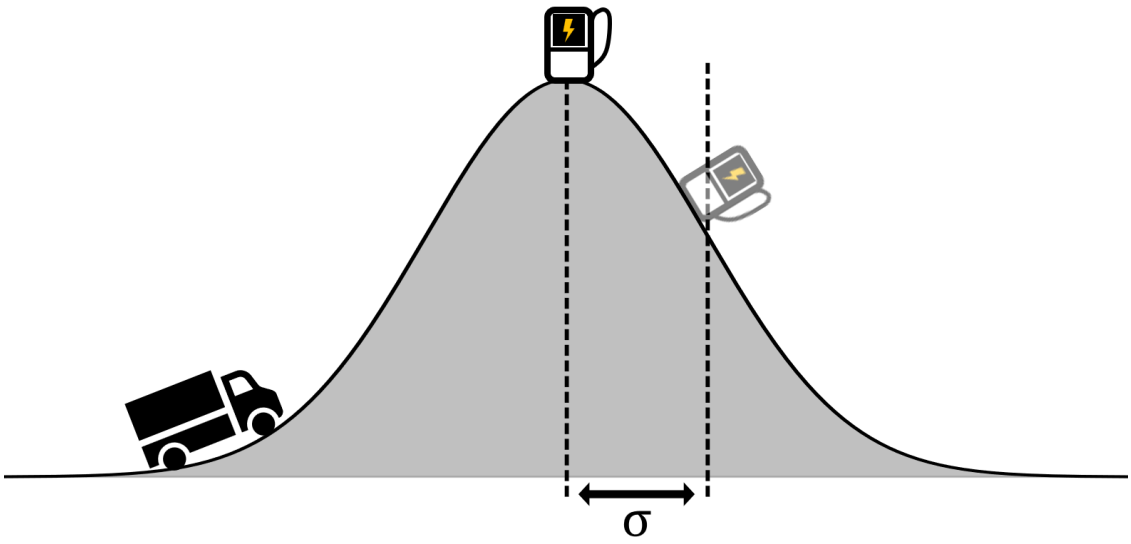
CHALMERS
UNIVERSITY OF TECHNOLOGY



Stochastic Charge Planning with Dynamic Programming

Master's thesis in Systems, Control and Mechatronics

OSKAR HÖGLUND
FILIP SANDSTRÖM

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

Master's thesis 2026

Stochastic Charge Planning with Dynamic Programming

OSKAR HÖGLUND
FILIP SANDSTRÖM

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
Gothenburg, Sweden 2026

Stochastic Charge Planning with Dynamic Programming

OSKAR HÖGLUND, FILIP SANDSTRÖM

Supervisor: Mohamed Abrash, Department of Electrical Engineering, Chalmers
Supervisor: Fatemeh Mohammadi, Volvo Group
Examiner: Nikolce Murgovski, Department of Electrical Engineering, Chalmers

Master's Thesis 2026
Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Conceptual truck driving on a normal distribution to the charging station located at mean, with a standard deviation denoted.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2026

Stochastic Charge Planning with Dynamic Programming

OSKAR HÖGLUND and FILIP SANDSTRÖM
Department of Electrical Engineering
Chalmers University of Technology

Abstract

The development of charge planning algorithms which extend further than those considering uncorrelated disturbance models and produce robust policies is an important subject. The freight sector is moving towards battery electric trucks where uncertainties can have a major impact on missions due to state of charge constraints. Therefore, this thesis investigates dynamic programming algorithms for use in charge planning. Disturbances are modeled as Gaussian processes, which for certain structures admits an equivalent transformation to an LTI SDE system. Using this transformation, the distribution along state trajectories are estimated using an unscented Kalman filter. The UKF showed good performance for the modeled disturbances, with a largest mean bias of 1.106% in a worst-case scenario. The proposed approximate dynamic based charge planning algorithm became robust under stochastic external uncertainties from wind and traffic by implementing chance constraints. The proposed planning algorithms achieved better performance than both a simpler deterministic dynamic programming algorithm and a simple heuristic planner. Computational complexity remains a key concern for real time implementations and is a crucial challenge when designing stochastic charge planning algorithms.

Keywords: charge planning, dynamic programming, approximate dynamic programming, unscented Kalman filter, state estimation, Gaussian process

Acknowledgements

Firstly, we would like to extend our gratitude to our supervisor at Chalmers, Mohamed Abrash. Without his support, ideas and knowledge, this thesis would not have been possible. His knowledge in the field of stochastic modeling was invaluable in this project. We also extend our gratitude and thanks to our supervisor at Volvo Group, Fatemeh Mohammadi, for her great feedback, ideas and support throughout the project. Her insight and experience in charge planning was a great resource to have in this project. We would also like to thank our examiner at Chalmers, Nikolce Murgovski. His ideas and support was a great source of inspiration throughout the project. We also want to thank Olof Lindgärde and the rest of the team at Volvo for their insight and perspectives. Finally, we want to thank the other master thesis students at Volvo Trucks for providing a friendly face to chat with during lunch.

Oskar Höglund and Filip Sandström, Gothenburg, June 2026

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

ADP	Approximate dynamic programming
BEV	Battery electric vehicle
DP	Dynamic programming
GP	Gaussian process
HDV	Heavy duty vehicle
LTI	Linear time invariant
MC	Monte Carlo (simulation)
PSD	Power spectral density
SDE	Stochastic differential equation
SoC	State of Charge
UKF	Unscented Kalman filter

Nomenclature

Below is the nomenclature of indices, sets, parameters, and variables that have been used throughout this thesis.

Indices

i	Indexing driving dynamics
k	Indexing charging stations
s_i	Location at index i
x_k	State at charging station k
x_k^+	State at charging station k after charging
x_M	State at the end of the route

Sets

$\mathcal{J}$	Set of chargers
$\mathcal{U}$	Permissible inputs
$\mathcal{X}$	Permissible states
X_s	Spatial points
X_t	Temporal points

Parameters

Δs	Space discretisation step
$a_1, a_2, \cdots, a_5$	Electric motor coefficients
c_{charging}	Cost of charging
c_{cut}	Charging power cutoff value
c_{der}	Charging power derating value
$c_{\text{low soc}}$	Cost on low state of charge
c_{time}	Cost of time
$C_d A_f$	Aerodynamic drag coefficient

C_r	Constant frictional coefficient
E_{bat}	Battery capacity
$\ell_s^{(t)}$	Spatial correlation length
$\ell_t^{(t)}$	Temporal correlation length
m	Truck mass
n_{GP}	Number of Gaussian processes in simulation
n_{std}	Number of standard deviations in chance constraint
P_{aux}	Auxiliary electric power
P_{max}	Maximum charging power
P_k	Maximum power of charger k
R_b	Internal battery resistance
ρ	Air density
σ_{traffic}	Standard deviation of traffic speed
σ_{wind}	Standard deviation of wind speed
soc_{min}	Minimum permissible state of charge
t_{deadline}	Maximum permissible arrival time
$t_k^{(\text{detour})}$	Detour driving time at station k
u_{oc}	Battery open terminal voltage

Variables

α	Road incline
a	Acceleration
C	Cumulative energy consumption
d_{charge}	Charging decision
E	Total energy consumption during mission
$f^{(t)}$	Random variable for traffic
$f^{(w)}$	Random variable for wind
M	Motor torque
N_{stop}	Number of stops during mission
P_{load}	Total electrical power load
P_{motor}	Motor power load
π	Heuristic policy
Q_k	Number of charging decisions at station k
soc	State of charge

Δsoc	State of charge increase
t	Time of day
t_{driving}	Driving time between chargers
t_{stop}	Total time for charging stop
T_{compute}	Compute time for simulation and planning
T_{stop}	Total mission time at charging stations
T_{tot}	Total mission time
u	Input
u^*	Optimal input
v	Velocity
v_{wind}	Wind velocity
w	Disturbance state
ω	Motor rotational velocity
$\text{SoC}_{\text{final}}$	State of charge at destination
SoC_{min}	Minimum state of charge during mission

Contents

List of Figures

List of Tables

1

Introduction

Heavy trucks and buses are responsible for 12.8 % of European carbon emissions [1]. Battery electric heavy duty vehicles (HDVs) have between 63 % and 76 % lower emissions per kilometer, when the entire vehicle life cycle is analyzed [2]. This assumes the current EU energy grid mix. If instead the entire grid used renewable energy, the equivalent CO_2 emission per kilometer decrease by 84% and 92% compared to a diesel HDV. Battery electric trucks represent only 2 % of the new truck sales in Europe during 2024 [3]. Comparing the total cost of ownership between diesel trucks and battery electric trucks, the heavy duty segment currently has the biggest cost disadvantage for battery trucks [4]. The authors of [4] also identified other non-monetary costs that hinder the adoption of battery electric HDVs, such as range anxiety and waiting times at charging stations.

Without intelligent charge planning, it can be difficult to accurately predict how a long range mission will take and how much energy will be consumed. This can hinder adoption of battery electric vehicles (BEV) trucks. A good charge planning algorithm can permit the use of smaller margins in the battery, which leads to a more effective use of the available battery capacity. Driving is an inherently stochastic activity. This means that both the energy consumption and the driving time become stochastic. Therefor, we implement a charge planner that can safely integrate uncertainty during planning. We still want the solution to be as close to optimal as possible, so the implementation has to be smart on how it handles uncertainties.

While planning with a large state of charge (SoC) margin is effective at reducing range anxiety and is a simple way to handle unknowns, it disregards knowledge about uncertainties. Because of this, the charge planning algorithm will end up suggesting more stops at charging stations than what would have been required if it did take uncertainties into consideration. In control theory, this could be equivalently compared to a robust solver versus a stochastic solver. One adding state margins, the other using state probabilities.

In this thesis a mathematical formulation for how to model disturbances using a Gaussian process is proposed. This is then implemented in a charge planner based on approximate dynamic programming. Chapter 2 is an introduction to the theoretical frameworks of Gaussian processes, unscented Kalman filters, and dynamic programming. In Chapter 3, the modeling assumption of the truck and route are introduced. Chapter 4 then details how the modeled uncertainties are propagated

forward and Chapter 5 details the charge planner implementations. Chapter 6 is simulation results, followed by a discussion of the result and method in Chapter 7. The final conclusion is found in Chapter 8.

1.1 Literature review

This section provides an overview on existing implementations in the literature. Firstly, algorithms for charge planning are introduced. This is followed by a short introduction to traffic modeling using Gaussian processes.

1.1.1 Charge planning algorithms

A significant constraint on real life trucking is the driver rest time regulations [5]. Based on these rules, [6] implements a charge planner as a mixed integer problem. In order to achieve feasible compute times an implementation of rollout is used. The rollout solution is compared with an exact optimal solution. The performance gap is on average less than 1 %, while maintaining compute times of less than 2 seconds.

One approach to handle uncertainties is to vary the minimum energy buffer, which is implemented by [7]. This paper considers charging a type of shortest path problem. In order to handle stochastic uncertainties, the paper introduces a minimum SoC level soc_{min}, which varies depending on the size of the uncertainty at a specific point along the route.

Wan et. al. [8] propose a solver using a genetic algorithm. This approach considers specific time and place dependent charging costs to minimize total trip cost. Using a similar technique, [9] investigates what the time savings can be when optimizing charging locations. The authors conclude that the effect of optimal charging strategies increase when charging power speeds are reduced. They also point out that accurately modeling SoC levels when driving is difficult and that uncertainties need to be taken into account to achieve robust planning.

Unknown waiting time at charging stations is a factor considered in [10], which uses probability distributions to model varying waiting times. This paper reports a 10 % increase in performance using the proposed mixed integer dynamic programming solver.

This thesis is closely related to two previous theses. Borghed implemented a charge planner based on dynamic programming in [11]. This planner included thermal management of the battery as a state to be optimized, which is not in the scope of this work. The charging problem was reformulated and solved using an external mixed integer non-linear program solver. A similar thesis on fuel cell vehicles [12] also used a two step planning approach to perform planning, where the vehicle speed was optimized on a low level first. Then, charge planning was performed on the fixed but optimized vehicle speed trajectory.

1.1.2 Modeling traffic with a Gaussian process

There exist models for traffic that use a Gaussian process as an underlying framework. In [13], a Gaussian process is used to learn traffic volume using a number of input parameters. It was shown that performance of the model increased when the number of input parameters was increased.

A Gaussian process was used in [14] to model traffic and compared to a conventional traffic model. The error margins for the Gaussian Process model were lower. The authors also highlighted that a standard deviation can easily be obtained from a GP model, which is not easily achievable with some type of models. Similar work was done in [15].

1.2 Purpose

The objective of the thesis is to minimize the total trip operational cost for BEV trucks. In this thesis, the optimization variables considered are where, when, and how much to charge for a given vehicle. Utilizing these decisions the total monetary cost, where travel time and charging costs are converted to Euros, over the entire route is minimized. The route of the journey is assumed to be pre-planned. The goal of this thesis is to develop a mathematical formulation for charge planning while handling process disturbances in a robust manner. The proposed planner is intended to be used in online operation, i.e. during a mission.

The charge planning problem is formulated as a dynamic programming, which is introduced in Section 2.4. This has multiple advantages. Firstly, the dynamic programming framework is flexible and can be used to model many different kinds of problems, such as the kind of mixed integer and non-linear problems that appear in this project. Secondly, there exists a large quantity of existing literature on formulations and strategies used to solve dynamic programming problems.

One challenge with solving any optimization problem is integer decisions, such as charging decisions, which can make the problem of finding minimums difficult or intractable. This master thesis will be exploring how to model disturbances and uncertainties such that they can be included in the trip planning formulation.

1.3 Scope and Limitations

At the end of the project, we intend to deliver a control formulation that can be used to solve the stochastic optimal trip planning problem. We also intend to analyze the effects of disturbances and how different strategies for modeling disturbances influence the results. Finally, we also intend to demonstrate performance evaluation of the proposed algorithm compared to already existing solutions.

This thesis will use simulation to evaluate the developed algorithms, with limited use of real-world data, and no testing using an actual truck will be carried out. Furthermore, this project will not aim to model disturbances that precisely match real-world expectations, but rather explore methods on how to integrate generic

disturbances in charge planning. As such, the solution may not be representative of real-world performance. However, adapting the model should in theory only be a question of parameter tuning, and as such is straightforward.

The route that the truck will take is fixed and will not be changed by our algorithm. This means that the most time or cost efficient routing might not be the shortest or most time efficient route as determined by whatever route planner is used. Integration of these components may be considered in future work, but will not be considered in this project.

This project will focus only on optimizing the decisions and actions of the truck and its driver. We will not consider other outside perspectives during optimization, i.e. charging will not be optimized from the perspective of a fleet of vehicles or the electric grid as a whole.

1.4 Research questions

Based on the purpose, scope, and limitations above, the following research questions are considered in this report

- How can the DP control framework with a mixed discrete-continuous action space, small and continuous disturbances, and large and intermittent disturbances, be formulated for effective disturbance handling?
- How can correlated disturbances be included in state dynamics, while also being able to retrieve the state distributions efficiently?
- How do you balance the tradeoff between solution optimality and computational feasibility when formulating the optimization problem for trip planning?

2

Theory

This chapter introduces the theory used in the remainder of the thesis. First, Gaussian processes are introduced, including sampling and conditioning on observations. Then, the Unscented Kalman Filter (UKF) is introduced as a Bayesian estimator. This is followed by a mathematical identity of a Gaussian process and a linear time invariant (LTI) stochastic differential equation (SDE), and how this permits the Bayesian estimation of a Gaussian process using a UKF. Finally, a short theoretical introduction to dynamic programming is given.

2.1 Gaussian processes

A Gaussian process (GP) is a distribution over functions, defined such that the function values of any finite collection of input points are jointly Gaussian distributed. This joint distribution is fully characterized by a mean function and a covariance function. The latter is parameterized by a small set of tunable parameters that control properties of the functions drawn from the distribution, such as their characteristic length scale and marginal variance. The GP therefore does not model a single function, but rather encodes prior beliefs about the space of plausible functions through these parameters. Given observations of the function at a set of input points, it is possible to condition this prior distribution on data, yielding a posterior distribution that updates the prior beliefs in a principled way while accounting for observation noise. Below is a light introduction to the GP framework, where details that are not used have been omitted. For a thorough treatment of the GP framework, see [16].

2.1.1 Definition of mean and kernel functions

A Gaussian process is an infinite collection of random variables, completely defined by a mean and covariance function, and any finite subset of which has a joint Gaussian distribution. For two inputs x, x' we define the mean function $m(x)$ and the covariance function, or more commonly known as the kernel function, $k(x, x', \theta)$ with hyperparameters θ of a real-valued process $f(x)$ as,

$$m(x) = \mathbb{E}[f(x)], \tag{2.1}$$

$$k(x, x', \theta) = \mathbb{E}[\Big(f(x) - m(x)\Big)\Big(f(x') - m(x')\Big)], \tag{2.2}$$

and we denote the GP by,

$$f(x) \sim \mathcal{GP}\Big(m(x), k(x, x', \theta)\Big). \tag{2.3}$$

For a defined process the hyperparameters θ do not change and will be omitted to simplify the notation. One of the key properties of a Gaussian process is called *consistency under marginalization*; any subset of variables from a jointly Gaussian distribution is still Gaussian with the corresponding mean vector and covariance matrix, as if we had taken the entire process into account. This is called the marginalization property.

Evaluating the GP on a finite set of inputs yields a multivariate normal distribution. Define the input X as a finite set of points $x^{(k)}, k = 1, \ldots, n$ for which we want to find the marginal Gaussian distribution. The marginal distribution $\mathbf{f}$ is given by evaluating the mean function $m(x)$ at all input points and the kernel function $k(x, x')$ at all pairs of input points,

$$\boldsymbol{\mu} = m(X) = \begin{bmatrix} m(x^{(1)}) & m(x^{(2)}) & \cdots & m(x^{(n)}) \end{bmatrix}^T, \tag{2.4}$$

$$K = \begin{bmatrix} k(x^{(1)}, x^{(1)}) & k(x^{(1)}, x^{(2)}) & \cdots & k(x^{(1)}, x^{(n)}) \\ k(x^{(2)}, x^{(1)}) & k(x^{(2)}, x^{(2)}) & \cdots & k(x^{(2)}, x^{(n)}) \\ \vdots & \vdots & \ddots & \vdots \\ k(x^{(n)}, x^{(1)}) & k(x^{(n)}, x^{(2)}) & \cdots & k(x^{(n)}, x^{(n)}) \end{bmatrix}, \tag{2.5}$$

which yields the multivariate normal distribution,

$$\mathbf{f} \sim \mathcal{N}\Big(\boldsymbol{\mu}, K\Big). \tag{2.6}$$

To simplify future notation, we define the covariance matrix function $K(\cdot, \cdot)$, working on input sets X, such that $K = K(X, X)$. The two input sets, in this case both being filled by X, can in general be different, and the resulting matrix is the kernel function $k(x, x')$ evaluated at all their pairs. From this we can see that the resulting K need not be square, as in the case of different sized input sets.

There also exists a special class of kernels that permit more efficient linear algebra. A kernel $k(x, x')$ is said to be separable if it factors into a product of kernels, each acting on a subset of the input dimensions. In the 2D case, with $x = (x_1, x_2)$, this reads,

$$k(x, x') = k_1(x_1, x_1')k_2(x_2, x_2'). \tag{2.7}$$

Let $X_1 = (x_1^{(1)}, x_1^{(2)}, \ldots, x_1^{(n_1)})$ and $X_2 = (x_2^{(1)}, x_2^{(2)}, \ldots, x_2^{(n_2)})$ be two real-valued ordered sequences of 1D input coordinates, along the first and second input dimension respectively. Then, let the full set of input points form the Cartesian product grid $X = X_1 \times X_2$. Defining K_d as the $n_d \times n_d$ covariance matrix with entries

$[K_d]_{ij} = k_d(x_d^{(i)}, x_d^{(j)})$, the separability of k together with the grid structure of X implies that the full covariance matrix is,

$$K = K_1 \otimes K_2, \tag{2.8}$$

where the Kronecker product is defined by $[A \otimes B]_{(i,k),(j,l)} = [A]_{i,j}[B]_{k,l}$. This structure greatly reduces the computational and memory complexity of sampling, as outlined in [17], and is further discussed in Section 2.1.2.

One of the primary features that the kernel provides is how it allows for varying the smoothness of the real-valued process by changing the hyperparameters. The selection of these hyperparameters can be done in a multitude of ways, but are typically selected by maximizing the log marginal likelihood of the observed data under the GP model, discussed in [16]. This selection finds the hyperparameters for which the prior assumptions, such as smoothness and correlation length, best describe the data.

2.1.2 Sampling from the Gaussian Process

Generating samples from a GP is necessary for the Monte Carlo methods used later in this thesis. Given this, we would like to generate n samples of a GP that form the Gaussian distribution $\mathbf{f}$ from (2.6) with mean $\boldsymbol{\mu}$ and covariance K. A possible way to draw samples from this distribution is by using Cholesky decomposition [16, p. 202], which allows a lower triangular matrix L to be formed such that $LL^T = K$. To perform the sampling of the full distribution, we draw a sample $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, I)$ which are n i.i.d. standard normal samples, and compute

$$\mathbf{f} = \boldsymbol{\mu} + L\mathbf{z}. \tag{2.9}$$

To show that the distribution obtained from this is equivalent to our expected distribution, we compute the mean and variance. The mean of $\mathbf{f}$ is $\mathbb{E}[\boldsymbol{\mu} + L\mathbf{z}] = \boldsymbol{\mu}$, which is the same. The covariance can similarly be calculated as,

$$\begin{aligned}
\mathrm{Var}(\mathbf{f}) &= \mathbb{E}[(\mathbf{f} - \boldsymbol{\mu})(\mathbf{f} - \boldsymbol{\mu})^T] = \mathbb{E}[(\boldsymbol{\mu} + L\mathbf{z} - \boldsymbol{\mu})(\boldsymbol{\mu} + L\mathbf{z} - \boldsymbol{\mu})^T] && (2.10)\\
&= \mathbb{E}[(L\mathbf{z})(\mathbf{z}^T L^T)] = L\mathbb{E}[\mathbf{z}\mathbf{z}^T]L^T = LL^T && (2.11)\\
&= K, && (2.12)\\
& && (2.13)
\end{aligned}$$

and also matches the desired distribution. Since the samples of $\mathbf{z}$ are normally distributed, and $L\mathbf{z}$ is a linear combination of $\mathbf{z}$, we know that $\mathbf{f}$ has a Gaussian distribution since a Gaussian stays Gaussian under linear transformations. Furthermore, a Gaussian distribution is fully characterized by its mean and covariance, and as such the distribution of (2.9) is equivalent with the distribution of (2.6).

As established in the previous section, a separable kernel on a Cartesian product grid $X = X_1 \times X_2$ yields $K = K_1 \otimes K_2$. Using the Cholesky decomposition this

can be written as $K = L_1 L_1^T \otimes L_2 L_2^T$, and it is then equivalently possible to take a sample using the individual components L_1, L_2 as with L by,

$$\mathbf{f} = \boldsymbol{\mu} + (L_1 \otimes L_2)\mathbf{z}. \tag{2.14}$$

This equivalence is not trivial, and to show it we consider the Kronecker property $(A \otimes B)(C \otimes D) = AC \otimes BD$. We write $K = L_1 L_1^T \otimes L_2 L_2^T = (L_1 \otimes L_2)(L_1 \otimes L_2)^T$, and note that we have two matrices, only differing by a transpose, which is very similar to our original Cholesky decomposition. There we had $K = LL^T$, and so substituting L by $L_1 \otimes L_2$ will also give the correct distribution since $K = (L_1 \otimes L_2)(L_1 \otimes L_2)^T$. Here, we have not shown that $L = L_1 \otimes L_2$, which is not true in general for two matrices $AA^T = BB^T$, but will be true in our specific case. This is not needed for the covariance structure to hold, so we will omit these details. Furthermore, this does not yet give us any computational savings if we still perform the Kronecker product before we multiply with $\mathbf{z}$. Importantly though, it does allow us to use the identity $\text{vec}(BZA^T) = (A \otimes B)\text{vec}(Z)$, where the vec() operator stacks the columns of a matrix as a column vector. Let $A = L_1 \in \mathbb{R}^{n_1 \times n_1}$, $B = L_2 \in \mathbb{R}^{n_2 \times n_2}$, $Z \in \mathbb{R}^{n_2 \times n_1}$, where Z is populated by random entries from the vector $\mathbf{z} \in \mathbb{R}^n, \text{vec}(Z) = \mathbf{z}$, so we can write $\text{vec}(L_2 Z L_1^T) = (L_1 \otimes L_2)\text{vec}(Z)$, giving us,

$$\mathbf{f} = \boldsymbol{\mu} + \text{vec}(L_2 Z L_1^T). \tag{2.15}$$

This provides an efficient way of computing the samples since we can avoid calculating the Cholesky decomposition for $K \in \mathbb{R}^{n \times n}$, and instead calculate it for $K_1 \in \mathbb{R}^{n_1 \times n_1}$, $K_2 \in \mathbb{R}^{n_2 \times n_2}$, and solving $L_2 Z L_1^T$ removes the need to compute the full covariance matrix K at all. Now, recall that n_1, n_2 came from the Cartesian grid dimensions, so we know that $n = n_1 n_2$. This sampling method reduces the computational complexity from $O(n^3) = O(n_1^3 n_2^3)$ to $O(n_1^3 + n_2^3 + n_1^2 n_2 + n_1 n_2^2)$, and reduces the memory complexity from $O(n^2)$ to $O(n_1^2 + n_2^2 + n_1 n_2)$, making computations and memory allocation tractable for large grids. The advantage of the latter is best seen if we imagine the optimal case where the number of points increase at the same rate in both directions, i.e. $n_1 = n_2$. Then, the memory complexity is reduced from $O(n_1^4)$ to $O(3n_1^2) = O(n_1^2)$. In the worst case, where $n_2 = 1$ and n_1 is varying, the memory complexity will equivalently be $O(n_1^2)$ both while utilizing and not utilizing the Kronecker structure. It is also clear that similar arguments hold for the computational complexity.

