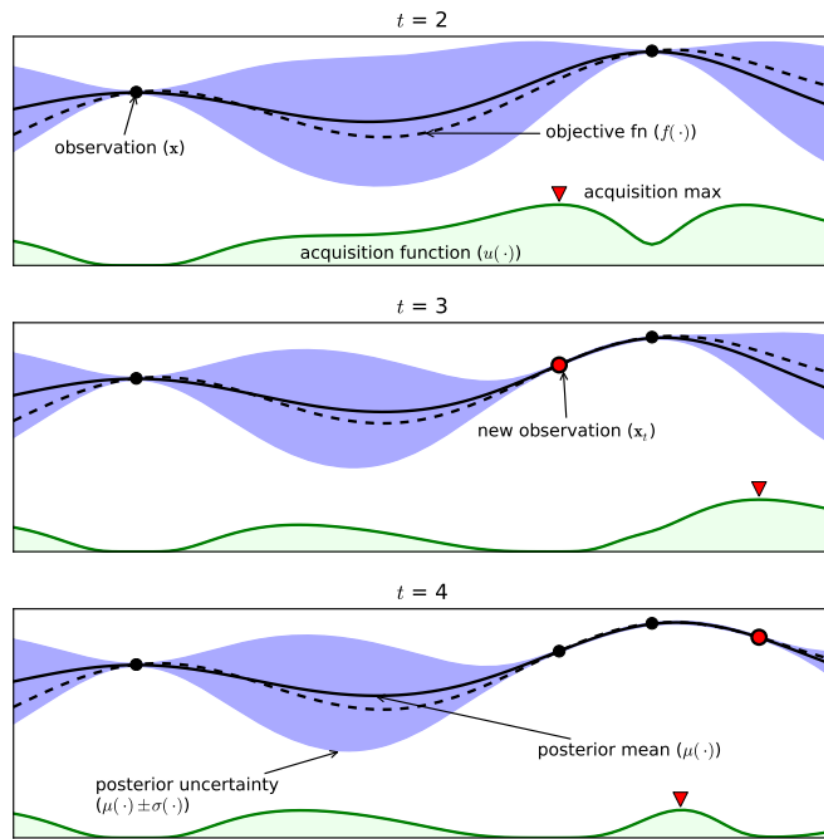


Figure 2.2: Bayesian optimization with a GP surrogate. Each panel ($t=2,3,4$) shows the posterior mean $\mu(x)$ (dashed) with uncertainty band $\mu(x) \pm \sigma(x)$ (blue), the unknown objective $f(x)$ (solid black), and an acquisition function $u(x)$ (green). The red triangle marks the maximizer of $u(x)$ (e.g. UCB). At $t = 2$ the acquisition peaks in a region that is promising and/or uncertain; querying there yields a new observation (red dot at $t = 3$), after which the GP is updated. Uncertainty shrinks around the new sample and the acquisition shifts ($t=4$), proposing the next informative query. Repeating this select–measure–update cycle drives evaluations toward the optimum while balancing exploration via $\sigma(x)$ and exploitation via $\mu(x)$. [30]

Programming, Sequential Quadratic Programming, Control Barrier Functions (CBF), and CBFs over a finite horizon.

2.5.1 Quadratic Programming

Quadratic Programming is one of the most widely used tools in control allocation and MPC. A general QP is defined as:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \frac{1}{2}x^\top Hx + g^\top x \\ \text{s.t.} \quad & Ax \leq b, \\ & Ex = d. \end{aligned} \tag{2.26}$$

where $H \in \mathbb{R}^{n \times n}$ is positive semidefinite, $g \in \mathbb{R}^n$ is a linear term, A, b represent inequality constraints, and E, d equality constraints. QP guarantees a globally optimal solution if $H \succeq 0$ and is computationally efficient, making it a standard choice in convex control allocation problems.

2.5.2 Sequential Quadratic Programming

Sequential Quadratic Programming is one of the most effective iterative methods for constrained nonlinear optimization. Following [31] the general problem it addresses is of the form

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{s.t.} \quad & g_i(x) \leq 0 \quad \text{for } i \in \mathcal{I}, \end{aligned} \tag{2.27}$$

where $f(x)$ is the nonlinear objective, and $g_i(x)$ represent equality and inequality constraints. The idea of SQP is to approximate this nonlinear problem locally by a QP subproblem, where the objective is modeled by a quadratic expansion and the constraints are linearized around the current iterate.

A crucial first step is the initialization. Following both the [32] and [31] theory, SQP requires a feasible starting point that satisfies all equality and inequality constraints. If such a point is not available, a Phase I problem can be formulated in which a slack variable γ is introduced to absorb constraint violations:

$$\begin{aligned} \min_{\gamma, x} \quad & \gamma \\ \text{s.t.} \quad & A_i x = b_i, \quad i = 1, \dots, m_e, \\ & A_i x - \gamma \leq b_i, \quad i = m_e + 1, \dots, m. \end{aligned} \tag{2.28}$$

Feasibility is recovered if the optimal solution satisfies $\gamma^* = 0$. This approach ensures that the algorithm can start even when the initial guess is infeasible.

Once an initial point is available, the core of SQP consists of repeatedly solving

a quadratic subproblem at iteration k . This subproblem is given by

$$\begin{aligned} \min_{d \in \mathbb{R}^n} \quad & q(d) = \frac{1}{2}d^\top H_k d + c_k^\top d \\ \text{s.t.} \quad & A_i d = b_i, \\ & A_i d \leq b_i \end{aligned} \tag{2.29}$$

where H_k is a quasi-Newton approximation to the Hessian of the Lagrangian, $c_k = \nabla f(x_k)$ is the gradient of the objective, and A_i denotes the Jacobian of the active constraints. This QP reflects the local model of the problem. Since active constraints must be respected, it is convenient to work in the null-space of A_k . By computing a matrix Z_k such that $A_k Z_k = 0$, the step can be expressed as $d = Z_k p$. Substituting into the quadratic model yields a reduced problem in terms of p :

$$q(p) = \frac{1}{2}p^\top Z_k^\top H_k Z_k p + (Z_k^\top c_k)^\top p. \tag{2.30}$$

Solving the reduced system

$$Z_k^\top H_k Z_k p = -Z_k^\top c_k \tag{2.31}$$

produces the search direction $\hat{d}_k = Z_k p^*$. This guarantees consistency with the active set of constraints.

The step length is then determined to ensure that no inequality constraints are violated. The maximum admissible α before hitting the next inactive constraint is

$$\alpha = \min_{i \notin \bar{A}_k, A_i \hat{d}_k > 0} \left\{ \frac{b_i - A_i x_k}{A_i \hat{d}_k} \right\}. \tag{2.32}$$

This constraint-aware update ensures that feasibility is maintained as the iterate moves.

Because nonlinear problems cannot be solved by quadratic approximations alone, SQP employs a line search based on a merit function. A common choice is

$$\Psi(x) = f(x) + \sum_{i \in \mathcal{E}} r_i |g_i(x)| + \sum_{j \in \mathcal{I}} r_j \max(0, g_j(x)), \tag{2.33}$$

where the penalty parameters r_i balance objective reduction with constraint satisfaction. They can be initialized as $r_i = \|\nabla f(x)\|/\|\nabla g_i(x)\|$ and updated using Powell's rule to maintain convergence guarantees. The step is accepted if the merit function decreases sufficiently:

$$\Psi(x_k + \alpha \hat{d}_k) < \Psi(x_k) - \gamma \alpha \cdot \text{sufficient decrease}. \tag{2.34}$$

At every iteration, optimality is assessed by checking the Karush–Kuhn–Tucker

(KKT) conditions. If the updated Lagrange multipliers satisfy

$$A_k^\top \lambda_k = c_k + H_k x_k, \quad (2.35)$$

and all inequality multipliers are nonnegative, the algorithm declares convergence. Otherwise, violated constraints are removed or added to the active set, and the process continues.

The curvature of the problem is incorporated by updating the Hessian approximation. The BFGS update is widely used:

$$H_{k+1} = H_k + \frac{q_k q_k^\top}{q_k^\top s_k} - \frac{H_k s_k s_k^\top H_k}{s_k^\top H_k s_k}, \quad (2.36)$$

where $s_k = x_{k+1} - x_k$ and $q_k = \nabla \mathcal{L}(x_{k+1}) - \nabla \mathcal{L}(x_k)$. This ensures that the positive-definiteness of H_k is preserved whenever possible, providing reliable curvature information for the quadratic model.

In summary, each iteration of SQP consists of solving a quadratic subproblem, computing a feasible step, applying a merit-function line search, checking KKT optimality, and updating both the active set and the Hessian approximation. The method terminates when the step size, constraint violations, and KKT residuals fall below tolerance. Due to this combination of second-order accuracy, constraint handling, and systematic convergence checks, SQP is considered one of the most robust algorithms for nonlinear constrained optimization.

While optimization ensures efficiency under constraints, it does not by itself always guarantee vehicle safety. To address this, we introduce Control Barrier Functions, which enforce safety by filtering or constraining torque allocations.

2.5.3 Control Barrier Functions: Theory and Safety Enforcement

Control Barrier Functions provide a formal framework to guarantee safety in nonlinear dynamical systems subject to state constraints. Literature like [20] and [21] to explain CBF consider the control-affine system:

$$\dot{x} = f(x) + g(x)u, \quad (2.37)$$

with state $x \in \mathbb{R}^n$ and control input $u \in \mathbb{R}^m$. Safety is encoded by defining a *safe set*

$$\mathcal{C} = \{x \in \mathbb{R}^n \mid h(x) \geq 0\}, \quad (2.38)$$

where $h : \mathbb{R}^n \rightarrow \mathbb{R}$ is a continuously differentiable function. Forward invariance of \mathcal{C} means that if $x(0) \in \mathcal{C}$, then $x(t) \in \mathcal{C}$ for all $t \geq 0$.

A control barrier function condition acts as a filter modifying the control only when the system risks leaving the safe set enforcing invariance by requiring that

along system trajectories

$$\dot{h}(x, u) + \alpha h(x) \geq 0, \quad \alpha > 0, \quad (2.39)$$

where $\dot{h}(x, u) = \nabla h(x)^\top (f(x) + g(x)u)$ is the Lie derivative of h along the system dynamics. This condition ensures that the system trajectory cannot cross the boundary $h(x) = 0$ into the unsafe set $h(x) < 0$. CBFs act as a filter: they only modify the control when the system risks leaving the safe set

The inequality is affine in the control input u , which allows it to be embedded directly as a linear constraint in a quadratic program. Thus, if the nominal control is obtained from a cost-minimizing controller, the CBF constraint acts as a filter, adjusting the control only when necessary to preserve safety:

$$\begin{aligned} \min_{u \in \mathbb{R}^m} \quad & \frac{1}{2} \|u - u_{\text{nom}}\|^2 \\ \text{s.t.} \quad & \nabla h(x)^\top (f(x) + g(x)u) + \alpha h(x) \geq 0. \end{aligned} \quad (2.40)$$

Here u_{nom} is the nominal input, and the CBF constraint guarantees that the chosen input maintains forward invariance of the safe set. In this way, CBFs integrate seamlessly with optimization-based controllers.

Different variants exist in the literature, such as *zeroing CBFs* (ZCBFs), which enforce strict invariance at the boundary, and *exponential CBFs* (ECBFs), which generalize the condition for higher relative degree systems.

2.5.4 Control Barrier Functions for Non Affine Systems

Classical CBF theory is typically developed for control-affine dynamics. However [22] provide a wider formulation considering nonlinear systems that evolve according to more general dynamics:

$$\dot{x} = f(x, u), \quad (2.41)$$

where the control input u appears nonlinearly. In this case the derivative of the barrier function is

$$\dot{h}(x, u) = \nabla h(x)^\top f(x, u), \quad (2.42)$$

so the safety condition

$$\nabla h(x)^\top f(x, u) + \alpha(h(x)) \geq 0 \quad (2.43)$$

is no longer affine in u . As a consequence, the resulting constraint is generally nonlinear, and quadratic programming is not sufficient to enforce it. Instead, one must resort to nonlinear programming or SQP approaches, where the constraint (2.43) is linearized at each iteration.

In digital control applications the system is usually modeled in discrete time [23],

$$x_{k+1} = F(x_k, u_k), \quad (2.44)$$

and forward invariance of the safe set $\mathcal{C} = \{x \mid h(x) \geq 0\}$ is enforced through the discrete-time CBF condition

$$h(F(x_k, u_k)) - h(x_k) + \alpha_d(h(x_k)) \geq 0, \quad (2.45)$$

with α_d a discrete class- \mathcal{K} function (e.g. $\alpha_d(s) = \kappa s$, $0 < \kappa < 1$). Constraint (2.45) guarantees that if $x_k \in \mathcal{C}$, then x_{k+1} remains in \mathcal{C} , thus preserving safety. Since (2.45) is typically nonlinear in u_k , it is enforced via nonlinear or SQP-based optimization rather than a simple QP.

This extension makes CBFs applicable to a wider class of nonlinear systems, while retaining their core role of certifying forward invariance of safety sets in real-time control.

2.5.5 Control Barrier Functions over a finite Horizon

Classical CBFs enforce safety instantaneously, but many applications require guarantees over a finite horizon. [21] extend the CBF concept by embedding barrier conditions into a finite-horizon problem,

Given the discrete dynamics $x_{k+1} = F(x_k, u_k)$ and the safe set $\mathcal{C} = \{x \mid h(x) \geq 0\}$, over a horizon of N_s steps, predict states x_{k+j} using $x_{k+0} = x_k$ and the dynamics

$$x_{k+j+1} = F(x_{k+j}, u_{k+j}), \quad j = 0, \dots, N_s - 1.$$

Safety is enforced by requiring the one-step CBF inequality at each predicted step:

$$h(x_{k+j+1}) - h(x_{k+j}) + \alpha_d(h(x_{k+j})) \geq 0, \quad j = 0, \dots, N_s - 1. \quad (2.46)$$

A practical finite-horizon optimization (solved as an NLP/SQP when F is non-affine in u) is:

$$\begin{aligned} \min_{\{u_{k+j}\}, \{\delta_j\} \geq 0} & \sum_{j=0}^{N_s-1} \frac{1}{2} \|u_{k+j} - u_{\text{nom},k+j}\|_R^2 \\ \text{s.t.} & x_{k+0} = x_k, \\ & x_{k+j+1} = F(x_{k+j}, u_{k+j}), \\ & h(x_{k+j+1}) - h(x_{k+j}) + \alpha_d(h(x_{k+j})) \geq 0, \\ & j = 0, \dots, N_s - 1 \end{aligned} \quad (2.47)$$

At runtime, apply only the first control u_k^* and repeat at $k+1$ (receding horizon).

Chapter 3

Methodology

This work presents a modular and structured framework designed to enable safe online learning and optimal torque allocation in AEV systems, specifically in scenarios involving a truck connected to an electric trailer with unknown powertrain characteristics. The key challenge lies in achieving real-time coordination without prior knowledge of the trailer's internal behavior. To address this, the framework continuously learns the trailer's power loss surface during operation and adapts the torque allocation strategy based on observed data, while ensuring safety at all times. A simple overview of the full methodology is illustrated in Figure 3.1, which highlights the interaction among the key components of the system.

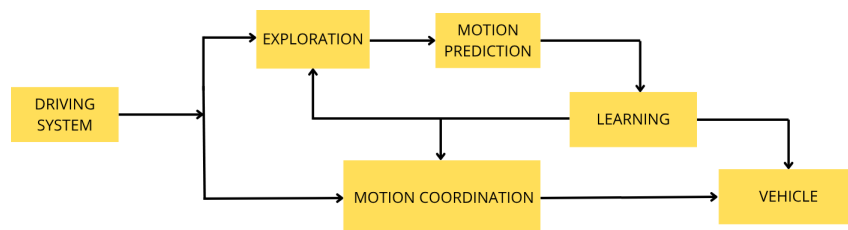


Figure 3.1: Schematic representation of the proposed methodology, showing the interaction among the four main modules: (1) Exploration, (2) Motion Prediction, (3) Learning, and (4) Motion Coordination.

The proposed framework is structured into four main functional blocks between the driving system request and the vehicle response :

1. Exploration: Drives the data collection strategy to expand knowledge of trailer behavior.
2. Motion Prediction (Safety Testing): Simulates short-term future trajectories to assess the safety of planned actions.

3. Learning: Builds and updates a model of the trailer's unknown power loss surface.
4. Motion Coordination (Exploitation): Computes optimal torque commands based on the learned characteristics and safety predictions.

Following the principles of Reinforcement Learning [33], this work employs an ε -greedy inspired strategy to balance the exploration and exploitation during online learning. A time-decaying probability function determines whether the system performs:

- Exploration: Testing alternative torque splits to learn the power loss model.
- Exploitation: Using the current learned model to minimize power loss safely.

To ensure sufficient model learning, particularly during the early stages, an intensive exploration phase is enforced. Simulation results indicated that at least 20 minutes of high exploration activity are required to achieve meaningful coverage of the torque–velocity space. After this period, the exploration rate ε begins to decay over time according to the following exponential schedule:

$$\varepsilon(t) = \varepsilon_{\min} + (\varepsilon_{\max} - \varepsilon_{\min}) \cdot e^{-\lambda t} \quad (3.1)$$

where, $\lambda = 0.001$, $\varepsilon_{\max} = 0.9$, and $\varepsilon_{\min} = 0.1$. At each control cycle, a random number $\zeta \in [0, 1]$ is drawn. If $\zeta < \varepsilon(t)$, the system performs exploration, otherwise, it exploits the current learned model.

Failure to maintain a sufficiently high ε early in training may result in inadequate model generalization and poor coverage of the operating space. This can lead to inaccurate power loss predictions and suboptimal control allocations. The temporal evolution of $\varepsilon(t)$ and its impact on decision-making is visualized in Figure 3.2.

To allow the development and evaluation of the proposed solution, a number of working assumptions have been made:

1. Sensor-Based Power Loss Measurements: In a real-world setting, power loss values are obtained directly from physical sensors embedded in the trailer's drivetrain system. In the simulation environment used for this study, these values are generated using proprietary software provided by Volvo, simulating different motor types under varying conditions.
2. Torque Estimation via Soft Sensors: In practice, trailer torque cannot be directly measured and is instead estimated through soft-sensor approaches. A full torque estimation model is beyond the scope of this thesis; torque is therefore treated as a standard sensor measurement.

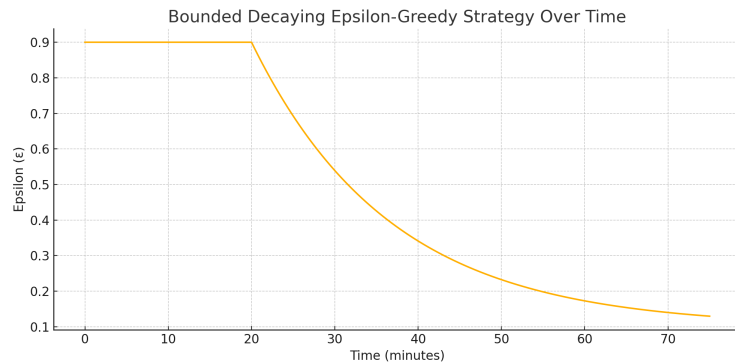


Figure 3.2: Temporal evolution of (t) . During the initial 20 minutes, the exploration rate was held at a high constant value to support model learning before beginning to decay.

3. **CAN Communication Protocol:** The trailer is assumed to support communication via a Controller Area Network (CAN) interface. It can transmit either precomputed power loss values from its onboard controller, or raw sensor data (torque, speed, current, voltage), which the truck's central unit processes.
4. **Electric Motor Types:** The e-trailer platform considered in this work is assumed to be equipped with either a Permanent Magnet Synchronous Motor (PMSM) or an Induction Motor (IM). Both types are supported in the simulation setup.
5. **Known Trailer Parameters:** For the purpose of model learning and safety evaluation, the mass and length of the trailer are assumed to be known and fixed during each test scenario.

In the following sections, each module of the framework is described in detail. Their interaction ultimately enables safe and adaptive control of the vehicle system under uncertainty.