2.1.3 Conditioning the prior Gaussian distribution on noisy observations

In practice, it is often of interest to condition the GP on observations to reduce uncertainty about the process at the observation points. To condition the GP on observations at input points X and evaluate the posterior at points X_*, we follow the derivation outlined in [16]. Let the observation and evaluation point sets X, X_* form the column vectors $\mathbf{x}, \mathbf{x}_*$, respectively. Assume we draw independent noisy observations y of the latent function f for all points $x \in X$. The observations are

given by $y = f(x) + \epsilon$, where the noise $\epsilon \sim \mathcal{N}(0, \sigma^2_{noise})$ is i.i.d. and independent of the latent process. Collecting the observations into a vector yields $\mathbf{y} = \mathbf{f} + \boldsymbol{\epsilon}$, where $\mathbf{f} = f(\mathbf{x})$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2_{noise} I)$. The observations $\mathbf{y}$ are the sum of the jointly Gaussian $\mathbf{f}$ and the i.i.d. Gaussian noise $\boldsymbol{\epsilon}$. The observations are thus also jointly Gaussian, $\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_y, K_{yy})$, with mean $\boldsymbol{\mu}_y = m(\mathbf{x})$. The covariance of $\mathbf{y}$ is partly the covariance of the latent process $\mathbf{f}$, and partly the covariance of the independent noise $\boldsymbol{\epsilon}$. Since the covariance of $\mathbf{f}$ is $K_{ff} = K(X, X)$, the noise covariance is $\mathrm{Cov}(\boldsymbol{\epsilon}) = \sigma^2_{noise} I$, and since the noise is independent of the process, the observation covariance is then $K_{yy} = K_{ff} + \sigma^2_{noise} I$. Their cross covariance K_{*y} can be computed by $K(X_*, X)$, again since the noise is uncorrelated with the process and has zero mean. Also, the mean and covariance of the prior $\mathbf{f}_* = f(\mathbf{x}_*)$ are $\boldsymbol{\mu}_* = m(\mathbf{x}_*)$, $K_{**} = K(X_*, X_*)$. Jointly between observation points $\mathbf{x}$ and evaluation points $\mathbf{x}_*$, we have,

$$\begin{bmatrix} \mathbf{f}_* \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}\Big(\begin{bmatrix} \boldsymbol{\mu}_* \\ \boldsymbol{\mu}_y \end{bmatrix}, \begin{bmatrix} K_{**} & K_{*y} \\ K_{y*} & K_{yy} \end{bmatrix} \Big). \tag{2.16}$$

To continue, we would like to find the conditional distribution $\mathbf{f}_*|\mathbf{y}$, and to this end we build a linear predictor of $\mathbf{f}_*$ in terms of $\mathbf{y}$ as,

$$\hat{\mathbf{f}}_* = \boldsymbol{\mu}_* + B(\mathbf{y} - \boldsymbol{\mu}_y), \tag{2.17}$$

with the residual $E = \mathbf{f}_* - \hat{\mathbf{f}}_*$. For the linear predictor to be optimal, we want to find the matrix B which makes the residual and observations uncorrelated. In a sense, this could be interpreted as having used all the available information from the observations $\mathbf{y}$, and what remains is the irreducible randomness of $\mathbf{f}_*|\mathbf{y}$. We first begin by verifying the mean prediction,

$$\mathbb{E}[E] = \mathbb{E}[\mathbf{f}_* - \hat{\mathbf{f}}_*] = \mathbb{E}[\mathbf{f}_*] - \mathbb{E}[\boldsymbol{\mu}_* + B(\mathbf{y} - \boldsymbol{\mu}_y)] = \boldsymbol{\mu}_* - \boldsymbol{\mu}_* = 0, \tag{2.18}$$

which verifies that our linear predictor does not have a bias. Next, we want to find the variance of E which is the remaining variance of our distribution $\mathbf{f}_*|\mathbf{y}$. However, as this depends on the optimality of our estimator we start by calculating B. Firstly, the optimal matrix B for our predictor is given by making the predictor uncorrelated with the observation, i.e. $\mathrm{Cov}(E, \mathbf{y}) = 0$, and is calculated as,

$$\mathrm{Cov}(E, \mathbf{y}) = \mathbb{E}\Big[\underbrace{\big((\mathbf{f}_* - \boldsymbol{\mu}_*) - (\hat{\mathbf{f}}_* - \boldsymbol{\mu}_*)\big)}_{\mathbf{f}_* - \boldsymbol{\mu}_* - B(\mathbf{y} - \boldsymbol{\mu}_y)} \big(\mathbf{y} - \boldsymbol{\mu}_y\big)^T \Big] = \tag{2.19}$$

$$= \mathbb{E}\Big[(\mathbf{f}_* - \boldsymbol{\mu}_*)(\mathbf{y} - \boldsymbol{\mu}_y)^T - B(\mathbf{y} - \boldsymbol{\mu}_y)(\mathbf{y} - \boldsymbol{\mu}_y)^T \Big] = \tag{2.20}$$

$$= \underbrace{K_{*y} - B K_{yy}}_{\text{Set to zero by orthogonality condition}} = 0, \tag{2.21}$$

$$\implies B = K_{*y} K_{yy}^{-1}, \tag{2.22}$$

giving us a closed form expression for B. Next we compute the now optimal variance of the posterior distribution $\mathbf{f}_*|\mathbf{y}$,

$$\begin{aligned}
\mathrm{Var}(E) &= \mathbb{E}[(\mathbf{f}_* - \boldsymbol{\mu}_* - B(\mathbf{y} - \boldsymbol{\mu}_y))(\mathbf{f}_* - \boldsymbol{\mu}_* - B(\mathbf{y} - \boldsymbol{\mu}_y))^T] = && (2.23)\\
&= K_{**} - K_{*y}B^T - BK_{y*} + BK_{yy}B^T = && (2.24)\\
&= K_{**} - K_{*y}(K_{yy}^{-1})K_{*y}^T - K_{*y}K_{yy}^{-1}K_{*y}^T + \underbrace{K_{*y}K_{yy}^{-1}K_{yy}(K_{yy}^{-1})K_{*y}^T}_{=K_{*y}(K_{yy}^{-1})K_{*y}^T} = && (2.25)\\
&= K_{**} - K_{*y}K_{yy}^{-1}K_{*y}^T = && (2.26)\\
&= K_{**} - K_{*y}(K_{ff} + \sigma_{noise}^2 I)^{-1}K_{*y}^T, && (2.27)
\end{aligned}$$

which gives the covariance matrix for the posterior distribution. Similarly, we can use the linear predictor to calculate the posterior mean as,

$$\begin{aligned}
\mathbb{E}[\mathbf{f}_*|\mathbf{y}] &= \boldsymbol{\mu}_* + K_{*y}K_{yy}^{-1}(\mathbf{y} - \boldsymbol{\mu}_y) = && (2.28)\\
&= \boldsymbol{\mu}_* + K_{*y}(K_{ff} + \sigma_{noise}^2 I)^{-1}(\mathbf{y} - \boldsymbol{\mu}_y), && (2.29)
\end{aligned}$$

The residual E between the prior $\mathbf{f}_*$ and the linear predictor $\hat{\mathbf{f}}_*$ is a linear combination of Gaussian random variables, and as such E is also Gaussian. From this and the above derivation, we conclude that the posterior distribution of $\mathbf{f}_*$ reads,

$$\mathbf{f}_*|\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_* + K_{*y}(K_{ff} + \sigma_{noise}^2 I)^{-1}(\mathbf{y} - \boldsymbol{\mu}_y), K_{**} - K_{*y}(K_{ff} + \sigma_{noise}^2 I)^{-1}K_{*y}^T). \quad (2.30)$$

The posterior distribution (2.30) involves three covariance terms, namely K_{**}, K_{*y}, and K_{yy}. When the number of evaluation points is large, the dominant challenge is storing and computing with $K_{**} \in \mathbb{R}^{n_* \times n_*}$ and $K_{*y} \in \mathbb{R}^{n_* \times n_{obs}}$, both of which grow with $n_* = n_{*,1}n_{*,2}$. If the evaluation grid X_* and observation grid X coincide, i.e. $X_* = X$, then it is possible to draw samples from the posterior distribution $\mathbf{f}_*|\mathbf{y}$ by utilizing the Kronecker structure. However, even when the observation set X_* is a subset of X, the Kronecker structure breaks and no efficient sampling is available directly from the posterior distribution. Fortunately, a surface equivalent to a draw from the posterior can be constructed using Matheron's rule, outlined in [18] [19]. They show that if we let $\mathbf{y}$ be the observed values at input locations X, and let $\tilde{\mathbf{f}}_*, \tilde{\mathbf{y}}$ be a joint prior of the GP evaluated at the evaluation locations X and observation locations X_*, then a sample from the posterior random variable $\mathbf{f}_*|\mathbf{y}$ is given by,

$$\mathbf{f}_* = \tilde{\mathbf{f}}_* + K_{*y}K_{yy}^{-1}(\mathbf{y} - \tilde{\mathbf{y}}). \quad (2.31)$$

In practice, for the joint prior draw $\tilde{\mathbf{f}}_*, \tilde{\mathbf{y}}$ to be efficient, the observation set X must be a subset of the evaluation set X_* which itself lies on a Cartesian product grid.

2.2 Filtering and Propagation

The Kalman filter is an optimal Bayesian estimator for Linear Time Invariant (LTI) systems with additive Gaussian noise, providing the minimum mean square error estimate of the posterior distribution by incorporating process noise, observation noise, and observations [20]. However, the standard framework does not extend to nonlinear dynamics or nonlinear observation models. The Unscented Kalman Filter (UKF), formulated in [21], addresses these shortcomings by propagating the state distribution through the true nonlinear dynamics.

2.2.1 Standard Kalman Filter

The standard formulation of the Kalman filter, outlined in [20], is defined for a linear time invariant system with additive Gaussian process and measurement noise as,

$$\dot{x} = Ax + Bu + w, \tag{2.32}$$
$$y = Cx + Du + v, \tag{2.33}$$
$$\tag{2.34}$$

with state x, control u, process noise w begin white noise with spectral density Q_c, and measurement noise $v \sim \mathcal{N}(0, R)$. The continuous system is transformed into its discrete counterpart with timestep Δt, such that from timestep k to $k+1$ the dynamics are written as,

$$x_{k+1} = \Phi x_k + \int_0^{\Delta t} e^{A\tau} B u_k \, d\tau + w_k, \quad w_k \sim \mathcal{N}(0, Q), \tag{2.35}$$
$$y_k = Cx_k + Du_k + v_k, \quad v_k \sim \mathcal{N}(0, R), \tag{2.36}$$

where $\Phi = e^{A\Delta t}$ and the process noise is integrated over the timestep as $Q = \int_0^{\Delta t} e^{\tau A} Q_c e^{\tau A^T} \, d\tau$. From some initial covariance P_0, the Kalman filter then alternates between a prediction step,

$$\hat{x}_{k+1|k} = \Phi \hat{x}_{k|k} + \int_0^{\Delta t} e^{A\tau} B u_k \, d\tau, \tag{2.37}$$
$$P_{k+1|k} = \Phi P_{k|k} \Phi^T + Q, \tag{2.38}$$

and an update step upon receiving observation y_{k+1},

$$K = P_{k+1|k} C^T (C P_{k+1|k} C^T + R)^{-1}, \tag{2.39}$$
$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K(y_{k+1} - C\hat{x}_{k+1|k}), \tag{2.40}$$
$$P_{k+1|k+1} = (I - KC) P_{k+1|k}. \tag{2.41}$$

The Kalman gain K weights the innovation $y_{k+1} - C\hat{x}_{k+1|k}$ by the ratio of predicted state uncertainty to total observation uncertainty, where $(CP_{k+1|k}C^T+R)^{-1}$ captures how much of the observed variance comes from measurement noise relative to the predicted state. The covariance update $P_{k+1|k+1} = (I-KC)P_{k+1|k}$ reduces the prior uncertainty by the factor $(I - KC)$, reflecting how much information was gained from the observation. This is structurally identical to the GP conditioning derived in section 2.1, where the posterior mean is shifted by a gain-weighted innovation and the posterior covariance is reduced by the same gain structure.

2.2.2 The Unscented Transform and its Sigma Point

The Unscented Transform is concerned with pushing a random variable $x \sim \mathcal{N}(\mu_x, P_{xx})$ through a nonlinear function $y = \mathcal{F}(x)$ and approximating the resulting distribution as a Gaussian. The transform constructs so-called sigma points in prominent locations around the mean of the distribution, which are then passed through the nonlinear map and used to reconstruct an estimated mean μ_y and covariance P_{yy}. For an n-dimensional random variable x, we define the sigma points as,

$$\mathcal{X}_0 = \mu_x, \quad \mathcal{X}_i = \mu_x + \sqrt{n+\lambda}\, L_i, \quad \mathcal{X}_{i+n} = \mu_x - \sqrt{n+\lambda}\, L_i, \quad i = 1, \dots, n, \tag{2.42}$$

where L_i is the i-th column of the Cholesky factor of P_{xx}, i.e. $LL^T = P_{xx}$, and λ is a scaling parameter controlling the spread of the sigma points. The Cholesky columns provide a natural set of directions for the sigma points, as each column encodes the residual variance in one direction after accounting for correlations with all previous directions, and together they span the full covariance structure of P_{xx}. Together with parameters α and β, the weights for reconstructing the transformed distribution are defined as,

$$W_0^{(m)} = \frac{\lambda}{n+\lambda}, \quad W_i^{(m)} = \frac{1}{2(n+\lambda)}, \quad i = 1, \dots, 2n, \tag{2.43}$$

$$W_0^{(c)} = \frac{\lambda}{n+\lambda} + (1 - \alpha^2 + \beta), \quad W_i^{(c)} = W_i^{(m)}, \tag{2.44}$$

and the propagated points $\mathcal{Y}_i = \mathcal{F}(\mathcal{X}_i)$, $i = 0, \dots, 2n$ are used to reconstruct the transformed distribution as,

$$\mu_y = \sum_{i=0}^{2n} W_i^{(m)} \mathcal{Y}_i, \tag{2.45}$$

$$P_{yy} = \sum_{i=0}^{2n} W_i^{(c)} (\mathcal{Y}_i - \mu_y)(\mathcal{Y}_i - \mu_y)^T. \tag{2.46}$$

To motivate this construction, we verify that the sigma points exactly recover the prior mean and covariance before any nonlinear map is applied. The mean is recovered as,

$$\sum_{i=0}^{2n} W_i^{(m)} \mathcal{X}_i = \frac{\lambda \mu_x}{n+\lambda} + \sum_{i=1}^{n} \frac{\mu_x + \sqrt{n+\lambda} L_i}{2(n+\lambda)} + \sum_{i=1}^{n} \frac{\mu_x - \sqrt{n+\lambda} L_i}{2(n+\lambda)} = \tag{2.47}$$

$$= \frac{\lambda \mu_x}{n+\lambda} + \sum_{i=1}^{n} \frac{2\mu_x}{2(n+\lambda)} = \frac{\lambda \mu_x}{n+\lambda} + \frac{n\mu_x}{n+\lambda} = \mu_x, \tag{2.48}$$

recovering the mean exactly. The variance can likewise be recovered as,

$$\sum_{i=1}^{2n} W_i^{(c)} (\mathcal{X}_i - \mu_x)(\mathcal{X}_i - \mu_x)^T = \tag{2.49}$$

$$= \sum_{i=1}^{n} \frac{(\sqrt{n+\lambda} L_i)(\sqrt{n+\lambda} L_i)^T}{2(n+\lambda)} + \sum_{i=1}^{n} \frac{(-\sqrt{n+\lambda} L_i)(-\sqrt{n+\lambda} L_i)^T}{2(n+\lambda)} = \tag{2.50}$$

$$= \sum_{i=1}^{n} \frac{2(n+\lambda) L_i L_i^T}{2(n+\lambda)} = \sum_{i=1}^{n} L_i L_i^T = P_{xx}, \tag{2.51}$$

where the zeroth term vanishes since $\mathcal{X}_0 - \mu_x = 0$. The rationale behind the transform is that weights which exactly recover the prior moments will also yield an accurate Gaussian approximation after the nonlinear map. For Gaussian distributions, this approximation is accurate to at least second order in the Taylor expansion of $\mathcal{F}$, with $\beta = 2$ being the optimal choice for minimizing higher-order errors. For further details, see [22].

2.2.3 Unscented Kalman Filter

The Unscented Kalman Filter (UKF) is an extension of the standard Kalman Filter that more closely estimates non-linear dynamics [21]. Similarly to the standard Kalman filter, a UKF alternates between the prediction step and the update step. The prediction step propagates the state x_k through the nonlinear dynamics $\mathcal{F}(x_k, u_k)$ using the unscented transform, and the update step conditions the predicted distribution on observations. The standard UKF takes the current prediction $\hat{x}_k$ and covariance $P_{k|k}$, calculates the corresponding sigma points, and propagates them through the dynamics as,

$$\mathcal{X}_i^{k+1} = \mathcal{F}(\mathcal{X}_i^k, u_k,) + w_k, \tag{2.52}$$

and construct the mean and covariance just like the unscented transform as,

$$\hat{x}_{k+1|k} = \sum_{i=0}^{2n} W_i^{(m)} \mathcal{X}_i^{k+1}, \tag{2.53}$$

$$P_{k+1|k} = \sum_{i=0}^{2n} W_i^{(c)} (\mathcal{X}_i^{k+1} - \hat{x}_{k+1|k})(\mathcal{X}_i^{k+1} - \hat{x}_{k+1|k})^T + Q, \tag{2.54}$$

where Q is the same additive process noise from the standard Kalman formulation. In that formulation, the covariance is propagated as $P_{k+1} = \Phi P_k \Phi^T + Q$, which assumes that the process noise is independent of the dynamics during the integration step, and is additive. However, for some systems the process noise enters the dynamics nonlinearly, and then it becomes necessary to extend this formulation. This is done by augmenting the state x_k with states for the process noise as $z_k = \begin{bmatrix} x_k^T & w_k^T \end{bmatrix}^T$, and the covariance matrix with the process noise,

$$P_k^a = \begin{bmatrix} P_k & 0 \\ 0 & Q \end{bmatrix}. \tag{2.55}$$

Then, for n states, n_w noise states, the sigma points are calculated in the same way but using the augmented state and covariance matrix,

$$\mathcal{X}_i^{k+1} = \mathcal{F}(\mathcal{Z}_i^k, u_k), \tag{2.56}$$

and construct the mean and covariance as,

$$\hat{x}_{k+1|k} = \sum_{i=0}^{2(n+n_w)} W_i^{(m)} \mathcal{X}_i^{k+1}, \tag{2.57}$$

$$P_{k+1|k} = \sum_{i=0}^{2(n+n_w)} W_i^{(c)} (\mathcal{X}_i^{k+1} - \hat{x}_{k+1|k})(\mathcal{X}_i^{k+1} - \hat{x}_{k+1|k})^T. \tag{2.58}$$

Note that the covariance is no longer added linearly in the prediction step, and is instead embedded in the covariance sigma point structure itself. To perform the update step, we use the nonlinear observation model $h(x)$, and pass the sigma points through it to get observations,

$$\mathcal{Y}_i^{k+1} = h(\mathcal{X}_i^{k+1}) + v_k, \tag{2.59}$$

and the observation mean and covariance are reconstructed as,

$$\hat{y}_{k+1|k} = \sum_{i=0}^{2(n+n_w)} W_i^{(m)} \mathcal{Y}_i^{k+1}, \tag{2.60}$$

$$P_{yy} = \sum_{i=0}^{2(n+n_w)} W_i^{(c)} (\mathcal{Y}_i^{k+1} - \hat{y}_{k+1|k})(\mathcal{Y}_i^{k+1} - \hat{y}_{k+1|k})^T + R, \tag{2.61}$$

$$P_{xy} = \sum_{i=0}^{2(n+n_w)} W_i^{(c)} (\mathcal{X}_i^{k+1} - \hat{x}_{k+1|k})(\mathcal{Y}_i^{k+1} - \hat{y}_{k+1|k})^T. \tag{2.62}$$

Having access to the covariance matrices P_{yy}, P_{xy}, we can proceed to calculate the posterior distribution conditioned on observations in the same manner as the standard Kalman filter,

$$K = P_{xy}P_{yy}^{-1}, \tag{2.63}$$
$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K(y_{k+1} - \hat{y}_{k+1|k}), \tag{2.64}$$
$$P_{k+1|k+1} = P_{k+1|k} - KP_{yy}K^T. \tag{2.65}$$

The cross-covariance P_{xy} has replaced the analytical expression PC^T, which measures how the state uncertainty projects onto the observation space. These two play exactly the same role, with the only difference being that P_{xy} has to be computed from the sigma points when using the UKF formulation.

2.3 Gaussian Process and LTI SDE Identity

This section considers a zero-mean, stationary, one-dimensional Gaussian process $f(t)$, for which the kernel and the power spectral density are Fourier-transform pairs. That is, a specific kernel corresponds to a specific power spectral density. For certain kernels, including the Matérn family, the process can be represented equivalently by a finite-dimensional linear time-invariant stochastic differential equation (LTI SDE) whose stationary output has the same covariance structure. In this section, we derive this correspondence for the Matérn kernel. A more general construction and the original work is found in [23].

2.3.1 Matching the Power Spectrum

The central idea behind the GP to SDE formulation is the matching of the power spectral density (PSD). For stationary Gaussian processes, the covariance function and the PSD are Fourier-transform pairs, meaning that the PSD uniquely determines the covariance structure of the process. Since a Gaussian process is completely characterized by its mean and covariance, two stationary Gaussian processes with identical PSDs are therefore equivalent in distribution. Consequently, if an LTI SDE can be constructed whose stationary output has the same PSD as the chosen GP kernel, then the two formulations describe the same stochastic process.

For the kernel families considered here, this equivalence permits a finite-dimensional state-space representation. The order m of this representation is determined by the smoothness of the process. More precisely, m is the smallest order for which the process can be represented by an m-dimensional companion-form LTI SDE driven by white noise. For Matérn kernels, this order is directly related to the kernel smoothness parameter and therefore to the mean-square differentiability of the process. To start, let us consider the process f for which we want the output to be equivalent to a first-order state-space LTI SDE system. In particular, we consider the following m-th order LTI SDE system,

$$\frac{d^m f}{dt^m} + a_{m-1}\frac{d^{m-1} f}{dt^{m-1}} + \cdots + a_1\frac{df}{dt} + a_0 f = w(t), \tag{2.66}$$

with white noise $w(t)$ which has spectral density q_c. We can write this as a first order Markov process on the companion form as,

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}\mathbf{x}(t) + \mathbf{L}w(t), \tag{2.67}$$

where $\mathbf{x}(t) = \begin{bmatrix} f & \dot{f} & \cdots & f^{(m-1)} \end{bmatrix}$, and $\mathbf{F} \in \mathbb{R}^{m\times m}$, $\mathbf{L} \in \mathbb{R}^{m\times 1}$ are on the following form

$$\mathbf{F} = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \\ -a_0 & \cdots & -a_{m-2} & -a_{m-1} \end{bmatrix}, \quad \mathbf{L} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}. \tag{2.68}$$

We want to find the Power Spectral Density (PSD) of $f(t)$, which can be obtained by replacing $\mathbf{x}(t)$ with $\mathbf{H}\mathbf{x}(t)$, where $\mathbf{H} = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{1\times m}$, allowing us to extract $f(t) = \mathbf{H}\mathbf{x}(t)$. But before applying this, let us take the Fourier Transform of our LTI SDE system, giving us,

$$i\omega X(\omega) = \mathbf{F}X(\omega) + \mathbf{L}W(\omega), \tag{2.69}$$
$$\implies X(\omega) = (i\omega I - \mathbf{F})^{-1}\mathbf{L}W(\omega). \tag{2.70}$$

To get the Fourier transform of $f(t)$ in particular, we use $\mathbf{H}$ to extract $f(t)$ as $f(t) = \mathbf{H}\mathbf{x}(t) \implies F(\omega) = \mathbf{H}X(\omega)$, which when used in the above equation yields

$$F(\omega) = \mathbf{H}X(\omega) = \mathbf{H}(i\omega I - \mathbf{F})^{-1}\mathbf{L}W(\omega). \tag{2.71}$$

The power spectrum $S(\omega)$ of $F(\omega)$ can be calculated as $S(\omega) = \mathbb{E}[F(\omega)F^\dagger(\omega)]$, where the $(^\dagger)$ operator is the complex conjugate, which gives us

$$S(\omega) = \mathbb{E}[\mathbf{H}(i\omega I - \mathbf{F})^{-1}\mathbf{L}W(\omega)W^*(\omega)\mathbf{L}^T((-i\omega I - \mathbf{F})^{-1})^T\mathbf{H}^T] = \tag{2.72}$$
$$= \mathbf{H}(i\omega I - \mathbf{F})^{-1}\mathbf{L}\mathbb{E}[W(\omega)W^*(\omega)]\mathbf{L}^T((-i\omega I - \mathbf{F})^{-1})^T\mathbf{H}^T = \tag{2.73}$$
$$= \mathbf{H}(\mathbf{F} + i\omega I)^{-1}\mathbf{L}q_c\mathbf{L}^T((\mathbf{F} - i\omega I)^{-1})^T\mathbf{H}^T. \tag{2.74}$$

Here, the form of $S(\omega) = G(i\omega)q_cG(-i\omega)$ is known, and to find it we consider the following. Since $\mathbf{F}$, $\mathbf{L}$, and $\mathbf{H}$ all have real entries, $G(i\omega)$ is a rational function of $i\omega$ with real coefficients. For the companion form in particular, $G(i\omega) = 1/D(i\omega)$, where $D(i\omega) = (i\omega)^m + a_{m-1}(i\omega)^{m-1} + \cdots + a_0$ is the characteristic polynomial of $\mathbf{F}$ evaluated at $i\omega$. For any polynomial $P(s)$ with real coefficients, the product $P(s)P(-s)$ is even in s and can therefore be written as a polynomial in s^2. Substituting $s = i\omega$ then yields a polynomial in $-\omega^2$, hence in ω^2. It follows that all odd powers of ω cancel in the denominator $D(i\omega)D(-i\omega)$, so that $S(\omega)$ is rational in ω^2,

i.e. $S(\omega) = G(i\omega)q_c G(-i\omega) = \frac{(\text{constant})}{(\text{polynomial in } \omega^2)}$. In particular, the Matérn kernel has a known power spectrum $S(\omega)$ and matching the coefficients of the polynomial in (2.74) would give an equivalent process in the LTI SDE form. The power spectrum for the Matérn kernel is,

$$S(\omega) = \sigma^2 \frac{2\pi^{1/2}\Gamma(\nu + 1/2)}{\Gamma(\nu)} \lambda^{2\nu} (\lambda^2 + \omega^2)^{-(\nu+1/2)}, \tag{2.75}$$

where $\lambda = \sqrt{2\nu}/\ell$, and Γ is the gamma function. For a Matérn $\nu = 3/2$ kernel, it is known that $\Gamma(2) = 1, \Gamma(3/2) = \sqrt{\pi}/2$, so the objective is to show how the power spectrum in (2.74) can equate to,

$$S(\omega) = \frac{4\sigma^2\lambda^3}{(\lambda^2 + \omega^2)^2}. \tag{2.76}$$

For the Matérn $\nu = 3/2$ kernel, the LTI SDE system on the companion form must have the same power spectrum for the process to be equivalent. The $m = 2$ state LTI SDE system can be written as,

$$\mathbf{H} = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} 0 & 1 \\ -a_0 & -a_1 \end{bmatrix}, \quad \mathbf{L} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{2.77}$$

and the power spectrum $S(\omega)$ is given by,

$$G(i\omega) = \mathbf{H}(\mathbf{F} + i\omega I)^{-1}\mathbf{L} = \frac{1}{a_0 + i\omega a_1 - \omega^2}, \tag{2.78}$$

$$S(\omega) = G(i\omega)q_c G(-i\omega) = \frac{q_c}{(a_0 + i\omega a_1 - \omega^2)(a_0 - i\omega a_1 - \omega^2)} \tag{2.79}$$

$$= \frac{q_c}{a_0^2 + (a_1^2 - 2a_0)\omega^2 + \omega^4}. \tag{2.80}$$

As was established in the beginning of this section, an equivalent process requires that the PSD from the LTI SDE system should be equal with the PSD from the GP given in (2.76), which yields,

$$S(\omega) = \frac{4\sigma^2\lambda^3}{\lambda^4 + 2\lambda^2\omega^2 + \omega^4} = \frac{q_c}{a_0^2 + (a_1^2 - 2a_0)\omega^2 + \omega^4}, \tag{2.81}$$

$$\implies q_c = 4\sigma^2\lambda^3, \quad a_0 = \lambda^2, \quad a_1 = 2\lambda, \tag{2.82}$$

This implies that the one dimensional GP process with a Matérn $\nu = 3/2$ kernel can equivalently be expressed with an LTI SDE system as,

$$\frac{d\mathbf{x}}{dt} = \begin{bmatrix} 0 & 1 \\ -\lambda^2 & -2\lambda \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w(t), \quad \mathbf{x} = \begin{bmatrix} f \\ f' \end{bmatrix}, \tag{2.83}$$

where the noise $w(t)$ has zero mean and spectral density $q_c = 4\sigma^2\lambda^3$ over all frequencies. This is fully determined by the kernel, since the kernel correlation length ℓ determines $\lambda = \sqrt{2\nu}/\ell = \sqrt{3}/\ell$.

2.3.2 Finding the Noise Covariance

The continuous-time white-noise input $w(t)$ is a generalized stochastic process, not a classical random variable, and it satisfies $\mathbb{E}[w(t)w(t')] = q_c\delta(t-t')$. The quantity used in the discrete-time model is therefore not q_c itself, but the covariance accumulated over one sampling interval, denoted by Q. In general, Q is given by

$$Q = \int_0^{\Delta t} e^{\tau\mathbf{F}}\mathbf{L}q_c\mathbf{L}^\top e^{\tau\mathbf{F}^\top}\, d\tau, \tag{2.84}$$

which is exact but cumbersome to evaluate directly for the companion-form system. For the stationary system considered here, it is more convenient to first derive the stationary covariance P_∞ from Itô's lemma and then recover Q from the discrete-time fixed-point relation. In general, the state covariance P_k for an LTI SDE system with a transition matrix $\Phi = e^{\Delta t\mathbf{F}}$ can be propagated from some initial covariance $P(0)$ using Q as,

$$P_0 = P(0), \quad P_{k+1} = \Phi P_k \Phi^T + Q, \tag{2.85}$$

It is also known that if the covariance of the system is propagated an infinite number of times, it will reach a stationary solution where $P_{k+1} = P_k$, i.e. Q is adding as much variance as the dynamics Φ are removing at each timestep. At this equilibrium, we conclude that

$$P_\infty = \Phi P_\infty \Phi^T + Q, \tag{2.86}$$

$$\implies Q = P_\infty - \Phi P_\infty \Phi^T, \tag{2.87}$$

so if P_∞ is known then Q can be computed in closed form. To find this, we consider the covariance of the state vector $P(t) = \mathbb{E}[\mathbf{x}(t)\mathbf{x}(t)^\top]$. Writing the SDE as $d\mathbf{x} = \mathbf{F}\mathbf{x}\,dt + \mathbf{L}\,dW$, where dW is a scalar Wiener increment with $\mathbb{E}[dW^2] = q_c\,dt$, an application of Itô's product rule to $\mathbf{x}\mathbf{x}^\top$ yields,

$$d(\mathbf{x}\mathbf{x}^\top) = (d\mathbf{x})\mathbf{x}^\top + \mathbf{x}(d\mathbf{x})^\top + (d\mathbf{x})(d\mathbf{x})^\top. \tag{2.88}$$

The Itô correction term evaluates to $(d\mathbf{x})(d\mathbf{x})^T = \mathbf{L}dW \cdot dW\mathbf{L}^T = \mathbf{L}q_c\mathbf{L}^T dt$, since all terms involving $dt \cdot dW$ vanish. Taking expectations and dividing by dt then yields the matrix differential equation,

$$\frac{dP}{dt} = \mathbf{F}P + P\mathbf{F}^\top + \mathbf{L}q_c\mathbf{L}^\top, \tag{2.89}$$

and the stationary solution to this equation is the Lyapunov equation,

$$\mathbf{F}P_\infty + P_\infty\mathbf{F}^T + \mathbf{L}q_c\mathbf{L}^T = 0. \tag{2.90}$$

From this, it is possible to find a closed-form expression of P_∞ for the system in (2.83) as,

$$\begin{aligned}
0 &= \begin{bmatrix} 0 & 1 \\ -\lambda^2 & -2\lambda \end{bmatrix}\begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix} + \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix}\begin{bmatrix} 0 & -\lambda^2 \\ 1 & -2\lambda \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} q_c \begin{bmatrix} 0 & 1 \end{bmatrix} = \\
&= \begin{bmatrix} p_{21} & p_{22} \\ -\lambda^2 p_{11} - 2\lambda p_{21} & -\lambda^2 p_{12} - 2\lambda p_{22} \end{bmatrix} + \begin{bmatrix} p_{21} & -\lambda^2 p_{11} - 2\lambda p_{12} \\ p_{22} & -\lambda^2 p_{21} - 2\lambda p_{22} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & q_c \end{bmatrix}, \\
\implies 0 &= p_{12} + p_{21}, \\
0 &= p_{22} - \lambda^2 p_{11} - 2\lambda p_{12}, \\
0 &= p_{22} - \lambda^2 p_{11} - 2\lambda p_{21}, \\
0 &= -\lambda^2(p_{12} + p_{21}) - 4\lambda p_{22} + q_c.
\end{aligned} \tag{2.91}$$

Since P is symmetric, we know that $p_{12} = p_{21}$, which yields

$$\left.\begin{aligned} 0 &= p_{12} - p_{21} \\ 0 &= p_{12} + p_{21} \end{aligned}\right\} \implies p_{12} = 0,\ p_{21} = 0, \tag{2.92}$$

$$\implies p_{22} = \frac{q_c}{4\lambda}, \quad p_{11} = \frac{q_c}{4\lambda^3}. \tag{2.93}$$

By substituting q_c with the known spectral density $4\sigma^2\lambda^3$ we finally conclude that the closed form expression for P_∞ is,

$$P_\infty = \sigma^2 \begin{bmatrix} 1 & 0 \\ 0 & \lambda^2 \end{bmatrix}. \tag{2.94}$$

This closed form expression for the steady-state covariance can then be used to calculate the additive noise covariance Q by using equation (2.87), instead of computing the integral in (2.84).

2.4 Dynamic Programming

This section will introduce the theoretical background and equations behind dynamic programming, which is the main framework used to formulate the charge planning algorithm in this thesis. For a more detailed introduction to the topic, consult Chapter 1 of [24]. The broad goal of dynamic programming is to perform a series of operations while minimizing some total cost function $J(x, u)$. This cost is based on the state trajectory x_k and inputs u_k from all subsequent stages k. The goal is to

select a sequence of inputs $u_{k=1,\cdots,n-1}$ such that the total cost J is minimized. The state dynamics is governed by the function $f(x, u)$ as

$$x_{k+1} = f_k(x_k, u_k) \tag{2.95}$$

and depends on the current state x_k and input u_k.

2.4.1 Backward and forward pass

Exact dynamic programming is solved as a two step process. First, a backward pass is performed where value functions for each stage is calculated. Value functions are functions that return the cost-to-go for optimal performance given a specific starting state. These calculations exploit a core concept in DP called the *principle of optimality*. "The principle of optimality is a fundamental aspect of dynamic programming, which states that the optimal solution to a dynamic optimization problem can be found by combining the optimal solutions to its sub-problems" [25, p. 1].

This means that the value function at stage k will depend only on the cost function for the current stage, and the value at the next stage $k + 1$. In the backward pass, this means starting at all possible final states, calculating the value function for these final states. Then, the value function at the second-to-last stage can be calculated as the summation of the stage-wise cost function for all possible states at the second-to-last stage plus the value function of the final stage. This can be carried backwards through the problem until the value function of the first stage is calculated. This recursive process is summarized by the following equation, which is known as the Bellman equation [24].

$$V_k(x_k) = \min_{u_k \in U_k(x_k)} g_k(x_k, u_k) + V_{k+1}(f_k(x_k, u_k)) \tag{2.96}$$

where $U_k(x_k)$ is the set of permissable inputs given the current state, $g_k(x_k, u_k)$ is the cost of taking a specific input u_k at the current state x_k. The computation of these value functions are preferably performed offline, as they have a compute time of $\mathcal{O}(K \times N)$, where K is the number of stages and N is the number of possible state values. If the state is multidimensional with n different states, the number of calculations of the value functions grows exponentially with n. This exponential growth, called the *curse of dimensionality*, is the reason why dynamic programming is rarely implemented directly in control programs without some kind of approximation [26].

When the value functions have been calculated, online planning can be performed as a forward pass. Compared to the computationally heavy backward pass, the forward pass can often be computed very quickly. In order to find the optimal input at stage k, depending on the state x_k, simply select the input with the lowest cost as

$$u_k^*(x_k) = \arg \min_{u_k \in U_k(x_k)} g_k(x_k, u_k) + V_{k+1}(f_k(x_k, u_k)), \tag{2.97}$$

where V_{k+1} is the previously computed value function that can now be retrieved via lookup.

2.4.2 Stochastic Dynamic Programming

The standard DP formulation introduced in the previous section is deterministic. This means that given a specific state and input combo x_k, u_k, the next state x_{k+1} is uniquely determined by the system dynamics equation $x_{k+1} = f(x_k, u_k)$. Many systems, including the one discussed in this thesis, are stochastic. Stochastic dynamic programming provides a framework to solve dynamic programming problems in a way which includes stochasticity in the problem formulation, which can lead to improved performance.

The stochastic elements at index k are denoted w_k and are considered similarly to the state x_k and the inputs u_k. The system dynamics equation now depends on w_k as

$$x_{k+1} = f_k(x_k, u_k, w_k) \tag{2.98}$$

where the noise term can not be affected. However, the distribution at w_k may be dependent on the current state x_k. Another viewpoint is to see the disturbances as additional inputs to the system, which occur after the control inputs u_k.

The value function V_k in stochastic dynamic programming is of course influenced by the noise term w_k. The disturbance enters the value function in two places; the cost of this stage, as well as at the dynamics for the next stage, as

$$V_k(x_k) = \min_{u \in U_k(x_k)} \mathbb{E}\left[g_k(x_k, u_k, w_k) + V_{k+1}(f(x_k, u_k, w_k)\right], \tag{2.99}$$

compare to (2.96) which does not include place the equation in an expectation. For the principle of optimality to hold in the stochastic case, the noise terms w_k must be independent and identically distributed [24]. Note the added expectation of the expression, which is used to handle to stochastic process. It is not obvious how to handle constraints. In (2.99), constraints are implicitly handled by the input set $U_k(x_k)$. In this thesis, chance constraints are used to handle uncertainties, see section 5.3.1. In general, stochastic dynamic programming is more computationally heavy compared to its deterministic counterpart. The curse of dimensionality applies to a greater degree, as the number of computations now also grow with a factor related to the dimension of the disturbance w_k.

2.4.3 Approximate Dynamic Programming

The exact DP approach of calculating all value functions quickly becomes intractable due to the curse of dimensionality. In order for DP to remain tractable for larger problems, approximations need to be introduced, which leads to the concept of approximate dynamic programming (ADP). Bertsekas [24] introduces three types of approximations that can be applied to reduce the computational complexity of a problem. These are approximation of value functions, reduction of the input space, and certainty equivalence.

The total cost function of the problem analyzed in this thesis can be split into a stage wise summation of the cost function $J(x_0|u_0, \cdots, u_{N-1})$ as

$$u_k^* = \arg\min_{u_k \in U_k} g_k(x_k, u_k) + \tilde{V}_k(f_k(x_k, u_k)) \tag{2.100}$$

where $\tilde{V}_k(f_k(x_k, u_k))$ is the approximated value function. It is in general difficult to describe the theory in ADP with too much detail, as it is very implementation dependent. There are may different methods of ADP, with individual benefits and drawbacks [27]. The exact details of the ADP implementation used in this thesis are explained in Section 5.2.

3

Models and assumptions

This chapter contains the models, and their related assumptions, that were utilized in this thesis. First, the models which describe the dynamics of the truck, both driving and charging, are outlined. Connecting to the truck model, the related environment structures and charger availability along the route is established. Second, we define the disturbance models which form the basis of uncertainty along state trajectories. Third, the state space models, both for driving and charging, and their related dynamics are described. Finally, based on the models and assumptions outlined, we conclude by formally stating the optimization problem. The following chapters, Chapter 4 and Chapter 5, then describe the proposed solution to the aforementioned problem.

3.1 Truck modeling

In this thesis, longitudinal dynamics are used to model driving. The forces involved are balanced using Newton's 2nd law of motion, from which the acceleration of the truck can be calculated. The acceleration of the truck $a(t)$ with mass m is then calculated based on the forces acting on the body as,

$$m \cdot a(t) = F_{\text{tr}}(t) - F_{\text{brk}}(t) - F_{\text{air}}(t) - F_{\text{g}}(t) - F_{\text{roll}}(t), \tag{3.1}$$

with forces; F_{tr} from the trucks electric motor, F_{brk} from the trucks mechanical brakes, F_{air} from air drag, F_{g} from potential energy, and F_{roll} from rolling resistances. The traction and braking force $F_{\text{tr}}, F_{\text{brk}}$ are the inputs to the system. Regenerative braking is done through the traction force.

The air drag term F_{air} is proportional to the square of the trucks relative wind velocity, according to

$$F_{\text{air}}(t) = \frac{1}{2}\rho C_d A_f \cdot (v(t) + v_{\text{wind}}(t))^2, \tag{3.2}$$

where $\rho C_d A_f$ represents the air resistance coefficient, $v(t)$ the truck velocity, and v_{wind} the wind velocity. Thus, positive wind speeds imply a headwind, and negative wind speeds imply a tailwind. The model assumes that the wind velocity, relative to a stationary observer, is aligned with the longitudinal direction of the truck. Thus, the trucks relative wind velocity is calculated by adding the truck velocity and the

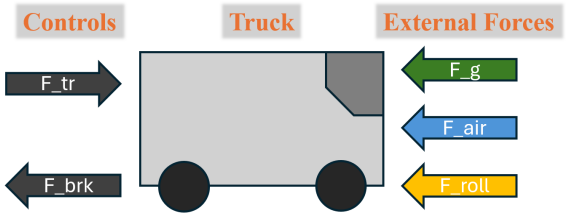


Figure 3.1: The control inputs and external forces considered for the trucks dynamical model. The traction force F_{tr} is generated by the trucks electric motor, and the braking force F_{brk} by the trucks mechanical brakes. The wind force F_{wind} is proportional to the trucks relative wind speed, and the gravitational and rolling forces $F_{\text{g}}, F_{\text{roll}}$ are proportional to the slope of the road.

external wind velocity. The air density ρ is fixed and is approximated as a constant along the route, regardless of elevation. The gravity term F_{g} represent the force from changing the potential energy of the truck and is modeled as,

$$F_{\text{g}}(t) = mg \cdot \sin(\alpha(t)), \tag{3.3}$$

where α is the incline of the road, and g is the gravity constant. The rolling resistance F_{roll} is given by the friction exerted by the road on the truck,

$$F_{\text{roll}}(t) = mgC_r \cdot \cos(\alpha(t)), \tag{3.4}$$

with C_r representing a constant frictional coefficient. All forces with their corresponding directions which are acting on the truck is shown in Figure 3.1.

3.1.1 Motor and battery modeling

The power consumption of the motor is based on a simple equation with parameters a_1, a_2, a_3, a_4 and a_5, and has been fitted based on measurements of a real electric motor, outlined in [28]. From these parameters, the motor power consumption for some operating point P_{motor}, specified from the motor angular velocity ω (rad/s) and torque M (N m), is given by,

$$P_{\text{motor}} = a_1 + a_2\omega + a_3\omega^3 + a_4\omega M + a_5\omega M^2. \tag{3.5}$$

The motor angular velocity ω and torque M are given by the gear ratio r and vehicle state, with the relations $\omega \cdot r = v$, and $M = F_{\text{tr}} \cdot r$. Additionally, a constant auxiliary power $P_{\text{aux}} = 5$ kW is added to the power drawn from the electrical system to emulate other essential systems. The total electrical load P_{load} on the battery can therefore be expressed as,

$$P_{\text{load}} = P_{\text{motor}} + P_{\text{aux}}. \tag{3.6}$$

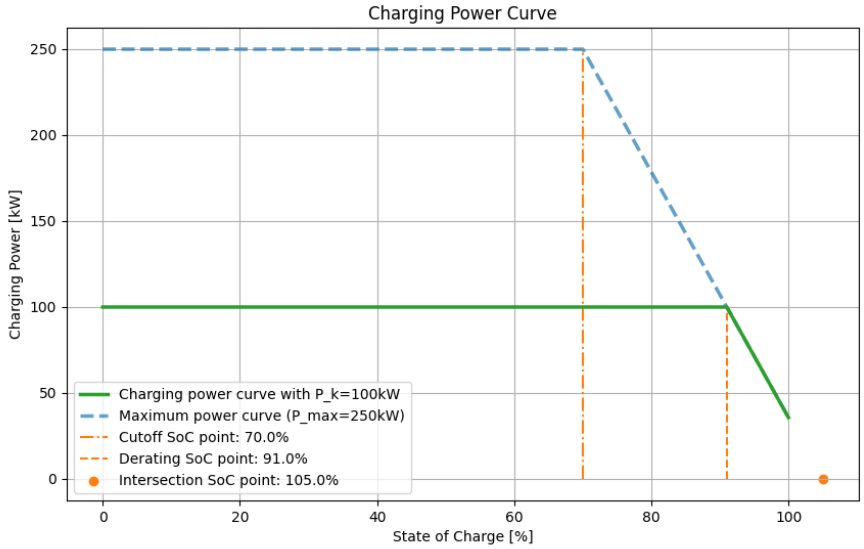


Figure 3.2: Illustration of the charging power $P(\text{soc})$ (*green*) as a function of SoC for a $P_k = 100\text{kW}$ charger at station k. The maximum charging power $P_{\max}$ (*blue*) is set to 250kW, and together with the cutoff point $c_{\text{cut}} = 0.7$ (*orange*) and the intersection point $c_{\text{intx}} = 1.05$ (*orange*), yield a derating SoC $c_{\text{der}} = 0.91$ (*orange*).

The battery is modeled using a constant internal resistance R_b, a constant open terminal voltage u_{oc}, which yields the equation for power consumed as,

$$P_{\text{b}}(P_{\text{load}}) = \frac{u_{\text{oc}}^2 - u_{\text{oc}}\sqrt{u_{\text{oc}}^2 - 4R_{\text{b}}P_{\text{load}}}}{2R_b}. \tag{3.7}$$

As such, the battery losses due to internal resistances are fully determined by the load on the electrical system at any given time. The battery power is subsequently used to determine the battery's SoC trajectory while the truck is driving.

3.1.2 Charging model

In order to preserve battery health as well as to abide by physical constraints imposed by the battery, a custom power curve as a function of SoC is derived. The physical constraint is a direct result of the fact that lithium ion battery, which is commonly used in a BEV's, do not exhibit the same charging characteristics throughout the charging process [29]. There are many ways to model this phenomenon. In the interest of simplicity, a simple linear model is used in this thesis. After the SoC passes a threshold c_{cut}, the charging power will decrease linearly. In turn, this implies that there will be an exponential growth in charging time as the SoC approaches full capacity. This model was chosen because it permits an analytical expression for charge time given an intial SoC soc_0 before charging and terminal SoC soc_1 after charging to be derived. The charging power as a function of the SoC value soc is given by,

$$P(\text{soc}) = \begin{cases} P_k, & \text{if soc} \leq c_{\text{der}}, \\ \frac{P_{\max}}{c_{\text{intx}} - c_{\text{cut}}}(c_{\text{intx}} - \text{soc}), & \text{if soc} > c_{\text{der}}, \end{cases} \tag{3.8}$$

where P_k is the rated charging power of the charger in use. The power limit $P_{\max}$ denotes the maximum power that the battery can physically receive, and the intersect parameter c_{intx} denotes where the charging power intersects the SoC axis. By utilizing this approach, the rate at which the charging curve decreases stays consistent, as when the truck is using a charger with less power it can utilize that power all the way to the derating line before hitting the battery charging rate limit at which point the power starts to decrease. This SoC point, where charging modes switch, is denoted as c_{der}. In addition to the power limit of the charger, there is a maximum charging power of the truck denoted $P_{\max}$, such that $P_k \leq P_{\max}$. The power curve, with an example charger P_k at 100kW and the maximum power $P_{\max}$ at 250kW, is shown in Figure 3.2.

From the charging power $P(\text{soc})$, the corresponding SoC derating point c_{der} given a charger with power P_k can be analytically calculated by,

$$c_{\text{der}} = -\frac{P_k(c_{\text{intx}} - c_{\text{cut}})}{P_{\max}} + c_{\text{intx}}. \tag{3.9}$$

Given the chosen structure, there are three cases for calculating the time t that charging from an initial SoC to a terminal SoC will consume. Given an initial SoC as soc_0 and a terminal SoC as soc_1, the battery capacity E_{bat}, the charging power P_k, and the derating point c_{der}, the charging time $\mathcal{T}_k$ at charger k is given by,

$$\mathcal{T}_k(\text{soc}_0, \text{soc}_1) = \begin{cases} \frac{E_{\text{bat}}(c_{\text{intx}} - c_{\text{cut}})}{P_{\max}}\Big(\ln(c_{\text{intx}} - c_{\text{der}}) - \ln(c_{\text{intx}} - \text{soc}_1)\Big) + \frac{E_{\text{bat}}}{P_k}(c_{\text{der}} - \text{soc}_0), \\ \qquad\qquad \text{if } \text{soc}_0 < c_{\text{der}} < \text{soc}_1. \\ \frac{E_{\text{bat}}}{P_k}(\text{soc}_1 - \text{soc}_0), \quad \text{if } \text{soc}_1 \leq c_{\text{der}}. \\ \frac{E_{\text{bat}}(c_{\text{intx}} - c_{\text{cut}})}{P_{\max}}\Big(\ln(c_{\text{intx}} - \text{soc}_0) - \ln(c_{\text{intx}} - \text{soc}_1)\Big), \\ \qquad\qquad \text{if } \text{soc}_0 \geq c_{\text{der}}. \end{cases} \tag{3.10}$$

We see that the solutions obtained are given based on what regions of the charging power curve were used. If the initial SoC is in the constant region, and the terminal SoC is in the derating region, the first formula is used. If the initial and terminal SoC are in the constant region, the second formula is used. Finally, if the initial and terminal SoC are in the derating region, the third formula is used.

3.1.3 Variable transformation to integration in space

The truck dynamics, outlined in equations (3.1) - (3.7), are based on Newton's 2nd law of motion and are expressed as functions of time. However, the driving dynamics

will be integrated in the space domain along the route, rather than the standard form of integration in the time domain. To make integration in the space domain possible, the variables must be transformed from functions of time $f(t)$ to functions in space $f(s)$. To do this, we consider the chain rule,

$$\frac{df}{ds} = \frac{df}{dt}\frac{dt}{ds} = \frac{df}{dt}\frac{1}{v(s)}, \tag{3.11}$$

where $\frac{dt}{ds} = \frac{1}{v(s)}$. Thus, changing the integration variable from time to space requires only dividing by the local velocity, permitting spatial rather than temporal integration.

3.1.4 Road and route assumptions

This thesis has made the assumption that the route which the truck is driving along is fixed, and route selection is not considered during optimization. As a consequence of this, the altitude and slope of the road is known prior to planning, which allows the incline of the road $\alpha(s)$ to be known for any spatial location s.

A number of charging stations exist along each route. Each station may feature multiple chargers. Each station has two features common among all chargers, its location along the route and the detour driving time from the main route. The chargers at the station have a maximum charging power, as well as a price per kWh. Once one of the available chargers at some station has been selected, the truck will never charge at less than the maximum power available from said charger, aside from the power reduction introduced in Section 3.1.2. Assuming that the electrical power of the charger always matches the specified power is a simplification, the effects of which could be an avenue for future work.

3.2 Disturbance models

The stochastic nature of driving invites the idea that the driving dynamics should be influenced by disturbances that are themselves random. This section will introduce the different types of disturbances and their structural details that are used in this thesis. The primary objective is to determine distributions for some metric over the plausible driving surface in both space and time, i.e. all the locations (spatio-temporal pairs) which might be visited during driving. To denote these locations, we define a spatial and temporal set X_s, X_t, containing n_s, n_t spatial and temporal points respectively. Let the full spatial and temporal point map be defined by their Cartesian product grid $X = X_s \times X_t$, with 2D points $x \in X$. For a given disturbance we then wish to find a process which defines the corresponding distributions for all points x in this grid X.