3.1 Learning of the Power Loss Surface

The objective of this section is to approximate the power loss surface, which is defined as a nonlinear function of torque and angular velocity (ω). Accurate estimation of this surface is critical for modeling and optimizing the performance of the powertrain system. During each iteration of the drive cycle, a measurement of the instantaneous power loss is obtained for a specific combination of torque and angular velocity required by the system. These measurements are sequentially collected, forming a dataset of operating points that is used to characterize the system's power

loss behavior over the entire operational limits of torque and ω .

To reconstruct the power loss surface from this data, two approaches were explored:

- **System Identification Approach:** A parametric method using recursive least squares (RLS).
- **Machine Learning Approach:** A data-driven method using Gaussian Process Regression (GPR).

A detailed discussion is provided in the subsequent sections; however, for clarity, a summary of the two approaches is presented in Table 3.1.

Table 3.1: Comparison of the main analyzed characteristics between System Identification and Machine Learning Approaches

Feature	RLS	GPR
Model Type	Parametric	Non-parametric
Uncertainty Estimation	Not available	Provides uncertainty
Scalability	low	high
Initial Requirement	None	Initialization necessary
Optimization	Suited for QP	Requires SQP (non-convex)
Update Frequency	Every control step	Every N steps (30 samples)
Interpretability	High — interpretable coefficients	Moderate

3.1.1 System Identification Approach via Recursive Least Squares

Under the assumption that drivetrain power losses exhibit quadratic dependence on torque, the system identification approach models these losses as a parametric surface fitted to measured torque–speed–loss data. Building on earlier Volvo work employing Ordinary Least Squares (OLS), the method adopts RLS to enable real-time, online parameter estimation. The loss model takes the form:

$$P_{\text{loss}}(T, \omega) = a(\omega)T^2 + b(\omega)T + c(\omega).$$

where the coefficients depend on the angular velocity and on the operating mode of the machine (acting as a motor for positive torques and as a generator for negative torques).

In Volvo’s previous implementation [8], OLS was used for parameter fitting providing a closed-form solution:

$$\theta = (\Phi^T \Phi)^{-1} \Phi^T y$$

However, this approach suffers from several drawbacks. First, it becomes computationally expensive and memory-intensive in an online setting, as new data

continuously accumulates and OLS does not naturally support incremental updates. Second, earlier work relied on heavily tuned piecewise polynomial fits for each parameter a, b, c over different angular velocity ranges. While this yielded high accuracy, it resulted in poor scalability and limited generalization, an unacceptable limitation when dealing with trailers whose dynamics are unknown.

To address the limitations of OLS, this work adopts the Recursive Least Squares algorithm, which allows efficient online updates as new data arrives. The RLS algorithm minimizes a regularized, exponentially weighted squared error:

$$\min_{\theta_k} \sum_{i=1}^k \lambda^{k-i} (y_i - \phi_i^\top \theta_k)^2$$

where θ_k is the estimated parameter vector at time step k , y_i is the observed power loss, ϕ_i is the input feature vector at time i and $\lambda \in (0, 1]$ is the forgetting factor, which determines how fast the algorithm forgets older data.

To provide more flexible fitting while maintaining real-time feasibility, two model structures were tested, starting with a linear parameter variation with respect to ω , that results in a polynomial with 6 coefficients θ :

$$a(\omega) = \theta_1 \omega + \theta_2$$

$$b(\omega) = \theta_3 \omega + \theta_4$$

$$c(\omega) = \theta_5 \omega + \theta_6$$

Then a quadratic parameter variation with respect to ω with 9 coefficients θ :

$$a(\omega) = \theta_1 \omega^2 + \theta_2 \omega + \theta_3$$

$$b(\omega) = \theta_4 \omega^2 + \theta_5 \omega + \theta_6$$

$$c(\omega) = \theta_7 \omega^2 + \theta_8 \omega + \theta_9$$

These structures offer a trade-off between simplicity and fitting flexibility, without relying on any prior knowledge of motor-specific behavior, maintaining compatibility with real-time learning constraints.

3.1.2 Gaussian Process Learning

While polynomial models may perform adequately for specific motor types, their accuracy can degrade under non-polynomial behaviors, such as component aging or in the presence of significant measurement noise. To achieve better scalability and robustness, this work adopts a non-parametric machine learning approach based on GPR. In contrast to models such as neural networks, which require extensive

offline training, GPR offers strong generalization capabilities with limited data and inherently provides uncertainty estimates, which is particularly valuable for modeling trailer power losses under sensor noise and model uncertainty.

The trailer power loss is modeled as a Gaussian Process over the input vector

$$x = [T \quad \omega]^\top,$$

with prior

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')),$$

where $m(x)$ is the mean function, initialized either to zero or to a low-degree polynomial, and $k(x, x')$ is the selected kernel that capture spatial correlations in the torque–velocity domain. Options explored include Squared Exponential and Polynomial kernels, tested in both isotropic (ISO) and automatic relevance determination (ARD) forms (further details in the results section 5.2.1.1).

Given n training samples $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, the joint distribution of the training outputs \mathbf{y} and a test output f_* at x_* is

$$\begin{bmatrix} \mathbf{y} \\ f_* \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} m(\mathbf{X}) \\ m(x_*) \end{bmatrix}, \begin{bmatrix} K(\mathbf{X}, \mathbf{X}) + \Sigma_n & K(\mathbf{X}, x_*) \\ K(x_*, \mathbf{X}) & K(x_*, x_*) \end{bmatrix} \right),$$

where $K(\cdot, \cdot)$ is the kernel matrix and $\Sigma_n = \text{diag}(\sigma_n^2(x_1), \dots, \sigma_n^2(x_n))$ is the diagonal matrix that contains the noise coming from the sensors measurements.

The kernel function is a core component of any Gaussian Process and picking the right one involves some trade-off considerations between the model complexity, since more complex kernels can overfit the data. Ultimately, the selection of the kernel was guided by accuracy validation and by the necessity of the derivability requested by the control optimization loop. In the final model, a squared exponential ISO kernel was selected due to its smoothness, differentiability and high accuracy results over different simulations.

When considering inference, two approaches were analyzed. Full GP (Exact Inference) is used when the dataset is below 5000-10000 points [34]. Provides high accuracy but incurs a computational cost of $\mathcal{O}(N^3)$ due to full covariance matrix inversion, making it unsuitable for real-time applications once the dataset grows beyond a few thousand points. Although the dataset used in each simulation typically ranges from 500 to 1000 points, early testing showed that even at the lower end of this range, full GP inference introduced latency during online control steps. To reduce computational time while preserving accuracy, a Sparse GP (FITC Approximation) was implemented using the Fully Independent Training Conditional (FITC) method was adopted using the GPML toolbox [35]. This approach reduces the computational complexity to $\mathcal{O}(nm^2)$, where n is the number of training points and $m \ll n$ is the number of inducing points (the different strategy adopted for the

choice of the inducing point are presented in the Appendix C). While the sparse approach demonstrated substantial improvements in computational efficiency, it suffered a degradation in accuracy. This reduction in performance was primarily due to the uneven distribution of training data, which impaired the sparse GP's ability to generalize across the input space. As a result, this work adopts the full GP formulation despite its higher computational cost, as it provides superior predictive fidelity and robustness, particularly important during the early stages of learning.

In an online setting, balancing model accuracy with computational efficiency is essential. The Gaussian Process model is updated periodically every 30 control steps (reaching at maximum 3 seconds when the dataset is large). During each update, the hyperparameters length scale and signal variance are re-optimized by maximizing the marginal likelihood. This allows the model to adapt to dynamic trailer behavior and changing sensor noise characteristics.

To accelerate convergence, initial hyperparameter values are heuristically set based on:

- **Length Scale:** Median of pairwise distances between initial training points,
- **Signal Variance:** Scaled to the empirical variance of power loss values,
- **Noise Variance:** Derived from sensor specifications and is updated using the two step approach.

To balance model accuracy with real-time feasibility, the GP is updated periodically every 30 control steps (0.7–3.0 seconds). Between updates, the most recently trained model is held fixed and used for inference in real time. This latency is acceptable within the 3-second update window and does not affect control performance, since inference between updates uses the most recent model.

Figure 3.3 shows the nominal evolution of the log length scale and log signal variance under good initialization, showing how the model progressively adapts as new observations are incorporated. A contrasting poor-initialization case is analyzed in the Results.

The modeling of electric trailer power loss under real-world conditions required accounting not only for the unknown system behavior (epistemic uncertainty) but also for the non-uniform, input-dependent measurement noise (aleatoric uncertainty) [36]. To address this, a two-step heteroscedastic Gaussian Process framework [29] was implemented.

The first GP models the latent power loss function $f(T, \omega)$. Observations follow the model:

$$y_i = f(x_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma_n^2(x_i)) \quad (3.2)$$

Here, $\sigma_n^2(x)$ captures the heteroscedastic (input-dependent) noise variance. This is incorporated into the GP via a composite kernel $k_{\text{obs}}(x_i, x_j)$.

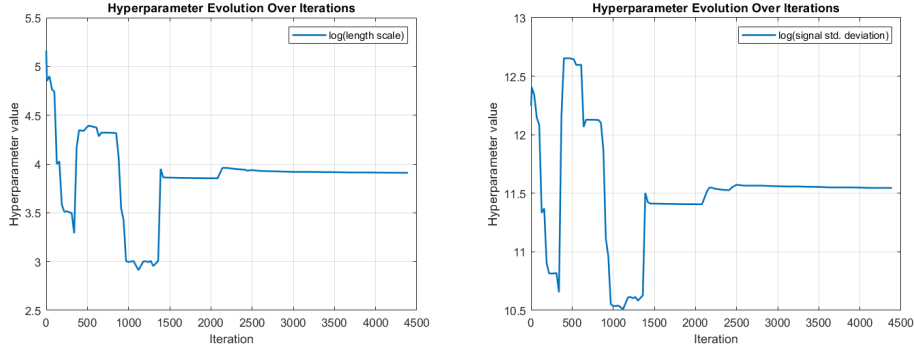


Figure 3.3: Temporal evolution of the Gaussian Process hyperparameters (length-scale and signal variance) during online learning. The plot illustrates how these parameters evolve as new trailer observations are incorporated.

In practice, this means that measurement noise is not treated as constant, and for this reason the likelihood noise is set to zero during training.

To estimate the input-dependent noise field $\sigma_n^2(x)$, a second GP is trained on the log-variance values:

$$z_i = \log \sigma_n^2(x_i) \quad (3.3)$$

$$z(x) \sim \mathcal{GP}(\mu_{\log}(x), k_{\log}(x, x')) \quad (3.4)$$

This allows the prediction of the aleatoric variance at any test input x_* as:

$$\hat{\sigma}_n^2(x_*) = \exp(\mu_{\log}(x_*)) \quad (3.5)$$

Finally, the full predictive distribution combines both uncertainty components:

$$y(x_*) \sim \mathcal{N} \left(\mu_f(x_*), \underbrace{\sigma_f^2(x_*)}_{\text{epistemic}} + \underbrace{\hat{\sigma}_n^2(x_*)}_{\text{aleatoric}} \right) \quad (3.6)$$

This decomposition enables safer and more robust decision-making in control and exploration, as it provides a calibrated estimate of both uncertainty from the model and from sensor noise as can be seen in the Figure 3.4. The full formulation for the heteroscedastic noise is presented in the Appendix B

3.1.3 Prior Knowledge and Normalization Strategy

In real-world applications, the absence of prior knowledge about the trailer's power loss behavior poses a significant challenge for data-driven learning. To address this, the Gaussian Process model is initialized using synthetic prior data derived from the truck's known power loss surface. This strategy provides a structured

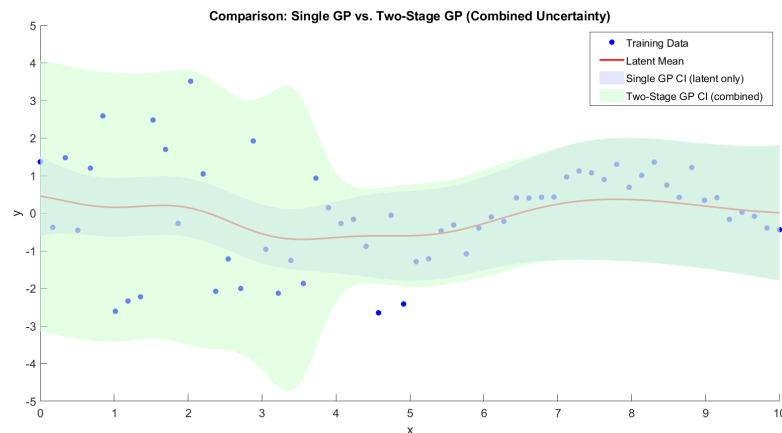


Figure 3.4: Comparison between standard single-GP regression and a two-stage heteroscedastic GP model in terms of uncertainty quantification. The red line represents the mean function predicted by both models. The blue dots indicate noisy training data. The shaded regions show the $\pm 2\sigma$ confidence intervals: the light blue region corresponds to the epistemic uncertainty from a single GP model, while the light green region reflects the combined uncertainty from a two-stage GP, where aleatoric noise is explicitly modeled as input-dependent via a second GP. Notably, around input regions with highly variable or sparse observations, the two-stage GP captures increased uncertainty due to both model uncertainty and measurement noise. This highlights the ability of the two-stage GP framework represent both epistemic (model) and aleatoric (data) uncertainty more effectively.

starting point for learning, especially in the early stages when real data from the trailer is sparse or noisy. The prior consists of 300 training points sampled from the truck's loss surface and scaled to match the trailer's torque and velocity limits. These points are uniformly distributed across the (T, ω) space, with the chosen number ensuring broad initial coverage. This initialization improves early model performance and accelerates convergence, while avoiding the need for precomputed trailer maps. A potential drawback of this initialization is that the truck's loss surface may not accurately reflect the trailer's characteristics, introducing bias in early predictions. This risk is mitigated by the proposed forgetting strategies, particularly heteroscedastic weighting, which gradually down-weights prior samples as more trailer-specific data becomes available. To enable the online adaptation and reduce over-reliance on the prior, two complementary forgetting strategies were tested:

- **Rolling Window Removal:** Every 35 iterations, two prior samples are removed from the training set. This gradually shifts the model's focus toward newly observed trailer data. However, this method can introduce instability if regions of the input space remain unexplored, as uncertainty may spike once all prior support is removed.

- **Heteroscedastic Weighting:** Prior samples are assigned artificially high noise variances during the GP update. This reduces their influence on the posterior prediction, allowing high-confidence measurements from the trailer to dominate the regression. Unlike hard removal, this soft-forgetting mechanism maintains stability while supporting flexible adaptation.

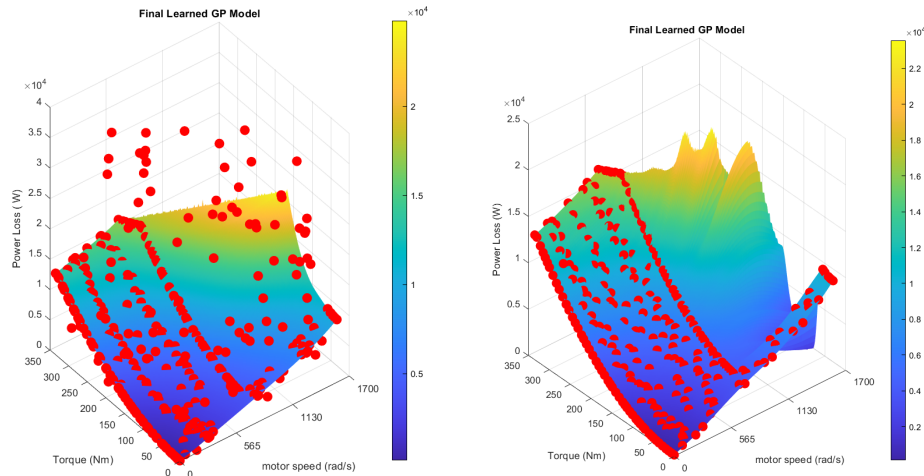


Figure 3.5: Visualization of the prior knowledge integration strategies used to initialize and adapt the Gaussian Process model.

Left: Heteroscedastic weighting assigns higher noise variance to prior samples, reducing their influence during model updates.

Right: Rolling window removal gradually deletes prior points every 35 iterations, allowing the model to shift focus to trailer-specific observations.

The heteroscedastic weighting provides a smooth transition from truck-based initialization to trailer-specific learning, enabling robust online adaptation to the new data, the main advantage compared to the rolling window is shown in the Figure 3.5.

3.2 Exploration Strategies

To effectively learn the trailer’s power loss characteristics, the system must actively explore the space of torque and speed commands (T, ω) . Exploration is critical for model generalization, especially in the early learning phase. Two alternative strategies were employed, depending on the learning model used:

- **Grid-Based Heuristic:** applied when using the parametric RLS learner, which does not provide uncertainty information. The grid ensures uniform coverage of the (T, ω) map and avoids overfitting to specific regions.

- **Uncertainty-Based Exploration (UCB):** applied when using the Gaussian Process learner, which supplies predictive variance. This allows actions to be chosen where uncertainty is high, following the principles of Bayesian Optimization.

These strategies are not combined sequentially; rather, they serve as model-specific alternatives. Grid-based exploration guarantees coverage for RLS, while UCB provides efficient uncertainty-driven exploration for GP. Both aim to improve learning accuracy while maintaining safety, when combined with motion prediction, but differ in their selection criteria and sensitivity to data distribution.

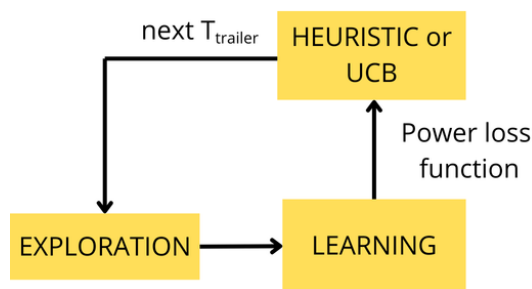


Figure 3.6: Schematic of the proposed exploration part of the framework. The Exploration block generates candidate torque split values. The selection of the next trailer torque command is then performed using either the grid-based heuristic or an Upper Confidence Bound criterion.

3.2.1 Grid-Based Heuristic Exploration

The torque velocity space (T, ω) is discretized into a 2D grid. Each cell tracks how often it has been visited. At each iteration, the system generates candidate torque splits by perturbing the previous one and selects the one landing in the least-visited grid cell. This ensures even coverage and avoids overfitting to specific operating points.

While naive random exploration can cause sharp transitions and inefficient coverage, the grid-based method improves smoothness and mechanical reliability. A trade-off exists between grid resolution and real-time feasibility: finer grids improve learning accuracy but increase computational and memory demands.

3.2.2 Uncertainty-Based Exploration

An uncertainty-driven strategy was employed using the mean and the predictive variance of the GP model. This approach is inspired by Bayesian Optimization and uses an Upper Confidence Bound acquisition function [15]. Here β is a tunable hyperparameter that sets the balance between exploration and exploitation.

To ensure compatibility with physical constraints particularly torque limits a constrained variant of UCB (C-UCB) was implemented. The optimization is performed under the following constraints:

$$\max_{T_2} \mu(T_2) + \beta \cdot \sigma(T_2)$$

subject to:

$$T_{\min} \leq T_2 \leq T_{\max}$$

$$|T_2 - T_{2,\text{prev}}| \leq \Delta T_{\max}$$

$$T_1 = \frac{rF_{\text{req}} - gr_2T_2}{gr_1}$$

$$T_{\min} \leq T_1 \leq T_{\max}$$

This ensures the selected trailer torque T_2 leads to a valid and safe truck torque T_1 , while also promoting exploration in uncertain regions. The constraint on the rate of change in T_2 further guarantees smoothness and prevents actuator stress due to abrupt transitions.

Candidate torques are first ranked using the UCB acquisition function. After this the highest-ranked proposals are then subjected to rollout-based safety checks, ensuring that only dynamically feasible and safe candidates are forwarded to the learning module.