3.2.1 Traffic process

One of the most obvious disturbances which intuitively will significantly impact driving is surrounding traffic. In this thesis we have thus considered how traffic

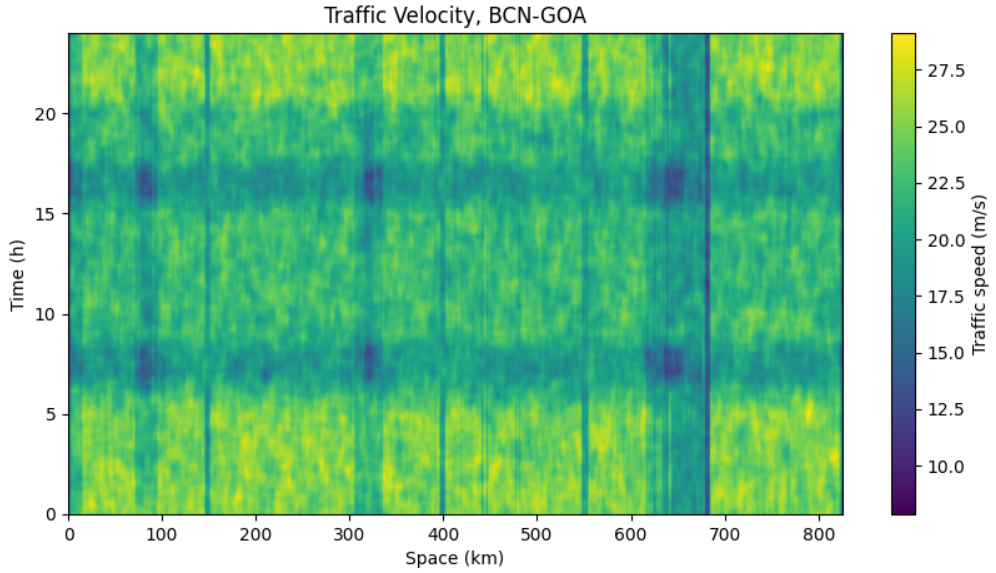


Figure 3.3: A sampled traffic velocity surface, for the route between Barcelona and Genoa. The process has a constant variance field with sigma at $\sigma_{\text{traffic}} = 1.0$ m/s, and spatio-temporal correlation lengths $\ell_s^{(\text{t})} = 5000$m, $\ell_t^{(\text{t})} = 1800$s. The intermediary cities that have been placed are Girona at 83 km, Montpellier at 320 km, and Nice at 645 km.

can be modeled and implemented into the charge planning formulation to emulate uncertainty of travel time and energy expenditure. Traffic models try to emulate how vehicle speeds change as a function of time on a given road, and can become very complex. To maintain scope, two primary objectives with the disturbance model were identified. Firstly, it should be general enough where it could approximately be fitted to real-world data by parameter tuning. Secondly, it should also be possible to find the state distributions along the trajectory. As a result of these considerations a Gaussian Process (GP) was selected because of its generality and well known structure, and will be denoted as $f^{(\text{t})}$.

One of the modeling assumptions imposed by the GP is that the modeled function values have a joint Gaussian structure. Using a GP to model traffic dynamics would inherently imply that the process could be described only by the mean and correlation offered by this structure. However, traffic speed distributions may not be inherently Gaussian, and correlation may not capture intricate traffic dynamics. Therefore, a GP may not model accurate traffic environments. These structural constraints are accepted as simplifications since the primary goal of the traffic model was to emulate traffic in a way which permits stage-wise distributions to be found.

More specifically, the traffic GP will have two input dimensions, namely space and time, and one output dimension of traffic velocity. A GP is defined by two structural choices; a mean function and a kernel function. The process itself is determined by the kernel function, and from this the final output is offset by the mean function.

In practice, the mean function is fitted to data of the average speed on the specified

road given a specific time of day. However, as such data was not available a manual construction was designed. The mean velocity value is based on 2 variables; type of road and the time of day. The road type is assigned into separate types depending on their location in relation to big cities. The placement of cities along the route was done manually for each route. Based on these placements one of three types is assigned to each road location along the route, namely city, urban, and highway. The city road type is assigned to road locations that are determined to be inside big cities, the urban road type is assigned to road locations that are on the outskirts of the big cities, and the highway road is assigned to all the remaining unassigned road locations. After road type assignment, the road is assigned a mean velocity value depending on what time of day it is. The velocity assignment is made to emulate phenomenons like rush-hour traffic, while also staying within the legal speed limit of the road. From this, we denote the constructed mean function for the traffic process as $m^{(\mathrm{t})}(x)$.

The specific kernel structure is chosen such that it can represent the patterns in the data, but there is generally not a standard way to decide on a specific kernel. For a chosen kernel, the parameters can be chosen in a multitude of ways, one of which is outlined in Section 2.1.1. However, since data was not available, the parameters were similarly to the mean also chosen by reason.

The chosen traffic kernel is called a separable product kernel and was paired with a stationary variance field, and are formally described in Section 2.1.1. The specific kernel component structure was chosen to be the same in both the spatial and temporal direction as Matérn $\nu = 3/2$ kernels, with correlation length $\ell_s^{(\mathrm{t})} = 5000\mathrm{m}$ in the spatial direction and correlation length $\ell_t^{(\mathrm{t})} = 1800\mathrm{s}$ in the temporal direction. The traffic standard deviation σ_{traffic} of the stationary variance field was chosen as $\sigma_{\text{traffic}} = 1\mathrm{m/s}$. Letting each point $x \in X$ have spatial and temporal component (s, t), the kernel components are defined as,

$$k_s^{(\mathrm{t})}(s, s') = \Big(1 + \frac{\sqrt{3}|s - s'|}{\ell_s^{(\mathrm{t})}}\Big) \exp\Big(- \frac{\sqrt{3}|s - s'|}{\ell_s^{(\mathrm{t})}}\Big), \tag{3.12}$$

$$k_t^{(\mathrm{t})}(t, t') = \Big(1 + \frac{\sqrt{3}|t - t'|}{\ell_t^{(\mathrm{t})}}\Big) \exp\Big(- \frac{\sqrt{3}|t - t'|}{\ell_t^{(\mathrm{t})}}\Big), \tag{3.13}$$

$$\tag{3.14}$$

where x' denotes a second, permitted to be either the same or a different, point in X. The kernel function determines the correlation of points, as illustrated in Figure 3.4. This yields the full product kernel for the traffic process as,

$$k^{(\mathrm{t})}(x, x') = \sigma_{\text{traffic}}^2 k_s^{(\mathrm{t})}(s, s') k_t^{(\mathrm{t})}(t, t'). \tag{3.15}$$

This specific kernel choice is motivated by the extensive analysis available for the Matérn kernel family, enabling later formulations outlined in Chapter 4. The real process $f^{(\mathrm{t})}$ now reads,

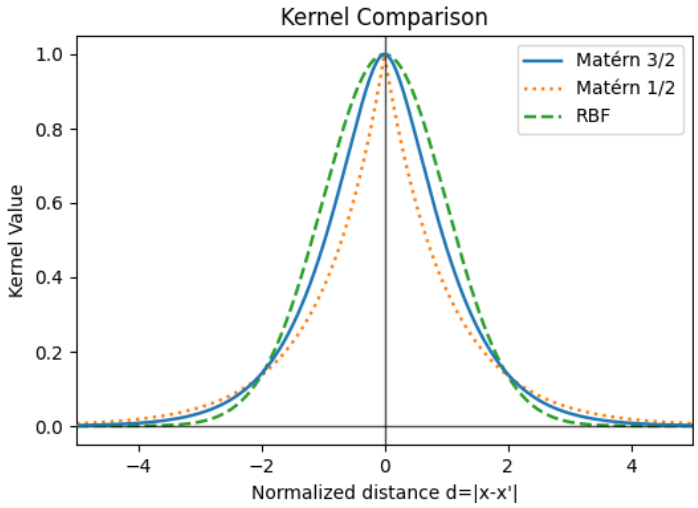


Figure 3.4: The function values of a Matérn $\nu = 3/2$ kernel, evaluated for ± 5 correlation lengths. A Matérn $\nu = 1/2$ kernel and a Radial Basis Function (RBF) kernel were also added for comparison. The x-axis was normalized with division by the correlation length.

$$f^{(\mathrm{t})}(x) \sim \mathcal{GP}\Big(m^{(\mathrm{t})}(x), k^{(\mathrm{t})}(x, x')\Big). \tag{3.16}$$

Finally, we earlier established that the disturbances should be defined for the discrete points $x \in X$. Let the stacked column vector for all points $x \in X$ be denoted as $\mathbf{x} \in \mathbb{R}^{n \times 2}$, where $n = n_s n_t$. Let the continuous traffic process $f^{(\mathrm{t})}$ be evaluated on the grid X, yielding the joint Gaussian random variable $\mathbf{f}^{(\mathrm{t})}$. The resulting multivariate Gaussian distribution is defined by its mean vector $\boldsymbol{\mu}^{(\mathrm{t})} = m^{(\mathrm{t})}(\mathbf{x})$ and covariance matrix $K^{(\mathrm{t})}(X, X) = K^{(\mathrm{t})}$, giving us the multivariate normal distribution,

$$\mathbf{f}^{(\mathrm{t})} \sim \mathcal{N}(\boldsymbol{\mu}^{(\mathrm{t})}, K^{(\mathrm{t})}). \tag{3.17}$$

A sampled surface from this process is seen in Figure 3.3. For large input grids X, e.g. as a result of long distance routes or fine input discretization, the covariance matrix becomes infeasible to form explicitly. However, because the chosen kernel is a separable product kernel and because the input is on a Cartesian grid, the Kronecker product can be used to greatly reduce computational and memory complexity, outlined in Section 2.1.2. This enables large grid structures to be used, which proved to be an essential component in this thesis.

3.2.2 Wind process

Another disturbance which intuitively affect state uncertainty is wind. Similarly to the traffic disturbance, the wind process is a GP formulation, and is denoted $f^{(\mathrm{w})}$. The wind kernel $k^{(\mathrm{w})}(x, x')$ is identical in structure, but differs in parameters, which makes the inclusion of this additional disturbance straight forward. Aside from introducing more state uncertainty, this disturbance also illustrates the ease at

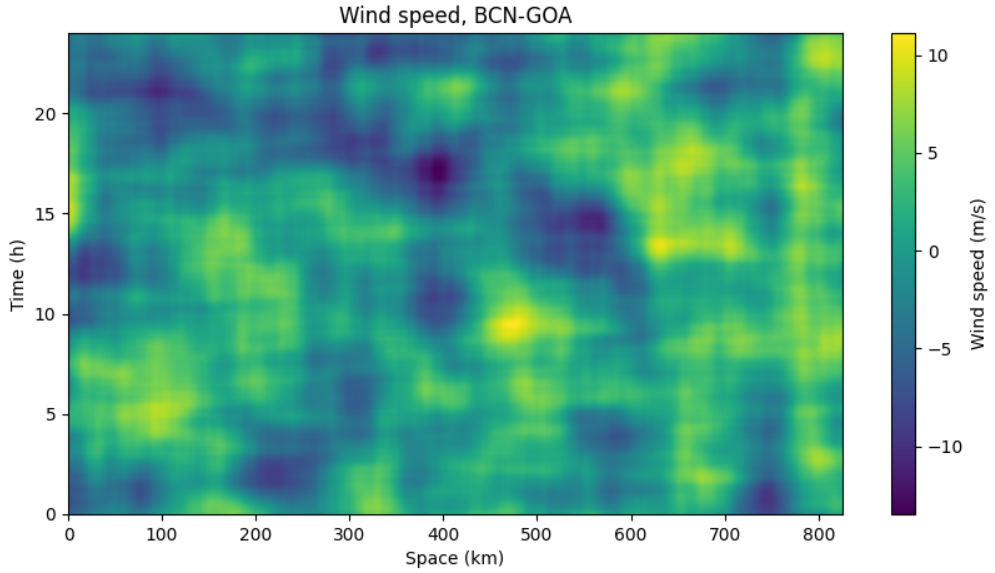


Figure 3.5: A sampled wind speed surface between Barcelona and Genoa. The process has a stationary variance field with sigma at $\sigma_{\text{wind}} = 4.0\text{m/s}$, and spatio-temporal correlation lengths $\ell_s = 50\,000\text{m}$, $\ell_t = 7200\text{s}$. The wind speed is set to have zero mean, which is true even for this random sample as shown by the color bar.

which new disturbances with similar structures to already established disturbances can be added.

The wind process is set to have zero mean, giving equal likelihood to wind in either direction. The parameters were chosen using the same method as described in the previous traffic section. The spatial and temporal correlation lengths $\ell_s^{(\text{w})}, \ell_t^{(\text{w})}$ were chosen as 50km, and 3 hours, respectively. The standard deviation for the stationary variance field was also chosen differently, at $\sigma_{\text{wind}} = 4.0\text{m/s}$. The joint Gaussian random variable for the wind process $f^{(\text{w})}$ evaluated on X is then,

$$\mathbf{f}^{(\text{w})} \sim \mathcal{N}\left(\mathbf{0}, K^{(\text{w})}\right), \tag{3.18}$$

where the wind mean vector $\boldsymbol{\mu}^{(\text{w})} = \mathbf{0}$ is zero, and the covariance matrix function for wind $K^{(\text{w})}(\cdot,\cdot)$ has been evaluate using the wind kernel $k^{(\text{w})}(x, x')$ on the input set to obtain the covariance matrix $K^{(\text{w})}$. The wind joint Gaussian random variable also uses the Kronecker structure described for traffic for efficient sampling.

3.3 State space models

This thesis includes the formulation of two state-space models. The first describes the driving dynamics, which is determined by the truck dynamics and GP disturbance models. The second describes the charge planning dynamics, and relies on the state trajectories of the first. Some trajectories will turn out to be stochastic,

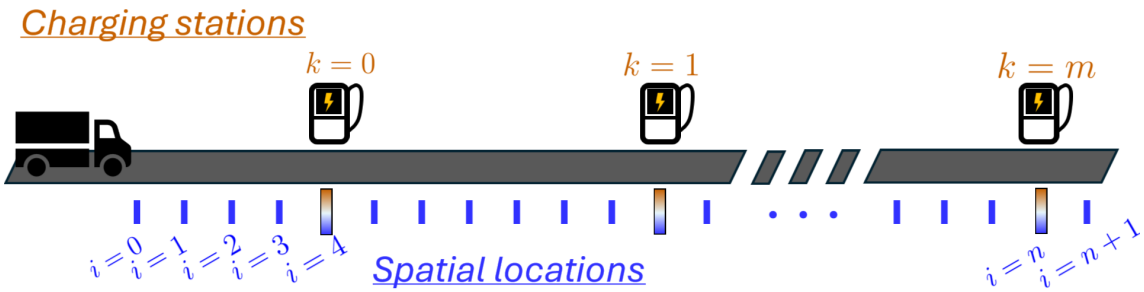


Figure 3.6: Illustration of the indexing systems for charging stations with index k and spatial locations with index i. In this case, it is seen that when $k = 0$, $i = 4$, the location in space s is aligned such that $s_{k=0} = s_{i=4}$. We can also see that every charger is located at some spatial location, i.e. charging locations form a subset of spatial locations such that $\{s_k\} \subseteq \{s_i\}$ always holds.

and their corresponding propagation will be handled in Chapter 4. The methods by which the trajectories are then used varies between charge planners, and will be discussed in Chapter 5.

Furthermore, it is also important to make a note on notation here. Throughout this thesis, k denotes charging station indexing and i denotes spatial location indexing, for which there are M charging stations and N spatial locations. When denoting locations in space s, both indexing systems may be used, and describe the position in meters along the route. Since charging stations form a strict subset of spatial locations, integrating the driving dynamics in space allows state trajectories to be determined precisely between and at chargers.

3.3.1 Driving dynamics

The state space model for driving is derived from the force balance in (3.1). The state vector consists of the time of day t, vehicle velocity v, and cumulative energy consumption C. Since integration is performed along the spatial dimension s with step Δs, all equations are transformed to vary in space, as outlined in Section 3.1.3. The full state vector x is then,

$$x = \begin{bmatrix} t \\ v \\ C \end{bmatrix}. \tag{3.19}$$

A key assumption made in this thesis is that the velocity of the truck v is not optimized. Instead, it is governed by three terms depending on the current state x_i. These are the speed of surrounding traffic, the road speed limit, and the maximum motor torque. Each term yields a candidate maximum-allowed next-state velocity v_{i+1}, the minimum of which is selected.

Two versions of the velocity policy are used in the thesis. The full policy, which enforces the speed limit and maximum motor power alongside traffic following, is used in simulation, where a specific disturbance sample makes these constraints

straightforward to apply. The propagation models in Chapter 4, however, use a simplified policy in which the truck velocity follows the traffic process $f^{(\mathrm{t})}(s,t)$ alone, since enforcing hard limits complicates trajectory calculations. The simplified policy was derived first, as it can be extended to the full policy with minor modifications. The derivation summary is provided in Appendix A.

The dynamics can be integrated either deterministically or stochastically, depending on the charge planner used. The deterministic driver propagates trajectories by evaluating the mean of all disturbances, i.e. traffic and wind, while the stochastic driver treats the dynamics as driven by the random processes $w = (f^{(\mathrm{t})}, f^{(\mathrm{w})})$, yielding stochastic state trajectories whose distribution propagation is covered in Chapter 4. In both cases the position state s can be omitted, as it evolves deterministically throughout. The discrete dynamics are then,

$$t_{i+1} = t_i + \frac{\Delta s}{f_i^{(\mathrm{t})}}, \tag{3.20}$$

$$v_{i+1} = f_{i+1}^{(\mathrm{t})} = f^{(\mathrm{t})}\Big(s_i + \Delta s,\, t_i + \frac{\Delta s}{f_i^{(\mathrm{t})}}\Big), \tag{3.21}$$

$$C_{i+1} = C_i + \Delta s \frac{P_b(f_i^{(\mathrm{t})}, f_{i+1}^{(\mathrm{t})}, f_i^{(\mathrm{w})})}{f_i^{(\mathrm{t})}}, \tag{3.22}$$

$$x_{i+1} = \mathcal{F}_{\mathrm{driving}}(x_i, w_i), \tag{3.23}$$

3.3.2 Charging dynamics

The charging dynamics govern the state evolution at charging stations. The relevant states are time of day t and battery state of charge soc. Vehicle position s is again omitted as it evolves deterministically. The state vector is then,

$$x = \begin{bmatrix} t \\ \mathrm{soc} \end{bmatrix}. \tag{3.24}$$

The control u_k at station k determines whether to charge, and if so, which charger to use and to what target SoC. Let station k have chargers $j \in \mathcal{J}_k$, where each charger j has rated power $P_k^{(j)}$, as well as admitting a discrete set of delta SoC values $\mathcal{C}$. The full set of admissible decisions at station k is then,

$$\mathcal{U}_k = \{(0,\,0)\} \cup \Big\{(j,\,\Delta\mathrm{soc}) : j \in \mathcal{J}_k, \Delta\mathrm{soc} \in \mathcal{C}\Big\}, \tag{3.25}$$

where $(0,\,0)$ denotes the decision not to charge. Since only one charger may be used at a time, and the battery can not be charged to more than the maximum capacity, the control is a single element $u_k \in \mathcal{U}_k(x_k)$ selected from the joint set of feasible controls. The feasible control set is found by ensuring that the delta SoC, Δsoc, is sufficient to keep the SoC above the lower bound $\mathrm{soc}_{\min}$ throughout the drive from station k to $k+1$, and does not exceed the battery capacity E_{bat}.

The state evolves in two steps between consecutive stations k and $k+1$. The charging decision u_k is applied first, for which a new SoC value soc_k^+ is obtained. For $u_k = (0,\, 0)$, the state is unchanged, and thus $x_k = x_k^+$. For $u_k = (j, \Delta\text{soc})$, the post-charging state $x_k^+ = \mathcal{F}_{\text{charging}}(x_k, u_k, w_k)$ is given by,

$$t_k^+ = t_k + \mathcal{T}_k^{(j)}\left(\text{soc}_k, \text{soc}_k + \Delta\text{soc}\right) + t_k^{(\text{detour})}, \tag{3.26}$$

$$\text{soc}_k^+ = \text{soc}_k + \Delta\text{soc}, \tag{3.27}$$

where $\mathcal{T}_k^{(j)}(\text{soc}_k, \text{soc}_k + \Delta\text{soc})$ follows from (3.10) with charger power $P_k^{(j)}$, and $t_k^{(\text{detour})}$ is any additional travel time incurred if the station lies off route. From the post-charging state x_k^+, the driving propagation $\mathcal{F}_{\text{driving}}$ advances the state to the next charging state $x_{k+1} = \mathcal{F}_{\text{driving}}(x_k^+, w_k)$. Since the driving trajectories may be stochastic, x_{k+1} is in general a random variable, and as such the feasible control set $\mathcal{U}_k(x_k)$ may accordingly also be random. To ensure that x_{k+1} remains feasible, the set $\mathcal{U}_k(x_k)$ has to be calculated carefully, as covered in Chapter 5.

3.4 Problem formulation

Given the provided setting, the optimization problem can be stated formally. The stage-wise cost $g_k(x_k, u_k)$, depending on the state x_k and input u_k, penalizes time and charging cost, and is deterministic given a state and control pair. Since the state x_k is stochastic, driven by the traffic and wind disturbances w, both the stage-wise cost and the cost-to-go must be evaluated in expectation. The optimal policy u_k^* then minimizes this expected cost-to-go at each stage k, subject to the feasibility sets imposed on both the state and control. Then, the problem can be expressed as,

$$u_k^*(x_k) = \arg\min_{u_k \in \mathcal{U}_k(x_k)} \mathbb{E}_w\Big[\sum_{j=k}^{M} g_j(x_j, u_j, w_j) + g_M(x_M)\Big], \tag{3.28}$$

$$\text{s.t. } x_k^+ = \mathcal{F}_{\text{charging}}(x_k, u_k, w_k), \tag{3.29}$$

$$x_{k+1} = \mathcal{F}_{\text{driving}}(x_k^+, w_k), \tag{3.30}$$

$$x_i \in \mathcal{X}_i \quad \forall i = 1, \ldots, N, \tag{3.31}$$

$$u_k \in \mathcal{U}_k(x_k) \quad \forall k = 1, \ldots, M. \tag{3.32}$$

4

Uncertainty Propagation

From the stochastic disturbance models and their impact on driving dynamics outlined in Chapter 3, it is of interest to determine the stage-wise distributions along the trajectory. These state distributions are then used to implement chance constraints, see Section 5.3.1, and is their primary function in this thesis. The stage-wise distributions would allow chance constraints to be imposed in the charge planning formulation to ensure the plan adheres to expected disturbances. Given the thesis problem formulation, outlined in (3.28), the expected value of the stage-wise cost function also had to be found. This section introduces the methods by which the state distributions were obtained, and how those were used to contribute to the problem formulation.

4.1 Uncertainty propagation with Monte Carlo simulations

Monte Carlo (MC) simulations is the process of using random disturbance sampling to simulate many possible behaviors of a system [30]. It is a general method that can be used to model the behavior of many different kinds of systems. The introduction of random sampling, performed using some kind of random number generation, allows statistical information to be obtained for the simulated system.

The primary advantage of MC simulations is the flexibility. Essentially any process can be simulated using MC, so long as the modeled probability distributions of the modeled system can be sampled. Another advantage of MC simulation method is that it can be used on black box models, or on other models where calculating resulting distributions analytically is either difficult or impossible [31]. The primary disadvantage is that you need to take many samples to get an accurate representation of the modeled problem. This becomes computationally expensive. In practice, there is a trade off between computational availability and accuracy of the result.

In this thesis, Monte Carlo simulations are used as a source of ground truth, especially when evaluating the Kalman filter introduced in Section 4.2. In our setting, surfaces from the wind and traffic processes are sampled, by which the dynamics are then forward simulated on. The mean and variance of the simulated trajectories can then be compared to the mean and variance values from the UKF. Finally, the MC simulations provide an easy way to calculate the expected value of the cost function, since it can be evaluated individually on each simulated result and then averaged

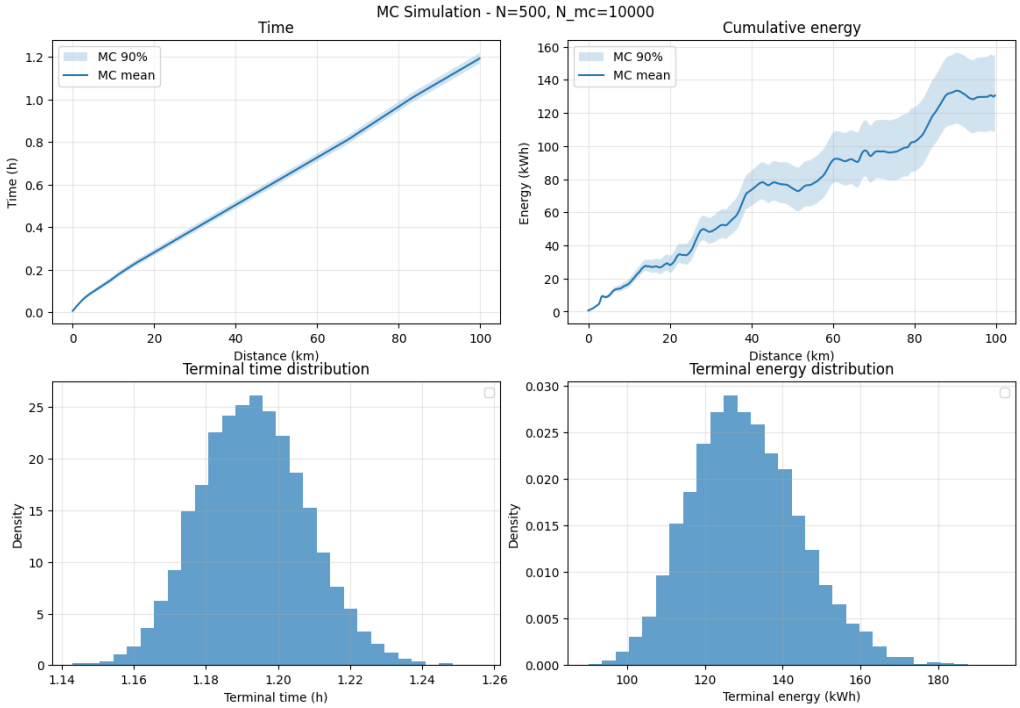


Figure 4.1: Terminal state distributions after running 10.000 MC simulations for a distance of 100km. The top figure shows the mean trajectory and a 90%CI band, and the bottom shows a histogram of the terminal states.

over all sampled distributions.