3.3 Motion Prediction

The solution of the exploration yields a set of candidate torque splits. This diversity of candidates is crucial: not all splits will be dynamically feasible or safe, and invalid proposals must be discarded in favor of safe alternatives. While prior exploration ensured compliance with motor limits and force constraints, it is also necessary to evaluate the dynamic safety of each proposed torque split over a short time horizon, necessary for the learning update. Specifically, a torque split must not induce unsafe behaviors such as jackknifing or trailer swing during execution.

3.3.1 Dynamical Model Description

To perform this safety assessment, a predictive vehicle model is employed. The model is based on a single-track (bicycle) representation of a truck-trailer system, selected for its balance of simplicity and fidelity. This model, derived from established vehicle dynamics literature [37], captures the key nonlinear behaviors of articulated vehicles, including tire slip dynamics, and consists of 24 coupled equations.

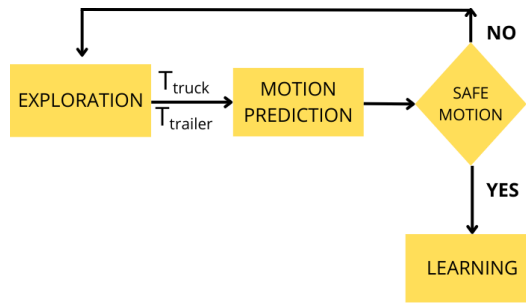


Figure 3.7: Schematic of the motion prediction stage within the torque split optimization framework. Candidate torque splits generated by the exploration module are evaluated in the motion prediction block over a short prediction horizon to assess dynamic safety. The safe motion check rejects torque splits that could lead to unsafe behaviors such as jackknifing or trailer swing. Only candidates passing this safety filter are forwarded to the learning module, ensuring that updates are based solely on dynamically feasible and safe operating points.

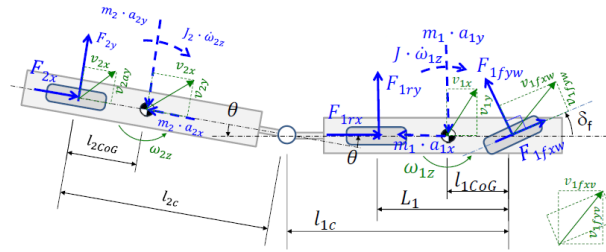


Figure 3.8: Dynamic model of the single track truck and trailer combination used for safety assessment. [37]

The full system is implemented in Simulink as a black-box FMU block, which receives as input:

- The torque applied to the trailer (single axle),
- The torque applied to the truck's rear axle (the only driven axle),
- The steering angle for the truck's front axle, which is provided by a predefined drive cycle.

A schematic of the simulation setup is shown in Figure 3.9 while the full mathematical description of the model, including state definitions, dynamic equations, and modeling assumptions, is provided in Appendix D.

3.3.2 Prediction Method

The framework employs two distinct mechanisms for safety assurance, depending on the control layer. During the exploration phase, candidate torque splits generated by

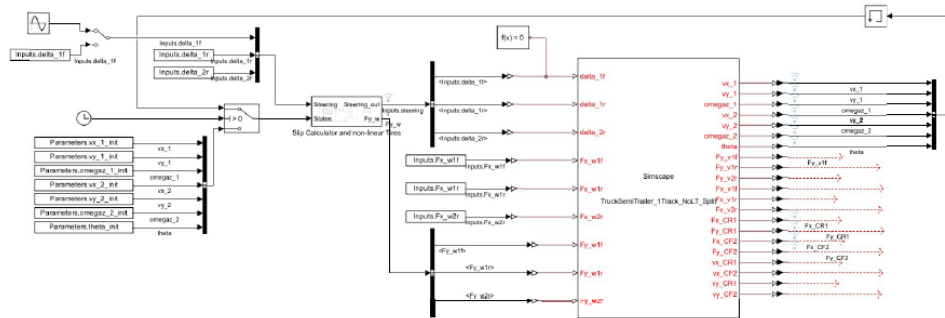


Figure 3.9: Implementation of the truck–trailer motion prediction in Simulink using a black-box FMU block. The FMU receives as inputs the trailer axle torque, the truck’s driven axle torque, and a prescribed steering angle profile from the drive cycle.

the C-UCB module are screened using rollout-based motion prediction. This choice is motivated by the need to evaluate not only the immediate control action, but also its multi-second consequences under the nonlinear articulated vehicle dynamics. Rollout simulation integrates the model forward in time from the current state, enabling the detection of gradual safety-critical evolutions that may not be apparent from instantaneous state checks. A purely constraint-based approach such as the control barrier function was not adopted here, as exploration requires rapidly testing a large set of hypothetical actions without the additional complexity of embedding them into an optimization loop. Rollouts thus provide a computationally direct and model-faithful method for filtering unsafe exploratory actions.

In contrast, in the optimal control phase, safety is enforced via a control barrier function (CBF) embedded directly into the sequential quadratic programming (SQP) formulation. Here, the focus is not on screening a wide range of candidate actions, but on solving for a single optimal torque allocation subject to hard safety guarantees.

Initial tests using the high-fidelity Simulink vehicle model showed that simulating a 3-second horizon via rollout required nearly 1 s of computation time, which is unsuitable for real-time use. To address this, an in-house symbolic implementation of the same model was adopted, based on prior work at Volvo[38]. This implementation reformulates the system into an implicit ODE structure and solves it using MATLAB’s `ode15i` solver. The resulting prediction is returned as a discretized state trajectory over the simulation horizon, reducing computation time to under 0.01 s per trajectory and making it feasible for iterative rollout safety screening.

3.3.3 Unsafe Behavior Detection Criteria

Direct derivation of dynamic safety boundaries for articulated trucks is a complex and application-specific task. Therefore, this work adopts criteria previously validated in an thesis done at Volvo [39]. The critical behaviors considered are jackknifing and trailer swing, with thresholds determined through extensive simulation and the construction of a safety envelope defining the safe operating region.

These behaviors are detected using velocity-based angular ratios α_{12} and α_{22} , defined as:

$$\alpha_{12} = \left| \frac{v_{y,12}}{v_{x,12}} \right|, \quad \alpha_{22} = \left| \frac{v_{y,22}}{v_{x,22}} \right| \quad (3.7)$$

In brief, α_{12} represents the absolute lateral-to-longitudinal velocity ratio at the truck's rear axle, and α_{22} at the trailer axle. Values of $\alpha_{12} > 0.8$ with $\alpha_{22} < 0.4$ and $|\theta| > 60^\circ$ correspond to high articulation risk (incipient jackknifing), while $\alpha_{12} < 0.4$ with $\alpha_{22} > 0.8$ and $|\theta| \geq 30^\circ$ corresponds to trailer swing onset. The detection conditions are summarized in Table 3.2.

Table 3.2: Criteria for detecting unsafe behavior.

Behavior	Condition	Angle Threshold
Jackknifing	$\alpha_{12} > 0.8, \alpha_{22} < 0.4$	$ \theta > 60^\circ$
Trailer Swing	$\alpha_{12} < 0.4, \alpha_{22} > 0.8$	$ \theta \geq 30^\circ$

If any proposed torque split leads to a trajectory where all three conditions for either jackknifing or trailer swing are met during the horizon, the split is deemed unsafe and rejected. The C-UCB algorithm then proposes the next candidate for evaluation.

To implement the safe exploration strategy described above, Algorithm 1 summarizes the step-by-step procedure used to select torque splits, evaluate their feasibility, and update the learning model.

3.4 Motion Coordination

The objective of the motion coordination module aims to minimize the total power losses of the AEV, leveraging the learned knowledge of the trailer's loss surface $P_{\text{loss,trailer}}$. At each time step, given the current wheel speed ω and longitudinal force demand F_{req} , the optimization seeks an instantaneous torque allocation:

$$\mathbf{u}^* = \begin{bmatrix} T_{\text{truck}} \\ T_{\text{trailer}} \end{bmatrix}$$

Algorithm 1 Pseudocode for Safe-Exploration Learning

- 1: GeorgiaPro-BoldInput: Current state x , force request F_{req} and requested ω
 - 2: GeorgiaPro-BoldOutput: Safe informative torque split $T_{\text{truck}}, T_{\text{trailer}}$
 - 3: **while** Exploration **do**
 - 4: Given the system's required F_{req} for a given ω
 - 5: Compute all possible $T_{\text{trailer},i}$ such that $T_{\text{truck},i} + T_{\text{trailer},i}$ satisfy the system request
 - 6: Filter out infeasible $(T_{\text{truck},i}, T_{\text{trailer},i})$ due to constraints
 - 7: Evaluate UCB for remaining candidates:
 - 8: $UCB_i = \mu(T_{\text{trailer},i}, \omega) + \beta \cdot \sigma(T_{\text{trailer},i}, \omega)$
 - 9: Sort candidates in descending order of UCB
 - 10: **for** each candidate in sorted list **do**
 - 11: Simulate trajectory $x_{0:N}$ using the dynamics model
 - 12: Compute safety indicators: $\alpha_{12}, \alpha_{22}, \theta$
 - 13: **if** safety violation (jackknife or trailer swing) **then**
 - 14: Reject candidate
 - 15: **else**
 - 16: GeorgiaPro-Boldapply $T_{\text{truck}}, T_{\text{trailer}}$ to the system
 - 17: **end if**
 - 18: **end for**
 - 19: Measure Power Loss value
 - 20: Update the learning model
 - 21: **end while**
-

that minimizes the total power losses of both the truck and trailer drivetrain:

$$\min_{\mathbf{u}} P_{\text{loss,truck}}(T_{\text{truck}}) + P_{\text{loss,trailer}}(T_{\text{trailer}})$$

To ensure feasible on the control actions, the following constraints are imposed:

- Force Balance Constraint:

$$B \cdot u = \nu$$

where $B = \begin{bmatrix} \frac{gr_1}{r} & \frac{gr_2}{r} \end{bmatrix}$ called the control effectiveness matrix, ensures the torques satisfy the required longitudinal force $\nu = F_{\text{req}}$. Here, gr_1 and gr_2 are the gear ratios of the truck and trailer respectively, and r is the wheel radius.

- Torque Bounds:

$$T_{\min,i} \leq T_i \leq T_{\max,i}, \quad i \in \{\text{truck, trailer}\}$$

These bounds are based on the motor type.

- Torque Sign Consistency:

$$\text{sign}(T_{\text{truck}}) = \text{sign}(T_{\text{trailer}}) \quad \text{or} \quad T_{\text{truck}} * T_{\text{trailer}} \geq 0$$

This enforces coordinated driving or braking, avoiding energy loss and mechanical stress due to counteracting torques.

3.4.1 Optimal control strategies

During the development of the project the two different learning scenarios required different strategies.

In the initial phase of control allocation, the RLS approach is employed under the assumption that the trailer's power loss can be approximated as a quadratic function of the applied torque. The convex nature of a quadratic cost function allows the use of Quadratic Programming (QP), which ensures global optimality and rapid convergence.

Given the structure of the power loss:

$$P_{\text{loss}}(T) = a(\omega)T^2 + b(\omega)T + c(\omega)$$

The matrix $H \in \mathbb{R}^{2 \times 2}$ is structured in a way that encodes the second-order coefficients of the power loss models, while the vector $\mathbf{g} \in \mathbb{R}^2$ contains the first-order coefficients. The optimization is subject to the constraints presented before:

$$\begin{aligned}
& \min_{\mathbf{u}=[T_{\text{truck}}, T_{\text{trailer}}]} \frac{1}{2} \mathbf{u}^\top \begin{bmatrix} 2a_{\text{truck}} & 0 \\ 0 & 2a_{\text{trailer}} \end{bmatrix} \mathbf{u} + \begin{bmatrix} b_{\text{truck}} \\ b_{\text{trailer}} \end{bmatrix}^\top \mathbf{u} \\
\text{s.t. } & T_{\min, \text{truck}} \leq T_{\text{truck}} \leq T_{\max, \text{truck}} \\
& T_{\min, \text{trailer}} \leq T_{\text{trailer}} \leq T_{\max, \text{trailer}} \\
& B \cdot \mathbf{u} = F_{\text{req}} \\
& \text{sign}(T_{\text{truck}}) = \text{sign}(T_{\text{trailer}})
\end{aligned}$$

To preserve the linearity of constraints in QP while enforcing that both actuators cooperate, the sign condition is handled via conditional torque bounds:

- If $F_{\text{req}} > 0$: set $T_{\min, i} = 0$
- If $F_{\text{req}} < 0$: set $T_{\max, i} = 0$

When the trailer's power loss surface is modeled using a GP, the resulting cost function becomes nonlinear and potentially non-convex. This violates the assumptions of convexity and quadraticity required for the QP approach. To handle this, a Sequential Quadratic Programming (SQP) method is adopted, capable of handling general nonlinear objectives and constraints. As illustrated in Figure 3.10, the GP-estimated mean function can exhibit high nonlinearity and irregularity, especially during the early stages of training.

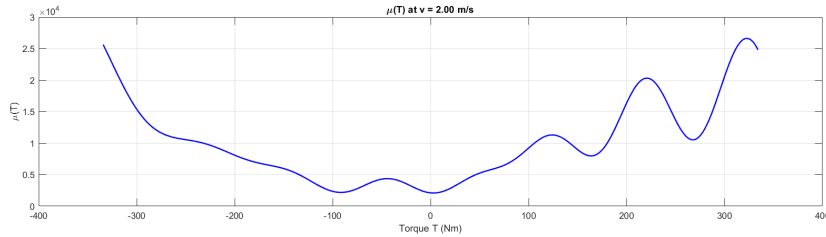


Figure 3.10: Example of a Gaussian Process mean function $\mu(T)$ estimated for a fixed velocity ($\omega = 230 \text{ rad/s}$) during the early stages of training. The irregular, non-smooth shape of the curve illustrates the nonlinearity and potential non-convexity of the GP-based power loss model. Such characteristics violate the convexity and quadraticity assumptions required by standard QP solvers, motivating the use of SQP.

The SQP algorithm solves a sequence of QP subproblems by iteratively linearizing the constraints and approximating the objective function locally. As detailed in Section 2.5.2, each iteration involves computing a null-space-based search direction, performing a merit-function-based line search, updating the

Hessian of the Lagrangian via BFGS, and checking the KKT conditions for convergence.

The torque allocation problem is formally defined as:

$$\begin{aligned}
 \min_{\mathbf{u}=[T_{\text{truck}}, T_{\text{trailer}}]} \quad & P_{\text{loss,truck}}(T_{\text{truck}}) + \hat{P}_{\text{loss,trailer}}(T_{\text{trailer}}) \\
 \text{s.t.} \quad & T_{\text{min,truck}} \leq T_{\text{truck}} \leq T_{\text{max,truck}} \\
 & T_{\text{min,trailer}} \leq T_{\text{trailer}} \leq T_{\text{max,trailer}} \\
 & Bu = \nu \\
 & \text{sign}(T_{\text{truck}}) = \text{sign}(T_{\text{trailer}})
 \end{aligned} \tag{3.8}$$

The first term in the cost is known analytically. The trailer loss term $\hat{P}_{\text{loss,trailer}}$ is a data-driven function represented by the predictive mean of a trained GP model. Since evaluating the GP prediction at each iteration can be computationally expensive and may not offer smooth gradients, the GP mean surface is pre-interpolated using cubic splines over a fixed torque grid. This results in a continuously differentiable approximation, enabling efficient and stable use within the SQP framework.

A key advantage of using Gaussian Processes is their ability to quantify uncertainty. This predictive uncertainty can be explicitly incorporated into the optimization to improve robustness and safety called uncertainty aware optimization.

Two scenarios are considered:

1. **Stochastic Scenario:** The optimization is performed using only the mean estimate of the GP:

$$\hat{P}_{\text{loss,trailer}}(T) = \mu(T)$$

2. **Worst-Case Scenario:** A penalty is added to account for uncertainty, promoting conservative decisions in high-uncertainty regions:

$$\hat{P}_{\text{loss,trailer}}(T) = \mu(T) + \beta \cdot \sigma(T)$$

where $\beta = 1.96$ corresponds to the 95% confidence interval under a Gaussian distribution.

3.4.2 Control Barrier Function

In the current optimal control formulation, a key element is missing: a safety constraint capable of identifying unsafe trajectory splits from the dynamic perspective of the truck-trailer system. As discussed in the exploration phase, a mechanism was needed to detect unsafe behaviors over a short prediction horizon. To address this, we introduce a Control Barrier Function (CBF) based approach.

CBFs are powerful tools for enforcing safety constraints, but they are typically designed for instantaneous (single-step) control systems. Inspired by [21], we extend the CBF concept to a prediction horizon framework, ensuring safety across multiple iterations of the system.

Due to the system's non-affine structure we employed MATLAB's `ode15i` solver to discretize the dynamics and applied the CBF constraints at each discrete time step. We define a discrete-time CBF [23] over a prediction horizon of length N . At each time step $k \in \{0, 1, \dots, N\}$, the system state x_k must satisfy the CBF constraint to ensure it remains within the admissible safe set.

Let the system state at time step k be denoted as $x_k \in \mathbb{R}^n$. We define the safe set $\mathcal{S} \subset \mathbb{R}^n$ as the set of all admissible states that satisfy dynamic stability conditions:

$$\mathcal{S} := \{x_k \in \mathbb{R}^n \mid g_i(x_k) < 0 \text{ for at least one } i \in \{1, \dots, m\}\}$$

where each $g_i(x_k)$ corresponds to a specific unsafe condition, such as excessive articulation or lateral slip.

Unsafe behavior is triggered only when *all* of a set of three conditions are simultaneously satisfied, as discussed in [39]. Since MATLAB's `fmincon` solver does not support logical "AND" constraints directly, we approximate this behavior using a smooth soft-minimum formulation of multiple safety functions [40].

Let $\{g_i(x_k)\}_{i=1}^m$ denote individual safety functions, where each function measures a specific dynamic property related to dynamic instability, such as articulation angle or lateral slip ratio. The system is considered unsafe when all $g_i(x_k) \geq 0$, and safe if at least one $g_i(x_k) < 0$. We define the smooth aggregated barrier function as:

$$h(x_k) = \frac{1}{\kappa} \log \left(\sum_{i=1}^m \exp(-\kappa g_i(x_k)) \right),$$

where $\kappa > 0$ is a smoothing parameter. As $\kappa \rightarrow \infty$, the approximation becomes exact, and the error between the soft-min and the true minimum is bounded above by $\frac{\log m}{\kappa}$, where m is the number of terms [41]. To ensure a sufficiently tight approximation while maintaining numerical stability, it was set $\kappa = 10m$. For instance, with $m = 3$, we use $\kappa = 30$.

Discrete-Time Barrier Condition

To enforce safety across a finite prediction horizon, the following discrete-time CBF condition is imposed at each time step k :

$$h(x_{k+1}) - h(x_k) + \alpha(h(x_k))\Delta t \geq 0,$$

where:

- $\Delta t = 0.01$ s is the simulation time step, corresponding to a 100 Hz sampling rate,
- $\alpha : \mathbb{R} \rightarrow \mathbb{R}$ is a class- \mathcal{K} function, i.e., a continuous, strictly increasing function with $\alpha(0) = 0$,
- in this work, a *linear* class- \mathcal{K} function $\alpha(h) = \gamma \cdot h$ is adopted, meaning $\gamma = 1 \text{ s}^{-1}$, which controls how strongly the system is driven away from the safety boundary $h(x) = 0$.

In classical CBFs, safety is enforced by $\dot{h}(x) + \gamma h(x) \geq 0$, which guarantees forward invariance of the safe set $\{x \mid h(x) \geq 0\}$. In our implementation we adopt the ‘fmincon’ convention $c(x) \leq 0$ and therefore define safety as $h(x) \leq 0$. The corresponding discrete barrier condition over a step Δt is written as

$$h(x_{k+1}) - h(x_k) + \gamma h(x_k) \Delta t \leq 0,$$

which is imposed at each step of the prediction horizon. (Equivalently, one may view this as using $\tilde{h} = -h$ with the classical sign; we keep $h \leq 0$ to match the solver’s inequality form.)

In the implementation two independent barrier functions are constructed to represent distinct unsafe behaviors:

- $h^A(x_k)$: Jackknifing prevention
- $h^B(x_k)$: Trailer swing prevention

Each is composed of three elemental conditions:

$$\begin{aligned} g_1(x_k) &= \alpha_{12} - 0.8, & g_2(x_k) &= -\alpha_{22} + 0.4, & g_3(x_k) &= \theta_1 - \frac{\pi}{3}, \\ g_4(x_k) &= -\alpha_{12} + 0.4, & g_5(x_k) &= \alpha_{22} - 0.8, & g_6(x_k) &= \theta_1 - \frac{\pi}{6}. \end{aligned}$$

The corresponding soft-minimum barrier functions at the current and next time step are:

$$\begin{aligned} h^A(x_k) &= \frac{1}{\kappa} \log \left(\exp(-\kappa g_1) + \exp(-\kappa g_2) + \exp(-\kappa g_3) \right), \\ h^B(x_k) &= \frac{1}{\kappa} \log \left(\exp(-\kappa g_4) + \exp(-\kappa g_5) + \exp(-\kappa g_6) \right), \\ h^A(x_{k+1}) &= \frac{1}{\kappa} \log \left(\exp(-\kappa g_1^{(k+1)}) + \exp(-\kappa g_2^{(k+1)}) + \exp(-\kappa g_3^{(k+1)}) \right), \\ h^B(x_{k+1}) &= \frac{1}{\kappa} \log \left(\exp(-\kappa g_4^{(k+1)}) + \exp(-\kappa g_5^{(k+1)}) + \exp(-\kappa g_6^{(k+1)}) \right). \end{aligned}$$

The final discrete-time safety constraints applied during optimization are:

$$\begin{aligned} h^A(x_{k+1}) - h^A(x_k) + \gamma h^A(x_k) \Delta t &\leq 0, \\ h^B(x_{k+1}) - h^B(x_k) + \gamma h^B(x_k) \Delta t &\leq 0. \end{aligned}$$

These constraints are enforced as nonlinear inequalities at each time step $k \in \{0, \dots, N - 1\}$ within the predictive optimization problem, guaranteeing dynamic safety against jackknifing and trailer swing.

3.4.3 Final Formulation

This section presents the complete mathematical formulation of the torque allocation and safety-constrained control problem developed in this thesis. The control allocation aims to minimize the total power loss of the AEV system while ensuring that the applied torques remain feasible and safe over a prediction horizon.

To enforce safety in the control layer, the CBF constraints are formulated as hard nonlinear inequality constraints. If the optimization problem becomes infeasible (e.g., due to model uncertainty or infeasibility in the safety), the solver terminates, and a predefined fallback strategy is triggered. This strategy allocates torque based on a conservative baseline policy that guarantees feasibility but may be suboptimal in energy efficiency.

$$\begin{aligned} \min_{\mathbf{u}=[T_{\text{truck}}, T_{\text{trailer}}]} \quad & P_{\text{loss, truck}}(T_{\text{truck}}) + \hat{P}_{\text{loss, trailer}}(T_{\text{trailer}}) \\ \text{s.t.} \quad & \forall k = 0, \dots, N - 1, \\ & x_0 \in \mathcal{S}, \\ & x_{k+1} = f(x_k, u), \\ & T_{\text{min, truck}} \leq T_{\text{truck}} \leq T_{\text{max, truck}} \\ & T_{\text{min, trailer}} \leq T_{\text{trailer}} \leq T_{\text{max, trailer}} \\ & \text{sign}(T_{\text{truck}}) = \text{sign}(T_{\text{trailer}}) \\ & Bu = \nu, \\ & h(x_{k+1}) - (1 - \gamma \Delta t)h(x_k) \leq 0 \end{aligned} \tag{3.9}$$

Here:

- \mathcal{S} is the safe set, inside it, all the possible torque split are considered safe.
- $x_{k+1} = f(x_k, u_k)$ is the dynamic simulated through the ode15i command in matlab.
- $h(x_k)$ is the control barrier function enforcing safety.

To complement the safe exploration procedure described in Algorithm 1, the Full Safe Online Learning and Control Framework is presented in Algorithm 2. While Algorithm 1 focuses exclusively on the exploration phase, screening candidate torque splits with safety checks and discarding unsafe actions such as jackknifing or trailer swing, Algorithm 2 integrates both exploration and exploitation in a unified control loop.

Following an ϵ -greedy inspired strategy, Algorithm 2 decides at each time step whether to:

- perform exploration (calling Algorithm 1 as a subroutine), or
- exploit the current GP-based model to allocate torques via SQP optimization under safety constraints.

Algorithm 2 Full Safe Online Learning and Control Framework

```

1: GeorgiaPro-BoldInput: Drive cycle data, truck parameters
2: GeorgiaPro-BoldOutput: Online learned trailer model and optimal torque
   allocations
3: for  $t = 1$  to  $t_{\text{end}}$  do
4:   Update  $\epsilon_t = \epsilon_{\text{min}} + (\epsilon_{\text{max}} - \epsilon_{\text{min}}) \cdot e^{-\lambda t}$ 
5:   if  $\text{rand} < \epsilon_t$  then
6:     Apply exploration policy
7:     Select torque split using GeorgiaPro-BoldSafe-Exploration Learning
   Algorithm
8:     Apply selected torque split to system
9:     Update power loss surface learned
10:  else
11:    GeorgiaPro-BoldUse learned power loss surface
12:    Apply safe-exploitation (SQP with CBF) policy
13:    Apply selected torque split to system
14:  end if
15: end for

```

Chapter 4

Results

This chapter presents the results obtained from the implementation and testing of the proposed online learning and torque allocation framework for articulated electric vehicles. The focus is on the learning accuracy, energy efficiency, safety performance, and optimal control behavior. The framework's components were integrated, tested and compared with industry-standard benchmark to validate improvements.

The research question guiding this thesis is: How can an articulated electric vehicle learn the power loss characteristics of an unknown trailer from limited data, and exploit this knowledge to achieve safe and efficient torque allocation?

To address this question, the evaluation in this chapter is organized around three sub-questions:

- (1) Can the Gaussian Process learn the trailer loss surface online with high accuracy?
- (2) Does improved model accuracy translate into measurable vehicle-level energy savings?
- (3) Can safety be guaranteed during both exploration and control?

The following subsections follow this order, ensuring a consistent link between model fidelity, efficiency gains, and safety assurance.

Before presenting the quantitative results, we first outline the simulation setup used to evaluate the framework. This includes the data generation process, the measurement assumptions, and the standardized drive cycles.

4.1 Data and Environment Setup

All experiments were conducted using MATLAB 2024b. The learning pipeline was implemented using MATLAB's "fitrgp" [42] function for initial Gaussian Process Regression (GPR) tests and later refined using the GPML Toolbox [35] to allow greater flexibility in kernel selection, hyperparameter tuning, and uncertainty

modeling. For the dynamic simulation of the truck trailer system, Simscape was initially used to develop and export a Functional Mock-up Unit (FMU) for integration within MATLAB Simulink, enabling model-based evaluation of safety conditions. However, due to the computational inefficiency and slow simulation times, a faster truck–trailer simulation model [38] was adopted for real-time testing. All simulations and optimization routines were executed on a workstation equipped with an Intel® Core™ Ultra 7 165H vPro® Enterprise CPU (16 cores), running Windows 11 Pro. The system was further supported by an NVIDIA® RTX™ 1000 Ada Generation GPU, 32 GB LPDDR5X memory, and a 1 TB SSD. This computational setup ensured that both Gaussian Process training and optimization-based safety evaluations could be executed within practical runtimes.

4.1.1 Measurement Steup

The data structure assumes tractor-trailer communication interface technology like CAN bus with the following available measurements include the wheel torque, the velocity, voltage of the battery and current.

These signals are subject to sensor noise derived from real datasheets. To simulate realistic drivetrain losses, noise was applied using bounds estimated via first-order Taylor expansion. Figure 4.1 shows a heatmap of estimated uncertainty across the torque–velocity operating space.

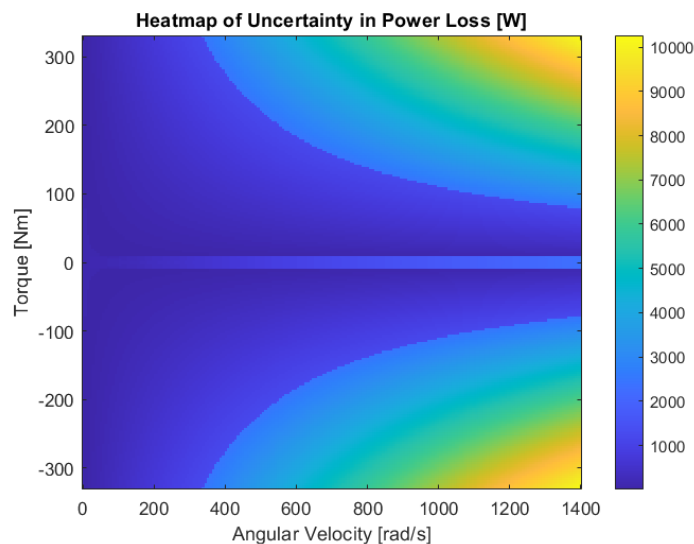


Figure 4.1: Heatmap of estimated uncertainty in power loss (in watts) across the torque– ω space. It can be seen how on the low torque area, the uncertainty is higher than in the other areas.

4.1.2 Driving System and Drive Cycles

To evaluate the proposed framework, two distinct drive cycles were employed. These cycles were selected to represent both real-world and standardized driving conditions:

Table 4.1: Overview of Drive Cycles

Drive Cycle	Type	Duration	Description
Göteborg–Borås cycle	Mixed	~75 minutes	Mainly Highway and straight road trajectory
WHVC cycle	Mixed	~30 minutes	Standardized mixed-condition profile

The selected cycles include diverse conditions such as stop-and-go traffic, highway cruising, accelerations, decelerations, and road gradients. This variety is essential because the efficiency of torque coordination strategies strongly depends on the driving scenario. For instance, frequent stop–start patterns stress the controller’s adaptability to rapidly changing torque demands, while long steady-state highway stretches emphasize sustained efficiency and stability. By combining a standardized cycle (WHVC), widely used in regulatory and research contexts, with a real-world short/medium-haul cycle (Göteborg–Borås), the evaluation achieves both comparability with existing literature and relevance for practical deployment.

Figure 4.2 illustrates the velocity and acceleration evolution of the WHVC cycle. Such profiles are directly translated into longitudinal force demands, forming the basis for torque allocation between truck and trailer. Hence, the drive cycles do not merely act as test scenarios but actively shape the operating points of the learning process. Their diversity ensures that the learning algorithms are exposed to a wide portion of the torque–velocity space, which is crucial for both model accuracy and the robustness of safety enforcement mechanisms.

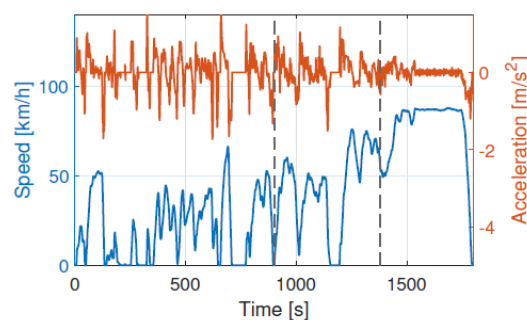


Figure 4.2: Variation of speed and acceleration over time for the World Harmonized Vehicle Cycle (WHVC) drive cycle

4.2 Evaluation Metrics and Benchmarks

Having established the simulation environment, we now define the evaluation metrics and to assess the effectiveness of the proposed framework, its results are bench marked against a rule-based strategy commonly used in the industry for scenarios involving systems with unknown or uncertain parameters. This reference method, as described in [16], provides a consistent foundation for comparing the proposed torque split solution with a baseline.

As part of the benchmarking process, the total longitudinal force request $F_{x,\text{req}}$ is distributed between two units (or components), based on their respective vertical loads. The allocation follows a proportional strategy, ensuring force distribution aligns with the vertical load each axle supports:

$$F_{x,\text{req},1} = \frac{F_{z,1}}{F_{z,1} + F_{z,2}} \cdot F_{x,\text{req}}$$

$$F_{x,\text{req},2} = F_{x,\text{req}} - F_{x,\text{req},1}$$

where:

- $F_{x,\text{req},1}$: Force requested from the first unit
- $F_{x,\text{req},2}$: Force requested from the second unit
- $F_{z,1}, F_{z,2}$: Vertical loads on the first and second axles, respectively
- $F_{x,\text{req}}$: Total longitudinal force demand from the vehicle system

This benchmark approach offers a robust reference point, facilitating quantitative evaluations of the proposed framework under realistic and varied operating conditions.

To evaluate the performance of the proposed safe online learning and control framework, two core criteria are used: (1) learning accuracy and (2) driving efficiency. These metrics provide complementary insights into the effectiveness of the method from both a modeling and control perspective.

1. Learning Accuracy. The accuracy of the trailer power loss surface estimation is evaluated using a normalized version of the Root Mean Squared Error (RMSE). Specifically, the RMSE is divided by the mean of the true power loss values to obtain a dimensionless percentage error relative to the typical magnitude of losses:

$$\text{RMSE}_{\%} = \frac{\sqrt{\frac{1}{N} \sum_{i=1}^N (\text{Ploss}_{\text{true},i} - \mu_i)^2}}{\text{Ploss}_{\text{true}}} \times 100$$

where $P_{\text{loss}_{\text{true},i}}$ is the true power loss at sample i , μ_i is the GP-estimated value, and $\overline{P_{\text{loss}_{\text{true}}}}$ is the mean of the true values. The accuracy metric is then defined as:

$$\text{Accuracy}_{\text{GP}} = 100 - \text{RMSE}_{\%}$$

This definition ensures that 100% indicates perfect prediction, and lower values correspond to proportionally larger average prediction errors relative to the mean true power loss.

2. Driving Efficiency. To quantify real-world benefit, the total energy consumed over the entire simulated drive cycle is computed for each tested strategy. The vehicle's electrical energy consumption E_{total} is obtained by integrating the input power drawn by the motors:

$$E_{\text{total}} = \int_{t=0}^{T_{\text{drive}}} P_{\text{truck}}(t) + P_{\text{trailer}}(t) dt$$

This metric captures the practical impact of learning: even with accurate models, the value lies in whether the resulting control actually saves energy. The learned strategy is considered successful only if it leads to a measurable energy reduction compared to the baseline.

3. Driving Safety. Safety is quantified using two indicators:

- **Intervention rate:** the fraction of torque proposals that are rejected during exploration or corrected by the CBF during exploitation. It quantifies how often the safety filter is required to intervene:

$$\text{Intervention Rate} = \frac{N_{\text{interventions}}}{N_{\text{proposals}}} \times 100\%,$$

where $N_{\text{proposals}}$ is the total number of torque commands considered and $N_{\text{interventions}}$ is the subset identified as unsafe. A lower rate indicates that the learning policy itself naturally respects safety constraints, while a higher rate reflects a heavier reliance on the filter.

- **Violation probability bound:** although no actual violations are observed, a statistical upper bound can still be reported. If zero unsafe events are recorded over N simulation steps, then with 95% confidence the true violation probability satisfies

$$p_{\text{viol}} \leq \frac{3}{N}.$$

This bound formalizes the safety guarantee obtained from repeated trials.

Together, these metrics provide a quantitative measure of how the framework maintains constraint satisfaction and how frequently safety mechanisms are activated to ensure robustness.

The framework is benchmarked against three control strategies:

1. **Benchmark A Load-Based Split:** Volvo’s standard proportional torque allocation, applied when trailer data is unavailable.
2. **RLS + Heuristic based Exploration:** The previous in-house method combining system identification with naive random torque proposals.
3. **Proposed Framework (GP + C-UCB + SQP):** The full methodology described in this thesis, combining uncertainty-aware exploration, Gaussian Process modeling, motion prediction, and safety-constrained SQP optimization.

Learning performance is central to this study. The next section compares alternative models with a focus on Gaussian Processes (GP), which, based on the results, constitute the preferred approach for online loss-surface learning in the framework.

4.3 Gaussian Process Learning Analysis

In the next subsections it will be presented the evaluation of different kernel choices and the trade-offs between sparse and full GP formulations. The metric used for the learning efficiency is the model accuracy in reconstructing the power loss surface.

Several kernel types were tested, including the ARD-SE, ISO-SE, Matérn 5/2 ARD, and a cubic polynomial ARD kernel. As shown in Figure 4.3 and Table 4.2, the ISO-SE kernel achieved the highest average accuracy of 97.46%, with ARD-SE following closely at 92.80%. While ARD-SE provided slightly better accuracy in localized anisotropic regions, its variable length scales occasionally produced non-smooth gradients, which destabilized the SQP optimization in early learning phases. ISO-SE was therefore selected for its smoothness, differentiability, and robust performance across operating conditions.

Table 4.2: Kernel Type Accuracy Comparison

Kernel Type	Accuracy (%)
ARD Squared Exponential Kernel	92.80
Isotropic Squared Exponential Kernel	97.46
Matérn 5/2 ARD Kernel	79.80
Cubic Polynomial ARD Kernel	74.00

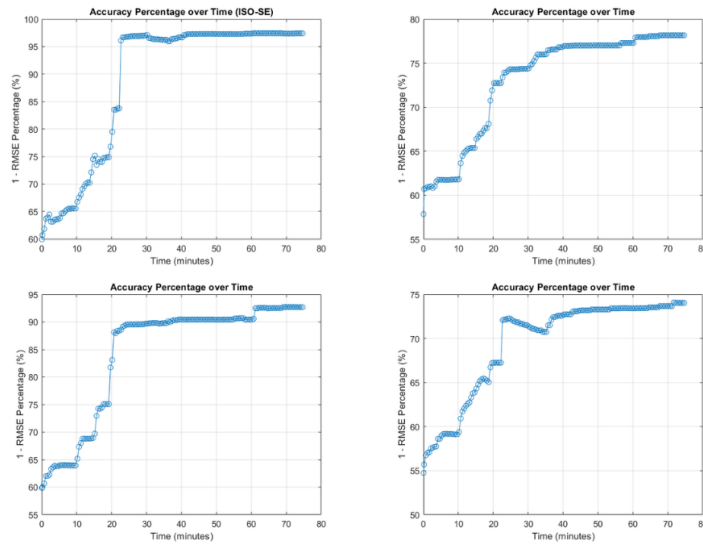


Figure 4.3: Learning accuracy over time for different GP kernels. Each curve reports $1 - \text{RMSE}$ (in %) computed online as the model is updated. Kernels compared: ISO-SE, ARD-SE, Matérn 5/2 (ARD), and cubic polynomial (ARD). ISO-SE attains the highest final accuracy and stabilizes fastest.

4.3.1 Computational Cost and Sparse Approximation

Besides prediction accuracy, update time impacts online applicability. Figure 4.4 reports per-update wall time versus dataset size. On the tested cycles, each full-GP update (hyperparameter re-optimization + retraining) required 0.5–3.0 s as n grew, consistent with $\mathcal{O}(n^3)$. FITC reduced time but incurred up to 8% accuracy loss. Given the downstream sensitivity of SQP to model fidelity, the full ISO-SE GP was retained here; scalable GP variants remain future work for embedded deployment.

Table 4.3: Estimated update time of full vs. sparse GP models. Sparse GP values are computed assuming $n = 100$ inducing points.

Training samples	Full GP [s]	Sparse GP [s]
300	0.80	0.19
500	1.70	0.27
750	2.80	0.75

Table 4.3 report the values for the training of the GP both in its full and sparse settings. Considering the results presented in [43], using the same GPML toolbox, show significantly smaller execution times for the Full GP across comparable dataset sizes. This discrepancy may be attributed to differences in computational setup, solver configuration, or implementation details, and highlights the sensitivity of GP training time to both software and hardware choices.