4.2 Uncertainty propagation with the Unscented Kalman Filter

The terminal state distributions obtained by driving between two chargers located 100 km apart are shown in Figure 4.1. The distributions closely resemble Gaussians, which motivates the use of a UKF. While the UKF does not require Gaussian distributions, it approximates them as such, and systems with near-Gaussian structure are therefore better suited to this estimation approach. The UKF provides a method by which such uncertainty can be propagated efficiently, in contrast to sampling-based methods such as MC. However, this requires the disturbance processes to be known in state-space form. As outlined in Section 2.3, a single-input GP with a Matérn kernel can be equivalently transformed to an LTI SDE. The proposed disturbances are not single-input, and the process noise is coupled through the dynamics. This section covers the proposed resolution for said difficulties, and the full UKF state space model.

4.2.1 Transformation to state-space form

The disturbances, outlined in Section 3.2, are modeled as Gaussian processes with spatio-temporal inputs. For a Gaussian process f with a Matérn kernel, an equiv-

alent LTI SDE form can be obtained by matching the power spectrum of the stationary process from the kernel with an LTI SDE model on the companion form. This was shown by [23], and the relevant background which is used in this thesis is outlined in Section 2.3.1. Specifically, when the Matérn kernel has $\nu = 3/2$, this equivalence is given by,

$$\frac{d\mathbf{x}}{dt} = \underbrace{\begin{bmatrix} 0 & 1 \\ -\lambda^2 & -2\lambda \end{bmatrix}}_{\mathbf{F}} \mathbf{x}(t) + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{\mathbf{L}} w(t), \quad \mathbf{x} = \begin{bmatrix} f \\ f' \end{bmatrix}. \tag{4.1}$$

where $\lambda = \sqrt{3}/\ell$, and ℓ is the Matérn kernel parameter. The use of separable product kernels allows the state space form to be found for both input dimensions independently. However, since the driving dynamics $\mathcal{F}_{\text{driving}}$ couple the spatial and temporal dimensions, the trajectory, i.e. the step length from i to $i+1$, is stochastic in the temporal direction. As such, the process noise Q_i for any given step is also stochastic, since the temporal step length is determined by the random process $f^{(\text{t})}$ itself. The combined state space is thus difficult to form for both disturbances.

To reduce complexity, let us consider the implications of approximating the temporal kernel of a spatio-temporal random process f with stationary variance as constant, corresponding to $\ell_t \to \infty$ and $k_t \approx 1$. The temporal kernel is removed rather than the spatial kernel since the temporal process is itself process dependent, whereas the spatial direction is integrated with a fixed step length and therefore remains deterministic in our formulation. In the proposed limit, a draw from f is coherent along the full temporal dimension for a fixed spatial point, and the variance growth of a process dependent state along the temporal direction is bounded by the stationary variance of the process. In the same way, for a given spatial correlation length ℓ_s, there will be a corresponding rate of variance growth along the spatial direction, also bounded by the same stationary variance field. We may therefore view these two rates as defining a variance surface, and a trajectory through the state dynamics may be interpreted as moving along a diagonal of this surface. Under the constant temporal approximation, the resulting uncertainty is then primarily governed by the spatial kernel.

From this, it is thought that there are two distinct cases when the UKF estimation will be aligned, and not be aligned, with the MC estimation. The first is given when ℓ_t is large in comparison to the certainty equivalent step length $\Delta s/\mu^{(\text{t})}$. Then, the added uncertainty along the temporal dimension is close to the largest value permitted by the stationary variance field, and thus also reasonably aligned with the $\ell_t \to \infty$ assumption. In that case, the UKF and MC estimates are expected to agree reasonably well independent of spatial correlation length. Secondly, the approximation is thought to become less accurate when ℓ_t is small, since the variance growth in the temporal direction is then mismatched by the $\ell_t \to \infty$ assumption. In this thesis, the simplification to remove the temporal kernel has been made to enable a UKF formulation. The implications, and the proposed interpretations, will be tested and evaluated in the results section. Other options have also been considered and will be discussed in the further works section.

This simplification yields a single input GP, and as such the same state space form as shown in (4.1) was used for both disturbances $f^{(\mathrm{t})}, f^{(\mathrm{w})}$. Finally, since the spatial discretization step is constant, the process noise Q can be calculated from the stationary covariance and the transition matrix as outlined in 2.3.2.

4.2.2 State augmentation and additional disturbances

Since the disturbances w enter the driving dynamics $\mathcal{F}_{\text{driving}}$ nonlinearly, the process noise cannot be added linearly to the propagated covariance. The noise must therefore be included in an augmented state vector, so that the covariance structure is mapped directly through the nonlinear dynamics via the unscented transform, as outlined in Section 2.2.3. With both disturbances on LTI SDE form, the augmented state vector is,

$$z_k = \begin{bmatrix} t_k \\ f_k^{(\mathrm{t})} \\ f_k^{'(\mathrm{t})} \\ C_k \\ f_k^{(\mathrm{w})} \\ f_k^{'(\mathrm{w})} \\ q_{k,1}^{(\mathrm{t})} \\ q_{k,2}^{(\mathrm{t})} \\ q_{k,1}^{(\mathrm{w})} \\ q_{k,2}^{(\mathrm{w})} \end{bmatrix} = \begin{bmatrix} x_k \\ q_k \end{bmatrix}, \qquad P_k^a = \begin{bmatrix} P_k & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & Q_k^{(\mathrm{t})} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & Q_k^{(\mathrm{w})} \end{bmatrix}, \tag{4.2}$$

where the dynamics for t_k, C_k follow from $\mathcal{F}_{\text{driving}}(x_k, w_k)$, the disturbances $w_k = (f^{(\mathrm{t})}, f^{(\mathrm{w})})$ follow from their respective LTI SDE models, and the noise $q_k^{(\mathrm{t})}, q_k^{(\mathrm{w})}$ is given by the unscented transform of $Q^{(\mathrm{t})}, Q^{(\mathrm{w})}$.

The proposed UKF framework can easily be extended to include more independent disturbances by extending the state space. The addition of an additional disturbance is illustrated by the wind process, as it simply required adding the equivalent LTI SDE system to the state space equations to be included in the formulation. From this, it is clear that the proposed method is easily adaptable to new disturbance processes, without having to change any parts of the planning framework.

4.2.3 Implications for planning

Compared to sampling-based methods, the UKF provides a computationally cheap way of estimating state distributions along the entire trajectory. Since stage-wise distributions are known, chance constraints can be imposed in the planning formulation, providing statistical guarantees with some confidence interval that the proposed plan will not violate the state constraints.

Since the cost functions used in all stochastic planners are linear, the expected cost-to-go reduces to $\mathbb{E}_w[g(x)] = g(\mathbb{E}_w[x])$, requiring only the stage-wise means from

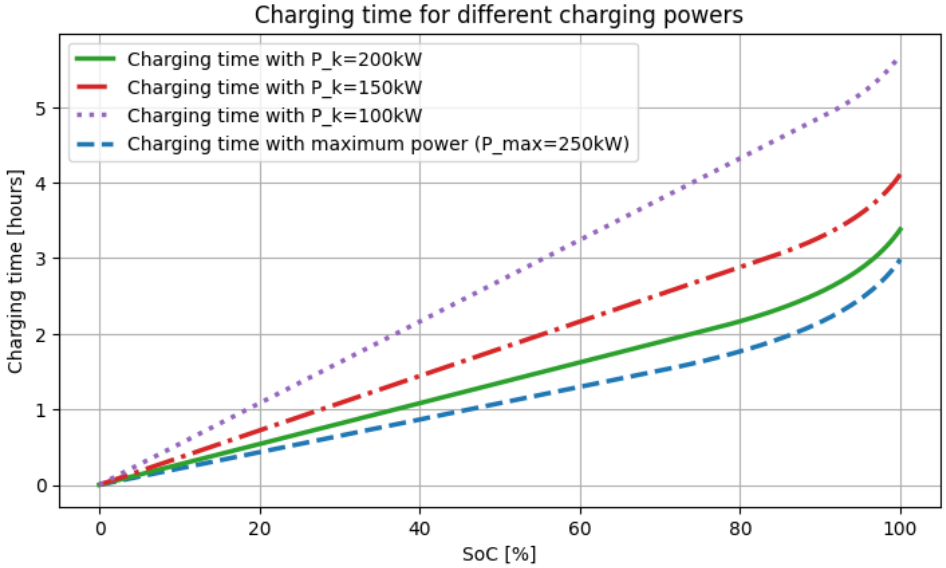


Figure 4.2: Charging times for the used power curve, shown for different charging powers. As the SoC level reaches near full capacity, the time to charge increases in a nonlinear fashion.

the UKF. The full expected cost-to-go $\mathbb{E}[J(x,u)]$, outlined in (3.28), is therefore equivalent to evaluating the cost function at the propagated stage-wise means.

During planning the states may also be propagated through the charging dynamics, and as such the corresponding state uncertainties have to be propagated accordingly. As charging decisions specify the amount of SoC to charge, the propagated SoC state will have the same uncertainty as before charging. However, because of how the power curve was designed the charging time between two SoC levels may not always be linear. If, for instance, the initial SoC is in the constant power range, and the terminal SoC is above the derating point c_{der}, then the time to charge between soc_0 to c_{der} will be stochastic. Equally the time to charge between c_{der} to $\text{soc}_0 + \Delta\text{soc}$ will also be stochastic, and as such the variance of the charging time $\text{Var}(\mathcal{T}(\text{soc}_0, \text{soc}_0 + \Delta\text{soc}))$ is non-zero. To simplify this, this thesis has made the assumption that the charging time is linear during variance propagation. Then for a given delta SoC, Δsoc, from an initial SoC, soc_0, the variance of charging time is calculate as,

$$\begin{aligned}
&\text{Var}(\mathcal{T}(\text{soc}_0, \text{soc}_0 + \Delta\text{soc})) = \\
&= \mathbb{E}\Big[\Big(\mathcal{T}(\text{soc}_0, \text{soc}_0 + \Delta\text{soc}) - \mathbb{E}[\mathcal{T}(\text{soc}_0, \text{soc}_0 + \Delta\text{soc})]\Big)^2\Big] \\
&= \mathbb{E}\Big[\frac{E_{\text{bat}}^2}{P_k^2}\Big(\text{soc}_0 + \Delta\text{soc} - \text{soc}_0 - \mathbb{E}[\text{soc}_0 + \Delta\text{soc} - \text{soc}_0]\Big)^2\Big] \\
&= \frac{E_{\text{bat}}^2}{P_k^2}\mathbb{E}\Big[(\Delta\text{soc} - \Delta\text{soc})^2\Big] \\
&= 0.
\end{aligned} \tag{4.3}$$

Thus, when assuming linear charging times the variance of the charging time is then seen to be zero. To illustrate the accuracy that this simplification may give, the charging times for different charging powers are shown in Figure 4.2. As seen in the figure, the charging time is approximately linear for a significant range of SoC values, but trends upward at the end. Because of this simplification, the variance may be reduced slightly compared to MC simulations, but is accepted as this effect is thought to be minor.

5

Charge planning with Dynamic Programming

The charge planning in this thesis is done using Dynamic programming (DP). This chapter will discuss the various planners and their implementation, using the theory from Section 2.4. All planers are implemented in Python with the array and matrix math using Numpy [32]. Finer implementation details, including discretization steps and similar topics, can be found in Appendix B.

5.1 Full horizon and exact deterministic DP planner

As a point of comparison, a deterministic planner is implemented as a baseline. This planner is implemented as a full horizon exact DP problem, solved with a full backwards and forwards pass. This planner does not consider driving time when evaluating charging policies, instead only considering the cost of charging and the time spent at each charging stop. The energy to charge, expressed in state of charge, is denoted Δsoc. The cost of charging is denoted $c_{\text{charging},k}$, note the index k indicating that this cost is charger dependent. Equivalently, the total stop time at each station is denoted t_{stop} with a cost of c_{time} per hour. The cost function g_k at each charging station k is given by

$$g_k(x_k, u_k) = c_{\text{charging},k} \cdot \Delta\text{soc} + c_{\text{time}} \cdot t_{\text{stop}} + c_{\text{low soc}} \sum \text{soc}_i^{-2}. \tag{5.1}$$

To compensate for uncertainties during driving, this planner utilizes a "range anxiety term" to penalize low SoC in the battery. This term is an inverse square summation of the SoC when driving. This means a low penalty is applied to the cost when driving with a high state of charge in the battery, and a high penalty is applied to the cost when driving with a low state of charge. This penalty term has coefficient $c_{\text{low soc}}$ as a tuning parameter.

Only the state of charge is used as a state in this planner, leading to the very simple state vector $x_k = \begin{bmatrix}\text{soc}\end{bmatrix}$, though the current charging station k is implicitly included as a state as well. The state evolution between charging stations can be split into two discrete parts where the state of charge x_{k+1} at the next station depend on the current state of charge x_k and charging decision u_k as

$$x_{k+1} = f(x_k, u_k) = f_{\text{charging}}(x_k, u_k) + f_{\text{driving}}(x_k^+). \tag{5.2}$$

The input u_k at each station is a two component vector, defined as

$$u_k = \begin{bmatrix} d_{\text{charge}} \\ \Delta\text{soc} \end{bmatrix} \tag{5.3}$$

where d_{charge} represents which specific charger at the station the truck will charge at and Δsoc, the energy to charge, is expressed in percent of the full battery. The charging component is determined by the sign of the selected charger d_{charge}, where $d_{\text{charge}} = 0$ implies not stopping at the k^{th} charging station. The state after charging x_{k^+} is expressed using the equation

$$x_{k^+} = f_{\text{charging}}(x_k, u_k) = \begin{cases} x_k + \Delta\text{soc} & \text{if } d_{\text{charge}} \neq 0 \\ x_k & \text{if } d_{\text{charge}} = 0 \end{cases}. \tag{5.4}$$

The state evolution when driving is also a simple equation, depending firstly on the battery SoC after charging followed by subtracting the energy consumed over one driving segment using Forward Euler, see (5.6). The indexes i_{s_k} and $i_{s_{k+1}}$ represent the distance-wise integration steps between charging stations k and $k+1$ respectively. The power consumption for spatial locations between chargers $P_{\text{b}}(s_i)$ is defined in (3.7), restated here with indexing in space s_i as

$$P_b(s_i) = \frac{u_{\text{oc}}^2 - u_{\text{oc}}\sqrt{u_{\text{oc}}^2 - 4R_b(P_{me}(v(s_i), v(s_{i+1}), v_{\text{wind}}(s_i)) + P_{aux})}}{2R_b \cdot v(s_i)}. \tag{5.5}$$

All in all, this gives the state dynamics from state x_k to state x_{k+1} as

$$x_{k+1} = f_{\text{driving}}(x_k^+) = x_k^+ + \sum_{i=i_{s_k}}^{i_{s_{k+1}}} P_b(s_i) \cdot \Delta s. \tag{5.6}$$

5.2 ADP with time as a state

Approximate dynamic programming (ADP) is a collection of methods used to reduce the compute time required to solve DP problems [24]. There are a number of approximations that can be made when solving a DP problem. While this may sound entirely like a drawback, using ADP allows a planner to process more states and make more informed decisions. In exact DP, the introduction of a second state would effectively square the computation time, see the discussion in Section 2.4. This is what is meant by the curse of dimensionality.

Recall that approximate dynamic programming uses a modified version of the Bellman equation to expedite computation, see Section 2.4.3. In the ADP case, the value function is not computed directly. Instead, the optimal input at stage k is computed as

$$u_k^* = \arg\min_{u_k \in \mathcal{U}_k} g_k(x_k, u_k) + \tilde{V}_k(f_k(x_k, u_k)) \tag{5.7}$$

where $\tilde{V}_k(x_k, u_k)$ is the approximated value function. This approximation can in theory be any function, but this thesis uses rollout as the approximation for the value function. In rollout, some other function is used to calculate the series of inputs from the next station $k+1$ until the final station M. This series of inputs is called the *policy* of the heuristic and is denoted π. This function, called a *base heuristic*, is simpler to compute than using exact DP to achieve the same sequence [24]. The series of inputs produced by the base heuristic may not be optimal, but the idea is that a sufficiently good heuristic produces inputs that yield a good final result. The algorithm for planning and simulating with rollout is defined in Algorithm 1.

Algorithm 1 Planning and simulating with rollout. $\mathcal{F}_k$ represent the combined state dynamics of charging and driving between stations k and $k+1$. $\hat{\mathcal{F}}$ represents the simulated driving dynamics using the UKF.

$k \leftarrow 0$
$x_0 \leftarrow \begin{bmatrix} \text{soc}_{\text{start}} & t_{\text{start}} \end{bmatrix}^\intercal$
$x_1 \leftarrow \mathcal{F}_{\text{driving}}(x_0)$ ▷ Do first driving step
for k = 1:M **do**
  $\text{cost}_k = \{\}$
  $\mathcal{U}_k \leftarrow$ Feaseable Inputs
  for n = 1:N **do**
    $u_k^n \leftarrow \mathcal{U}_k(n)$ ▷ Get the n-th possible input
    $x_{k+1}^n \leftarrow \hat{\mathcal{F}}_k(x_k, u_k^n)$ ▷ Simulate state forward until next station
    $\pi_{k+1}^n \leftarrow \text{rollout}(x_{k+1}^n)$ ▷ Generate the rollout policy
    $\text{cost}_n \leftarrow g_k(x_k, u_k^n)$ ▷ Calculate cost of first exact stage
    $x_j \leftarrow x_{k+1}^n$
    for j = k:M **do**
      $\text{cost}_n \leftarrow \text{cost}_n + g_j(x_j, \pi_{k+1}^n(j, x_j))$ ▷ Cost of applying the heuristic
      $x_{j+1} \leftarrow \hat{\mathcal{F}}_j(x_j, \pi_{k+1}^n(j, x_j))$
    end for
    $\text{cost}_k \leftarrow \text{cost}_k \cup \{\text{cost}_n\}$
  end for
  $n^* \leftarrow \arg\min \text{cost}_k$ ▷ Get the input with the lowest total cost
  $x_{k+1} \leftarrow \mathcal{F}_k(x_k, u_k^{n^*})$ ▷ Apply the best input and repeat
end for

Figure 5.1 shows an illustrative example of ADP with rollout. To select which of the inputs is the optimal in x_k, the approximated value function is takes as the sum of costs from the next state x_{k+1} to the end as

$$\tilde{V}_{k+1}(x_{k+1}|\pi) = \mathbb{E}_w[\sum_{j=k+1}^{N-1} g_j(x_j, \pi_j(x_j))] \tag{5.8}$$

where g_j is the stage wise cost function, x_j and u_j are the states at stage j and π_n

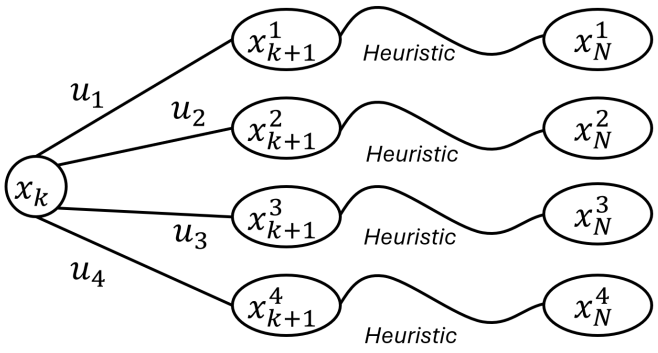


Figure 5.1: Conceptual explanation of rollout. At the current state x_k, the inputs $u_1 \cdots u_4$ can be selected. Each input leads to a different state at $k+1$. From these next states, the rollout heuristic generates a series of inputs that lead to different terminal staes x_N.

is the is the heuristic policy that follows from input u_n at stage k. The policy π_n is the policy from taking n^{th} possible input at station k.

At this next state x_{k+1}, a base heuristic is used to produce a valid set of inputs to get to the end of the route, with the terminal state x_N. This heuristic will produce a non-optimal route towards the end, and the cost of this route produced by the heuristic will be used as the approximated value function $\tilde{V}_k$. The primary algorithm presented in this thesis simply uses the deterministic planner described in Section 5.1. The ADP planner proposed in this thesis evaluates the heuristic on the full horizon, i.e. all the way to the end. Experimentally, it was determined that the charge planner has poor performance when the rollout heuristic only evaluated on a partial route, rather than on the full route to the end. The policy produced from the deterministic planner is evaluated on the the entire route using the ADP cost function in (5.9).

Since the deterministic planner only has SoC as a state, a cost function with the objective of minimizing drive time can not be constructed. By introducing the time of day as an extra state, time dependent disturbances can be introduced. This in turn permits drive time and energy consumption to be affected by the time of day rather than only by location. The extended state x therefor becomes $x = \begin{bmatrix} \text{soc} & t \end{bmatrix}^\intercal$. The inclusion of time as a state allows for the objective to be changed, minimizing both energy consumption and time consumption, by defining the cost function as,

$$g_k(x_k, u_k) = \mathbb{E}_w[c_{\text{charging},k} \cdot \Delta\text{soc} + c_{\text{time}} \cdot (t_{\text{stop}} + t_{\text{driving}})], \tag{5.9}$$

which does not include the range anxiety term but does include the drive time t_{driving}. The expectation in the cost function g_k is effectively ignored during computation, other than to mean that the simulated states use the mean value from the UKF. This can be done because the cost terms are linear, which means that the following holds for the cost function:

$$\mathbb{E}[f(x)] = f(\mathbb{E}[x]). \tag{5.10}$$

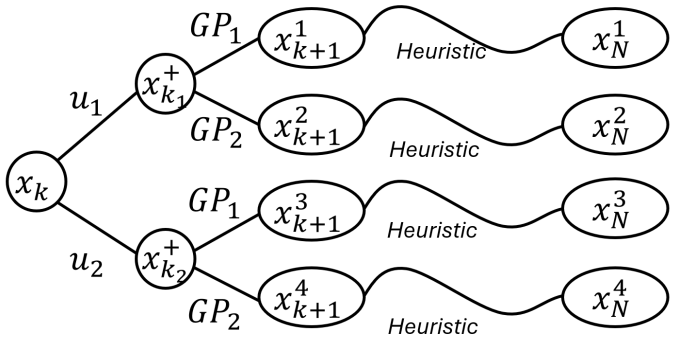


Figure 5.2: Conceptual overview of ADP with rollout using Gaussian processes to sample disturbances. From the current state x_k, the inputs u_1 and u_2 can be taken. After selecting each input, from the post charging states x_k^+, a different GP is sampled and the system is forward simulated with a sampled GP. The input u and sampled GP to land in a total of four different possible next states x_{k+1}. From each such state, the rollout heuristic is forward simulated to reach four possible final states x_N.

To convert this deterministic ADP formulation to a stochastic ADP formulation, disturbance terms w_k are introduced. In theory, the only difference between stochastic DP and deterministic DP is the inclusion of an expected value in the Bellman equation as

$$V_k(x_k) = \mathbb{E}_w \left[\min_{u_k \in \mathcal{U}_k(x_k)} g_k(x_k, u_k) + V_{k+1}(f_k(x_k, u_k, w_k)) \right]. \tag{5.11}$$

5.2.1 Sampling disturbances with Gaussian processes

One method to calculate distributions for uncertainties is to Monte Carlo sample different GP surfaces. Figure 5.2 shows a simple graph explaining how this works. A predefined number n_{GP} of GP surfaces are sampled at the beginning of planning. Each possible input $(u_1, \cdots, u_{Q_k})$ at each charging station k, is then applied when driving with speeds from each of the n_{GP} GP surfaces. This implies that there will be $n_{\text{GP}} \times Q_k$ different state trajectories from charging station k to the next charging station k+1. In Figure 5.2, this is represented by each state x^i_{k+1}, $i \in 1, \ldots, n_{\text{GP}} \times Q_k$, at station $k+1$.

Using this method, the input $u^* \in \mathcal{U}_k(x_k)$ is selected by averaging the cost of the current action as well as the cost of the heuristic is averaged across GPs. Additionally, only policies whose states stay within the allowed state set $\mathcal{X}_{k:M}$ for all stages until the terminal stage are considered during the selection. Ultimately, this method is not practically feasible as it takes too much time to sample enough GPs to get an accurate statistical representation of possible state evolutions.

5.3 ADP with chance- and soft constraints

In order to achieve robust performance to disturbances, the mean and variance trajectories from the UKF introduced in Chapter 4 is used to represent the statistical properties of the mission's uncertainties. These uncertainties are handled more specifically in terms of a chance constraint on the minimum SoC level. In addition, the planning algorithms gains flexibility by converting the cost function component of time to a soft constraint.

5.3.1 Chance constraint

To ensure we do not violate the minimum SoC constraint along the route, the plan has two countermeasures to ensure it has margin against the minimum SoC constriant. The first of these is a chance constraint of energy. The standard deviation in energy σ at each point is computed using the UKF introduced in Section 4.2. The minimum SoC constraint is then set such that n_{std} standard deviations from the mean is greater than the minimum SoC constraint of 10%, see (5.12).

$$\text{SOC}_i - n_{\text{std}}\sigma_i > \text{SOC}_{\text{min}} \tag{5.12}$$

This formulation permits the charge planner to use larger SoC margins where there is more state uncertainty, and smaller SoC margins where there is less uncertainty. For the ADP formulation specifically, this means the charge planner will be cautious with a long horizon, but then relax its bound on SoC as the truck gets closer to the end of the journey. Practically, this is implemented by modifying the permissible input set $\mathcal{U}_k(x_k)$ such that the chance constraint is not violated.

5.3.2 Soft constraint on time

A soft chance constraint on time is introduced to eliminate a tuning parameter, namely the price on time c_{time}. The goal with this formulation is to enable two types of charge planning, without retuning any part of the algorithm. By setting a maximum permissible arrival time t_{cutoff}, the behavior of the algorithm can be adjusted. If the the predicted arrival time t_N is before the cutoff time, the soft constraint is not active. This in turns means the total cost function minimizes the total charging cost of the mission. However, if the predicted arrival time is after latest allowed time of arrival t_{cutoff}, then the soft constraint is activated. By choosing a sufficiently high cost of delayed arrival c_{delay}, the cost function becomes dominated by this delay component, so the total cost function acts as a time minimizer. The stage-wise cost function $g_{0:M-1}$ then reduces to,

$$g_k(x_k, u_k) = \mathbb{E}_w[c_{\text{charging},k} \cdot \Delta\text{SOC}] \tag{5.13}$$

This means the soft constraint is implemented as a terminal cost, rather than as an optimization constraint. The implementation in practice puts all constraint handling in the cost function, instead of handling them separately. There is now a terminal

cost function g_M implemented as

$$g_M(x_M) = c_{\text{delay}} \cdot \max(t_M - t_{\text{deadline}}, 0). \tag{5.14}$$

This implementation effectively permits the charge planner to operate in different 'modes'. If the fastest possible arrival is desired, then the arrival time t_{deadline} can be set to $t_{\text{deadline}} = 0$. This turns the problem into a time minimization problem. The opposite also holds. If cheap charging is desired, setting t_{deadline} to a very high value means that time is not considered in the cost at all.