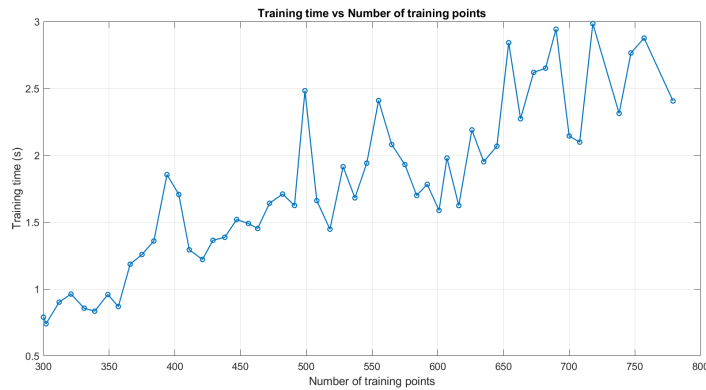


Figure 4.4: Evolution of Gaussian Process training time as a function of dataset size. Each update includes hyperparameter re-optimization and posterior retraining. Training time grows from 0.7 s at 300 training samples to nearly 3 s with over 750 training samples, reflecting the cubic complexity of exact GP inference.

To evaluate computational trade-offs, sparse GP approximations were tested using different techniques for the inducing points: k-means clustering, farthest point sampling (FPS), and a hybrid approach combining 30% k-means with 70% FPS. As shown in Figure 4.5 and Table 4.4, the hybrid model offered better coverage and smoother learning behavior than k-means alone, which suffered from instability. However, all sparse methods underperformed compared to the full GP in terms of peak accuracy.

Table 4.4: Sparse vs. Full Gaussian Process Comparison

Model Type	Maximum Accuracy (%)	Computational Complexity
Full Gaussian Process	97.46	$\mathcal{O}(n^3)$
Sparse Gaussian Process	89.70	$\mathcal{O}(nm^2)$

Given that the full GP reached 97.46% maximum accuracy versus 89.70% for the sparse variants, and considering the stability observed under varying operating conditions, the full ISO-SE GP was adopted despite its $\mathcal{O}(n^3)$ cost, as it provided the accuracy needed to reliably drive the SQP allocation.

4.3.2 Hyperparameter Optimization and Model Adaptation

While the core hyperparameter initialization and update scheme were described in the methodology, the results here show how those parameters evolved over time and influenced performance under different initialization.

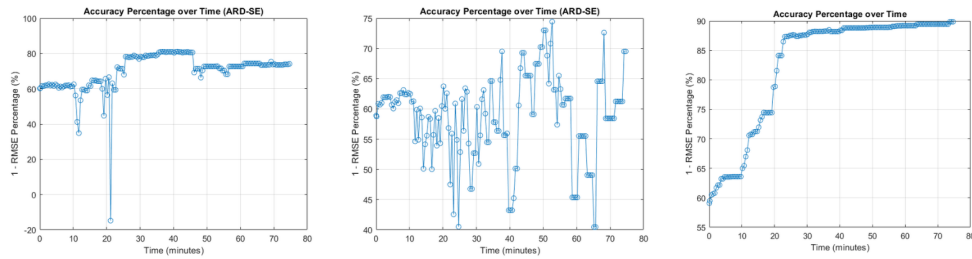


Figure 4.5: Learning accuracy over time for sparse GP (FITC) approximations using three inducing-point selection strategies: (a) FPS, (b) K-means, (c) hybrid. The metric is $1 - \text{RMSE}$ (in %). Sparse configurations converge more slowly and can oscillate compared with the full GP; the best sparse (hybrid) attains the highest accuracy among sparse variants but still plateaus below the full model.

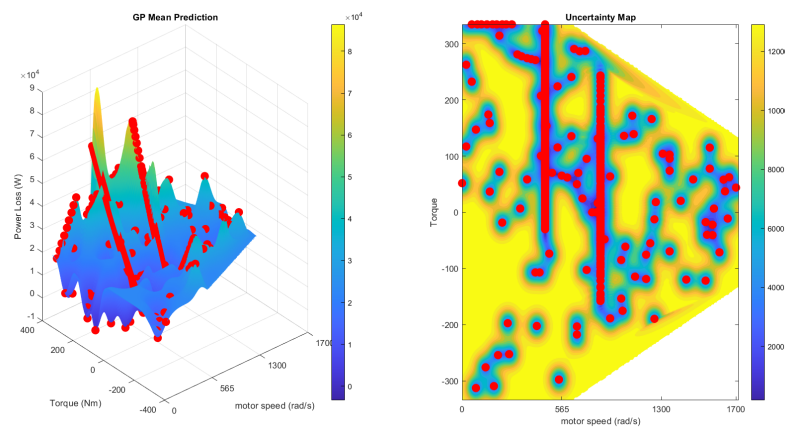


Figure 4.6: Example of overfitting caused by poor hyperparameter initialization and lack of regularization. Left: GP mean surface $\mu(T, \omega)$ with spurious high-curvature ridges. Right: predictive standard deviation $\sigma(T, \omega)$ with training samples (red) showing sparse regions where extrapolation inflates uncertainty.

To complement the nominal traces reported in Figure 3.3, Figure 4.7 shows a deliberately poor-initialization run. Without suitable priors or bounds, the length scale and signal variance drift and can fail to stabilize, producing the overfitting in Figure 4.6. This behavior will explain part of the transient energy overhead observed during learning and disappears once the hyperparameters are properly initialized and constrained.

Overall, the periodic optimization strategy ensured that, when appropriately initialized, hyperparameters adapted smoothly, especially after the first 20 minutes of exploration. This reinforced the GP's ability to generalize and safely guide control decisions.

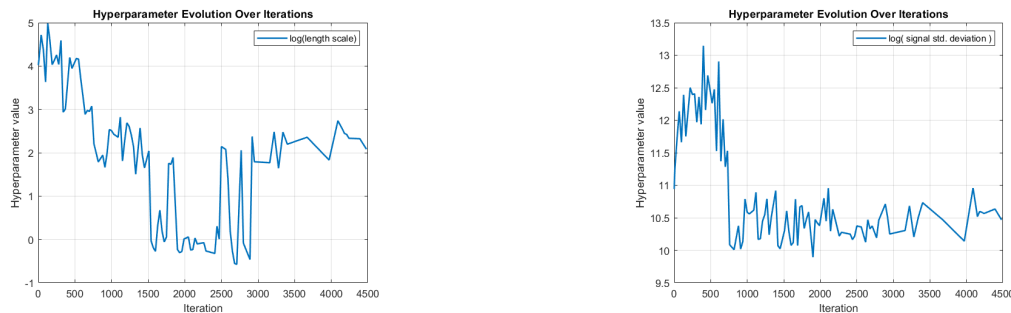


Figure 4.7: Hyperparameter evolution over online updates. Left: log length-scale. Right: log signal variance. Both traces drift and may not stabilize without priors or bounds, leading to unrealistic fits (cf. Figure 4.6).

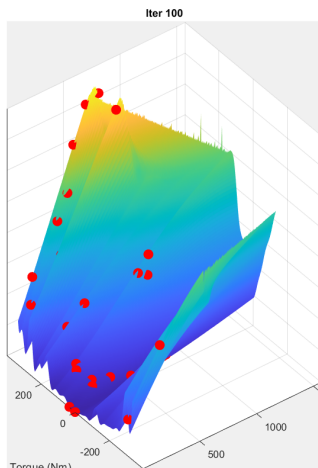
4.3.3 Surface Evolution over Time

To validate the progressive learning behavior of the GP model, we visualize the evolution of the predicted power loss surface across time. These snapshots illustrate how the model initially interpolates based on sparse, noisy measurements and gradually converges to a smoother, more accurate estimate as more data becomes available.

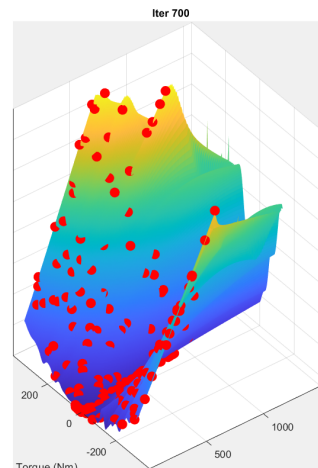
Figure 4.8 shows the GP-predicted power loss surface at four key time points: 2 minutes, 12 minutes, 22 minutes and 28 minutes. Each surface is plotted over the torque–velocity (T, ω) space, with color representing the predicted power loss in watts.

Initially (Figure 4.8a), the surface exhibits local overfitting due to insufficient data and poor hyperparameter initialization. As the learning process continues (Figures 4.8b and 4.8c), the GP smooths out irregularities and begins to capture the general trends of the power loss function. By 30 minutes (Figure 4.8d), the surface has stabilized and closely aligns with the true system behavior, especially in regions of the (T, ω) space that were frequently visited during exploration.

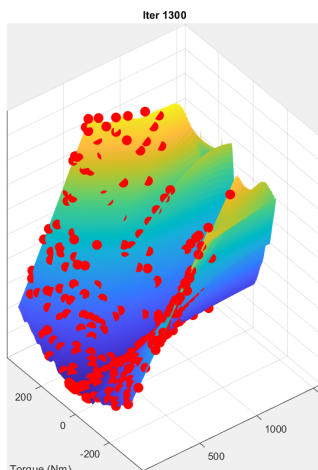
This qualitative evolution of the model supports the accuracy trends previously reported, and visually confirms the effectiveness of the GP in learning non-parametric trailer dynamics online under noise and uncertainty. To provide a balanced algorithm comparison, we next evaluate Recursive Least Squares. While less flexible than GP, RLS remains attractive due to its simplicity, low computational cost, and established use in related applications. The following section analyzes its performance across motor types and forgetting factor choices.



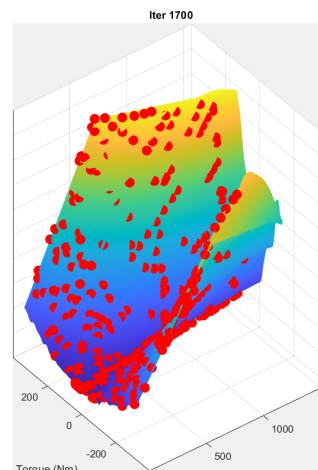
(a) 2 minutes



(b) 12 minutes



(c) 22 minutes



(d) 28 minutes

Figure 4.8: Gaussian Process power loss surface predictions at different time intervals during online learning. Surfaces converge as model accuracy improves and exploration progresses.

4.4 Recursive Least Squares Learning Analysis

The RLS method was implemented to model the power loss surface in real time using a polynomial regression model. This section presents its performance across different motor types, sensitivity to noise, and the influence of forgetting factors.

The RLS method is a fixed polynomial approach, so it differs from the GP in this sense, since the results change substantially from the type of motor tested. The two

motor configurations tested are the Permanent Magnet Synchronous Motor and the Induction Motor. As illustrated in Figure 4.9, the model achieved faster convergence and higher accuracy with PMSM. Specifically, the RLS reached over 90% accuracy within the first 10 minutes of operation. This rapid improvement was linked to the near-quadratic nature of PMSM power losses, which matched the polynomial structure of the RLS model.

Conversely, RLS performance degraded significantly under Induction Motor settings due to more complex loss behavior. Polynomial mismatches introduced systematic bias and increased residual errors.

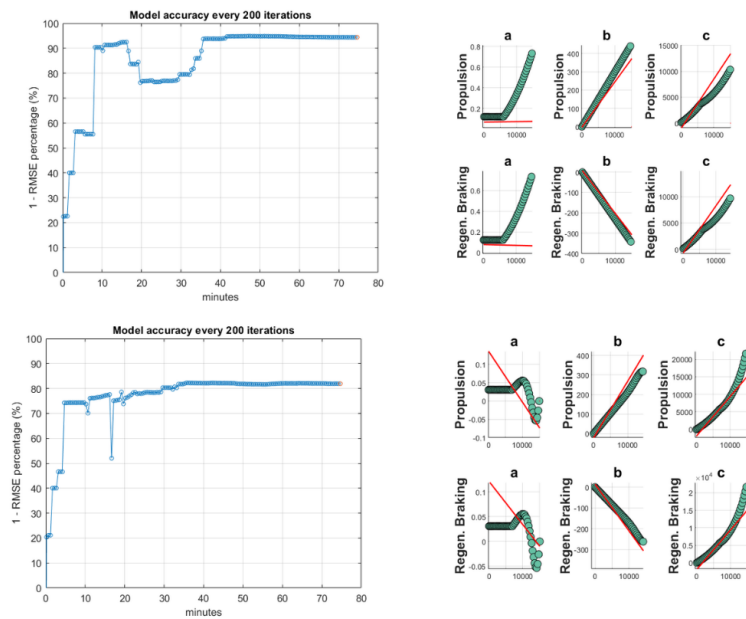


Figure 4.9: Comparison of RLS model learning accuracy and residual behavior for PMSM and Induction motors. PMSM surfaces fit well to quadratic models (top figure), while induction models suffer from structural mismatches (bottom figure).

Table 4.5 reports final accuracy using both linear and quadratic RLS surfaces for each motor type. PMSM saw high fidelity from both models (especially linear), while IM exhibited an accuracy drop under both linear quadratic fitting.

Table 4.5: Accuracy of RLS across motor types and polynomial structures

Motor Type	Polynomial Type	Accuracy (%)
PMSM	Linear	93.75
PMSM	Quadratic	92.00
Induction Motor	Linear	82.60
Induction Motor	Quadratic	81.00

The forgetting factor λ plays a critical role in balancing reactivity and stability. Figures 4.10 compare performance under $\lambda = 0.95$ and $\lambda = 1.0$. With $\lambda = 0.95$, the model adapted quickly to changing conditions, albeit with minor instability in early learning. On the other hand, $\lambda = 1.0$ yielded a smoother convergence curve but was slower to respond to new input distributions.

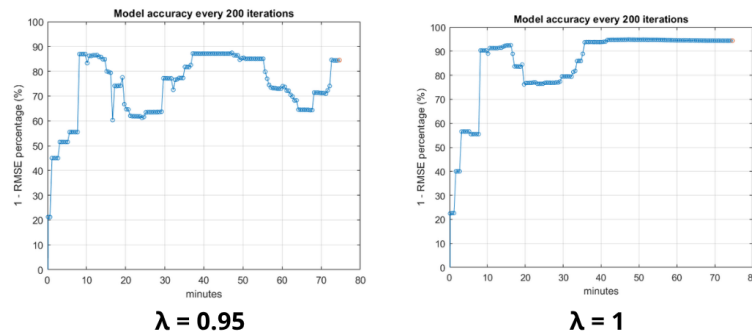


Figure 4.10: Effect of the forgetting factor on RLS learning accuracy. With $\lambda = 0.95$ (left), the model adapts more quickly to new data but exhibits oscillations and instability. With $\lambda = 1$ (right), the model is more stable but slower to adapt, highlighting the trade-off between responsiveness and robustness.

4.4.1 Impact of Noise on RLS

Sensor noise was shown to strongly impact RLS stability, especially without explicit noise modeling. The surface predictions became erratic in low-data regions and under fluctuating torque-speed commands. Figure 4.11 demonstrates how RLS fits deviate from the true surface under noise, while the residual errors appear visually consistent despite being statistically biased.

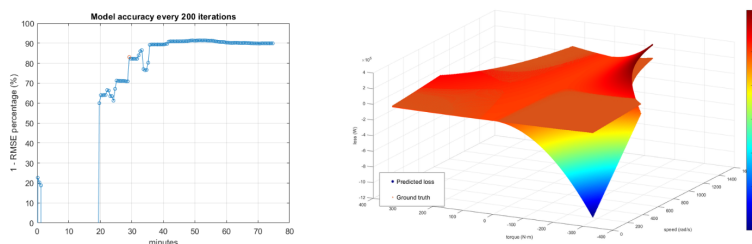


Figure 4.11: Influence of sensor noise on RLS fitting. Left: learning accuracy degrades under noisy observations, with fluctuations in convergence. Right: reconstructed loss surface becomes distorted in sparsely sampled and noisy regions, indicating reduced reliability.

Compared to the Gaussian Process approach, RLS lacks uncertainty quantification, making it more fragile in high-noise regions. The inability to represent aleatoric

uncertainty contributed to larger generalization gaps, especially in early learning.

4.5 Exploration Analysis

Beyond model learning accuracy, the performance of the exploration strategies was analyzed to evaluate their ability to cover the torque–velocity space and balance exploration with exploitation.

4.5.1 Grid-Based Exploration

A trade-off exists between grid resolution and real-time feasibility: finer grids improve learning accuracy but increase computational and memory demands.

In this work, a resolution of 30 torque bins and 10 velocity bins was selected, providing wide coverage while maintaining computational efficiency. Figure 4.12 illustrates the resulting coverage maps across two representative drive cycles. Coverage strongly depends on the drive cycle, as vehicle velocity is externally prescribed and cannot be directly influenced by the exploration strategy.

4.5.2 UCB Parameter Sensitivity

For the uncertainty-driven exploration, the β parameter in the UCB criterion governs the exploration–exploitation trade-off. A sensitivity analysis was conducted for $\beta \in \{0.5, 2.0, 4.5, 10.0\}$. Lower values favored exploitation and faster initial convergence but left unexplored regions under-sampled. Higher values promoted broader exploration but delayed convergence to efficient splits. Based on these experiments, $\beta = 10.0$ was selected as a balanced choice, ensuring both coverage and stable allocations.

Synthesis The learning analysis supports the choice of Gaussian Processes over RLS. Across kernels, the isotropic squared–exponential GP attains the highest and most stable reconstruction accuracy of the loss surface and provides smooth gradients that are reliable for downstream optimization, whereas alternative kernels converge more slowly or less smoothly. RLS, being a fixed polynomial fit, reforms well only when the true losses are near-quadratic but degrades markedly when the operating behavior departs from that structure, as observed under induction-motor settings. Moreover, GP learning offers calibrated predictive uncertainty, which is essential to drive principled exploration and to avoid overconfident allocations in poorly sampled regions; RLS provides no uncertainty quantification and is therefore more fragile in high-noise zones. The GP formulation also admits explicit noise modeling (including heteroscedastic noise), which improves generalization in regions of sparse data and prevents the instability seen with RLS when measurement

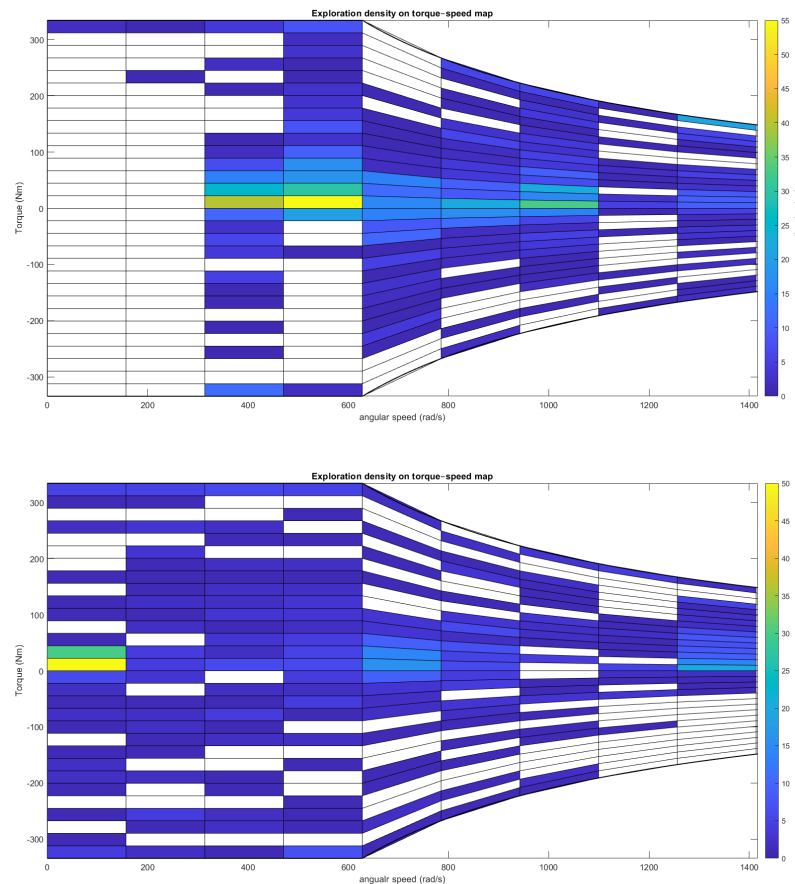


Figure 4.12: Coverage maps of the torque–velocity space (T), obtained using the grid-based exploration heuristic for two different drive cycles. Each cell corresponds to a grid bin in the space, with color intensity indicating the number of visits during exploration. The heuristic perturbs the previous torque split and selects the candidate landing in the least-visited cell, promoting uniform coverage and avoiding overfitting to specific operating points. Differences between the two maps illustrate that exploration coverage strongly depends on the drive cycle, as the vehicle velocity is fixed by the cycle and cannot be directly influenced by the exploration algorithm.

noise increases. In online operation, the exploration schedule ensures sufficient coverage during the initial learning window, after which the GP model stabilizes and its uncertainty contracts in the frequently visited torque–speed cells, yielding a dependable surface for control.

Crucially, the advantage of the GP is not only statistical but also functional: higher fidelity and smoother gradients enable the SQP controller to identify more efficient torque allocations, while the polynomial bias of RLS often leads to suboptimal or unstable splits in nonlinear regions. In this sense, improved modeling

accuracy is expected to directly translate into better vehicle-level energy outcomes, a hypothesis tested in the following section.

4.6 Energy Efficiency Analysis

Having established in the previous section that the proposed GP-based framework achieves higher modeling accuracy than the baseline RLS approach, we now turn to the question of whether this accuracy translates into tangible vehicle-level benefits. In particular, the following analysis evaluates energy efficiency across the two representative drive cycles. The goal is to assess whether improved learning not only reduces prediction error but also results in measurable reductions in electrical energy consumption. For comparison, the learning-based strategies are benchmarked against consistent rule-based baselines. To contextualize achievable performance, an idealized full-knowledge case is also considered, where the controller has perfect access to the loss map. The results are summarized in Table 4.6.

Table 4.6: Comparison of electrical energy consumption (kJ) across drive cycles for baseline rule-based control and an idealized full-knowledge case.

Strategy	WHVC cycle	Göteborg–Borås cycle
Rule-based baseline	142.23	406.00
Full-knowledge (oracle)	137.01	389.27

A post-learning evaluation is included because it isolates the steady-state quality of the learned policy from the transient exploration cost. Re-running the same drive cycle after the model has converged quantifies the deployable performance once adaptation is complete, which is the operating regime of interest for deployment. Reporting both the during-learning and the post-learning numbers, and benchmarking them against a full-knowledge oracle, provides a fair picture: the online cost incurred while identifying the system and the asymptotic efficiency once the controller exploits the learned map. This separation is standard in data-driven control and learning, where asymptotic (after-convergence) performance is assessed independently from the learning transients.

4.6.1 Energy Consumption

This subsection quantifies the electrical energy drawn over each drive cycle for the rule-based baseline, the learning-based controllers during online learning, and their post-learning counterparts. The post-learning condition is defined as a re-run of the same cycle after convergence of the learned loss model to a target accuracy of

at least 82%, so that the controller exploits the identified map without additional exploration.

Table 4.7: Electrical energy consumption on the WHVC cycle for RLS- and GP-based controllers, reported both during online learning and after convergence (post-learning). Values are compared against the rule-based baseline and the full-knowledge oracle.

strategy	energy (kJ)	ratio vs. rule-based	ratio vs. full-knowledge
RLS	143.25	100.7%	104.5%
GP	143.59	100.9%	104.8%
RLS (post-learning)	140.10	98.5%	102.2%
GP (post-learning)	138.92	97.6%	101.3%

On the WHVC cycle (Table 4.7), both learning-based controllers show a small transient overhead relative to the 142.23 kJ baseline: 143.25 kJ for RLS and 143.59 kJ for GP (about 0.7–0.9%). The slightly higher GP cost during learning is expected, since WHVC excites a near-quadratic operating region where the parametric RLS surface and its convex QP exploit well, while the GP stack performs uncertainty-driven exploration with periodic retraining (and a more conservative SQP when uncertainty is considered), adding temporary overhead. After convergence, the trend reverses: RLS (post-learning) reaches 140.10 kJ (98.5%) and GP (post-learning) 138.92 kJ (97.6%). In absolute terms, GP saves 3.31 kJ versus rule-based and remains 1.91 kJ above the full-knowledge reference of 137.01 kJ (about 1.3%), confirming that the brief RLS advantage is a learning-phase artefact rather than a steady-state property.

Table 4.8: Electrical energy consumption on the real-world Göteborg–Borås cycle. Results are shown for RLS and GP controllers during learning and after convergence, benchmarked against rule-based and full-knowledge references.

strategy	energy (kJ)	ratio vs. rule-based	ratio vs. full-knowledge
RLS	415.19	102.2%	106.7%
GP	410.46	101.1%	105.4%
RLS (post-learning)	397.52	97.9%	102.1%
GP (post-learning)	391.40	96.4%	100.5%

On the Göteborg–Borås trajectory (Table 4.8), the same pattern appears with larger absolute magnitudes due to the longer horizon and richer operating envelope. During learning, RLS and GP consume 415.19 kJ and 410.46 kJ against the 406.00 kJ baseline, i.e., overheads of about 2.2% and 1.1%. post-learning, RLS

achieves 397.52 kJ (97.9% of baseline) and GP achieves 391.40 kJ (96.4%), yielding absolute savings of 8.48 kJ and 14.60 kJ relative to rule-based. The GP post-learning value is 2.13 kJ above the full-knowledge case of 389.27 kJ, corresponding to a gap of about 0.5%.

Comparing strategies, the GP controller outperforms the RLS controller both during and post-learning. The post-learning advantage is about 1.2 kJ on WHVC and about 6.1 kJ on Göteborg–Borås. This behavior is consistent with a nonparametric model that captures non-quadratic loss characteristics and leverages calibrated uncertainty during allocation. The residual gap to the oracle is attributable to remaining model bias and uncertainty in sparsely visited torque-speed regions and to safety and rate constraints that limit instantaneous optimal reallocation.

Across cycles, relative improvements are larger on the real-world trajectory than on WHVC. The longer duration and broader distribution of operating points improve identification and create additional opportunities to exploit the learned loss map, while the rule-based baseline remains conservative on segments where the learned controller can re-distribute torque more efficiently. In steady state, the GP controller operates within one to two percent of the full-knowledge bound on WHVC and within about half a percent on Göteborg–Borås, while the transient overheads incurred during online learning remain within one to two percent. From a deployment perspective, reporting both during-learning and post-learning results is informative: the former quantifies the temporary energy cost of information acquisition, whereas the latter quantifies the steady-state efficiency available once the trailer map is identified and exploited.

4.6.2 Energy Losses

In addition to evaluating total energy consumption, it is essential to quantify the energy losses occurring during vehicle operation. These losses provide insight into system inefficiencies and the impact of different control strategies on overall energy utilization. Table 4.9 presents the energy losses associated with the rule-based and full-knowledge scenarios, which serve as reference points. These values are compared to the results obtained using both RLS and GP strategies, during and after the learning, as shown in Tables 4.10 and 4.11, respectively for the WHVC and Göteborg–Borås cycle.

Table 4.9: Reference energy losses for the WHVC and Göteborg–Borås cycles. The rule-based strategy serves as baseline, while the full-knowledge oracle represents the lower bound on achievable losses.

	WHVC cycle	Göteborg–Borås cycle
Rule-based baseline	68.58	178.04
Full-knowledge (oracle)	63.49	160.36

Table 4.10: Energy losses on the WHVC cycle for RLS- and GP-based controllers, during learning and after convergence (post-learning). Ratios quantify improvements relative to the rule-based baseline and the oracle.

strategy	energy (kJ)	ratio vs. rule-based	ratio vs. full-knowledge
RLS	66.13	96.4%	104.2%
GP	65.85	96.0%	103.7%
RLS post-learning	65.50	95.5%	103.2%
GP post-learning	64.28	93.7%	101.2%

Loss trends mirror the consumption results, so we focus on post-learning deltas and residual gaps to the oracle. On WHVC, RLS (post-learning) reduces losses from 68.58 kJ to 65.50 kJ (gap to oracle 2.01 kJ), while GP (post-learning) reaches 64.28 kJ (gap 0.79 kJ), closing about 85% of the baseline-to-oracle gap. During learning, losses are 66.13 kJ (RLS) and 65.85 kJ (GP), i.e., a small transient overhead relative to the oracle.

Table 4.11: Energy losses on the Göteborg–Borås real-world cycle for RLS and GP controllers. Both learning and post-learning cases are reported, with comparisons to rule-based and full-knowledge references.

strategy	energy (kJ)	ratio vs. rule-based	ratio vs. full-knowledge
RLS	185.94	104.4%	116.0%
GP	177.00	99.4%	110.4%
RLS post-learning	168.52	94.7%	105.1%
GP post-learning	162.49	91.3%	101.3%

On the Göteborg–Borås cycle, gains are larger: RLS (post-learning) reduces losses from 178.04 kJ to 168.52 kJ (residual 8.16 kJ), and GP (post-learning) reaches 162.49 kJ with only 2.13 kJ above the oracle (about 1.3%). GP already matches or

slightly improves the baseline during learning (177.00 kJ), whereas RLS increases losses (185.94 kJ).

Overall, the nonparametric GP consistently achieves lower losses than RLS, with the largest advantage on the Göteborg–Borås cycle. The small residual to the oracle reflects remaining model uncertainty in sparsely visited torque–speed regions and safety/rate limits that restrict instantaneous reallocation.

4.6.3 Learning and Exploration Cost

During the exploration phase of the learning process, energy consumption is expected to be higher due to the need for diverse and informative actions that guide the learning agent toward an accurate understanding of the environment. Table 4.12 reports the energy consumed during exploration for both the RLS and GP strategies.

Table 4.12: Additional energy consumed during the learning phase due to exploratory torque allocations. Reported for RLS with heuristic exploration and GP with uncertainty-driven (UCB) exploration.

strategy	energy consumptions (kJ)
RLS with Heuristic based	43.86
GP with UCB	44.80

Among the methods evaluated, the GP-based exploration using the UCB criterion demonstrated a favorable trade-off between exploration efficiency and operational safety. This strategy ensured comprehensive coverage of the torque–velocity space while minimizing unnecessary energy expenditure and avoiding unsafe control behaviors. These characteristics make UCB-based exploration particularly suitable for real-world applications where performance and reliability must be carefully balanced.

Synthesis. Across both drive cycles, online learning introduces a small transient overhead (WHVC: +0.7–0.9%; Göteborg–Borås: +1.1–2.2%) that disappears after convergence. In steady state, the GP controller consistently outperforms RLS and the rule-based baseline, reducing total energy by about 2.3% on WHVC (142.23 kJ → 138.92 kJ) and 3.6% on Göteborg–Borås (406.00 kJ → 391.40 kJ), and operating within roughly 1.3% and 0.5% of the full-knowledge reference, respectively. Losses mirror this trend: GP (post-learning) lowers WHVC losses by 6.3% (68.58 kJ → 64.28 kJ) and real-world losses by 8.7% (178.04 kJ → 162.49 kJ), with residuals of about 1.2–1.3% to the oracle. The brief WHVC advantage of RLS during learning is a transient, model-class alignment effect; after convergence, GP is the preferred strategy. The remaining gap to full knowledge is explained by unavoidable model

uncertainty in sparsely visited torque-speed regions and by safety/rate limits that preclude instantaneous optimal reallocations.

These efficiency gains, however, are meaningful only if they can be realized without compromising vehicle stability. We therefore proceed to evaluate the framework’s ability to ensure safety during both learning and control execution in the next section.

4.7 Safety Evaluation

An essential component of the proposed framework is its capacity to detect and prevent unsafe behaviors arising during both learning and control execution. Such behaviors are primarily linked to highly unbalanced torque allocations between the truck and trailer, which can compromise vehicle stability particularly in adverse conditions or during aggressive maneuvers.

For all safety evaluations, the prediction horizon was set to 3 s. This duration was selected based on empirical testing: it is long enough to capture the delay between torque application and the next GP model update, while short enough to maintain computational efficiency. In practice, a 3 s rollout provided sufficient foresight to reveal critical instabilities without incurring prohibitive computation times.

4.7.1 Detection of Unsafe Behavior

Across all evaluated drive cycles, no actual safety violations were observed, confirming that the proposed framework successfully enforces constraint satisfaction. Safety can therefore be quantified by the frequency of rejected torque allocations during exploration.

Table 4.13 reports the total number of torque proposals and the subset rejected as unsafe. In the WHVC cycle, 25 out of 1801 proposals were rejected, corresponding to an intervention rate of 1.4%. These events were primarily linked to high acceleration or deceleration combined with aggressive steering, which was artificially introduced to probe unsafe regions. In contrast, during the Göteborg–Borås cycle, no unsafe allocations were detected, resulting in a zero intervention rate.

Table 4.13: Rejected torque allocations during exploration

Drive cycle	Proposals N_{prop}	Rejected N_{rej}
WHVC	1801	25
Real-world	4980	0

Since no safety violations occurred, the statistical upper bound on violation probability can be derived. With 95% confidence, the true violation probability satisfies $p_{\text{viol}} \leq 3/N$, which yields $p_{\text{viol}} \leq 0.0017$ for the WHVC cycle and $p_{\text{viol}} \leq 0.0006$ for the Göteborg–Borås cycle. This formalizes the safety guarantees obtained under the tested conditions.

Overall, these results demonstrate that the framework reliably prevents unsafe torque allocations: in WHVC only a small fraction of exploratory actions were filtered, while in the Göteborg–Borås cycle all proposals were accepted without requiring intervention. The absence of unsafe actions in the latter case is explained by the characteristics of the drive cycle itself: it primarily consists of highway operation, where steering angles remain small and vehicle speeds are relatively conservative, so unsafe conditions do not arise.

The intervention events primarily occurred during the exploration phase, when uninformed torque allocations were proposed to improve map coverage.

To stress-test the safety mechanisms, unsafe conditions were artificially introduced by combining aggressive steering inputs with high acceleration or deceleration. Although rollout simulation is used during exploration and CBF is embedded in exploitation, both were evaluated under identical unsafe scenarios to verify that they consistently detect and reject the same unsafe behaviors.

Figure 4.13 illustrates these cases: red crosses mark unsafe torque splits rejected by rollout simulation, and green circles mark those identified by the CBF. Both approaches flagged the same unsafe behaviors, with no difference in rejection outcome.

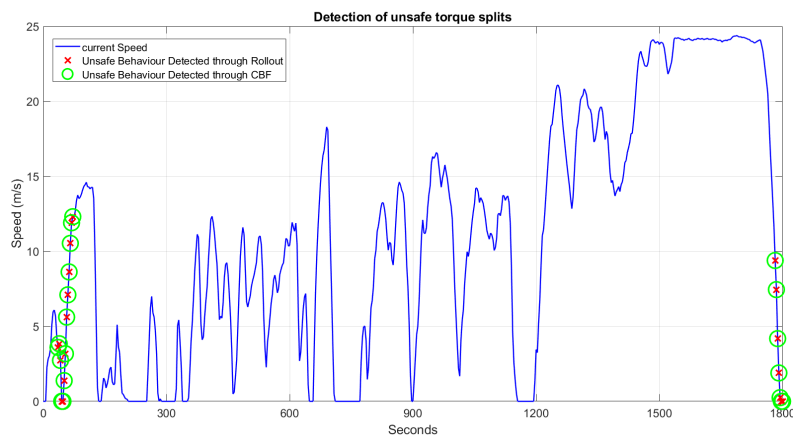


Figure 4.13: Unsafe torque allocations detected and rejected in real time during the WHVC drive cycle. Red crosses denote unsafe behavior identified via rollout simulation, and green circles denote unsafe behavior by the CBF.

It is worth noting that some rejected allocations would have led to slightly more energy-efficient splits if safety constraints were ignored. Thus, the safety

layer occasionally sacrifices marginal efficiency to preserve stability. Nevertheless, the GP-based controller still achieves measurable energy savings compared to baselines, showing that safety enforcement filters out only unsafe allocations without eliminating the overall efficiency benefit.

4.7.2 Safety Mechanisms: Rollout Simulation and Control Barrier Function

Two primary safety mechanisms were employed to evaluate torque allocations: rollout simulation during exploration and the CBF during exploitation. While both methods serve the common goal of ensuring safe behavior, they differ fundamentally in their approach and timing of detection. To illustrate their operational differences, a jack-knifing scenario is shown in Figure 4.14, which visualizes an unsafe vehicle behavior approximately 1.6 seconds after the application torque input.

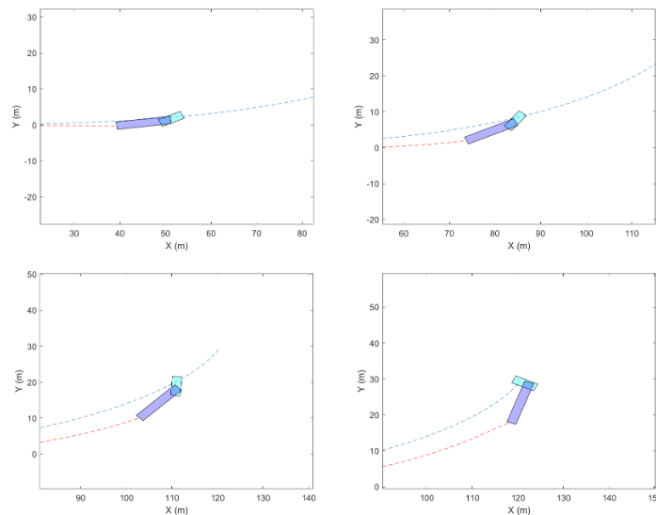


Figure 4.14: Example of a jack-knifing event occurring 1.6 seconds after the application of an unsafe torque split. The vehicle trajectory departs from the safe envelope, illustrating the type of instability that motivates the use of safety filters.

The rollout simulation mechanism assesses future system states by simulating the evolution of the system under the proposed control input. As shown in Figure 4.15, this method flags unsafe behavior only when all safety constraints are violated simultaneously. This makes it a reactive method with limited predictive capability.

In contrast, the Control Barrier Function acts as a predictive safety filter by evaluating the future evolution of safety-related constraints. As depicted in Figure 4.16, the CBF anticipates unsafe behavior and intervenes before the unsafe state is reached (at 1.12 seconds) providing a more proactive and reliable safety guarantee.

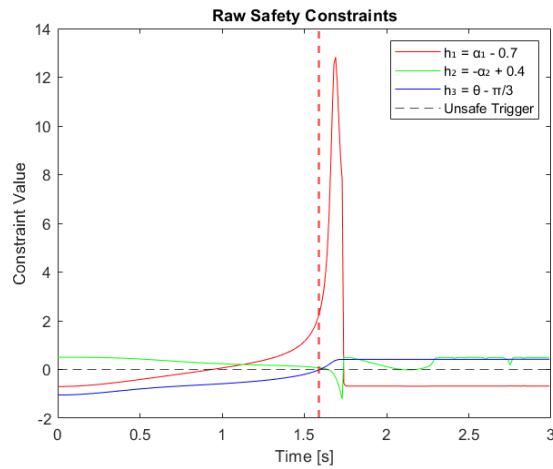


Figure 4.15: Safety-constraint evolution under rollout: violation detected at $t = 1.6$ s (reactive detection).

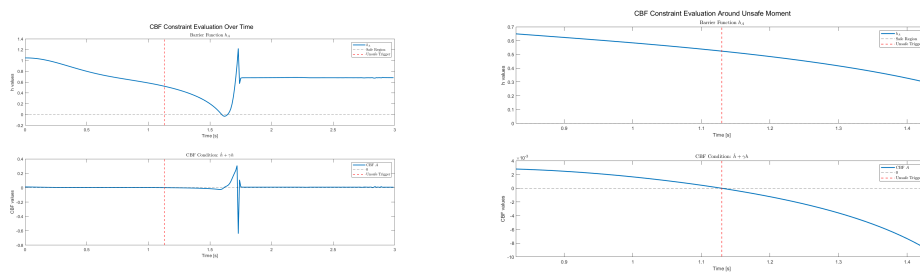


Figure 4.16: . Unsafe behavior predicted and prevented by the CBF. The intervention occurs about 0.5 seconds earlier than rollout detection, providing a more proactive and computationally efficient safety guarantee.