5.4 Naive heuristic planner

As a point of comparison, a naive heuristic planner is implemented as a comparative baseline. This charging heuristic does not produce a plan using DP. Instead, this planner charges the battery to 90% capacity at the first station after dropping below 30% SoC, using the fastest charger at each station. The charging quantity Δsoc at each charging station k is simply expressed as

$$\Delta\text{soc}_k = \begin{cases} 0.9 - \text{soc}_k & \text{if soc}_k < 0.3 \\ 0 & \text{else} \end{cases}. \tag{5.15}$$

This charger makes no consideration of what future energy consumption along the route will look like and only considers what the current state of charge is. If the heuristic charges, the planner will select the individual charger u_0 at the station which has the highest charging power.

6

Results

The following section presents the setup and evaluation of the proposed methods. The UKF formulation is evaluated on the Saarbrücken to Burgos route, for which two different disturbance scenarios are used. The unscented Kalman filter is compared against the Monte Carlo simulation baseline along the entire route. The charge planning formulations are evaluated on two different routes, for which four different disturbance scenarios are used. The two disturbance scenarios are; normal with $\sigma_{\text{traffic}} = 1.0\text{m/s}$, $\sigma_{\text{wind}} = 4.0\text{m/s}$, and large with $\sigma_{\text{traffic}} = 2.0\text{m/s}$, $\sigma_{\text{wind}} = 8.0\text{m/s}$.

6.1 Unscented Kalman filter evaluation

The UKF formulation is evaluated on the Saarbrücken, Germany to Burgos, Spain route (1419.4 km), considering both traffic and wind disturbances $f^{(\text{t})}, f^{(\text{w})}$. Driving is initiated at 6 a.m., and is evaluated on the entire route as if the battery had an infinite capacity. The two disturbance scenarios introduced above (normal, large) are considered, and in both cases the UKF is evaluated against a corresponding 2500 MC simulations while using the correlation lengths suggested in the method section. The hope is that this test will reveal the general performance of the filter in normal circumstances.

To verify the impact of removing the temporal kernel, a sweep over temporal correlation lengths and a sweep over spatial correlation lengths were conducted separately against 5000 MC runs. As mentioned previously in Section 4.2.1, the proposed hypothesis says that for sufficiently large temporal correlation lengths the filter performance will be accurate, regardless of spatial correlation length. Also, when the temporal correlation length is small, the filter should over estimate the variance.

Table 6.1: Terminal state comparison for the two disturbance scenarios, shown as mean ± 1 standard deviation. Evaluated on the Saarbrücken to Burgos route, with 2500 MC simulations for each disturbance. Normal correlation lengths were used.

Scenario	State	MC mean ± 1σ	UKF mean ± 1σ	bias [%]	std ratio
Normal	t [h]	18.182 ± 0.057	18.184 ± 0.068	0.010	1.183
	C [kWh]	1821.208 ± 59.928	1823.218 ± 66.325	0.110	1.107
Large	t [h]	18.259 ± 0.116	18.292 ± 0.135	0.136	1.164
	C [kWh]	1884.705 ± 121.333	1894.013 ± 134.213	0.494	1.106

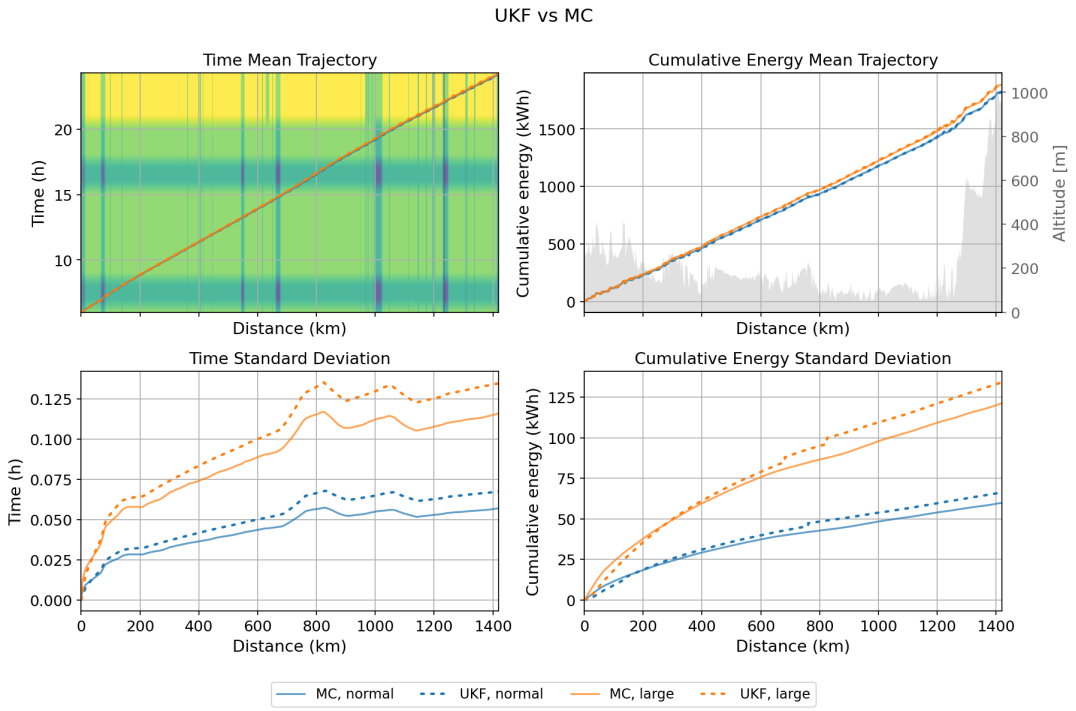


Figure 6.1: State propagation using UKF evaluated against 2500 MC simulations acting as the ground truth. The filter was tested for both disturbances, where MC is shown in solid lines and UKF is shown in dotted lines. The filter estimates the mean with small bias, while the standard deviations show larger discrepancies.

In Figure 6.1, we see that the proposed filter aligns closely with the MC simulation results. This is further confirmed in Table 6.1, where the largest mean bias is 0.494%, occurring in cumulative energy under large disturbances. The standard deviation of time shows a larger discrepancy than that of cumulative energy, though the means remain well aligned. In absolute terms, the time standard deviation is small throughout. The cumulative energy standard deviation is more substantial, particularly under large disturbances, where one standard deviation amounts to 134 kWh. This corresponds to 7.12% of the total cumulative energy consumed along the route, or 24.85% of the battery capacity reserved for charge planning. We can also note the compute times, as the UKF formulation predicted the full trajectory in 870.0 milliseconds, while MC took roughly 10 minutes to compute each trajectory.

The figure also shows how the standard deviation of time increases and decreases as the traffic wave reaches temporal bands, i.e. horizontal lines of similar traffic conditions. This is likely a modeling artifact, and will be elaborated on in the discussion.

The tests that were conducted to verify the implications of removing the temporal kernel are seen in Figure 6.2, Figure 6.3, Table 6.2, and Table 6.3. These confirm the expected result, and show that as the temporal correlation lengths are large, the estimated variance aligns closely with the true variance. When the spatial correlation length is then kept constant and the temporal correlation length is swept, a clear degenerative behavior can be observed as the temporal correlation length decreases.

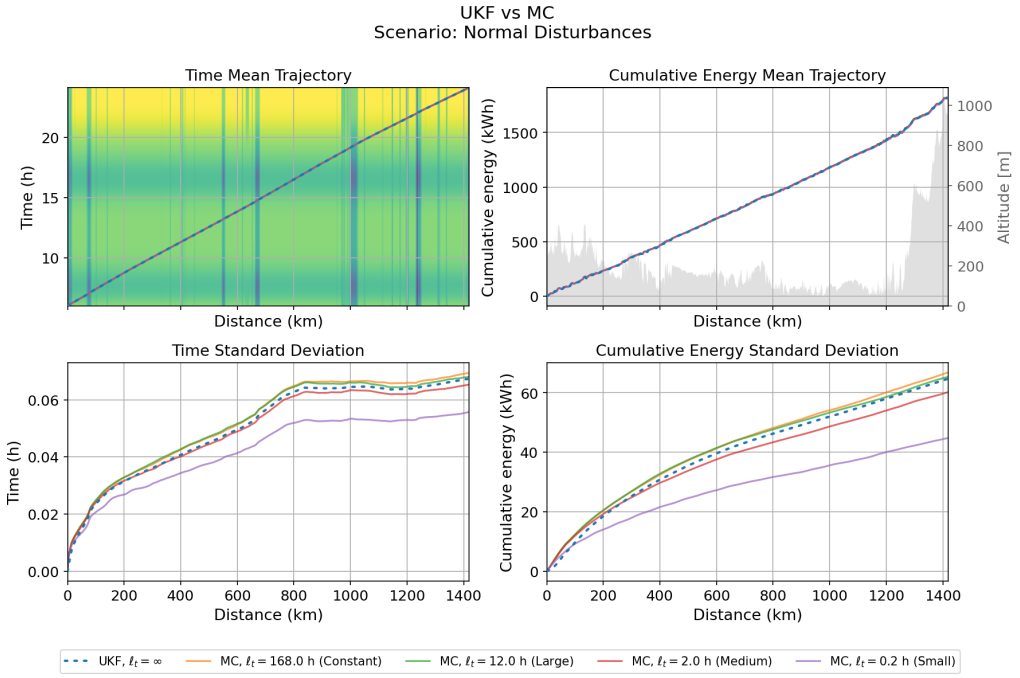


Figure 6.2: Estimation performance with the UKF for a varying temporal correlation length. The filter shows a degradation in performance as the temporal correlation decreases, which is in line with expected behavior.

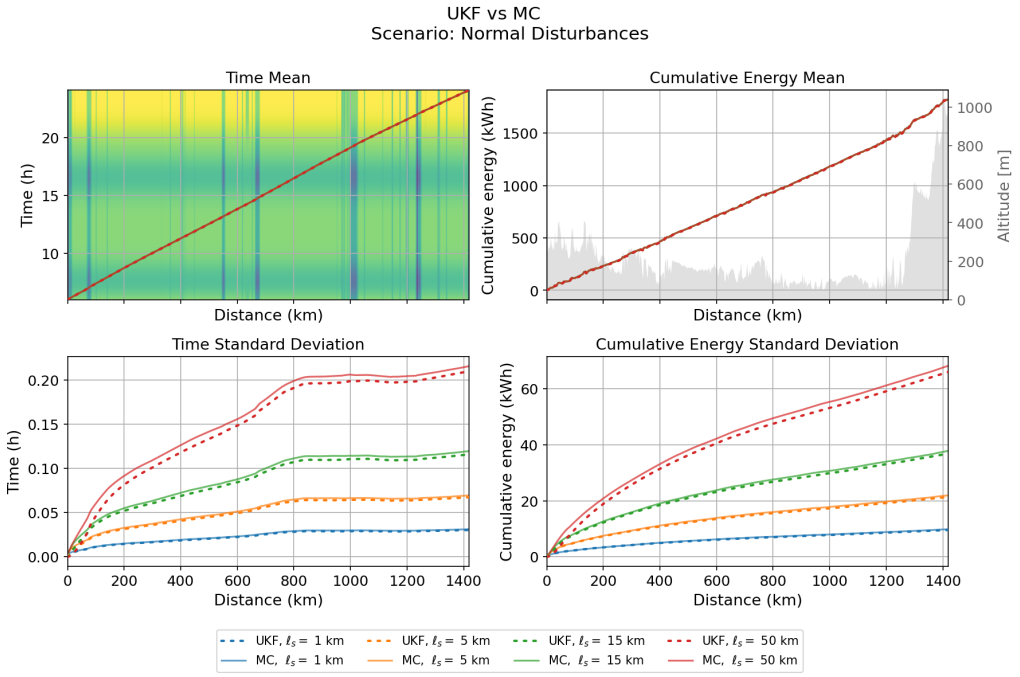


Figure 6.3: Estimation performance with the UKF for a varying spatial correlation length. The temporal correlation length was set to 168 hours, to align with the proposed simplification. The filter shows consistent performance over all spatial correlation lengths, which is in line with expected behavior.

Table 6.2: Terminal state comparison for the temporal-correlation run, shown as mean ± 1 standard deviation. Evaluated on the Saarbrücken to Burgos route, with 5000 MC simulations for each disturbance and correlation setting. The spatial correlation was fixed for traffic and wind at $\ell_s^{(\mathrm{t})} = 5$ km and $\ell_s^{(\mathrm{w})} = 50$ km. For each scenario, the UKF result is constant and thus listed once at the top. Bias is computed as $(\mu_{\mathrm{UKF}} - \mu_{\mathrm{MC}})/\mu_{\mathrm{MC}}$ and the std ratio as $\sigma_{\mathrm{UKF}}/\sigma_{\mathrm{MC}}$.

Scenario	ℓ_t	State	Mean ± 1σ	bias [%]	std ratio
Normal	UKF reference	t [h]	18.13 ± 0.07	–	–
	168.0 [h]	t [h]	18.14 ± 0.07	-0.016	0.97
	12.0 [h]	t [h]	18.14 ± 0.07	-0.011	0.99
	2.0 [h]	t [h]	18.13 ± 0.07	-0.002	1.03
	0.2 [h]	t [h]	18.12 ± 0.06	0.051	1.21
	UKF reference	C [kWh]	1821.5 ± 64.7	–	–
	168.0 [h]	C [kWh]	1822.2 ± 66.8	-0.037	0.97
	12.0 [h]	C [kWh]	1822.8 ± 65.4	-0.068	0.99
	2.0 [h]	C [kWh]	1820.3 ± 60.2	0.069	1.08
	0.2 [h]	C [kWh]	1814.6 ± 44.8	0.382	1.45
Large	UKF reference	t [h]	18.24 ± 0.14	–	–
	168.0 [h]	t [h]	18.25 ± 0.14	-0.041	0.95
	12.0 [h]	t [h]	18.24 ± 0.14	-0.030	0.97
	2.0 [h]	t [h]	18.23 ± 0.13	0.018	1.01
	0.2 [h]	t [h]	18.20 ± 0.11	0.230	1.19
	UKF reference	C [kWh]	1889.8 ± 129.5	–	–
	168.0 [h]	C [kWh]	1892.5 ± 135.9	-0.141	0.95
	12.0 [h]	C [kWh]	1893.6 ± 132.9	-0.199	0.97
	2.0 [h]	C [kWh]	1885.0 ± 122.2	0.256	1.06
	0.2 [h]	C [kWh]	1861.5 ± 90.6	1.521	1.43
UKF compute time [s], mean ± 1σ			1.28 ± 0.35		
MC compute time [s], mean ± 1σ			940.8 ± 30.7		

Table 6.3: Terminal state comparison for the spatial-correlation run, shown as mean ± 1 standard deviation. Evaluated on the Saarbrücken to Burgos route, with 5000 MC simulations for each disturbance and correlation setting. The temporal correlation length was fixed at $\ell_t = 168$ h.

Scenario	ℓ_s [km]	State	MC mean ± 1σ	UKF mean ± 1σ	bias [%]	std ratio
Normal	1	t [h]	18.14 ± 0.03	18.13 ± 0.03	−0.016	0.97
	5	t [h]	18.14 ± 0.07	18.13 ± 0.07	−0.016	0.97
	15	t [h]	18.14 ± 0.12	18.13 ± 0.12	−0.016	0.97
	50	t [h]	18.14 ± 0.22	18.13 ± 0.21	−0.018	0.97
	1	C [kWh]	1823.2 ± 9.8	1823.1 ± 9.6	−0.008	0.98
	5	C [kWh]	1822.3 ± 21.9	1822.0 ± 21.3	−0.015	0.97
	15	C [kWh]	1822.2 ± 37.8	1821.8 ± 36.8	−0.019	0.97
	50	C [kWh]	1822.1 ± 68.1	1821.4 ± 65.9	−0.040	0.97
Large	1	t [h]	18.25 ± 0.06	18.24 ± 0.06	−0.037	0.95
	5	t [h]	18.25 ± 0.14	18.24 ± 0.14	−0.041	0.95
	15	t [h]	18.25 ± 0.25	18.23 ± 0.23	−0.064	0.94
	50	t [h]	18.25 ± 0.44	18.23 ± 0.42	−0.123	0.94
	1	C [kWh]	1896.1 ± 20.0	1897.0 ± 19.4	0.049	0.97
	5	C [kWh]	1892.3 ± 44.6	1891.7 ± 42.7	−0.032	0.96
	15	C [kWh]	1892.1 ± 76.8	1890.2 ± 75.1	−0.102	0.98
	50	C [kWh]	1892.2 ± 138.6	1890.0 ± 138.1	−0.115	1.00
UKF compute time [s], mean ± 1σ				1.36 ± 0.22		
MC compute time [s], mean ± 1σ				1028.9 ± 229.1		

6.2 Charge planner evaluation setup

This section explains the setup used to simulate and evaluate the DP charge planners. The routes, simulations scenarios, and the metrics used for evaluation are explained.

6.2.1 Routes

The charge planner is evaluated on two different routes. The first route is Barcelona, Spain to Genoa, Italy, this route is 825.4 km long. The second route is Saarbrücken, Germany to Burgos, Spain, this route is 1419.4 km long. Figures 6.4 and 6.5 features the time and space dependent speed limit of the routes, as well as the altitude of the road along the respective routes. The red dots represent charging stations.

6.2.2 Simulation scenarios

Multiple simulation scenarios are run on each of the routes introduced in the previous section. The first of these is a deterministic experiment. This means there is no variance in either the traffic or in wind speed. The mean speed reduction from driving in cities and during rush hour is still active in this simulation, however.

The second scenario has "normal" disturbance for both the traffic and wind. This means a standard deviation in traffic speed σ_{traffic} of 1.0 m/s and a standard deviation

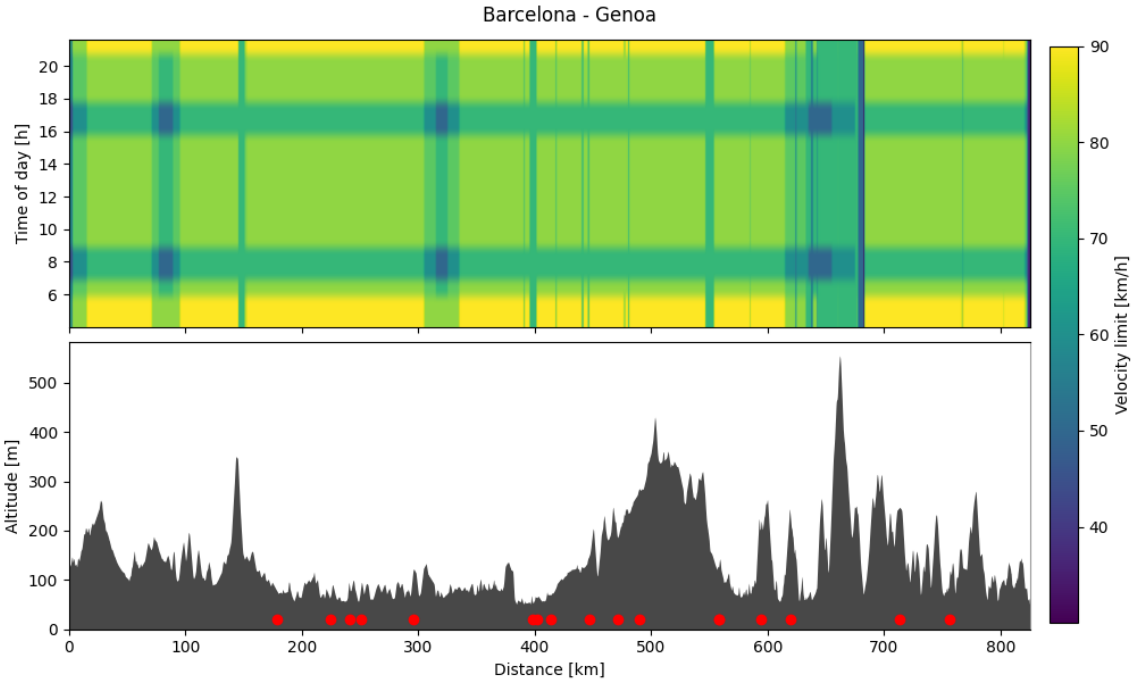


Figure 6.4: Time dependent speed limit and altitude plot of the Barcelona - Genoa route. The intermediary cities that have been placed are Girona at 83 km, Montpellier at 320 km, and Nice at 645 km. The red dots represent charging stations along the route.

of wind speed σ_{wind} of 4.0 m/s, as mentioned previously. These standard deviation values are known by the controller. This scenario represents our average variance that can be expected along any route when charge planning.

The third scenario has a larger disturbance for both traffic and wind. As mentioned previously, the standard deviation on traffic speed σ_{traffic} is 2.0 m/s and the deviation on wind speed σ_{wind} is 8.0 m/s for the large disturbance scenario. A standard deviation on wind of 8.0 m/s represent a very windy route, but still falls in the realm of possibility.

The fourth and final scenario represents a model mismatch in the disturbance. While the actual simulated standard deviation are the larger values from the third scenario, the ADP planners are programmed with the lower values from the normal scenario. This scenario tests how well the planners handle parameter errors in the disturbance values. Poor performance in this scenario means that the planners must be programmed with accurate distributions to provide a good charging policy.

6.2.3 Evaluation metrics

Each simulation is evaluated on a number of metrics to determine the performance of the planner. The first of these is the total mission time T_{tot}, measured from the departure to the arrival at the destination. The second of these is the stoppage time T_{stop}. This includes the detour drive time off the route and to the charging station, a dead time representing queuing, the time to connect and disconnect from the

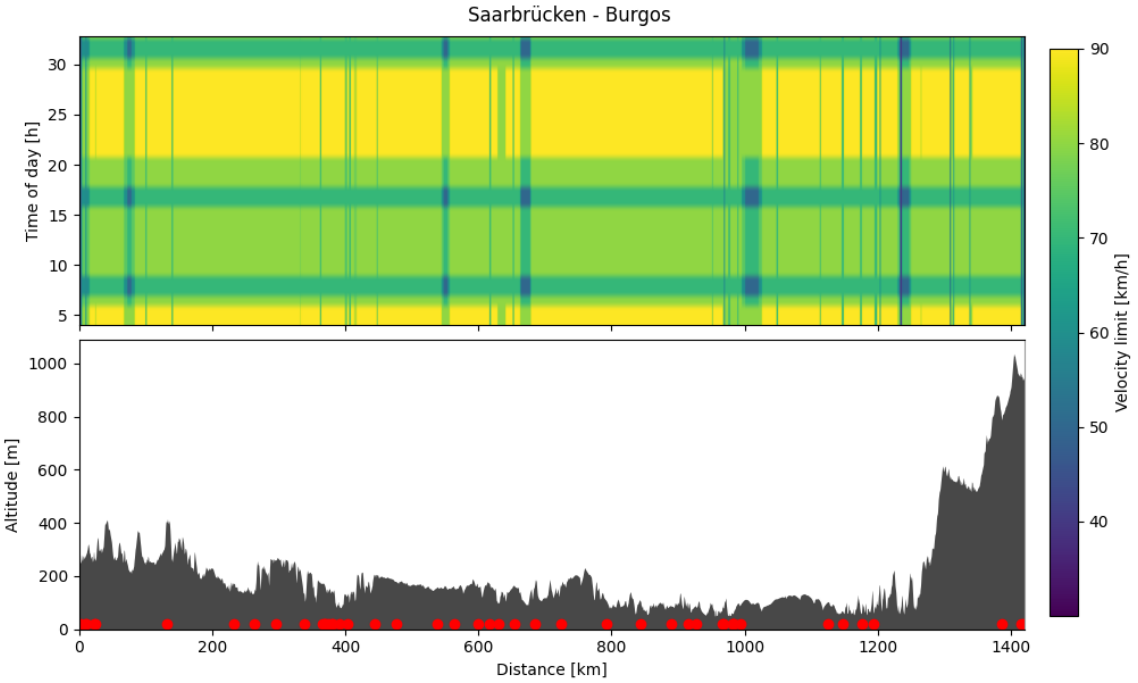


Figure 6.5: Time dependent speed limit and altitude plot of the Saarbrücken - Burgos route. The intermediary cities that have been placed are Metz at 75 km, Orleans at 550 km, Tours at 670 km, Bordeaux at 1010 km, and San Sebastian at 1239 km. The red dots represent charging stations along the route.

charging, as well as the actual time connected to the charger. The thrid evaluation metric is the total energy expenditure E, which includes all energy consumed from the battery during the trip.

The fourth evaluation metric is the number of stops N_{stop}. While the number of stops is not directly a measure of performance, the number of stops can often help explain the time performance of the planner. Finally, two constraint metrics are evaluated. The minimum SoC along the route SoC_{min} and the state of charge at the destination $\text{SoC}_{\text{final}}$. As a reminder, the planners are constrained to never dropping below 10% SoC as a minimum safety margin. In addition, the planners are constrained to arrive at the final destination with at least 50% SoC remaining.

Finally, the time to compute a calculation and complete the forward simulation T_{compute} is reported for the deterministic scenario. This time in principle only depend on the route and planner used, and is the same regardless of the size of the disturbance.

6.3 Charge planner evaluation

The results of the simulation is presented in the tables below. There is one table per route and scenario introduced in Section 6.2.2. In addition, representative examples is highlighted for select simulation scenarios. The method names in the table corresponds as follows: *Full DP* represents the one state planner introduced in Sec-

tion 5.1, *Heuristic* represents the naive heuristic planner introduced in Section 5.4. There are two implementations of the ADP planners tested. Firstly, *ADP 1* represents the ADP formulation as stated in Section 5.2. Secondly, *ADP 2* represents an ADP planner using the same framework as *ADP 1* but with the naive heuristic planner used as the rollout heuristic, instead of the one state exact DP planner. This section will present select results that are representative of all results. More tables with simulation results can be found in Appendix C.

6.3.1 Barcelona - Genoa

The first results in Table 6.4 shows that the main ADP planner is able to complete the route 90 minutes quicker that the full DP counterpart that does not consider time as a state. The naive heuristic planner is the quickest of all planners, but it violates the end-of-route SoC constraint by over 25 % SoC. As there is no disturbance, the deterministic scenario is only evaluated once. Table 6.5 evaluates the

Table 6.4: Deterministic scenario results for the Barcelona - Genoa route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$	T_{compute} [s]
Full DP	1021	378	1026	5	0.4838	0.4922	0.41
ADP 1	930	268	1012	3	0.1787	0.5107	253
Heuristic	895	236	1010	2	0.2468	0.2861	0.40
ADP 2	1095	456	1034	5	0.4850	0.5194	123

planning outcomes of the Barcelona - Genoa route when using normal disturbances. The plus minus values in the table represent the standard deviation of the result. Each individual simulation in simulations with disturbances uses a different fixed seed to generate the disturbances which means the simulation runs differ from each other while still being reproducible. For each scenario, 7 runs are completed. The complete ADP implementation, denoted ADP 1, saves around 90 minutes on average compared to the deterministic planner without time as a state. The minimum SoC column shows that this is achieved by the ADP planner by permitting itself to reach a lower state of charge. The cost of low SoC parameter w_{soc}, see (5.1), is tuned in such a way that the exact DP formulation clears all constraints on all routes which means it becomes conservative on the comparatively easier route of Barcelona to Genoa. The compute time significantly longer for the ADP planners while being nearly instant for the full DP and heuristic planners.

Table 6.5: Normal disturbance scenario results for the Barcelona - Genoa route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$
Full DP	1022 ± 16.1	377 ± 15.8	1052 ± 41.4	4 ± 0.4	0.4107 ± 0.0133	0.4887 ± 0.0245
ADP 1	932 ± 5.8	266 ± 5.8	1019 ± 26.1	3 ± 0.0	0.1754 ± 0.0315	0.4896 ± 0.0137
Heuristic	959 ± 105.1	300 ± 107.9	1029 ± 36.7	2 ± 0.4	0.2251 ± 0.0517	0.3918 ± 0.1857
ADP 2	1095 ± 32.7	449 ± 34.6	1040 ± 47.6	6 ± 0.7	0.4134 ± 0.0384	0.5221 ± 0.0235

An example simulation of the ADP planner with normal disturbance is in Figure 6.6. The x-axis represents distance along the route. In the top plot, the y-axis

represents the time of day, the red line represents driving through time and space and the background colors represent the simulated traffic speed on the road. In the bottom plot, the blue line represents the state of charge, the gray background the altitude of the road, the green dots represents the charging station where the truck stopped to charge and the red dots represents the charging station that the truck drove past without stopping. The height of the green dots represent how much the truck charged.

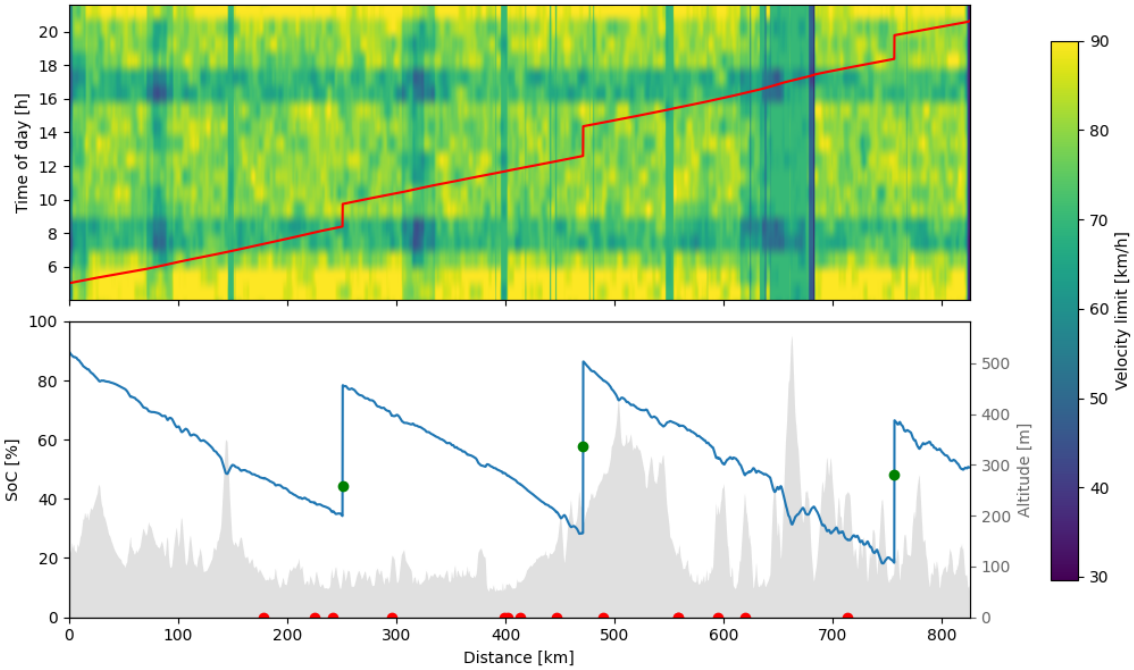


Figure 6.6: Charge planning simulation for the Barcelona - Genoa route with the primary ADP planner and normal disturbances.

Table 6.6: Large disturbance scenario results for the Barcelona - Genoa route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$
Full DP	1038 ± 29.2	390 ± 27.0	1102 ± 87.7	4 ± 0.4	0.3772 ± 0.0292	0.4719 ± 0.0540
ADP 1	955 ± 28.8	294 ± 38.6	1074 ± 84.4	4 ± 0.9	0.1576 ± 0.0208	0.4792 ± 0.0383
Heuristic	996 ± 110.5	335 ± 114.9	1061 ± 73.1	2 ± 0.4	0.1749 ± 0.0700	0.3798 ± 0.1718
ADP 2	1130 ± 54.8	483 ± 59.2	1069 ± 85.4	6 ± 1.2	0.4210 ± 0.0738	0.5280 ± 0.0425

The results are similar for the large disturbance test, see Table 6.6. The time gain is now on average 83 minutes while saving energy on average for the ADP compared to the single state full DP planner. The simple heuristic planner and the 2nd ADP planner with the simple heuristic as its rollout heuristic produce poor results on all evaluated scenarios. The simple heuristic in ultimately unaware of the SoC constraints both during the route and at the destination, and therefor violates these constraints. The ADP planner with a simple heuristic does not violate constraints, but still has worse performance in terms of total mission time T_{tot}.

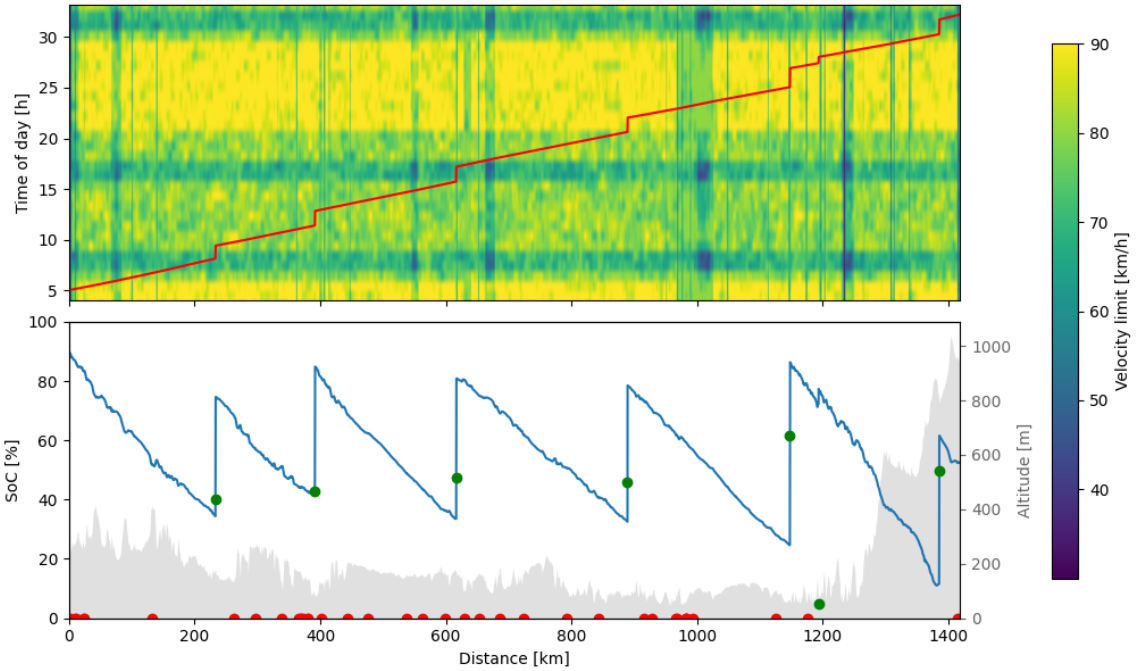


Figure 6.7: Charge planning simulation for the Saarbrücken - Burgos route with the primary ADP planner and normal disturbances. The charge planner makes a small charging stop at km 1200 in order to fulfill the SoC constraint along the entire route.

6.3.2 Saarbrücken - Burgos

The results for the Saarbrücken to Burgos route largely agree with the results from the Barcelona to Genoa route. Table 6.7 has the results for the Saarbrücken - Burgos simulation with normal disturbance. There is an improvement of 69 minutes on average for the ADP planner compared to the DP planner. There is a small constraint violation for the ADP planner, and the standard deviation indicates there are worse outliers. Again, the ADP planner with naive rollout, ADP 2, has the worst performance among all planners in terms of total time T_{tot} and number of stops, N_{stop}.

Table 6.7: Normal disturbance scenario results for the Saarbrücken - Burgos route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$
Full DP	1707 ± 26.2	629 ± 22.1	1849 ± 57.5	7 ± 0.5	0.2167 ± 0.0245	0.5289 ± 0.0075
ADP 1	1638 ± 32.6	572 ± 31.0	1864 ± 36.4	7 ± 1.0	0.0997 ± 0.0256	0.5146 ± 0.0157
Heuristic	1736 ± 18.9	665 ± 16.4	1845 ± 39.0	5 ± 0.0	0.1604 ± 0.0576	0.7963 ± 0.0039
ADP 2	1769 ± 38.9	684 ± 39.4	1828 ± 55.4	10 ± 1.1	0.1185 ± 0.0219	0.5185 ± 0.0059

Figure 6.7 shows a charge planning simulation using the same plot description as 6.6. The charge planner underestimates the energy consumption when climbing the final hill and must make a small additional charging stop towards the end. Charging stops 1 and 3 both happen during rush hour. This reduces the total mission time by charging when the truck otherwise would be driving through slow traffic.

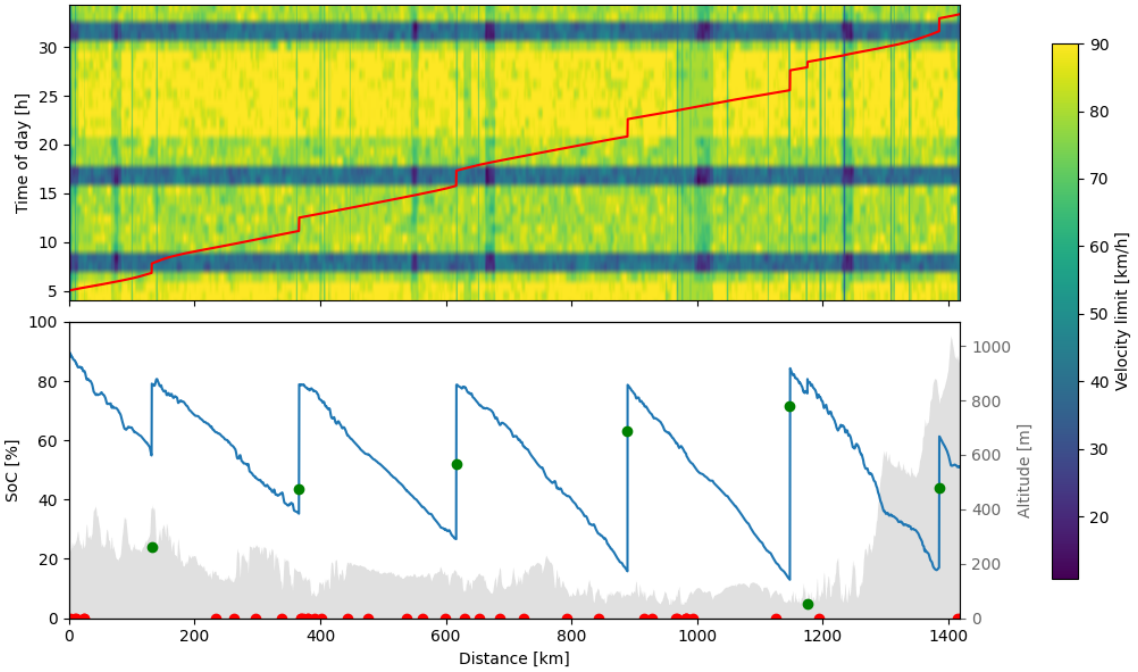


Figure 6.8: Saarbrücken - Burgos evaluated with the ADP planner with reduced speeds in rush hour. Note that there are three charging stops during different sections of rush hour traffic, which saves on the total mission time of 28 hours and 23 minutes.

For the large disturbance case, the ADP planner becomes worse that the base DP planner, see Table 6.8. The standard deviation for time T_{tot} is the largest among all test cases for both ADP planners in this specific scenario, which follows from the large variance in the number of stops.

Table 6.8: Large disturbance scenario results for the SAR-BUR route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$
Full DP	1666 ± 6.2	595 ± 3.5	1842 ± 8.9	7 ± 0.0	0.1375 ± 0.0133	0.5458 ± 0.0058
ADP 1	1767 ± 69.9	676 ± 63.1	1771 ± 14.7	10 ± 1.7	0.1540 ± 0.0419	0.5328 ± 0.0131
Heuristic	1818 ± 5.2	717 ± 3.3	1704 ± 8.0	5 ± 0.0	0.2372 ± 0.0125	0.8965 ± 0.0009
ADP 2	1801 ± 67.5	716 ± 70.8	1881 ± 21.2	10 ± 1.8	0.1062 ± 0.0591	0.5257 ± 0.0106

To further evaluate the differences when planning with time as a state, a test was made evaluating on worse-than-standard rush hour traffic. The speed limit in rush hour in cities was reduced to 20 km/h, in urban areas to 30 km/h, and to 40 km/h on the open freeway. The ADP planner completed the route in 28 hours and 23 minutes, while the one state DP planner completed the route in 30 hours and 17 minutes minutes. Studying Figures 6.8, the ADP planner, and comparing it with Figure 6.9, the DP planner, clearly reveals the time saving. The ADP planner is able to stop and charge during rush hour, while the DP planner is unaware it will hit traffic and therefor drives through slower traffic and instead charges when the road is clear and the speed is high.

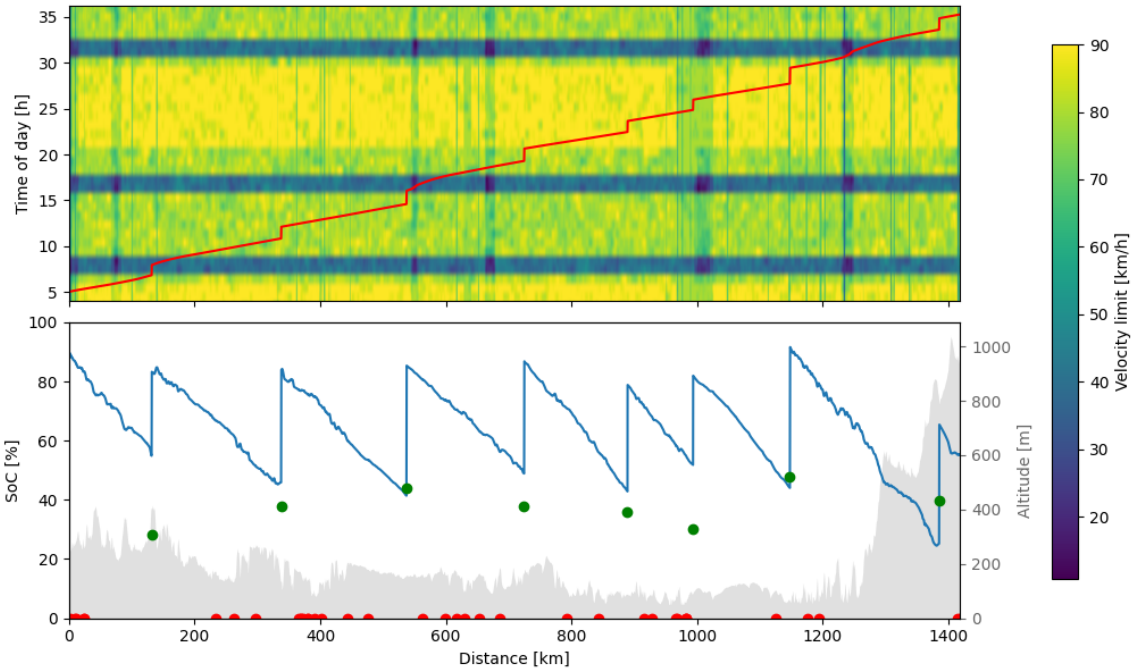


Figure 6.9: Saarbrücken - Burgos evaluated with the one state DP planner with reduced speeds in rush hour. Note that only charging stop happens during rush hour traffic. Because the truck has to drive through rush hour traffic, the total driving time increases to 30 hours and 17 minutes.

7

Discussion

In this chapter, we will discuss the methods and results used in this thesis. Potential shortcomings and avenues for future work are also discussed.

7.1 Discussion on propagation

From the results, it is clear that there is a substantial difference between computational cost for the two approaches. A single MC trajectory requires approximately 200 milliseconds, and the state distributions converge after approximately 2000 MC simulations. This means that a representative distribution is calculated in approximately 6.5 minutes. In comparison, the UKF filter computes all distributions along a trajectory in 870 ms. In the same time, the MC simulation produces four samples. Four samples are wholly insufficient to characterize a distribution, and the number of runs required for convergence pushes the effective MC cost to several orders of magnitude higher.

The proposed simplification of removing the temporal kernel from the GP is well justified by the results. By sweeping the temporal and spatial correlation lengths, we see that for sufficiently large temporal correlation lengths the UKF estimation is accurate regardless of spatial correlation length. We also see that small temporal correlation length degrade filter performance. This is consistent with estimating the temporal correlation as constant, as this is conceptually equivalent with replacing it with an infinite correlation length. For reasonable temporal correlation lengths, the filter shows a systematic tendency to overestimate the standard deviation. Also, when the temporal correlation length is unreasonably large, the UKF estimate somewhat underestimates the standard deviation as well. We believe that the filter underestimations are due to sigma point calculations as this is only accurate to the second moment. We also believe that the overestimates is due to the variance reduction effect which the temporal kernel has in the ground truth simulations. The figures also show that there appears to be some variance limit, which is consistent with the proposed explanation. Given that the proposed simplification substantially reduces model complexity, the trade-off is acceptable for the traffic conditions studied.

The validity of the Gaussian approximation underlying the UKF depends on whether the state distribution remains compact and approximately symmetric throughout the trajectory. Consider an ensemble of drivers departing under identical initial

conditions. Since the dynamics are integrated spatially, uncertainty accumulates along the route, and at constant mean speed the noise distributes arrival times as a roughly Gaussian wave in space, seen in Figure 4.1. This wave can be sheared by two mechanisms. Firstly, a steep velocity gradient intersecting only part of the ensemble, splitting the distribution at a congestion boundary. Secondly, when variance accumulation over long distances, where tails encounter systematically different conditions and the distribution skews. Whether either effect becomes significant depends on the ensemble width relative to the spatial scale of the velocity field. In the simulations, the state standard deviation remains small throughout, keeping the ensemble compact and the Gaussian structure approximately intact. However, this is conditional on the stationary variance of the modeled traffic, as well as the temporal and spatial mean velocity bands. Real traffic may not exhibit those properties, so the range over which the distribution remains Gaussian, and whether the UKF continues to track it accurately, should be evaluated against real data with a more representative mean surface and a non-stationary variance model. The evaluation would also clarify whether a full GP-to-state-space transformation is necessary, or whether estimating the traffic process by one input dimension is sufficient.

7.2 Discussion on planning

There is an improvement in drive time when using the full ADP implementation with chance constraints compared to using the full DP planner using the $c_{\text{low soc}}$ penalty term as the sole mitigation against uncertainties. The energy usage between the different planners remains relatively consistent. The largest discrepancies in power consumption between the planners come from driving in headwind or tailwind. Using the naive heuristic planner as a rollout heuristic produces worse result than just using the base DP planner in many cases. This highlights the need for a robust rollout heuristic in this setting. Using just the base heuristic as a planner performs poorly across all evaluated settings. This is consistent with Borghed's previous thesis in the field [11].

Charge planning necessarily makes a lot of assumptions. We have only done single step look ahead exactly. As can be seen in Figure 6.8 for example, the charge planning occasionally implements micro charging, i.e. a small charging stop near another larger charging stop. The time loss from this may not be excessively bad however, given that the charging time increases significantly past 80% SoC, see Figure 3.2. In fact, some micro charging events may rather be that the time lost from charging at a lower power is greater than the time lost from driving to another charger.

Another problem with the ADP planner is the significant increase in compute time as implemented, compared to the one state DP planner. While ADP provides an improvement in compute time when compared like-for-like to an equivalent exact DP formulation, this property does not hold when implementing additional states into the ADP planner. Real time implementation is not possible with the current formulation of the ADP planner, it is simply too slow. A compute time of over 11 minutes on a standard laptop would prove challenging to implement in real time.

This presents a serious obstacle from increasing the number of exact look-ahead steps that are feasible to compute. You effectively end up with compute time complexity $\mathcal{O}(n^s)$, where s is the number of steps you choose to solve exactly. Significant optimization is required to make multi-step look ahead feasible.

Our planner becomes very dependent on the type of disturbances you choose to model. An advantage to the inclusion of a generic penalty of low SoC, like in the exact DP planner, is that the planner becomes robust to unknown disturbances regardless of what they are or how they affect the energy consumption or drive time of the route. A combination of chance constraint and penalty on low SoC may contribute to a further increase in planner robustness.

7.3 Implications of modeling

Compared to just adding constant Gaussian noise, independent at each timestep, the GP, and subsequent UKF formulation allows specific processes to be modeled. The GP is tuned from fitting directly from data, and is easier to extend to more types of disturbances.

The UKF filter shows a time standard deviation with a non-uniform growth pattern, alternating between contraction and expansion of the traffic wave. This is thought to be governed by the velocity gradient in the temporal direction. This is best illustrated by imagining an ensemble of drivers, who's probability mass is Gaussian distributed. When this ensemble approaches a horizontal negative gradient band, i.e. the velocity is decreasing, as seen when the mean moves from normal traffic to rush-hour traffic, the drivers who are already delayed will enter the gradient zone first. These drivers will then spend more time driving the same discretization distance in space, and as such the standard deviation of the whole ensemble will grow. When the whole ensemble has fully passed through the gradient, the standard deviation process will continue to grow, now at a slightly increased rate since traffic is moving slower. It will take comparatively longer to traverse the same distance, and more noise can then enter the system. The opposite happens when driver ensemble is passing through a positive gradient zone, i.e. speed is increasing, and as such the standard deviation shrinks. However, the patterns are thought to be exaggerated in the simulation as the gradient zones in the temporal direction are artificially large. This is caused by the sparse distributional means given in the temporal direction, as the values are smoothed between those in a uniform-like way.

In this thesis we have considered disturbances from traffic and wind. These disturbances affect the driving time and energy consumption while driving and are continuous in nature. These will affect the system in small and large amounts, and we have chosen to model these uncertainty sources as Gaussian processes. This is a modeling structure that does not fit well for other kinds of uncertainties, such as a sudden and wholly unexpected congestion on the road, like what would follow for example a major accident. Disturbances during charging, such as wait times and power discrepancies at chargers are also not modeled as sources of uncertainty. In order to extend the controller to handles these kind of uncertainties, the UKF filter

may have to be extended with some other method of estimation.

7.4 Outlook

From the results and discussion, a few interesting future directions to investigate have been selected. As of now, the deterministic planner is tuned by a range anxiety term, which has to be tuned manually. An idea is to instead incorporate the uncertainty from the UKF into the DP planner, as these will approximately give bounds for some set statistical chance. If the UKF distributions are accurate, this could provide a formulation which does not require tuning, as the uncertainty is based on the distance between chargers directly from the disturbance processes.

Currently, there is only one exact step evaluated with all states in the ADP planner. Further work could investigate the impact of multi-step look ahead. This would need to investigate in combinations with method to speed up computations for the ADP planner. This could include Monte Carlo tree search [24] or other similar techniques.

The thesis has also investigated two generic disturbances, namely traffic and wind. Moving forward, it would be important to identify the primary sources of uncertainty and model those specifically to improve real-world alignment. This in turn may lead to better planning as the SoC margins likely closely reflect actual uncertainty bounds. Some of these disturbances that are thought to have a major impact on driving dynamics are related to charging, as the queue time uncertainty and stated power discrepancies may be substantial.

Finally, the proposed UKF filter has been shown to work well in simulations. However, as discussed previously, there are concerns for performance degradation over long distances and sharper mean surfaces. To validate the filter framework, the UKF should be evaluated on a traffic process fitted to real traffic data, as this might indicate the true limitations of both propagation distance and the framework simplifications. The inclusion of non local observations while planning is also yet to be addressed, and is seen as another substantial area of further work.

8

Conclusion

This thesis has investigated how to formulated a DP framework for effective disturbance handling. One of the primary results is that an ADP solver using chance constraints from the UKF filter provides an adequate plan to avoid constraint violations. This was done by using one step exact DP for checking chance constraint violations while also incorporating time as a state to avoid traffic congestion. The remaining stages were evaluated using a rollout heuristic to approximate the cost-to-go from the subsequent stage. The proposed planner effectively produces policies that avoids congestion, and for larger congestion also shows major savings on mission time compared to baseline planners. However, the ADP planner is prone to micro-charging if tuning is not done carefully, likely due to policy mismatch between the ADP planner and its heuristic. Nonetheless, the proposed ADP approach is promising, and should be a starting point for future work.

The inclusion of correlated disturbances was also considered, permitting a larger class of disturbances than uncorrelated noise to be used in modeling. This thesis investigated the use of Gaussian processes as disturbance models, as they provide a very generic framework to model continuous disturbances with. Gaussian processes also permit an equivalent LTI SDE form for certain structures. The identity between a GP and an LTI SDE system was exploited in a UKF filter, which showed good state prediction for the modeled disturbances over the entire trajectory. Compared to retrieving state distributions using MC simulations, a UKF filter is computationally very efficient, and its use was seen to be essential for the ADP formulation. However, it is thought that the disturbance models considered may not have been fully representative of real world conditions, and as such the UKF formulation has to be further validated using real data to find the valid operating range.