In driving scenarios where none of the critical safety constraints are violated, neither the rollout simulation nor the CBF mechanisms are triggered. This validates the selective nature of the framework’s safety layer. Figure 4.17 confirms that under safe operating conditions, both safety modules remain inactive, avoiding unnecessary interventions.

4.8 Optimal Control Performance

Having established in the energy analysis that post-learning operation relies on two distinct controllers: RLS paired with quadratic programming, and GP paired with sequential quadratic programming and a control barrier function, this section examines their control-level behavior in more detail. Whereas the previous section quantified overall efficiency, here the focus shifts to the new mechanism paired

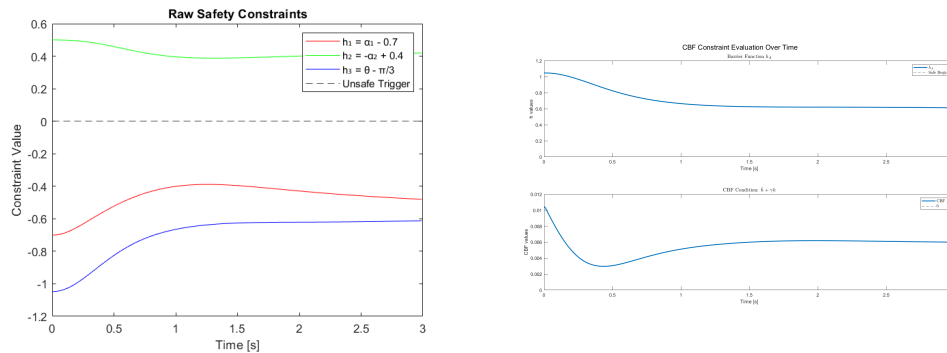


Figure 4.17: Validation under nominal conditions. Neither rollout nor CBF mechanisms are triggered when the vehicle remains within safe regions, confirming that the safety layer avoids unnecessary interventions.

with the GP learning, that makes allocation decisions under uncertainty and safety constraints.

Aligning the control analysis with the energy experiments ensures that the mechanisms examined, directly explain the aggregate outcomes reported earlier. The comparison highlights a deliberate trade-off: the RLS+QP controller prioritizes simplicity and computational economy but lacks formal treatment of uncertainty and safety, whereas the GP+SQP+CBF controller explicitly manages both at modest additional complexity.

4.8.1 Uncertainty-Aware Optimization

As detailed in the methodology, the GP model represents both the mean behavior of the loss surface and its predictive uncertainty, which decreases as data accumulates in frequently visited regions. Early in learning, uncertainty is high across large parts of the torque–velocity space, but after exploration the model converges to a smooth, reliable representation of trailer losses. This evolution is illustrated in Figure 4.18, where the initial surface (left) displays strong uncertainty artifacts, while the post-learning surface (right) is considerably smoother and more stable.

To further disentangle the sources of uncertainty, Figure 4.19 compares residual sensor noise against the combined model+sensor uncertainty. Once the GP is sufficiently trained, residual uncertainty is dominated by measurement noise rather than modeling error. This distinction is critical: allocating torque in regions with large epistemic uncertainty (poorly explored states) may lead to systematically biased or unsafe control, while residual sensor noise mainly adds variance around otherwise reliable predictions.

To manage predictive uncertainty in control, two cost formulations were evaluated within the GP controller:

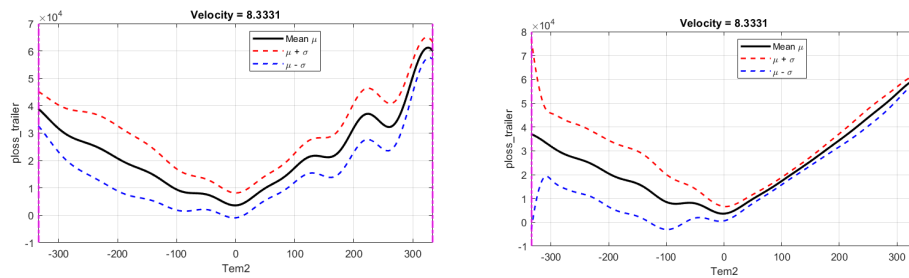


Figure 4.18: Evolution of the learned power loss surface in the positive torque region. Left: high uncertainty during early learning. Right: reduced uncertainty and smoother gradients after sufficient exploration.

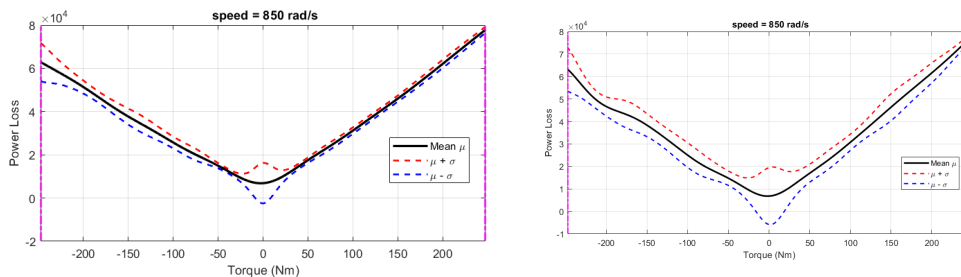


Figure 4.19: Uncertainty decomposition. Left: contribution from sensor noise only. Right: total predictive uncertainty (model + sensor). After convergence, residual uncertainty is dominated by sensors.

1. Mean-based: $J(T) = \mu(T)$, optimizing only the predicted mean loss.
2. Risk-aware: $J(T) = \mu(T) + \beta\sigma(T)$, penalizing allocations with high predictive variance.

The risk-aware formulation discourages torque splits in regions of poor model confidence, effectively trading marginal efficiency for robustness. Table 4.14 shows that this design choice incurred only a minor overhead in energy consumption, while ensuring safer and more reliable decisions in uncertain operating regimes.

Table 4.14: Energy consumption: mean-based vs risk-aware GP control.

Drive Cycle	Mean-based GP (kJ)	Risk-aware GP (kJ)	Ratio (Risk/Mean)
WHVC	138.92	139.22	100.2%
Göteborg–Borås	391.40	393.31	100.5%

This result is consistent with the broader literature on cautious learning-based MPC. Hewing et al. [17] show that penalizing predictive variance yields

more reliable closed-loop behavior in GP-MPC. Similarly, McKinnon et al. [44] demonstrate how explicitly shaping the cost to include model error avoids trajectories prone to unexpected losses. By incorporating $\sigma(T)$ into the cost, the present framework achieves the same principled balance: avoiding over-optimistic exploitation of poorly explored regions, while still enabling efficient operation in regions well-supported by data.

Synthesis. The analysis demonstrates that while mean-based optimization maximizes short-term efficiency, the risk-aware formulation provides a more robust control policy under uncertainty. Importantly, the CBF occasionally rejects torque allocations that could have been more energy-efficient but unsafe, underscoring that safety takes precedence over marginal efficiency gains. Overall, the GP + SQP + CBF strategy achieves a balanced compromise: near-optimal energy efficiency combined with consistent safety guarantees and robustness to uncertainty, outperforming the baseline RLS + QP approach in both stability and adaptability.

Chapter 5

Conclusion and Future work

The results presented in this thesis demonstrate the feasibility of learning trailer-specific power loss characteristics in real time using only limited sensor data, and their subsequent exploitation for energy-efficient torque coordination. Two complementary approaches were investigated: Recursive Least Squares as a lightweight parametric baseline and Gaussian Process regression as a non-parametric, uncertainty-aware method.

The learning models comparison between RLS and GP highlights the classical trade-off between computational efficiency and modeling flexibility. RLS offers fast updates and interpretability, but its polynomial structure restricts its ability to capture complex loss surfaces, particularly for induction motor configurations analyzed in this thesis. In contrast, GP regression adapts flexibly to nonlinear behaviors and measurement noise, yielding superior accuracy across the operational envelope. These findings are consistent with prior work showing the effectiveness of GPs in data-efficient regression and control [24, 17].

A distinctive contribution of this thesis is the extension of GP regression to explicitly handle heteroscedastic sensor noise, a feature rarely supported in standard toolboxes [29, 28]. By designing a custom two-stage GP with a composite kernel, both epistemic (model) and aleatoric (sensor) uncertainties were represented, improving calibration and ensuring reliable safety-aware exploration. This extension addresses a common limitation in online learning for cyber-physical systems, where noise properties vary across operating conditions [36]. The main drawback of GP regression remains its cubic computational complexity in the dataset size, $\mathcal{O}(n^3)$. Sparse approximations such as FITC [27] were tested, but accuracy degradation and instability limited their suitability for safety-critical control. Although this limitation does not undermine the exploratory, simulation-based nature of this work, it highlights the importance of efficient sparse approximations for real-world deployment [34].

From a vehicle-level perspective, the learned models translated into measurable

efficiency improvements. Post-learning evaluations showed energy savings of 2–4% compared to a rule-based baseline, with GP-based allocation operating within 1% of the oracle case with full trailer knowledge. These results strength earlier findings that uncertainty-aware machine learning models can approach the performance of model-based controllers even under partial knowledge [17, 18].

Importantly, the results also underline the cost of exploration. During short trips, the initial exploration phase can lead to slightly higher consumption relative to the baseline. This is a well-known property of online learning methods [15, 14]: information gathering incurs a temporary cost that is only amortized over longer driving horizons. The trade-off resembles the classical exploration–exploitation dilemma in reinforcement learning, here manifested in a real-world energy efficiency context. In practice, this implies that the framework is best suited for long-haul applications where sufficient operation time justifies the transient overhead.

Safety was ensured through a two-layer mechanism: rollout-based motion prediction during exploration and a CBF constraint embedded in the SQP formulation during exploitation. The predictive CBF formulation [21] extended safety guarantees from instantaneous enforcement to finite-horizon trajectories, effectively preventing jackknifing and trailer swing. This aligns with recent advances in safety filters for learning-based control [20, 23], while adapting them to the specific dynamics of articulated vehicles. Notably, the safety layer occasionally rejected torque allocations that might have yielded marginal energy savings. This reflects a fundamental trade-off: maximizing efficiency cannot come at the expense of vehicle stability. Similar efficiency–safety compromises have been observed in GP-MPC studies [17], confirming that robust safety mechanisms inevitably reduce the achievable optimum, but ensure reliability under uncertainty.

5.1 Contributions to Literature

Beyond empirical performance, several contributions emerge:

Framework design: The modular integration of learning, exploration, safety simulation, and control allocation provides a blueprint for safe online adaptation in heavy-duty transport.

Heteroscedastic Gaussian Processes: A novel kernel-based extension enabled input-dependent noise modeling, bridging a gap in practical GP implementations on MATLAB, given that both `fitrgp` and `GPML Toolbox` lack of this feature [29, 28].

Finite-horizon CBFs for non-affine systems: By embedding finite horizon CBFs in SQP optimization, this thesis extended safety enforcement to nonlinear articulated dynamics, complementing earlier control-affine formulations [21, 22].

Application to e-trailer torque coordination: While prior work on torque allocation assumed full model knowledge [5, 6], this thesis demonstrates that

comparable performance can be achieved under strict data limitations through online learning.

5.2 Future Work

Several limitations should be acknowledged. First, the evaluated drive cycles do not provide steering angle information by default. In this work, steering profiles were artificially constructed based on the shape of the drive cycle and prior analysis, which limits the realism of the safety evaluation. While this allowed stress-testing of the rollout and CBF filters, future validation with measured steering profiles in real-world drive cycles is necessary to fully assess safety under realistic articulated dynamics. Second, while GP regression achieved high accuracy, its scalability remains a bottleneck for future real-time use. Recent work on large-scale exact and approximate GPs [34] could be leveraged to address this. Third, the safety framework relies on non-affine dynamics; future work could explore affine approximations with perturbation methods to simplify optimization [23].

Looking ahead, several directions are promising. Stochastic MPC could be integrated as an alternative to SQP, offering a natural way to encode uncertainty and constraints in a receding horizon framework [17]. The framework could also be extended to learn higher-level trailer parameters such as mass or length, which directly affect dynamics and power loss [7]. Finally, real-time adaptation under hardware constraints represents a crucial step toward deployment.

In conclusion, this thesis shows that safe and efficient online learning for articulated electric vehicles is achievable through Gaussian Processes enriched with uncertainty modeling, guided exploration, and predictive safety constraints. While computational challenges and real-world validation remain open, the presented framework advances the state of the art in data-driven torque coordination. By combining learning, exploration, safety, and optimization in a modular structure, it bridges the gap between theoretical machine learning tools and the operational requirements of safety-critical transport systems.

Appendix A

Force Request Computation

The velocity and acceleration data from these drive cycles are used to determine the actual force demand on the truck and trailer combination. This is done using the following set of physics-based equations:

$$\begin{aligned}F_{\text{mass}} &= m \cdot a(t) \\F_{\text{roll}} &= m \cdot C_{\text{roll}} \cdot g \cdot \cos(\alpha_{\text{slope}}) \\F_{\text{air drag}} &= V(t)^2 \cdot \rho \cdot A \cdot C_d \cdot 0.5 \\F_{\text{slope}} &= m \cdot g \cdot \sin(\alpha_{\text{slope}}) \\F_{x,\text{req}} &= F_{\text{mass}} + F_{\text{roll}} + F_{\text{air drag}} + F_{\text{slope}}\end{aligned}$$

Where:

- m : Mass of the vehicle
- $a(t)$: Instantaneous acceleration
- C_{roll} : Rolling resistance coefficient
- g : Gravitational acceleration
- α_{slope} : Road slope angle
- $V(t)$: Instantaneous vehicle speed
- ρ : Air density
- A : Frontal area of the vehicle
- C_d : Aerodynamic drag coefficient

This model captures the total longitudinal force ($F_{x,\text{req}}$) required to follow the drive cycle velocity profile by accounting for inertial, rolling, aerodynamic, and slope-related resistances.

Appendix B

Heteroscedastic noise

In real-world applications involving physical systems, measurement noise is rarely uniform across the input space. To capture this input-dependent variability—referred to as heteroscedastic noise this work employs a two-step Gaussian Process (GP) approach that explicitly models both epistemic and aleatoric uncertainty.

B.1 Composite Kernel Formulation

The heteroscedastic noise is incorporated into the GP by modifying the kernel to account for input-dependent variance. The resulting composite kernel is defined as:

$$k_{\text{obs}}(x_i, x_j) = k_{\text{SE}}(x_i, x_j) + \delta_{ij}\sigma_n^2(x_i) \quad (\text{B.1})$$

where:

- $k_{\text{SE}}(x_i, x_j)$ is the squared exponential kernel,
- δ_{ij} is the Kronecker delta (i.e., 1 if $i = j$, 0 otherwise),
- $\sigma_n^2(x_i)$ is the input-dependent noise variance at training point x_i .

Given this structure, the GP prior becomes:

$$y(x) \sim \mathcal{GP}(0, k_{\text{obs}}(x, x')) \quad (\text{B.2})$$

Posterior Prediction

The predictive mean and variance for a test input x_* are given by:

$$\mu_f(x_*) = k_*^\top K^{-1} y \quad (\text{B.3})$$

$$\sigma_f^2(x_*) = k_{\text{SE}}(x_*, x_*) - k_*^\top K^{-1} k_* \quad (\text{B.4})$$

where:

- k_* is the kernel vector between x_* and training inputs,
- $K = K_{\text{SE}} + D$ is the covariance matrix with noise added on the diagonal,
- $D = \text{diag}(\sigma_n^2(x_1), \dots, \sigma_n^2(x_n))$.

Since noise is handled directly through D , the likelihood noise term is set to zero during training.

B.2 Modeling the Noise Variance

To estimate the input-dependent noise $\sigma_n^2(x)$, a second Gaussian Process is trained on the log-variance values. Let:

$$z_i = \log \sigma_n^2(x_i) \quad (\text{B.5})$$

$$z(x) \sim \mathcal{GP}(\mu_{\log}(x), k_{\log}(x, x')) \quad (\text{B.6})$$

The estimated aleatoric variance at a test input x_* is then computed as:

$$\hat{\sigma}_n^2(x_*) = \exp(\mu_{\log}(x_*)) \quad (\text{B.7})$$

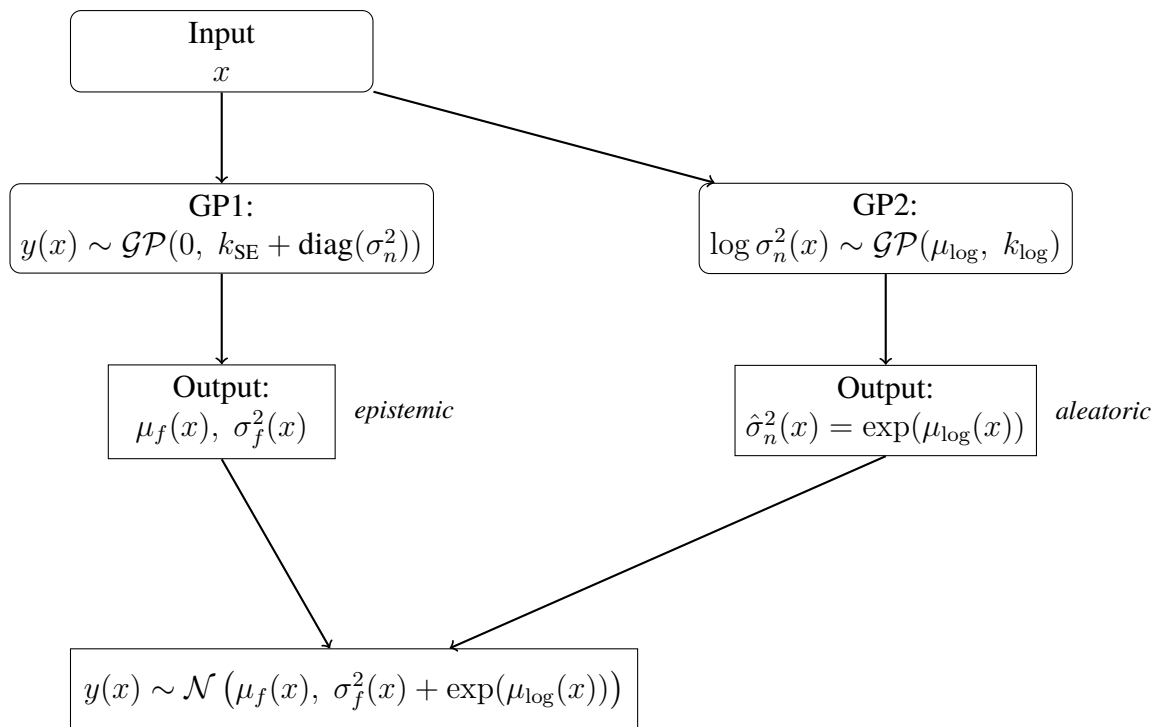


Figure B.1: Vertical two-GP architecture. The first GP models the observation function with known input-dependent noise embedded in the kernel. The second GP learns the noise structure as a smooth log-variance field. Their outputs are summed to compute the final predictive variance.

Appendix C

Induction point selection

The quality and placement of inducing points are critical for preserving model accuracy while reducing computational cost [45]. Several strategies were explored:

- Random Sampling: Provides a fast baseline but often results in poor coverage.
- k-means Clustering: Clusters the input space and places inducing points at the cluster centers. This provides good coverage in dense areas but may overrepresent high-density regions and is computationally expensive [46].
- FPS (Furthest Point Sampling): Selects points that maximize coverage by placing each new point as far as possible from existing ones. Better suited for uniform coverage.
- Hybrid Method (30% FPS / 70% k-means): Combines the strengths of k-means (data-adaptive) and FPS (coverage), mitigating the weaknesses of each.

Appendix D

Dynamic Vehicle Model

The motion prediction module relies on a detailed dynamic model of the truck–trailer system, extended with articulation and nonlinear tire slip behavior. The equations form a Differential-Algebraic Equation (DAE) system composed of 24 equations. This model, implemented in Simscape and exported as an FMU, is used during each control cycle to simulate and validate the safety of torque commands.

D.1 Equations of the vehicle

Let the state vector \mathbf{x} contain the following core dynamic states:

$$\mathbf{x} = [v_{1x}, v_{1y}, \omega_{1z}, \omega_{2z}, \theta]^T$$

Let the known control inputs be:

$$\mathbf{u} = [\delta_f, F_{1fxw}, F_{1rx}, F_{2x}]^T$$

D.1.1 Equations of Motion of the Truck

Equilibrium Equations:

$$\begin{aligned} m_1 (\dot{v}_{1x} - \omega_{1z} v_{1y}) &= \cos(\delta_f) F_{1fxw} - \sin(\delta_f) F_{1fyw} + F_{1rx} + P_{1x} \\ m_1 (\dot{v}_{1y} + \omega_{1z} v_{1x}) &= \sin(\delta_f) F_{1fxw} + \cos(\delta_f) F_{1fyw} + F_{1ry} + P_{1y} \\ J_1 \dot{\omega}_{1z} &= (\sin(\delta_f) F_{1fxw} + \cos(\delta_f) F_{1fyw}) l_{CoG} - F_{1ry} (L_1 - l_{CoG}) \\ &\quad + (-P_{1y}) (l_c - l_{CoG}) \end{aligned}$$

Kinematic Compatibility:

$$\begin{aligned}v_{1fyv} &= v_{1y} + \omega_{1z} \cdot l_{CoG} \\v_{1ry} &= v_{1y} - \omega_{1z} \cdot (L_1 - l_{CoG}) \\v_{1cy} &= v_{1y} - \omega_{1z} \cdot (l_c - l_{CoG})\end{aligned}$$

Velocity Components:

$$\begin{aligned}v_{1fxw} &= \cos(\delta_f)v_{1x} + \sin(\delta_f)v_{1fyv} \\v_{1fyw} &= -\sin(\delta_f)v_{1x} + \cos(\delta_f)v_{1fyv}\end{aligned}$$

Slip and Force Models:

$$\begin{aligned}s_{1fy} &= \frac{v_{1fyw}}{|v_{1fxw}|}, & F_{1fyw} &= -C_{1f}s_{1fy} \\s_{1ry} &= \frac{v_{1ry}}{|v_{1x}|}, & F_{1ry} &= -C_{1r}s_{1ry}\end{aligned}$$

D.1.2 Equations of Motion of the Trailer**Equilibrium Equations:**

$$\begin{aligned}m_2 (\dot{v}_{2x} - \omega_{2z}v_{2y}) &= F_{2x} + P_{2x} \\m_2 (\dot{v}_{2y} + \omega_{2z}v_{2x}) &= F_{2y} + P_{2y} \\J_2\dot{\omega}_{2z} &= -F_{2y}l_{CoG} + P_{2y}(l_c - l_{CoG})\end{aligned}$$

Kinematic Compatibility:

$$\begin{aligned}v_{2cy} &= v_{2y} + \omega_{2z}(l_c - l_{CoG}) \\v_{2ay} &= v_{2y} - \omega_{2z}l_{CoG}\end{aligned}$$

Slip and Force Model:

$$s_{2y} = \frac{v_{2ay}}{|v_{2x}|}, \quad F_{2y} = -C_2s_{2y}$$

D.1.3 Coupling Equations**Equilibrium in 1st Unit Coordinates:**

$$\begin{aligned}P_{1x} + \cos(\theta)P_{2x} + \sin(\theta)P_{2y} &= 0 \\P_{1y} - \sin(\theta)P_{2x} + \cos(\theta)P_{2y} &= 0\end{aligned}$$

Coupling Compatibility:

$$\begin{aligned}v_{1x} &= \cos(\theta)v_{2x} + \sin(\theta)v_{2cy} \\v_{1cy} &= -\sin(\theta)v_{2x} + \cos(\theta)v_{2cy} \\ \dot{\theta} &= \omega_{1z} - \omega_{2z}\end{aligned}$$

D.1.4 Model Summary

This system is implemented as a DAE model using MATLAB's `ode15i` solver to simulate trajectory predictions over a 3-second horizon. It enables evaluation of safety constraints such as jackknifing and trailer swing.

Assumptions:

- Road grip is assumed constant: $\mu = 0.8$
- Braking occurs passively via torque sign reversal (no ABS or EBS modeled)
- Steering angle δ_f is predefined from the drive cycle
- Articulation is rigid; no backlash or compliance modeled

Appendix E

Gaussian Process Derivatives

Let $f(x) \sim \mathcal{GP}(m(x), k(x, x'))$ with a differentiable kernel k . Then the derivative process $\frac{\partial f}{\partial x_j}(x)$ is also a GP with mean and covariance obtained by differentiating the kernel.

E.1 Joint distribution with derivatives

For any test inputs x, x' , we have

$$\text{cov}\left(f(x), \frac{\partial f}{\partial x_j}(x')\right) = \frac{\partial}{\partial x'_j} k(x, x'), \quad (\text{E.1})$$

$$\text{cov}\left(\frac{\partial f}{\partial x_i}(x), \frac{\partial f}{\partial x_j}(x')\right) = \frac{\partial^2}{\partial x_i \partial x'_j} k(x, x'). \quad (\text{E.2})$$

Thus, if we augment the training set with derivatives, the joint GP prior remains Gaussian, with additional blocks defined by the differentiated kernels.

E.2 Squared Exponential kernel

For the isotropic SE kernel

$$k(x, x') = \sigma_f^2 \exp\left(-\frac{\|x - x'\|^2}{2\ell^2}\right), \quad (\text{E.3})$$

the cross-covariance between a function value and a derivative is

$$\text{cov}\left(f(x), \frac{\partial f}{\partial x_j}(x')\right) = \frac{(x_j - x'_j)}{\ell^2} k(x, x'), \quad (\text{E.4})$$

and the covariance between two partial derivatives is

$$\text{cov}\left(\frac{\partial f}{\partial x_i}(x), \frac{\partial f}{\partial x_j}(x')\right) = \left[\frac{\delta_{ij}}{\ell^2} - \frac{(x_i - x'_i)(x_j - x'_j)}{\ell^4}\right] k(x, x'), \quad (\text{E.5})$$

where δ_{ij} is the Kronecker delta. Analogous formulas hold for other differentiable kernels such as Matérn- ν .

E.3 Use in control

These results imply that gradient information can be computed in closed form from the kernel, without numerical differentiation. In optimization-based control, this allows the GP surrogate to be embedded seamlessly in solvers like SQP that rely on analytic derivatives.

Appendix F

Solver Implementation

The optimization was carried out using MATLAB's `fmincon` function configured with the `'sqp'` algorithm. The nonlinear constraint $T_{\text{truck}} \cdot T_{\text{trailer}} \geq 0$ and the CBF constraints were implemented via a custom function passed to `nonlcon`, while box constraints were applied directly through lower and upper bounds. The solver configuration included:

- `TolFun` = 10^{-6} ,
- `TolCon` = 10^{-6} ,
- `MaxIterations` = 100.

References

- [1] M. Surcel, S. Mercier, and A. Bonsi, “Energy efficiency evaluation of a hybrid electric tractor-semi-trailer prototype,” SAE International, SAE Technical Paper 2024-01-4319, 2024. [Online]. Available: <https://doi.org/10.4271/2024-01-4319> [Pages 1 and 2.]
- [2] Volvo Trucks. (2025) Our electric truck range. [Online]. Available: <https://www.volvotrucks.com/en-en/trucks/electric.html> [Page 2.]
- [3] Z. Gao, “Exploring class 8 long-haul truck electrification: Key technology evaluation and potential challenges,” SAE International, SAE Technical Paper 2024-01-2812, April 2024, presented at WCX SAE World Congress Experience. [Online]. Available: <https://doi.org/10.4271/2024-01-2812> [Page 2.]
- [4] TruckingInfo.com. (2023) Mesilla valley says range energy trailers deliver 36% fuel economy boost. [Online]. Available: <https://www.truckinginfo.com/10210920/mesilla-valley-says-range-energy-trailers-deliver-36-fuel-economy-boost> [Page 2.]
- [5] H. Na, S.-H. Hwang, K. Cho, Y. J. Chang, and S.-H. You, “Optimal torque split strategies for dual motor ev,” in *Proceedings of the 33rd Electric Vehicle Symposium (EVS33)*. Portland, Oregon, USA: Electric Vehicle Symposium, June 2020, presented at EVS33, June 14–17, 2020. [Pages 2, 7, 13, and 80.]
- [6] J. Torinsson, M. Jonasson, D. Yang, and B. Jacobson, “Energy reduction by power loss minimisation through wheel torque allocation in electric vehicles: A simulation-based approach,” *Vehicle System Dynamics*, vol. 60, no. 5, pp. 1488–1511, 2020. doi: 10.1080/00423114.2020.1858121. [Online]. Available: <https://doi.org/10.1080/00423114.2020.1858121> [Pages 2, 3, 7, 13, and 80.]
- [7] S. Janardhanan, E. Gelso, L. Laine, M. Jonasson, and B. Jacobson, “Reviewing control allocation using quadratic programming for motion control and power coordination of battery electric vehicles,” in *2022*

- IEEE Vehicle Power and Propulsion Conference (VPPC)*, November 2022. doi: 10.1109/VPPC55846.2022.10003297 pp. 1–8. [Online]. Available: <https://doi.org/10.1109/VPPC55846.2022.10003297> [Pages 3, 13, and 81.]
- [8] U. Erdinc, O. A. S. Ali, J. Tian, E. Gelso, and M. S. Kati, “Learning Efficiency Maps of Electric Trailers for Power Loss Minimization,” in *Proceedings of the 29th IAVSD International Symposium on Dynamics of Vehicles on Roads and Tracks*, Shanghai, China, Aug. 2025, august 2025. [Online]. Available: https://iavsd2025.tongji.edu.cn/symposium_program/iavsd2025Programme.pdf [Pages 3, 5, 7, and 30.]
- [9] M. Law. (2022, Oct.) What are jackknife accidents caused by? [Accessed: Sep. 16, 2025]. [Online]. Available: <https://www.montagnalaw.com/blog/jackknife-accident-causes/> [Page 4.]
- [10] G. R. Lencione and F. J. V. Zuben, “Online multi-task learning with recursive least squares and recursive kernel methods,” 2024, arXiv preprint arXiv:2308.01938. [Online]. Available: <https://arxiv.org/abs/2308.01938> [Pages 5, 7, and 14.]
- [11] J. Wang, P. Filippi, S. Haan, L. Pozza, B. Whelan, and T. F. Bishop, “Gaussian process regression for three-dimensional soil mapping over multiple spatial supports,” *Geoderma*, vol. 446, p. 116899, 2024. doi: <https://doi.org/10.1016/j.geoderma.2024.116899>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0016706124001289> [Pages 5 and 7.]
- [12] L. Deng, W. Li, and Y. Pan, “Data-efficient gaussian process online learning for adaptive control of multi-dof robotic arms,” *IFAC-PapersOnLine*, vol. 55, no. 2, pp. 84–89, 2022. doi: <https://doi.org/10.1016/j.ifacol.2022.04.174> 14th IFAC Workshop on Intelligent Manufacturing Systems IMS 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S240589632201756> [Pages 6 and 7.]
- [13] G. P. Kontoudis and M. Otte, “Adaptive exploration-exploitation active learning of gaussian processes,” in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Detroit, MI, USA, 2023. doi: 10.1109/IROS55552.2023.10342130 pp. 9448–9455. [Online]. Available: <https://doi.org/10.1109/IROS55552.2023.10342130> [Page 6.]
- [14] E. Contal, D. Buffoni, A. Robicquet, and N. Vayatis, *Parallel Gaussian Process Optimization with Upper Confidence Bound and Pure Exploration*. Springer Berlin Heidelberg, 2013, p. 225–240. ISBN 9783642387098.

- [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40988-2_15 [Pages 6, 7, 20, and 80.]
- [15] L. Papenmeier, N. Cheng, S. Becker, and L. Nardi, “Exploring exploration in bayesian optimization,” 2025. [Online]. Available: <https://arxiv.org/abs/2502.08208> [Pages 6, 7, 20, 37, and 80.]
- [16] S. Janardhanan, L. Henderson, M. Jonasson, B. Jacobson, and E. R. Gelso, “Simulation-based assessment of wheel torque allocation strategies on heavy vehicles with drivetrains on multiple axles,” *Transportation Engineering*, vol. 20, p. 100322, 2025. doi: <https://doi.org/10.1016/j.treng.2025.100322>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2666691X25000223> [Pages 7 and 54.]
- [17] L. Hewing, J. Kabzan, and M. N. Zeilinger, “Cautious model predictive control using gaussian process regression,” *IEEE Transactions on Control Systems Technology*, 2020, extended from arXiv preprint 2017. [Pages 7, 19, 76, 79, 80, and 81.]
- [18] J. Maritz, F. Lubbe, and L. Lagrange, “A practical guide to gaussian process regression for energy measurement and verification within the bayesian framework,” *Energies*, vol. 11, no. 4, 2018. doi: 10.3390/en11040935. [Online]. Available: <https://www.mdpi.com/1996-1073/11/4/935> [Pages 7, 19, and 80.]
- [19] A. Candelieri, “Mastering the exploration-exploitation trade-off in bayesian optimization,” 2023. [Online]. Available: <https://arxiv.org/abs/2305.08624> [Pages 7 and 19.]
- [20] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control barrier functions: Theory and applications,” 2019. [Online]. Available: <https://arxiv.org/abs/1903.11199> [Pages 8, 24, and 80.]
- [21] K. P. Wabersich and M. N. Zeilinger, “Predictive control barrier functions: Enhanced safety mechanisms for learning-based control,” arXiv preprint arXiv:2105.10241, 2022, available: <https://arxiv.org/abs/2105.10241>. [Pages 8, 24, 26, 46, and 80.]
- [22] A. Agrawal and K. Sreenath, “Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation,” 07 2017. doi: 10.15607/RSS.2017.XIII.073 [Pages 8, 25, and 80.]
- [23] Y. Xiong, D.-H. Zhai, M. Tavakoli, and Y. Xia, “Discrete-time control barrier function: High-order case and adaptive case,” *IEEE Transactions on Cybernet-*

- ics*, vol. 53, no. 5, pp. 3231–3239, 2023. doi: 10.1109/TCYB.2022.3170607 [Pages 8, 26, 46, 80, and 81.]
- [24] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006. ISBN 026218253X © 2006 Massachusetts Institute of Technology. [Online]. Available: <http://www.GaussianProcess.org/gpml> [Pages 15, 17, 18, 19, and 79.]
- [25] H. Wang and X. Zhou, “Explicit estimation of derivatives from data and differential equations by gaussian process regression,” 2020. [Online]. Available: <https://arxiv.org/abs/2004.05796> [Page 16.]
- [26] T. Galy-Fajou and M. Opper, “Adaptive inducing points selection for gaussian processes,” 2021. [Online]. Available: <https://arxiv.org/abs/2107.10066> [Page 18.]
- [27] H. Bijl, J. W. Wingerden, T. Schön, and M. Verhaegen, “Online sparse gaussian process regression using fitc and pitc approximations,” vol. 48, 12 2015. doi: 10.1016/j.ifacol.2015.12.212 pp. 703–708. [Pages 18, 19, and 79.]
- [28] M. Lázaro-Gredilla and M. Titsias, “Variational heteroscedastic gaussian process regression,” in *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011, pp. 841–848. [Pages 18, 19, 79, and 80.]
- [29] K. Kersting, C. Plagemann, P. Pfaff, and W. Burgard, “Most-likely heteroscedastic gaussian process regression,” vol. 227, 06 2007. doi: 10.1145/1273496.1273546 pp. 393–400. [Pages 18, 19, 33, 79, and 80.]
- [30] E. Brochu, V. M. Cora, and N. de Freitas, “A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning,” *ArXiv*, vol. abs/1012.2599, 2010. [Online]. Available: <https://api.semanticscholar.org/CorpusID:1640103> [Page 21.]
- [31] F. E. Curtis, T. Mitchell, and M. L. Overton, “A bfgs-sqp method for nonsmooth, nonconvex, constrained optimization and its evaluation using relative minimization profiles,” *Optimization Methods and Software*, vol. 32, no. 1, pp. 148–181, 2017. doi: 10.1080/10556788.2016.1208749 [Page 22.]
- [32] The MathWorks, Inc., “Constrained nonlinear optimization algorithms,” <https://it.mathworks.com/help/optim/ug/constrained-nonlinear-optimization-algorithms.html>, 2025, accessed: 22-Aug-2025. [Page 22.]

- [33] H. N. and P. G., “A brief study of deep reinforcement learning with epsilon-greedy exploration,” *International Journal of Computing and Digital Systems*, vol. 11, pp. 541–551, Jan. 2022. doi: 10.12785/ijcds/110144 [Page 28.]
- [34] K. A. Wang, G. Pleiss, J. R. Gardner, S. Tyree, K. Q. Weinberger, and A. G. Wilson, “Exact gaussian processes on a million data points,” *arXiv preprint arXiv:1903.08114*, 2019. [Online]. Available: <https://arxiv.org/abs/1903.08114> [Pages 32, 79, and 81.]
- [35] C. E. Rasmussen and H. Nickisch, *GPML Matlab Code*, 2024, accessed: 2025-07-11. [Online]. Available: <https://gaussianprocess.org/gpml/code/matlab/doc/> [Pages 32 and 51.]
- [36] Z. B. Patel, N. Batra, and K. Murphy, “Uncertainty disentanglement with non-stationary heteroscedastic gaussian processes for active learning,” *arXiv preprint arXiv:2210.10964*, 2022. [Online]. Available: <https://arxiv.org/abs/2210.10964> [Pages 33 and 79.]
- [37] B. Jacobson *et al.*, “Vehicle motion engineering,” Göteborg, Sweden, Oct. 2024, latest draft. [Online]. Available: <https://www.dropbox.com/scl/fi/76jtg7b3erlkylhf0p7srw/VehicleMotionEngineering.pdf?rlkey=hv3wntgs7kops474wsgb1g3c&dl=0> [Pages 38 and 39.]
- [38] T. Ghandriz, B. Jacobson, P. Nilsson, L. Laine, and N. Fröjd, “Computationally efficient nonlinear one- and two-track models for multitrailer road vehicles,” *IEEE Access*, vol. 8, pp. 203 854–203 875, 2020. doi: 10.1109/ACCESS.2020.3037035 [Pages 40 and 52.]
- [39] E. Andersson and A. Hansson, “Safe operating envelope for electric semi-trailer: On safe driving conditions with electric trailer propulsion for heavy truck combinations using coordinated vehicle motion control,” Master’s thesis, Chalmers University of Technology, Gothenburg, Sweden, 2022. [Online]. Available: <https://www.chalmers.se> [Pages 41 and 46.]
- [40] A. Safari and J. Hoagg, “Safe navigation in unmapped environments for robotic systems with input constraints,” 10 2024. [Page 46.]
- [41] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004. [Page 46.]
- [42] MathWorks, *fitrgp*, 2024, accessed: 2025-07-11. [Online]. Available: <https://it.mathworks.com/help/stats/fitrgp.html> [Page 51.]

- [43] C. Lyu, X. Liu, and L. Mihaylova, “Review of recent advances in gaussian process regression methods,” 2024. [Online]. Available: <https://arxiv.org/abs/2409.08112> [Page 57.]
- [44] C. D. McKinnon and A. P. Schoellig, “Context-aware cost shaping to reduce the impact of model error in receding horizon control,” in *Robotics: Science and Systems (RSS)*, 2020. [Page 77.]
- [45] T. Galy-Fajou and M. Opper, “Adaptive inducing points selection for gaussian processes,” *arXiv preprint arXiv:2107.10066*, 2021. [Online]. Available: <https://arxiv.org/abs/2107.10066> [Page 88.]
- [46] D. Arthur and S. Vassilvitskii, “K-means++: The advantages of careful seeding,” in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2007. doi: 10.1145/1283383.1283494 pp. 1027–1035. [Page 88.]

TRITA-EECS-EX-2025:950
Stockholm, Sverige 2024

www.kth.se