The drawback of the proposed charge planning method is the significantly increased compute time. This presents a hurdle for future implementation in a real product, where this plan needs to be created nearly instantaneously. The relatively minor gains are a result of the specific disturbances modeled, and other disturbances would likely affect the result in a different way. Notably, optimizing charging decisions under the constraint of driver rest time regulations is a strong reason to implement the ADP planner with time as a state.

Bibliography

[1] Eurostat, *Greenhouse gas emissions by source sector*, 2025. [Online]. Available: https://ec.europa.eu/eurostat/databrowser/bookmark/c1e9a542-d24d-4de7-b448-59ad63eb6d8b.

[2] A. O'Connell, N. Pavlenko, G. Bieker, and S. Searle, *A Comparison of the Life-cycle Greenhouse Gas Emissions of European Heavy-Duty Vehicles and Fuels*, en, Feb. 2023. [Online]. Available: https://theicct.org/wp-content/uploads/2023/02/Lifecycle-assessment-trucks-and-buses-emissions-Europe.pdf.

[3] IEA, "Global EV Outlook 2025," en, Jul. 2025. [Online]. Available: https://www.iea.org/reports/global-ev-outlook-2025/trends-in-heavy-duty-electric-vehicles (visited on 05/04/2026).

[4] R. Danielis, A. M. K. Niazi, M. Scorrano, M. Masutti, and A. M. Awan, "The Economic Feasibility of Battery Electric Trucks: A Review of the Total Cost of Ownership Estimates," en, *Energies*, vol. 18, no. 2, p. 429, Jan. 2025, ISSN: 1996-1073. DOI: 10.3390/en18020429. [Online]. Available: https://www.mdpi.com/1996-1073/18/2/429 (visited on 05/04/2026).

[5] Transportstyrelsen, *Regler om kör- och vilotider - Transportstyrelsen*, sv, Oct. 2025. [Online]. Available: https://www.transportstyrelsen.se/sv/vagtrafik/yrkestrafik/kor-och-vilotider/regler-om-kor--och-vilotider/ (visited on 05/04/2026).

[6] T. Bai, Y. Li, K. H. Johansson, and J. Mårtensson, "Rollout-Based Charging Strategy for Electric Trucks With Hours-of-Service Regulations," *IEEE Control Systems Letters*, vol. 7, pp. 2167–2172, 2023, ISSN: 2475-1456. DOI: 10.1109/LCSYS.2023.3285137. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10147895 (visited on 05/04/2026).

[7] G. Huber, K. Bogenberger, and H. Van Lint, "Optimization of Charging Strategies for Battery Electric Vehicles Under Uncertainty," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 760–776, Feb. 2022, ISSN: 1524-9050, 1558-0016. DOI: 10.1109/TITS.2020.3027625. [Online]. Available: https://ieeexplore.ieee.org/document/9226119/ (visited on 05/04/2026).

[8] Y. Wan, Z. He, Y. Gao, and Y. Xue, "Long-haul truck charging planning problem considering time flexibility and energy flexibility," *Energy*, vol. 306, p. 132 361, Oct. 2024, ISSN: 0360-5442. DOI: 10.1016/j.energy.2024.132361. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360544224021352 (visited on 05/04/2026).

[9] M. Zähringer, S. Wolff, J. Schneider, G. Balke, and M. Lienkamp, "Time vs. Capacity—The Potential of Optimal Charging Stop Strategies for Battery Electric Trucks," en, *Energies*, vol. 15, no. 19, p. 7137, Jan. 2022, ISSN: 1996-1073. DOI: 10.3390/en15197137. [Online]. Available: https://www.mdpi.com/1996-1073/15/19/7137 (visited on 05/04/2026).

[10] P. K. Bhat, Z. Wu, and B. Chen, "Long Trip Charging Planning of Battery Electric Vehicle Considering Vehicle Waiting Time Forecast at Fast Charging Stations: A Mixed-Integer Dynamic Programming Approach," *IEEE Access*, vol. 13, pp. 52 100–52 113, 2025, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2025.3552730. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10933962 (visited on 05/04/2026).

[11] I. Borghed, "Hierarchical optimization for charge planning and thermal management of battery electric trucks," M.S. thesis, Chalmers University of Technology, 2024.

[12] I. Kuangaliyev and O. Mark, "Mission Management for Fuel Cell Electric Trucks - a Bilevel Approach," en, M.S. thesis, Chalmers University of Technology, Gothenburg, Sweden, 2024. [Online]. Available: http://hdl.handle.net/20.500.12380/308774.

[13] Z. Liu, C. Lyu, Z. Wang, S. Wang, P. Liu, and Q. Meng, "A Gaussian-Process-Based Data-Driven Traffic Flow Model and Its Application in Road Capacity Analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 2, pp. 1544–1563, Feb. 2023, ISSN: 1558-0016. DOI: 10.1109/TITS.2022.3223982. [Online]. Available: https://ieeexplore.ieee.org/document/10025463 (visited on 05/04/2026).

[14] Y. Xie, K. Zhao, Y. Sun, and D. Chen, "Gaussian Processes for Short-Term Traffic Volume Forecasting," en, *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2165, no. 1, pp. 69–78, Jan. 2010, ISSN: 0361-1981, 2169-4052. DOI: 10.3141/2165-08. [Online]. Available: https://journals.sagepub.com/doi/10.3141/2165-08 (visited on 05/04/2026).

[15] Z. Cheng, X. Wang, X. Chen, M. Trépanier, and L. Sun, "Bayesian Calibration of Traffic Flow Fundamental Diagrams Using Gaussian Processes," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 763–771, 2022, ISSN: 2687-7813. DOI: 10.1109/OJITS.2022.3220926. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9943806 (visited on 05/04/2026).

[16] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning* (Adaptive computation and machine learning), en, 3. print. Cambridge, Mass.: MIT Press, 2008, ISBN: 978-0-262-18253-9.

[17] Y. Saatci, "Scalable Inference for Structured Gaussian Process Models," en, PhD, University of Cambridge, Dec. 2011. [Online]. Available: https://mlg.eng.cam.ac.uk/pub/pdf/Saa11.pdf.

[18] B. Do, N. A. Ajenifuja, T. A. Adebiyi, and R. Zhang, "Sampling from Gaussian processes: A tutorial and applications in global sensitivity analysis and optimization," en, *Structural and Multidisciplinary Optimization*, vol. 69, no. 2, p. 47, Feb. 2026, ISSN: 1615-147X, 1615-1488. DOI: 10.1007/s00158-025-

04245-y. [Online]. Available: https://link.springer.com/10.1007/s00158-025-04245-y (visited on 04/30/2026).

[19] J. T. Wilson, V. Borovitskiy, A. Terenin, P. Mostowsky, and M. P. Deisenroth, "Efficiently sampling functions from Gaussian process posteriors," in *Proceedings of the 37th International Conference on Machine Learning*, ser. ICML'20, JMLR.org, 2020.

[20] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," en, *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, Mar. 1960, ISSN: 0021-9223. DOI: 10.1115/1.3662552. [Online]. Available: https://asmedigitalcollection.asme.org/fluidsengineering/article/82/1/35/397706/A-New-Approach-to-Linear-Filtering-and-Prediction (visited on 05/08/2026).

[21] E. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, Lake Louise, Alta., Canada: IEEE, 2000, pp. 153–158, ISBN: 978-0-7803-5800-3. DOI: 10.1109/ASSPCC.2000.882463. [Online]. Available: http://ieeexplore.ieee.org/document/882463/ (visited on 04/24/2026).

[22] S. Julier, "The scaled unscented transformation," in *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, Anchorage, AK, USA: IEEE, 2002, 4555–4559 vol.6, ISBN: 978-0-7803-7298-6. DOI: 10.1109/ACC.2002.1025369. [Online]. Available: http://ieeexplore.ieee.org/document/1025369/ (visited on 05/09/2026).

[23] J. Hartikainen and S. Sarkka, "Kalman filtering and smoothing solutions to temporal Gaussian process regression models," in *2010 IEEE International Workshop on Machine Learning for Signal Processing*, Kittila, Finland: IEEE, Aug. 2010, pp. 379–384, ISBN: 978-1-4244-7875-0. DOI: 10.1109/MLSP.2010.5589113.

[24] D. Bertsekas, *A Course in Reinforcement Learning: 2nd Edition*. Athena Scientific, Aug. 2025. [Online]. Available: https://www.mit.edu/~dimitrib/RLCOURSECOMPLETE%202ndEDITION.pdf.

[25] B. Light, "The principle of optimality in dynamic programming: A pedagogical note," *Operations Research Letters*, vol. 57, p. 107 164, Nov. 2024, ISSN: 0167-6377. DOI: 10.1016/j.orl.2024.107164. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167637724001007 (visited on 04/27/2026).

[26] W. B. Powell, "What you should know about approximate dynamic programming," en, *Naval Research Logistics (NRL)*, vol. 56, no. 3, pp. 239–249, Apr. 2009, ISSN: 0894-069X, 1520-6750. DOI: 10.1002/nav.20347. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/nav.20347 (visited on 04/27/2026).

[27] M. R. K. Mes and A. P. Rivera, "Approximate Dynamic Programming by Practical Examples," in *Markov Decision Processes in Practice*, R. J. Boucherie and N. M. Van Dijk, Eds., vol. 248, Series Title: International Series in Operations Research & Management Science, Cham: Springer International Publishing, 2017, pp. 63–101, ISBN: 978-3-319-47766-4. DOI: 10.1007/978-3-319-47766-

4_3. [Online]. Available: http://link.springer.com/10.1007/978-3-319-47766-4_3 (visited on 05/08/2026).

[28] A. Hamednia, N. K. Sharma, N. Murgovski, and J. Fredriksson, "Computationally Efficient Algorithm for Eco-Driving Over Long Look-Ahead Horizons," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6556–6570, Jul. 2022, ISSN: 1524-9050, 1558-0016. DOI: 10.1109/TITS.2021.3058418. [Online]. Available: https://ieeexplore.ieee.org/document/9359536/ (visited on 06/01/2026).

[29] T. Li, W. Zhang, G. Huang, *et al.*, "Real-world data-driven charging strategies for incorporating health awareness in electric buses," en, *Journal of Energy Storage*, vol. 92, p. 112 064, Jul. 2024, ISSN: 2352152X. DOI: 10.1016/j.est.2024.112064. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2352152X24016505 (visited on 01/29/2026).

[30] R. L. Harrison, C. Granja, and C. Leroy, "Introduction to Monte Carlo Simulation," Bratislava (Slovakia), 2010, pp. 17–21. DOI: 10.1063/1.3295638. [Online]. Available: https://pubs.aip.org/aip/acp/article/1204/1/17-21/866186 (visited on 05/12/2026).

[31] K. J. Preacher and J. P. Selig, "Advantages of Monte Carlo Confidence Intervals for Indirect Effects," en, *Communication Methods and Measures*, vol. 6, no. 2, pp. 77–98, Apr. 2012, ISSN: 1931-2458, 1931-2466. DOI: 10.1080/19312458.2012.679848. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/19312458.2012.679848 (visited on 05/12/2026).

[32] C. R. Harris, K. J. Millman, S. J. Van Der Walt, *et al.*, "Array programming with NumPy," en, *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020, ISSN: 0028-0836, 1476-4687. DOI: 10.1038/s41586-020-2649-2. [Online]. Available: https://www.nature.com/articles/s41586-020-2649-2 (visited on 02/13/2026).

A

Driving dynamics derivation

This thesis mainly considers the discrete version of the dynamics and uses forward Euler integration for simplicity. The discrete dynamics map for driving is denoted $\mathcal{F}$, and is fully determined by the state x_i since the driver is now assumed to follow the traffic process $f^{(\mathrm{t})}(s,t)$.

Concretely, by using the force balance from (3.1) and solving for the control $u(t) = F_{\mathrm{tr}}(t)$, we can find a policy which makes the vehicle follow a desired acceleration. The braking force F_{brk} is omitted to simplify the calculations. Since the aim is to find the policy which satisfies that the acceleration $a(s,t)$ is equal to the spatial derivative of the traffic process $f^{(\mathrm{t})}$, we get,

$$a(s,t) = \frac{\partial f^{(\mathrm{t})}}{\partial t}(s,t) = \frac{u(s,t) - F_{\mathrm{air}} - F_{\mathrm{g}} - F_{\mathrm{roll}}}{m}, \tag{A.1}$$

where $F_{\mathrm{air}} = \frac{1}{2}\rho C_d A_f \Big(f^{(\mathrm{t})}(s,t) + f^{(\mathrm{w})}(s,t)\Big)^2$, and $F_{\mathrm{g}} + F_{\mathrm{roll}} = mg\Big(\sin(\alpha(s)) + C_r\cos(\alpha(s))\Big)$. Note that $a(s,t)$ describes the rate of change of the velocity while time varies, i.e. $a = dv/dt$. We would like to transform this to vary as a function of space instead, but would then get $dv/ds = a/v$ which is numerically ill-posed for velocities close to zero. To avoid this, we consider the change of variables $E = v^2/2$, and differentiate w.r.t. time. This yields,

$$\begin{aligned} &\frac{d}{dt}(E) = \frac{d}{dt}(\frac{v^2}{2}) = \frac{dv}{dt}v, \quad \frac{dE}{ds} = \frac{dE}{dt}\frac{dt}{ds} = \frac{dE}{dt}\frac{1}{v}, \\ \implies &\frac{dE}{ds} = \frac{dE}{dt}\frac{1}{v} = (\frac{dv}{dt}v)\frac{1}{v} = \frac{dv}{dt} = a, \end{aligned} \tag{A.2}$$

so by transforming the velocity to a variable proportional to the kinetic energy of the system, the numerically ill-posed division by zero can be avoided. To continue, we consider the Forward Euler discretization,

$$\begin{aligned} &E_{i+1} = E_i + \Delta s \cdot a_i, \quad E_i = \frac{(f_i^{(\mathrm{t})})^2}{2}, \\ \implies &a_i = \frac{1}{\Delta s}\Big(\frac{(f_{i+1}^{(\mathrm{t})})^2}{2} - \frac{(f_i^{(\mathrm{t})})^2}{2}\Big), \end{aligned} \tag{A.3}$$

And by substituting a_i from (A.1) we derive that,

$$\begin{aligned}\frac{\partial f_i^{(\mathrm{t})}}{\partial t} &= \frac{1}{\Delta s}\Big(\frac{(f_{i+1}^{(\mathrm{t})})^2}{2} - \frac{(f_i^{(\mathrm{t})})^2}{2}\Big),\\ u_i &= \frac{m}{2\Delta s}\Big((f_{i+1}^{(\mathrm{t})})^2 - (f_i^{(\mathrm{t})})^2\Big)\\ &\quad + \frac{1}{2}\rho C_d A_f\Big(f_i^{(\mathrm{t})} + f_i^{(\mathrm{w})}\Big)^2\\ &\quad + mg\Big(\sin(\alpha_i) + C_r\cos(\alpha_i)\Big).\end{aligned} \tag{A.4}$$

The subscript here is short for the inputs required for said variable at stage i, e.g. $f_i^{(\mathrm{t})} = f^{(\mathrm{t})}(s_i, t_i)$. It may appear like the control at stage i depends on the next state $i+1$, but by writing $f_{i+1}^{(\mathrm{t})} = f^{(\mathrm{t})}(s_i + \Delta s, t_i + \Delta s/f_i^{(\mathrm{t})})$ we find that it is determined by the current state through the dynamics. Thus, we have found a control policy $u_i = u(x_i)$ which follows the traffic process s.t. $f_i^{(\mathrm{t})} = v_i$. Given this policy, we would like to calculate the both the time t and cumulative energy C trajectories for our system. The time dynamics w.r.t space can easily be found by using the time to space transformation outlined in (3.11),

$$\frac{dt}{dt} = \frac{dt}{ds}\frac{ds}{dt} = \frac{dt}{ds}v \implies \frac{dt}{ds} = \frac{1}{f^{(\mathrm{t})}}, \tag{A.5}$$

which yields our time trajectory as a function of the traffic process. The cumulative energy consumption C is obtained by considering the power drawn from the battery P_b, outlined in (3.7). To start, we transform the power to vary in space instead of time as,

$$P_b = \frac{dW}{dt} = \frac{dW}{ds}\frac{ds}{dt} \implies \frac{dW}{ds} = P_b\frac{1}{f^{(\mathrm{t})}}. \tag{A.6}$$

The battery power P_b is a function of the electrical load P_{load}, which is a function of the motor and auxiliary power $P_{\text{motor}}, P_{\text{aux}}$. As described previously, the auxiliary power is a constant and as such can just be added to the total load. The motor power is a function of the motor angular rate ω and the motor torque M. The policy u describes the force required to maintain the traffic velocity $f^{(\mathrm{t})}$, so the angular rate and torque can be equivalently written as $\omega = f^{(\mathrm{t})}/r,\ M = u \cdot r$ where r is the gear ratio between the wheels and motor. From this, we write the motor power as,

$$P_{\text{motor}} = a_1 + a_2\frac{f^{(\mathrm{t})}}{r} + a_3\frac{(f^{(\mathrm{t})})^3}{r^3} + a_4 f^{(\mathrm{t})}u + a_5\frac{f^{(\mathrm{t})}u^2}{r}. \tag{A.7}$$

Then, the rate of change of the cumulative energy C, which includes the battery losses, motor power, and auxiliary power, becomes,

$$\frac{dW}{ds} = \frac{P_{b,i}}{f_i^{(\mathrm{t})}} = \frac{u_{\mathrm{oc}}^2 - u_{\mathrm{oc}}\sqrt{u_{\mathrm{oc}}^2 - 4R_b(P_{\mathrm{motor},i} + P_{\mathrm{aux}})}}{2R_b f_i^{(\mathrm{t})}}, \tag{A.8}$$

and is fully determined by the traffic and wind processes $f^{(\mathrm{t})}, f^{(\mathrm{w})}$. Finally, using forward Euler the discrete dynamics for cumulative energy is expressed as,

$$C_{i+1} = C_i + \Delta s \frac{P_{b,i}}{f_i^{(\mathrm{t})}} \tag{A.9}$$

$$= C_i + \Delta s \frac{u_{\mathrm{oc}}^2 - u_{\mathrm{oc}}\sqrt{u_{\mathrm{oc}}^2 - 4R_b(P_{\mathrm{motor},i} + P_{\mathrm{aux}})}}{2R_b f_i^{(\mathrm{t})}}, \tag{A.10}$$

where $P_{\mathrm{motor},i}$ can be written as a function of the disturbances as,

$$\begin{aligned} P_{\mathrm{motor},i} &= P_{\mathrm{motor}}(f_i^{(\mathrm{t})}, f_{i+1}^{(\mathrm{t})}, f_i^{(\mathrm{w})}) \\ &= a_1 + a_2\frac{f_i^{(\mathrm{t})}}{r} + a_3\frac{(f_i^{(\mathrm{t})})^3}{r^3} + a_4 f_i^{(\mathrm{t})} u_i + a_5\frac{f_i^{(\mathrm{t})} u_i^2}{r}. \end{aligned} \tag{A.11}$$

B

Implementation details

This appendix lays out more implementation details that complement Chapter 5. These details are more practical in nature.

B.1 Discretization steps

The backward pass used in the exact DP formulation requires discretization of the state x = soc. The step length is 1% soc, which means there is a total of 100 discretization steps in state. The inputs are discretized at 2% soc. The number of inputs that are evaluated depend on the current state of charge, but is never greater than 45. In order to prevent micro charging, the planner will never consider charging the battery than with less than 5% soc.

B.2 Interpolation of cost functions

When calculating the backwards pass for the exact DP solution, the state of charge trajectory is calculated between two charging stations. In order to calculate the value for the current step k with a given state x_k and input u_k pair, according to

$$V_k(x_k) = \min_{u_k \in U_k(x_k)} g_k(x_k, u_k) + V_{k+1}(f_k(x_k, u_k)), \tag{B.1}$$

the value function of the next state must be known. The value function of the next state x_{k+1} has been calculated at the discrete state values as outlined in the section above. However, forward simulating the next state using the system dynamics equation f_k will not necessarily lead to a SoC value coinciding with a discretization point. Therefor, the value function at the next charging station V_{k+1} is evaluated using linear interpolation between the two nearest state values as

$$V_{k+1}(x_{k+1}) = \frac{(x^+_{k+1} - x_{k+1})V_{k+1}(x^-_{k+1}) + (x_{k+1} - x^-_{k+1})V_{k+1}(x^+_{k+1})}{x^+_{k+1} - x^-_{k+1}}, \tag{B.2}$$

where x^-_{k+1} and x^+_{k+1} is the previous and next evaluated states in the value function, respectively.

B.3 Charging station pre-processing

In order to speed up computation, pre-processing was performed on the charging stations to remove chargers that the planners are unlikely to select. Experimentally, the chargers never stopped at chargers slower than 100 kW when planning. If a charging station had no chargers with a maximum power level $P_{\max} > 100$ kW, it was removed from the route entirely.

C

Simulation results

This appendix consists of all simulated scenarios for all routes. For an explanation of the scenarios and routes, see Section 6.2.

C.1 Barcelona - Genoa

Table C.1: Deterministic scenario results for the Barcelona - Genoa route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$	T_{compute} [s]
Full DP	1021	378	1026	5	0.4838	0.4922	0.41
ADP 1	930	268	1012	3	0.1787	0.5107	253
Heuristic	895	236	1010	2	0.2468	0.2861	0.40
ADP 2	1095	456	1034	5	0.4850	0.5194	123

Table C.2: Normal disturbance scenario results for the Barcelona - Genoa route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$
Full DP	1022 ± 16.1	377 ± 15.8	1052 ± 41.4	4 ± 0.4	0.4107 ± 0.0133	0.4887 ± 0.0245
ADP 1	932 ± 5.8	266 ± 5.8	1019 ± 26.1	3 ± 0.0	0.1754 ± 0.0315	0.4896 ± 0.0137
Heuristic	959 ± 105.1	300 ± 107.9	1029 ± 36.7	2 ± 0.4	0.2251 ± 0.0517	0.3918 ± 0.1857
ADP 2	1095 ± 32.7	449 ± 34.6	1040 ± 47.6	6 ± 0.7	0.4134 ± 0.0384	0.5221 ± 0.0235

Table C.3: Large disturbance scenario results for the Barcelona - Genoa route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$
Full DP	1038 ± 29.2	390 ± 27.0	1102 ± 87.7	4 ± 0.4	0.3772 ± 0.0292	0.4719 ± 0.0540
ADP 1	955 ± 28.8	294 ± 38.6	1074 ± 84.4	4 ± 0.9	0.1576 ± 0.0208	0.4792 ± 0.0383
Heuristic	996 ± 110.5	335 ± 114.9	1061 ± 73.1	2 ± 0.4	0.1749 ± 0.0700	0.3798 ± 0.1718
ADP 2	1130 ± 54.8	483 ± 59.2	1069 ± 85.4	6 ± 1.2	0.4210 ± 0.0738	0.5280 ± 0.0425

Table C.4: Mismatched disturbance scenario results for the Barcelona - Genoa route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$
Full DP	1043 ± 31.8	395 ± 31.3	1094 ± 97.2	5 ± 0.5	0.3963 ± 0.0463	0.4833 ± 0.0551
ADP 1	958 ± 28.4	300 ± 32.4	1082 ± 76.0	4 ± 0.5	0.2122 ± 0.0405	0.4711 ± 0.0329
Heuristic	996 ± 110.5	335 ± 114.9	1061 ± 73.1	2 ± 0.4	0.1749 ± 0.0700	0.3798 ± 0.1718
ADP 2	1135 ± 52.5	489 ± 57.5	1078 ± 84.8	6 ± 1.1	0.4138 ± 0.0796	0.5305 ± 0.0320

C.2 Saarbrücken - Burgos

Table C.5: Deterministic scenario results for the Saarbrücken - Burgos route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$	T_{compute} [s]
Full DP	1726	643	1841	8	0.2287	0.5270	2.06
ADP 1	1650	586	1858	7	0.1405	0.5049	690
Heuristic	1730	664	1859	5	0.1583	0.7921	1.99
ADP 2	1750	665	1843	9	0.0916	0.5052	452

Table C.6: Normal disturbance scenario results for the Saarbrücken - Burgos route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	SoC_{f}
Full DP	1707 ± 26.2	629 ± 22.1	1849 ± 57.5	7 ± 0.5	0.2167 ± 0.0245	0.5289 ± 0.0075
ADP 1	1638 ± 32.6	572 ± 31.0	1864 ± 36.4	7 ± 1.0	0.0997 ± 0.0256	0.5146 ± 0.0157
Heuristic	1736 ± 18.9	665 ± 16.4	1845 ± 39.0	5 ± 0.0	0.1604 ± 0.0576	0.7963 ± 0.0039
ADP 2	1769 ± 38.9	684 ± 39.4	1828 ± 55.4	10 ± 1.1	0.1185 ± 0.0219	0.5185 ± 0.0059

Table C.7: Large disturbance scenario results for the Saarbrücken - Burgos route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$
Full DP	1721 ± 41.4	638 ± 37.9	1875 ± 108.4	8 ± 0.5	0.2182 ± 0.0369	0.5348 ± 0.0097
ADP 1	1788 ± 31.6	695 ± 22.6	1840 ± 102.2	10 ± 0.9	0.1812 ± 0.0455	0.5228 ± 0.0084
Heuristic	1761 ± 49.0	681 ± 39.8	1875 ± 102.1	5 ± 0.0	0.1402 ± 0.0771	0.8151 ± 0.0338
ADP 2	1797 ± 81.9	713 ± 88.5	1843 ± 73.9	10 ± 1.8	0.1394 ± 0.0626	0.5281 ± 0.0107

Table C.8: Mismatched disturbance scenario results for the Saarbrücken - Burgos route.

Method	T_{tot} [min]	T_{stop} [min]	E [kWh]	N_{stop}	SoC_{min}	$\text{SoC}_{\text{final}}$
Full DP	1761 ± 35.0	672 ± 38.7	1863 ± 116.6	8 ± 0.5	0.3497 ± 0.0456	0.5427 ± 0.0099
ADP 1	1682 ± 60.6	605 ± 52.5	1898 ± 86.3	8 ± 1.4	0.1273 ± 0.0566	0.5162 ± 0.0084
Heuristic	1761 ± 49.0	681 ± 39.8	1875 ± 102.1	5 ± 0.0	0.1402 ± 0.0771	0.8151 ± 0.0338
ADP 2	1836 ± 82.3	750 ± 89.0	1884 ± 110.4	12 ± 2.4	0.1392 ± 0.0554	0.5183 ± 0.0102

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY