UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

PROGNOSTIC BASED REAL-TIME DECISION MAKING APPROACH FOR THE DYNAMIC AND STOCHASTIC SHORTEST PATH PROBLEM FOR ELECTRIC VEHICLES

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN
CIENCIAS DE LA INGENIERÍA, MENCIÓN ELÉCTRICA

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELÉCTRICO

HERALDO FELIPE ROZAS OVANDO

PROFESOR GUÍA:
MARCOS ORCHARD CONCHA

PROFESOR CO-GUÍA:
DIEGO MUÑOZ CARPINTERO

MIEMBROS DE LA COMISIÓN:
DORIS SÁEZ HUEICHAPAN
FERNANDO AUAT CHEEIN

SANTIAGO DE CHILE
2019

RESUMEN DE LA TESIS PARA OPTAR AL GRADO DE
MAGÍSTER EN CIENCIAS DE LA ING., MENCIÓN ELÉCTRICA Y AL
TÍTULO DE INGENIERO CIVIL ELÉCTRICO
POR: HERALDO FELIPE ROZAS OVANDO
FECHA: 2019
PROF. GUÍA: DR. MARCOS ORCHARD CONCHA

PROGNOSTIC BASED REAL-TIME DECISION MAKING APPROACH FOR THE DYNAMIC AND STOCHASTIC SHORTEST PATH PROBLEM FOR ELECTRIC VEHICLES

Traffic congestion is becoming increasingly common in big cities, which leads to a significant increase in travel times. Routing-systems work finding the best paths for either a vehicle or a fleet, allowing a more efficient operation. Thus they play an important role in avoiding delays caused by traffic congestion. These routing-systems will make better decisions if they incorporate all available information (current and historical); more specifically, the optimal path needs to be calculated considering both the current traffic state and a dynamic model that characterizes the stochasticity of the future traffic evolution (based on historical data). Typically, this problem is called Dynamic-Stochastic Shortest Path Problem in the transport systems literature.

The development of routing-systems for Electric Vehicles is particularly important because they are considered the most promising alternative to internal combustion engine vehicles. Finding the optimal path for an Electric Vehicle is more challenging than for conventional vehicles since, in general, the fastest path for electric vehicles is not necessarily the optimal path from the energetic point of view due to they can recover energy during braking; thus, the variables of travel time and energy consumption both need to be incorporated in the decision-making process.

In this thesis, a real-time solution strategy for the Electric Vehicle Dynamic Stochastic Shortest Path Problem (DSSPP) is proposed, which is based on a Prognosis Decision Making approach. It allows the optimization of the travel time, energy consumption, or both. The proposed decision-making algorithm consists of three major steps. In the first step, a set of path candidates are selected using an ad-hoc heuristic. In the second step, each path candidate is evaluated using a Prognosis-based Particle Filter; it allows to estimate the probability density functions for travel time and energy consumption, and incorporates the current traffic state and a characterization of the future evolution. Finally, the optimal path is chosen as the minimum of a function of cost that incorporates the expected value of travel time and energy consumption.

The proposal is analyzed by a simulation study, which shows that the DSSPP can be properly solved by the proposed routing strategy. In addition, the results show that periodic path updates, which are calculated en-route, can generate a reduction in both travel time and energy consumption. In other words, the incorporation of real-time traffic information into the decision-making process allows a more efficient electric vehicle operation. Moreover, the results show a trade-off (with regard to mean values) between the travel time and energy consumption because, generally, the fastest path has the highest energy consumption.

RESUMEN DE LA TESIS PARA OPTAR AL GRADO DE
MAGÍSTER EN CIENCIAS DE LA ING., MENCIÓN ELÉCTRICA Y AL
TÍTULO DE INGENIERO CIVIL ELÉCTRICO
POR: HERALDO FELIPE ROZAS OVANDO
FECHA: 2019
PROF. GUÍA: DR. MARCOS ORCHARD CONCHA

PROGNOSTIC BASED REAL-TIME DECISION MAKING APPROACH FOR THE DYNAMIC AND STOCHASTIC SHORTEST PATH PROBLEM FOR ELECTRIC VEHICLES

La congestión en el tráfico vehicular es cada día más común en las grandes ciudades, lo que conduce a un significativo incremento en los tiempos de viaje. Los sistemas de ruteo trabajan en la búsqueda de rutas óptimas (minimizando tiempos de viaje o costos en general), ya sea para un vehículo o una flota. Por consiguiente, los sistemas de ruteo juegan un rol fundamental para evitar los grandes retrasos causados por la congestión. Dichos sistemas de ruteo deben ser capaces de incorporar toda la información disponible (actual e histórica) en el proceso de toma de decisiones; es decir, decidir una ruta óptima no solo considerando la situación actual, sino también posibles modelos de evolución basados en datos históricos. Típicamente en la literatura de transporte este problema es llamado Problema Dinámico-Estocástico de ruta más corta. Particularmente, es de interés desarrollar sistema de ruteos para vehículos eléctricos; debido a que estos son considerados la alternativa más prometedora al recambio de los vehículos convencionales. Esto conlleva un desafío, ya que en el caso de vehículos eléctricos, la ruta más rápida no necesariamente es la más óptima desde el punto de vista energético; por lo cual ambas variables (tiempo de viaje y consumo energético) deben ponerse en la balanza al momento de realizar la toma de decisiones.

En el presente trabajo una estrategia en tiempo real para el Problema Dinámico-Estocástico de ruta más corta para un vehículo eléctrico es presentada. Esta estrategia consiste en un enfoque de toma de decisiones basadas en pronóstico. El algoritmo propuesto funciona por etapas. En la primera etapa, un conjunto de rutas candidatas son seleccionadas mediante una heurística diseñada ad-hoc para este propósito; ésta garantiza una rápida ejecución y diversidad en las soluciones. Luego, cada candidata es evaluada mediante Pronóstico basado en Filtro de Partículas; esto permite realizar una predicción del tiempo de viaje y consumo energético incorporando información actual del tráfico y una caracterización de la evolución futura del tráfico. Finalmente, la ruta óptima es decidida minimizando un funcional de costo que incorpora en el valor esperado del tiempo de viaje y consumo energético.

La estrategia de ruteo es analizada mediante un estudio por simulaciones, el cual corrobora que el problema en estudio puede ser resuelto efectivamente mediante la por la estrategia propuesta. Además, los resultados muestran que realizando toma de decisiones en ruta (es decir, actualización de ruta óptima periódicamente) se puede reducir los tiempos de viaje y el consumo energético. En otras palabras, las actualizaciones en ruta permiten una operación más eficiente de los vehículos eléctricos. Adicionalmente, los resultados advierten un trade-off entre el tiempo de viaje y el consumo eléctrico. Ello pues en general, la ruta más rápida conlleva un mayor gasto energético. Por lo último, se resalta la importancia de tener claridad respecto a que es más prioritario al momento de tomar decisión de ruta: tiempo de viaje o consumo de energía, pues aquello definirá la ruta óptima a seguir.

Para mi Mamá y mi Papá, por el esfuerzo conjunto que han hecho a lo largo de sus vidas para que mi hermano, mi hermana y yo seamos profesionales.

Acknowledgments

En primer lugar, agradecer a mi familia, por siempre creer en mí y apoyarme en este largo tiempo de estudio. A mi Mamá y a mi Papá, les agradezco haberme enseñado que con trabajo duro, se puede estar mucho más cerca de cumplir nuestros sueños. A mi hermana y hermano, les agradezco su infinito apoyo durante toda la vida. Quizá no seamos una familia del todo convencional, pero les amo y me siento muy feliz de ser parte de ustedes.

Quiero agradecer también a Giselle, quien ha sido mi amor y confidente durante estos años de estudio. Gracias por el apoyo incondicional, y por ayudarme cada día a ser una mejor persona.

Además, agradecer tantos buenos momentos junto a mis compañeros de universidad (no quiero individualizar pues son muchos y puede que se me pase alguno): los del plan común, los DIE y los del fútbol.

Aprovechar el espacio para agradecer al profesor Marcos Orchard, por su apoyo y orientación; a los integrantes del Laboratorio de Control Avanzado I, por su ayuda en mi formación como investigador y por el buen ambiente de trabajo. Agradezco a Diego Muñoz, por su buena disposición para responder consultas.

Finalmente, agradezco al Gobierno de Chile por haber financiado mi educación: en pregrado a través de la Beca Gratuidad y en magíster a través de la Beca de Magíster Nacional CONICYT-PFCHA/MagisterNacional/2018-22180232.

En resumen, ¡agradezco a cada persona e institución que ayudó a que hoy esté finalizando mis estudios!

Contents

List of Tables

List of Figures

List of Abbreviations

DSP	Daily Speed Profile
DSSPP	Dynamic and Stochastic Shortest Path Problem
EBT	Extra Buffer Time
EV	Electric Vehicle
EV-DSSPP	Electric Vehicle Dynamic and Stochastic Shortest Path Problem
EV-ECM	EV energy consumption model
HWFET	Highway fuel economy
IIFS	Incident Impact Factor by Segment
K-SPP	K-Shortest Path Problem
MDP	Markov Decision Process
NY	New York
PDF	Probability Density Function
PDM	Prognostic Decision Making
PETT	Percentage of Edges with Typical Traffic
RSP	Recurrent Speed Profile
SOC	State of Charge
SPP	Shortest Path Problem
TT	Travel Time

Chapter 1

Introduction

In road networks, factor such as traffic congestion, traffic incidents, and weather lead to an increase in travel times in urban areas [1]. One study [2] showed that, on average, Americans spend about 34 hours every year in traffic congestion; this has an estimated economic opportunity cost of about $ 124 billion every year. In this context, using offline information (i.e. historical data) and/or online information (i.e. real-time information on the current state of the network) that is obtained from an Intelligent Transportation System, it is possible to develop routing-systems that allow significant reductions in travel times [3–5]. These routing systems work finding the best paths for either one vehicle or one fleet, so they can be employed for diverse applications including emergency vehicles, delivery vehicles, or cabs. These routing strategies need to be able to incorporate all available information (such as current network states, traffic dynamics and traffic stochasticity) in order to find the best possible solutions, and solve the optimization problem in real-time; thus designing routing-system is not easy tasks. The problem with finding the optimal path, while considering the dynamic and stochasticity of the system is called Dynamic and Stochastic Shortest Path Problem (DSSPP) in the transport literature.

Particularly, it is relevant the development of routing systems for Electric Vehicles (EVs) because they are considered the most promising alternative to internal combustion engine vehicles in the endeavor towards a cleaner transportation sector [6]. In this context, EVs are starting to become a reality in some countries, such that Norway and Iceland [7]. Even more, the total number of EVs around the world has been increasing in the last year, this fact is shown in Figure 1.1. In spite of this positive trend, there are two barriers that limit the Electric Vehicle (EV) adoption: autonomy and anxiety range [8–10]. The autonomy or driving range can be defined as the maximum distance allowed by a fully charged EV battery, whereas the range anxiety is the driver fear that the EV battery will deplete while driving [8]. To address these issues an interesting alternative is the development of strategies that help to extend driving range (e.g. strategies of minimization of energy consumption) [11]. Additionally, proving driving range predictions (before departure) can help to reduce the range anxiety due to the driver will know in advance how much energy the EV may spend to reach the destination [11].

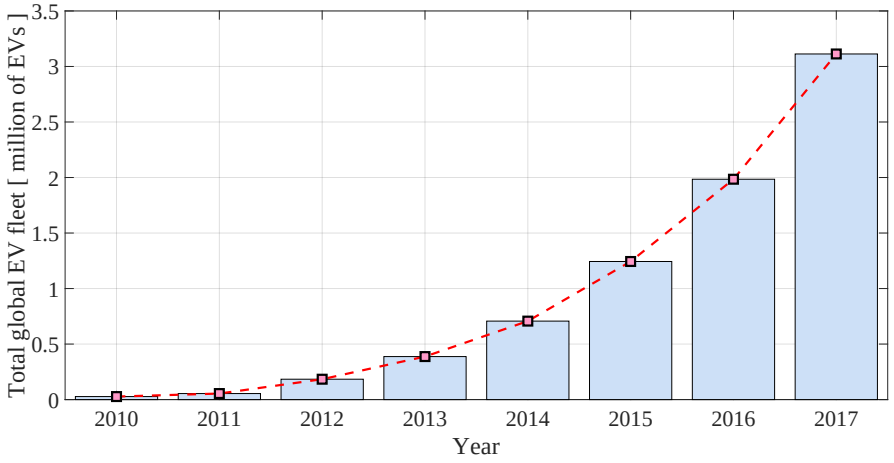


Figure 1.1: Total global EV fleet (excluding electric buses) in 2010 – 2017 (adapted from [12]).

An EV routing system should allow the efficient operation of EVs, helping to reduce travel times and electric energy consumption. Finding the optimal path for an EV is more challenging than for conventional vehicles since, in general, the fastest path for electric vehicles is not necessary the optimal path from the energetic point of view due to they can recover energy during braking [13]; thus, the variables of travel time and energy consumption both need to be incorporated in the decision making process. In addition, both EV and road models play an important for EV routing systems because the path consumption is highly dependent on the EV characteristics; elevation profile; velocity profile; and acceleration profile due to EVs can even recover some energy during braking. Therefore, the incorporation well-detailed EV and road models are key in the decision making process.

It is essential to note that depending on the context or application field, the drivers can be interested in to minimize the travel time and/or the energy consumption. For instance, when emergency vehicles have to attend emergencies, they are solely interested in to minimize the travel times, whereas when a delivery driver is transporting a package with enough time, probably the driver choose the path that minimize the energy consumption. Even the same user in different context can choose different objective. For example, let suppose a driver who is driving to its work, if the driver has enough time, probably he chooses minimize energy consumption, whereas if he is late, he will choose minimize the travel time. The examples commented above showed that the routing systems should be able to achieve the minimization of the travel time and/or the energy consumption, and so users can choose their objective according to their priorities.

Furthermore, there are two approaches that can be used to address the DSSPP, which differentiate in the time when the solution algorithm is computed. The first approach is to compute the solution path before departure, whereas the second is to compute (and update) the solution path en-route [14]. Basically, in the second approach the solution path is continuously being updated as long the path is traveled, thus the computing time is more limited and real-time execution algorithms are needed. The main motivation to re-compute the optimal path en-route is that the availability of new traffic information because it may

enable to find a better path. For instance, if a non-recurring event has been occurred which may generate delay in the travel time, it is possible to compute a new path en-route in order to avoid this possible congestion.

Considering the increasing importance of reducing travel times and of optimizing EV operation, this works is focused on to treat both issues jointly. In the present work, the problem will be solved by a Prognostic Decision Making approach.

1.1 Hypotheses

This work focuses on the study and design of a real-time strategy for the Electric Vehicle Dynamic and Stochastic Shortest Path Problem (EV-DSSPP), based on a Prognostic Decision Making (PDM) approach. Thus, the following hypothesis will be tested:

1. The real-time EV-DSSPP can be solved effectively by a PDM approach incorporating consumption model, a road model, historical and real-time traffic information (recurrent and non-recurrent traffic congestion).
2. Updating the paths en-route offers higher performance in terms of travel time and energy consumption than solely using the optimal solution computed before departure.

1.2 Objectives

1.2.1 General objective

To design, implement and test a real-time strategy for EV-DSSPP based on a PDM approach that incorporates information from: EV energy consumption model, road model, historical and real-time traffic information (recurrent and non-recurrent traffic congestion).

1.2.2 Specific objectives

The general objective can be separated into the following specific objectives:

1. To design and implement a model for stochastic traffic simulation incorporating recurrent and non-recurrent congestion.
2. To design and implement an EV energy consumption model that incorporates road information (e.g. inclination, velocity, distance, road type) and EV characteristics.
3. To design, implement and test a real-time strategy for the EV-DSSPP based on a PDM approach.

1.3 Thesis organization

This thesis is organized as follows. Chapter 2 provides a theoretical framework of Graph Theory and Prognostics that are essential for the understanding of the present work. In addition, it presents a brief literature review of EV-DSSPP, which is useful to motivate the present work and to highlight which are the main contributions. Chapter 3 introduces the models employed to develop this work: road networks, traffic and EV energy consumption model. Chapter 4 presents in depth the proposed PDM approach for EV-DSSPP. Chapter 5 shows the results and analysis of the simulation experiments. Finally, Chapter 6 presents the work's conclusions and perspectives for future research.

Chapter 2

Theoretical Framework

This chapter presents a brief introduction to several theoretical concepts and methods that constitute relevant background for this work.

2.1 Graph Theory

Graph Theory focuses on the study of abstract objects called graphs, and applications thereof in the modeling of real issues. There are a great variety of application fields including, operations research; transports; electrical networks; chemistry; and computer science [15, 16]. In this section some important concepts of Graph Theory related to Road Network applications will be introduced because, this application field is directly linked to the present work. The following concepts will be introduced using the definitions presented in [17, 18].

A graph G consists of an ordered pair $G = (V, E)$, where V is a finite set of nodes (also called vertices) and E is a finite set of edges. The edges are connections between two nodes; thus any edge e can be represented by an ordered pair (v_x, v_y), where v_x and v_y are the two nodes that are connected by e.

If each edge has a direction, then the graph is called a directed graph. In addition, for an edge $e = (v_x, v_y)$ in a directed graph, we say that v_x is the *start* of the edge, and v_y is the *finish*. Usually, the directed graph are represented by diagrams in which the nodes are circles, and the edges are arrows (see Figure 2.1).

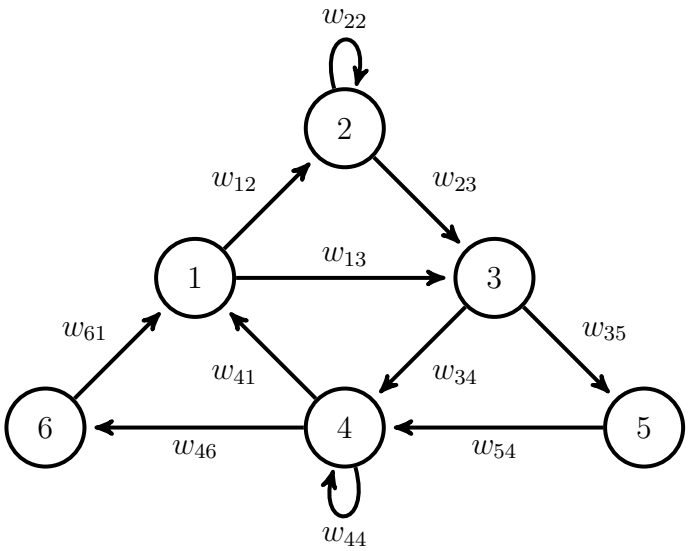


Figure 2.1: Graphical representation of directed graphs. The nodes are represented by enumerated circles, while the edges are represented by arrows, where the arrow head indicates the edge direction.

If $(v_x, v_y) \in E$, it is said that v_x and v_y are *adjacent*. Thus, the adjacency matrix of G denoted as $A(G)$ can be defined as follows:

$$a_{\mathrm{i}j} = \begin{cases} 1 & \text{if } x_{\mathrm{i}} \text{ is adjacent to } x_j \\ \\ 0 & \text{Otherwise} \end{cases} \tag{2.1}$$

In many applications, it is essential to define a positive function $W : E \to \mathrm{R}^+ \cup \{0\}$, with the aim of assigning a weight to each edge. The weight can model different quantities, such as, travel time, cost, or energy consumption. Particularly, if W is not time-variant, W is defined as a fixed matrix, as follows:

$$w_{\mathrm{i}j} = \begin{cases} W(v_{\mathrm{i}}, v_j) & \text{if } v_{\mathrm{i}} \text{ is adjacent to } v_j \\ \\ 0 & \text{Otherwise} \end{cases} \tag{2.2}$$

In the context of directed graphs, a *path* P is defined as a sequence of nodes $P = (v_1, \cdots, v_n)$, where $(v_k, v_{k+1}) \in E, \forall k = 1, \cdots, n-1$. In addition, it can be described as a sequence of edges $S = (\mathrm{e}_1, \cdots, \mathrm{e}_{n-1})$, where $\mathrm{e}_k = (x_k, x_{k+1})$. We say that P is a *loopless* (also called *simple*) path if it does not have repeated nodes. The distance of a path P is defined as:

$$\mathrm{d}(P) = \sum_{\mathrm{e} \in P} W(\mathrm{e}) \tag{2.3}$$

Note that the distance can be represented by other quantities, such as travel time or cost, among others.

The concepts mentioned above will be illustrated in Figure 2.2. In this figure, two possible loopless paths that go from 1 to 4 are highlighted, one in red color and the other in blue. The distance of the red path can be calculated as:

$$\begin{aligned} C_{\text{red}} &= \sum_{\text{e}\in\{(1,2),(2,3),(3,4)\}} w(\text{e}) \\ &= w_{12} + w_{23} + w_{34} \\ &= 3 + 2 + 1 \\ &= 6 \end{aligned} \tag{2.4}$$

Using the same method of calculation, the weight of the blue path is 9.

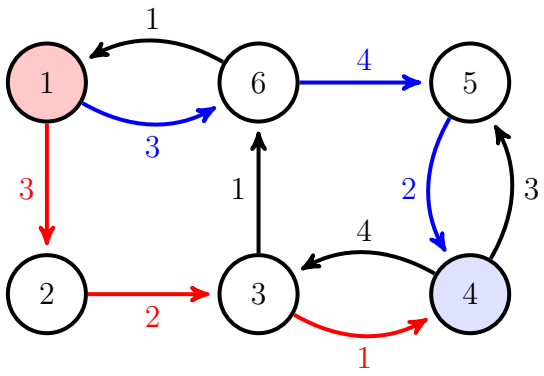


Figure 2.2: An example of a paths in a directed graph. Two possible paths that go from node 1 to node 4 are presented (by the red and blue sequences of edges, respectively).

Road Networks are usually modeled by a directed graph, where the nodes represent intersections or crossroads; the edges represent road segments; and the weight edge can represent any quantity of interest (e.g. travel time, the distance, the cost or flow in the edge).

The above-reviewed example of Figure 2.2 shows that the distance to go from one node to another depends on the path; therefore, in order to find the best path, under specific criteria (e.g., cost or time), an optimization problem can be formulated. In the following subsections, some useful optimization problems for the present work will be reviewed.

2.2 Shortest Path Problem

In this section some work settings of Shortest Path Problem will be reviewed.

2.2.1 Shortest Path Problem: a basic formulation

Shortest Path Problem (SPP) is on finding a path with minimum distance, time or cost from the source node to the destination. In this case the length of each arc is treated as a deterministic and static quantity, namely, each value is a known and is not time-dependent. More formally this problem can be described as follows.

Let be $G = (V, A, W)$ a directed graph , where V is the set of nodes, A is the set of edges, and W is the edge weight matrix. The set of outgoing and incoming edges of node i are denoted as δ^-(i) and δ^+(i), respectively. The aim of the SPP is to calculate the shortest path between two nodes. A standard integer programming formulation to determine the shortest path from node s to node t is the following [19]:

$$\text{Min } z = \sum_{\forall (i,j) \in E} w_{\mathrm{i}j} x_{\mathrm{i}j} \tag{2.5}$$

subject to

$$\sum_{\forall (\mathrm{i},j) \in \delta^+(\mathrm{i})} x_{\mathrm{i}j} - \sum_{\forall (\mathrm{i},j) \in \delta^-(\mathrm{i})} x_{j\mathrm{i}} = \begin{cases} 1 & \text{if } i = s \\ \text{-1} & \text{if } i = t \\ 0 & \text{else} \end{cases} \qquad \forall i \in V \tag{2.6}$$

$$x_{\mathrm{i}j} \in \{0, 1\} \qquad \forall (i, j) \in A \tag{2.7}$$

where, $w_{\mathrm{i}j} \geq 0$ is the edge weight, and x_{ij} are binary edge variables that take the value 1 if the edge (i, j) belongs to the path. Constraints 2.6 are flow conservation constraints. Note that this formulation of SPP is general, in the sense that the optimum can be measured in different ways (depending on the domain of $w_{\mathrm{i}j}$), such as time, distance or cost.

One of the most important and useful methods to solve the SPP is Dijkstra's Algorithm [20]; it is shown by a pseudo-code in Algorithm 1. The lines 1-5 are the initiation, where $dist[v]$ is an estimate distance from s to v and $pred[v]$ is the predecessor of v. The lines 6-13 are the main loop, where at each step (Line 7) the node with the shortest distance, u, is extracted from Q (being Q a priority queue that is ordered by minimum distance $dist[\cdot]$),then it is checked whether the current paths for all the neighbors v of u can be improved by the new route $(s, ..., u, v)$ (Lines 10-13). This procedure is repeated until $u = t$ (Line 8). Finally, in the lines 15-20, the optimal path is rebuilt using the save information of the predecessors for each visited node. To complement the understanding of Dijkstra's Algorithm, an illustrative applications example can be reviewed in [21].

The main advantage of this formulation is the easy implementation and fast computing. The main disadvantage is that in real-word setting, the street networks are not deterministic and not static. Therefore, in practice applications solving this SPP offers only sub-optimal solutions.

Algorithm 1 Dijkstra's Algorithm

Input : Set of nodes V, Weight Matrix W, Starting-point s, Destination d
Output: Optimal path $path^*$, distance of optimal path $dist^*$

Initiation:
1: **for** each vertex $v \in V$ **do**
2: $dist[v] \leftarrow \infty$
3: $pred[v] \leftarrow NIL$
4: $dist[s] \leftarrow 0$
5: $Q \leftarrow V$ ▷ Q is a priority queue that is ordered by minimum distance $dist(\cdot)$

Main Loop:
6: **while** $Q \neq \emptyset$ **do**
7: $u \leftarrow$ Extract-Min(Q) ▷ extract the first element of Q
8: **if** u=t **then**
9: Go to Line 14
10: **for** each edge (u, v) **do**
11: **if** $dist[v] > dist[u] + W(u, v)$ **then**
12: $dist[v] \leftarrow dist[u] + W(u, v)$
13: $pred[v] \leftarrow u$

Rebuild the optimal path:
14: Insert t at the beginning of $path^*$
15: **while** $pred[u] \neq s$ **do**
16: Insert u at the beginning of $path^*$
17: $u \leftarrow pred[u]$
18: Insert s at the beginning of $path^*$
19: $dist^* \leftarrow dist[t]$
20: **return** $path^*, dist^*$

2.2.2 Dynamic Shortest Path Problem

The Dynamic Shortest Path Problem or also called Time-Dependent Shortest Path Problem is on finding a path with minimum distance, time or cost from the source node to the destination in a directed graph $G = (V, A, W)$. This problem considers that the edge weight matrix W is time-dependent and its time dependence is known deterministically [22]. The formulation allows to incorporate the weight time dependency inherent to the network, and thus it can be used in different fields, such that Transportation management, telecommunication and logistics management [23]. Particularly, in the case of street networks, usually the travel time of an arterial is time-variant (see Figure 2.3) due to the high traffic at rush hours generates a significant increment in travel times.

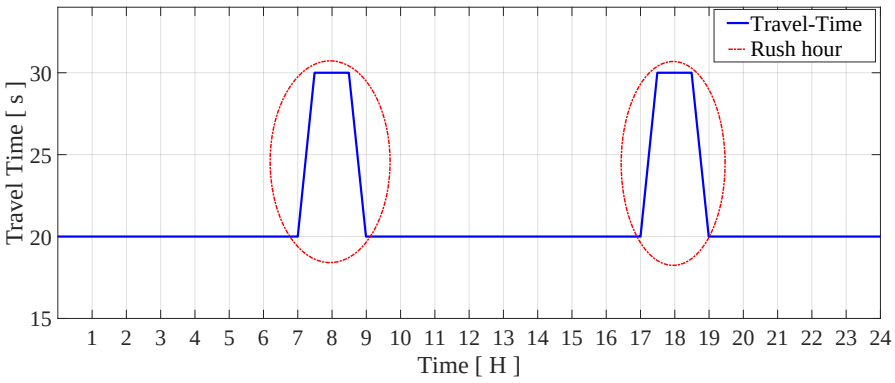


Figure 2.3: Example of dynamic travel time in an arterial segment during 24 hours.

There are a great number of publications to treat the Dynamic SPP (e.g. [22–26]). The main advantage of the time dependency incorporation is that enables to find better solutions than SPP. For instance, in [26], the authors showed that it was possible to reduce a 36% the travel time if the Dynamic Shortest Path Problem was solved instead of the static setting. Nevertheless, the addition of dynamic weights brings about an increment in the computational cost of the solution algorithms, due to for each time variation new solutions can be reached and they should be explored for the algorithms. In addition, depending on the system an accurate characterization of the dynamic of the weights may be not an easy task. For example, in the case of a street network, the characterization of travel time dynamic during the day needs an extensive database.

2.2.3 Stochastic Shortest Path Problem

The Stochastic Shortest Path Problem is on finding a path with minimum distance, time or cost from the source node to the destination in a directed graph $G = (V, A, W)$. This problem considers that the edge weight matrix W is stochastic. The main aim of this formulation is to capture the inherent randomness of the network due to failures, maintenance or other reasons. For instance, in the case of street networks (see Figure 2.4) there are random events such as incidents, vehicle breakdown or bad weather, which greatly affect the reliability of the network [27].

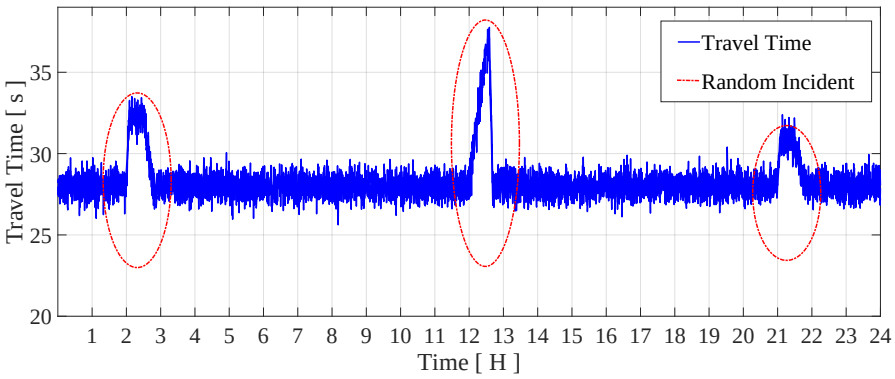


Figure 2.4: Example of a realization of the travel time of a street segment during a day. In general, the travel time can vary randomly around a certain mean values, nevertheless, suddenly random incidents can occur, which increment the travel time.

This problem has been extensively studied and there is a vast quantity of publications (e.g. [27–29]). Similarly with the case of Dynamic Shortest Path Problem, the main advantage of the stochastic formulation is that enables to find better solutions than the SPP. Nevertheless, the addition of weight stochasticity increment the computational cost of the solutions due to different scenarios can be occurred depending on the realizations of the random variables and they must be explored in the searching of optimal path.

2.2.4 Dynamic and Stochastic Shortest Path Problem

According to [4] the problem of calculating the best path to go from one origin to one destination in a directed graph $G = (V, A, W)$ incorporating the dynamics and stochasticity of the traffic dynamic is known in the literature as the Dynamic and Stochastic Shortest Path Problem (DSSPP). In other words, this problem considers that W is a random variable and its distribution is time-dependent.

In the context of road networks, the travel times of road segments are stochastic and their distributions are time-dependent (due to the daily traffic congestion) [30]. As a result, for a day the travel times by hour may be similar to the example presented in Figure 2.5, where for off-peak hours the travel times have a mean value and they fluctuate in the mean neighborhood, nevertheless, at rush hours the mean increases. Moreover, sometimes it can occur random incidents that suddenly change the travel time mean. As a result of the inherent stochasticity and time dependence of the road networks, the DSSPP has been widely studied in this arena. In the following paragraphs, different works that address the DSSPP will be reviewed; first the focus is placed on a general problem (DSSPP) and then, it will be on Electric Vehicle Dynamic and Stochastic Shortest Path Problem (EV-DSSPP).

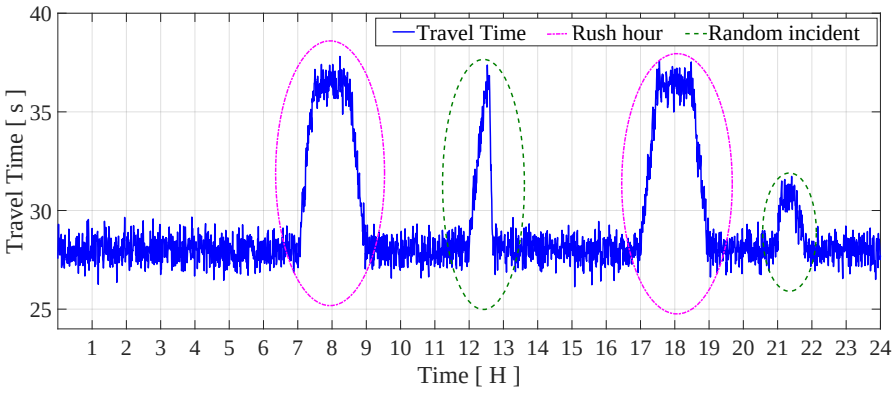


Figure 2.5: Example of a realization of dynamic and stochastic travel times in a road segment. Note that rush hours and random incidents generate a significant increment in travel times.

Kim et al. in [31] formulated the DSSPP as a Markov Decision Process (MDP) and it is solved via backward induction. They not only studied the shortest path problem, but also the optimal departure time. The algorithm was tested with real traffic data in a road network that considers only highways. Their results showed that incorporating real-time traffic information and historical traffic data into the decision making process can significantly reduce expected total costs. Nevertheless, for larger networks, the formulation becomes intractable due in part to the amount of data available that may be useful for optimal route selection. This issue has been treated in [32] by applications of procedure for identifying traffic data that have no decision making value. In [33], the authors studied the DSSPP considering traffic incidents. They assumed a known characterization of incident duration. Then, they modeled the problem as an MDP, and they solved it using anticipative policies. The underlying idea is if the decision-maker has information about the occurrence of an incident, the driver can change the current path in advance to avoid the congestion. Additionally, if the incident is too far, the driver can evaluate if the congestion will be dissipated before that he or she arrives to this area. If so, the driver continues by the same path; otherwise, changes to other path. By simulations, they showed that the optimal anticipatory policy has a higher performance (in terms of expected cost) than the reactive policies. Nevertheless, in the formulation only two states of congestion were considered, and the authors mentioned that the complexity of the algorithm increases exponentially with the number of congestion states.

The work presented in presented in [34] incorporates both recurring and non-recurring congestion into the DSSPP. The authors formulated the problem as MDP and it is solved by dynamic programming. For finding the solutions, they took into account historical traffic data, as well as real-time traffic information to solve the problem. They showed that the solutions that incorporate real-time traffic information have higher performance than static path planning methods. Nevertheless, they studied the problem only for a vehicle that every day has the same starting-point and destination; consequently, they can save computational resources by precalculating the optimal path for different departure time under recurrent congestion. On the other hand, under non-recurring congestion, the problem cannot be solved in real-time using their solution, so they proposed to select the optimal path by a subgraph that allows a faster execution of their algorithm. Similarly, in [3] the problem is

studied considering both recurring and non-recurring congestion. It is formulated as an MDP and is solved by dynamic programming. Additionally, they evaluated the effect of online and offline information on the performance of the routing decisions. Their results showed that considering a partial knowledge of the network is enough to obtain satisfactory results in terms of performance and computing time. Nevertheless, it is only tested in small network. Also, the computing time increases exponentially with the number of disruption states, thus it hinders the real-time implementation. This works has been continued in [4], where the authors tried to solve the dimensionality problem using approximate dynamic programming. The algorithm is tested in a 100-node network considering five levels of disruption and the computing time is 0.6 minutes; consequently, the results may be not promising for a real-time application.

A different approach is presented in [35]. In this case the problem is formulated as a multiobjective optimization problem whose objectives are: minimize the expected travel time, variance and the probability of cover the path in a maximum time. This is solved by a genetic algorithm. The proposal is novel in the sense that a multiobjective is optimized, nevertheless, the solution algorithm has a high-computational cost, which complicates a real-time implementation.

Other relevant works are presented in [36], where a survey about dynamic and stochastic vehicle routing problems is presented.

As it just has seen, the DSSPP has been extensive studied. Nevertheless, in the case of EV-DSSPP, the literature is rather scarce. In the following paragraph, different works related to EV-DSSPP will be reviewed.

2.2.4.1 Dynamic and Stochastic SPP for EVs

In [37], EV-DSSPP is studied considering the option of en-route recharging. Intention-Aware Routing System is proposed, where the idea is to take into account the intention of other drivers to avoid congestion at charging stations. The problem is formulated as an MDP, and the optimal policy is computed using dynamic programming. Their results showed that the proposed algorithm led to an over 80 % improvement in waiting times at charging stations. This paper studied travels in a road network with long routes (longer than $50[km]$), thus the real-time execution is not studied in depth due to it is not priority for their application. In addition, they used a simplified EV consumption model, for instance, the authors assumed that the energy consumption is not affected by the edge travel time; nevertheless, it does not occur in practice.

In [38], the authors studied the problem of finding an optimal EV path in a network where trips may require multiple recharging stops. The considered a network where on each node there is one charging station. Additionally, both the stochasticity of availability of recharging station and the waiting time are incorporated. In this environment, they looked for an optimal recharging policy by different heuristic methods. Even though their positive results, they only are focused on the recharge problem for a long trip and factors such as recurring or non-recurring congestion were not considered.

In [14], the authors addressed the problem of online charging and routing of a single EV in a road network with stochastic and time-variant travel times. Their aim is to minimize the expected cost, which is a weighted sum of travel time and charging cost. The problem was formulated as an MDP, and a solution based on backward recursion was developed. Additionally, the online proposal was contrasted against an offline solution. Both proposals were tested on randomly generated networks. Their results showed that using the online algorithm was a better performance (in terms of travel cost) than offline algorithm. Furthermore, they studied the computing time of both algorithms as a function of the network size, it showed that for a 900-node network the computing time of the online algorithm was 2 minutes. Thus, it may complicate a real-time implementation of the algorithm in a large network.

In [39], the authors treated the eco-routing problem for a EV whose objective is to finding the route with minimum energy consumption. The consumption model was modelled as a random variable. In addition, the authors studied different EV models and simulating of winding speed profiles. The problem is formulated as a stochastic programming model whose objective is to minimize the expected energy consumption. Then, the relaxed formulation is solved and the solution are reconstructed by two different, called Local and Global. Their results showed depending on the EV models or wind profiles, the optimal path was different. Furthermore, they evaluated the computing time for different network size. It depicted a fast increment of the computing time as a function of the nodes of the network; consequently, this fact may hinder a real-time implementation in a large network.

2.2.4.2 Discussion of reviewed researches and contributions of the present work thesis

In most of reviewed works, the DSSPP is formulated as an MDP and it is solved by algorithms based on dynamic programming. Nevertheless, this approach brings about dimensionality problems for large network due to the number of scenarios increases rapidly. This fact is evidenced by some authors who reported their computing time [4, 14]. The above may complicate the implementation of this type strategies for real-time applications. Given this issue, in the present work a strategy will be designed considering the real-time execution as a priority requirement. The proposed strategy is based on a PDM approach and it has two main steps. In the first step a set of path candidates is calculated by a proposed heuristic for K-Shortest path with diversity, whereas in the second step each path candidate is evaluated in terms of travel travel time and energy consumption in order to decide which is the optimal path. Each step is designed considering a predefined fixed maximum computation time; as a result the maximum computation time of the proposed strategy is bounded.

Some works include recurring and non-recurring congestion [3, 4, 34], whereas other are solely concentrated non-recurring congestion. In spite of their work settings, most of reviewed works lead to the same conclusion, which is updating decision en-route allows to reduce the travel times.

On the other hand, in general, the main focus of the reviewed works is to study solution algorithms for the DSSPP, whereas the energy vehicle consumptions are not explained in depth. Therefore, if any user is interested in to implement any of the reviewed works in a real-

word applications, he or she may have problems, due to the works lack of a depth discussion of how the energy consumption model should be tuned to his/her car. Considering the above, in this work, not only the proposed optimization algorithm will be described, but also the EV consumption model and how it is incorporated into the decision making process; thereby providing a clearer picture about how the solution algorithm can be tuned in a real-word application.

In the case of EV-DSSPP, the reviewed works are concentrated on solving the problem considering the recharge option. They treat the EV-DSSPP in a work setting where long trips need to be accomplished, thus, multiple recharges will be necessary. In this arena, to solve issues related to when or where to recharge the EV are priority. The works not only incorporate the stochasticity of travel times, but also the stochasticity associate with recharge station availability and waiting time in station. On the other hand, EV-DSSPP for short city trips has not received the same attention, thus in the present work this setting will be considered. Particularly, recurring and non-recurring congestion will be considered in the street network. Additionally, factors such as distance and inclination will be taken in to account in the consumption model.

Summarizing, the main contribution of the present work is:

- The formulation of a routing strategy for EV-DSSPP where real-time execution is a high priority requirement. This strategy is based on a PDM approach and consists of two main steps. In the first step a set of path candidates is generated by a proposed heuristic. In the second step, these candidates are evaluated by a prognostic technique in order to find which has the best performance in terms of travel time and/or energy consumption. Note that each step is executed with a predefined fixed maximum computation time, which guaranties that the complete strategy execution has a bounded computation time. Additionally, factors as: recurring and non-recurring traffic; EV consumption model that incorporates distance, velocity and inclination are included into decision making process.

In addition, the minor contributions are:

- The formulation of a fast-computing heuristic for solving K-Shortest Path Problem with diversity.
- A detailed description of the employed consumption model in terms of the parameters of the EV, and how this consumption model is incorporated into the decision making process.

2.3 K-Shortest Loopless Path Problem

Let $G = (V, A, W)$ be a directed graph, where V is the set of nodes, A is the set of edges, and W is the matrix of edge weights. The aim of the K-Shortest Loopless Path Problem is to calculate the K shortest loopless paths to go from s to t. More specifically, it computes a set of ranked paths $\mathcal{C} = \{P_1, ..., P_k\}$, where $\mathrm{d}(P_i) \leq \mathrm{d}(P_{i+1}) \forall i = 1, ..., k-1$. A known solution

for this problem is Yen's Algorithm presented in [40]. This algorithm works iteratively. It is initialized calculating the shortest path, denoted as P_1. Then, the other paths ($P_2, \cdots, P_K$) are calculated considering the fact that the i, denoted as P_i, is a deviation from some of the previously calculated paths ($P_1, \cdots, P_{i-1}$). Therefore, in each iteration in order to obtain P_i, it is necessary to look for all shortest deviation of the previous calculated path and to choose the one that has the shortest weight.

This problem is of interest when a set of possible paths is required for instance, for robustness purposes. Nevertheless, in some practical cases, the set of generated paths as the solution of this problem are highly similar due to a large number of shared edges among the shortest paths [41]. To address this issue, the K-Shortest Path Problem with diversity is formulated. In this case, one constraint is added to achieve that the set of resultant paths have a minimum diversity. This problem can be solved by brute force, that is applying the Yen's algorithm many times until to generate a set of K path that accomplishes the diversity requirement. Other solution is presented in [41], where the algorithm looks for the K paths iteratively discarding in advance paths whose similarity does not accomplish the diversity requirement.

2.4 Understanding Prognostic based on Particle Filter

In this section Prognostic based on Particle Filter will be introduced. For this purpose, first, the focus is placed on Particle Filter will and then, it will be on Prognosis based on Particle Filter. Both algorithms assume that the system under study can be described in space-state variables. Thus, the following system model structure is assumed as known:

Equation of transition:

$$x_k = f(x_{k-1}, u_{k-1}, w_{k-1}) \tag{2.8}$$

Equation of observation:

$$y_k = g(x_k, v_k) \tag{2.9}$$

where, $x_k \in \mathrm{R}^{n_x}$ is the state at time k; $y_k \in \mathrm{R}^{n_y}$ is the observation at time k; $f(\cdot)$ is the state transition function; $g(\cdot)$ is the state observation function; u_k is the input of the system; and w_k and v_k are the process noise and observation noise, respectively. In a general way, $f(\cdot)$ and $g(\cdot)$ are non-linear functions, while w_k and v_k are non-Gaussian noises.

Even tough both use the same system model representation, the aim of each one is different. In the following subsections each one will be analyzed.

2.4.1 Particle Filter

The PF is a Bayesian processor, and its main goal is to sequentially approximate the posterior Probability Density Function (PDF) of x_k, by a set of weighted particles. As a result, at time k the estimate of the posterior PDF of x_k is given by:

$$\hat{p}(x_k|y_{1:k}) = \sum_{\mathrm{i}=1}^{N_p} w_k^i \delta(x_k - x_k^i) \tag{2.10}$$

where $\{x_k^i, w_k^i\}_{i=1}^{N_p}$ is a set of weighted-particles; $y_{1:k} = \{y_1, ..., y_k\}$ is set of observations; N_p is the number of particles; x_k^i is position of particle i; w_k^i is the weight of particle i; and $\delta(x) = 1$ if $x = 0$ and $\delta(x) = 0$ otherwise. Note again that at time k the set $y_{1:k}$ is known because it is the set of past measurements, on the other hand, x_k is a random variable whose PDF is unknown, and thus the aim will be estimate its PDF conditional to $y_{1:k}$, namely its posterior PDF $\hat{p}(x_k|y_{1:k})$. The estimation process has two stages (see Figure 2.6): prediction and update.

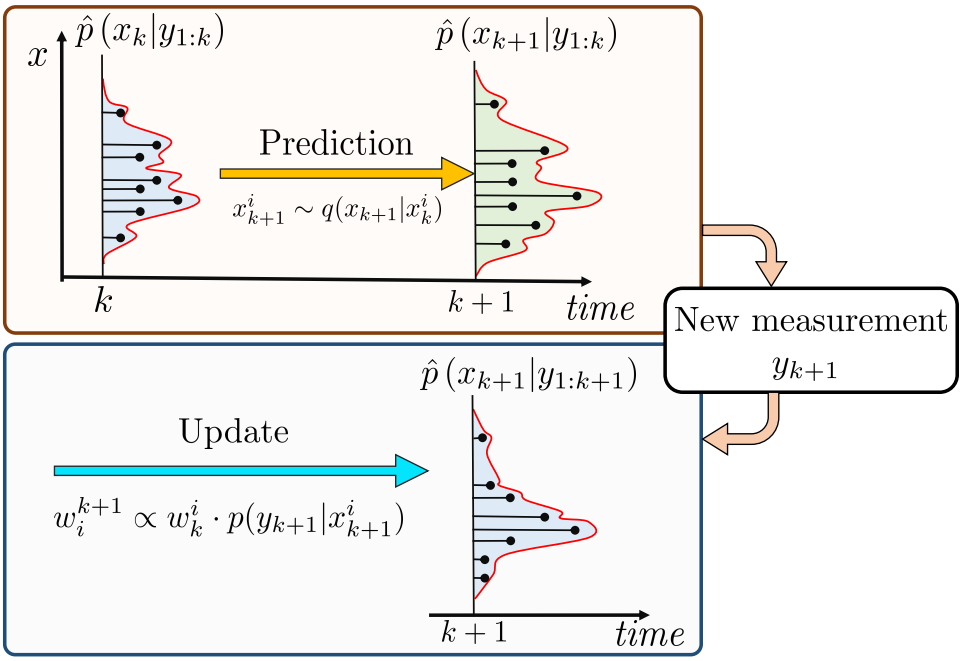


Figure 2.6: Step of PF: Prediction (top) and Update (below).

1. **Prediction**: each particle is propagated one time ahead using the transition model:

$$x_{k+1}^{\mathrm{i}} \sim q(x_{k+1}|x_k^{\mathrm{i}}) \tag{2.11}$$

where $q(x_{k+1}|x_k^{\mathrm{i}})$ is the conditional distribution of the states at time $k+1$ given the state at time k, and is calculated using the state transition equations (see Equation 2.8). In other words, x_{k+1}^{i} is sampled from $f(x_k^{\mathrm{i}}, u_{k-1}, w_{k-1})$; note that the PDF $f(x_k^{\mathrm{i}}, u_{k-1}, w_{k-1})$ depends solely of w_{k-1} because the x_k^{i} and u_{k-1} are fixed. An example of prediction step is shown in the top Figure 2.6.

2. **Update**: a new measurement arrives y_{k+1}, and it is employed to update the weight of each particle:

$$w_{k+1}^i \propto w_k^i \cdot p(y_{k+1}|x_{k+1}^i) \tag{2.12}$$

where $p(y_{k+1}|x_{k+1}^{\mathrm{i}})$ is the likelihood function, it is calculated using the equation of observation (see Equation 2.9). Note that $p(y_{k+1}|x_{k+1}^{\mathrm{i}})$ is the evaluation of the mea-

surement y_{k+1} in the likelihood function, thus it is a known value. Finally, the weights are normalized. An example of update step is shown in the below Figure 2.6.

Summarizing, PF works by steps. In the first, the transition model is used for prediction. Then, in the second, the observation model is used to update the prediction. Basically, the update stage makes a cross validation between the estimation using only the transition model and the measurement. If they are consistent, the particle weight will be incremented, otherwise it will be punished. The previous concepts and other implementations related to PF are discussed in depth in [42, 43].

2.4.2 Prognostic based on Particle Filter

Prognostic based on PF algorithm presented in [44] uses the PF, but for a different goal, and thus has some remarkable differences. Prognostic schemes intend to characterize the future state evolution of a dynamic system based on long-term predictions. In particular, in the case of Prognostic based on PF, the central concept is to model the propagation of uncertainty toward in time based on: a stochastic state-space model of the system, a probabilistic characterization of future operating profiles, and a PF-based estimate of the state. The central ideas of Prognostic based on PF are illustrated in Figure 2.7. In the following paragraphs, this prognostic algorithm will be introduced more formally.

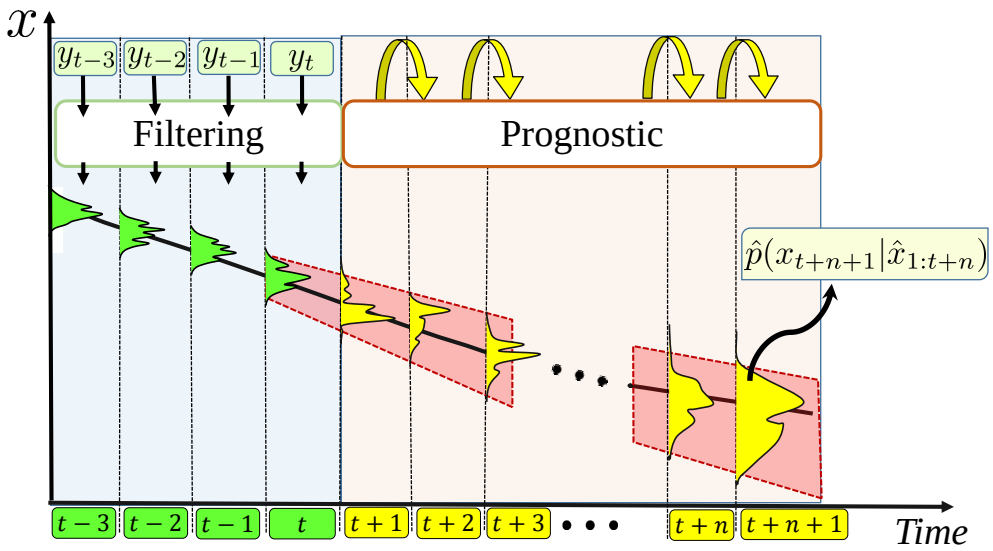


Figure 2.7: Example of Prognostic based on PF. In the illustration, the current time is t and the posterior PDF of x_t has been estimated by PF (green). The aim of Prognostic based on PF algorithm is to estimate the future evolution of the state $x_{t+k},\ k = 1, 2, \cdots, t+n+1$ (yellow). Remark that measurements solely are available until time t, therefore PF cannot be applied directly for prognostic purpose. In consequence, this prognostic algorithm proposes a methodology to estimate the future evolution, which only depends of the stochastic state-space model of the system, the probabilistic characterization of future operating profile and the initial estimate state.

Let be a time t, where the computation of the conditional PDF for prognostic at time $t+k$ $(k=1,...,n)$ is required. Using Prognostic based on PF algorithm, this PDF is computed in a sequential way using the last step prediction conditioning, that is $\hat{p}(x_{t+k}|\hat{x}_{1:t+k-1})$. Consequently, it is assumed that at time $t+k$ the algorithm starts with a set of weighted particles $\{x^{\mathrm{i}}_{t+k-1}, w^{\mathrm{i}}_{t+k-1}\}_{\mathrm{i}=1}^{N_p}$ resulting from estimate PDF at time $t+k-1$. Afterwards, the particles are propagated step by step according to the state transition equation. To calculate the PDF of the one-step propagated particles conditional to the previous state, the law of total probabilities is applied:

$$\hat{p}(x_{t+k}|\hat{x}_{1:t+k-1}) \approx \sum_{\mathrm{i}=1}^{N_p} w^{\mathrm{i}}_{t+k-1}\hat{p}(x^{\mathrm{i}}_{t+k}|\hat{x}^{\mathrm{i}}_{t+k-1}), \qquad \forall t \in \{1,...,n\} \tag{2.13}$$

Note that $\hat{p}(x^{\mathrm{i}}_{t+k}|\hat{x}^{\mathrm{i}}_{t+k-1})$ is related to the state transition equations thus a characterization of the futures inputs of the system is necessary. An update of the particle weights is also necessary, but as mentioned earlier, this cannot depend on the availability of new measurements in future time instants because they are unknown. One approach to circumvent this issue, and that has proven particularly useful in large prediction horizons [44], is based on the regularized PF algorithm [45]. Instead of updating particle weights in each prediction step, the uncertainty is represented by a re-sampling of the predicted state. This algorithm will be presented in the following paragraphs.

Using regularized PF for prognostic, the predicted pdf presented in 2.13 changes due to a Kernel is introduced as follows:

$$\hat{p}(x_{t+k}|\hat{x}_{1:t+k-1}) \approx \sum_{\mathrm{i}=1}^{N_p} w^{\mathrm{i}}_{t+k-1} K_h\left(x_{t+k} - \mathrm{E}\left\{x^{\mathrm{i}}_{t+k}|\hat{x}^{\mathrm{i}}_{t+k-1}\right\}\right) \tag{2.14}$$

$$K_h = \frac{1}{h^{n_x}} K\left(\frac{x}{h}\right), \quad h_{opt} = A \cdot N^{-\frac{1}{n_x+4}}, \quad A = \left(8c^{-1}_{n_x} \cdot (n_x+4) \cdot (2\sqrt{\pi})^{n_x}\right)^{\frac{1}{n_x+4}} \tag{2.15}$$

where $\dot{K()}$ is a kernel density function, which may correspond to the process noise pdf, a Gaussian kernel or a rescaled version of the Epanechnikov kernel; c_{n_x} is the volume of the unit sphere in R^{n_x}. The steps for regularization process are presented in the following:

1. Apply modified inverse transform resampling procedure. For $\mathrm{i}=1,...,N_p$ $w^{\mathrm{i}}_{t+k} = N_p^{-1}$.
2. Calculate $\hat{S}_{t+k}$, the empirical covariance matrix of $\left\{E\left[x^{\mathrm{i}}_{t+k}|\hat{x}^{\mathrm{i}}_{t+k-1}\right], w^{\mathrm{i}}_{t+k}\right\}_{\mathrm{i}=1}^{N_p}$
3. Compute $\hat{D}_{t+k}$ such that $\hat{D}_{t+k}\hat{D}^T_{t+k} = \hat{S}_{t+k}$
4. For $\mathrm{i}=1,...,N$, draw $\varepsilon^{\mathrm{i}} \sim K$, the Epanechnikov kernel and assign $\hat{x}^{(\mathrm{i})*}_{t+k} = \hat{x}^{(\mathrm{i})}_{t+k} + h^{opt}_{t+k}\hat{D}_{t+k}\varepsilon^{\mathrm{i}}$, where h^{opt}_{t+k} is computed as in 2.15.

Prognostic based on PF has been widely employed in diverse applications, including time of discharge prognostic in batteries [46], fault prognostic in railway tracks [47] and rotating machines [48]. Furthermore, this algorithm is currently considered the state-of-the-art for

model-based prognostic by many researchers within the Prognostic and Health Management community [49]. Being its main advantages the suitable uncertainly propagation and the real-time execution.

Chapter 3

Description of Models

In this chapter, the models in the simulation framework and those employed within the proposed routing strategy will be presented. Firstly, the road network and traffic model, which incorporate recurring and non-recurring traffic event, will be introduced in Section 3.1. Finally, the EV consumption model will be presented and analyzed in Section 3.2.

3.1 Road networks and traffic information model

Road networks and traffic information models are fundamental for a DSSPP strategy because both provide valuable information for the decision making process [3]. Therefore, their modeling should be clear respect to which information is available at each time. In the following subsections, both models will be described.

3.1.1 Modeling a Road Network

The road network is modeled as a directed graph, $G = (V, E)$, where V is a finite set of nodes or vertices and E is a finite list of edges. The distance and inclination of the edges are assumed as fixed quantities. On the other hand, the travel times of the edges are modeled as random variables that depend on traffic states.

For simulation purposes, a real road network database is employed; its geographic distribution is shown in Figure 3.1. The database has information about average travel times in the city divided by road segments. These average travel times are sampled every 30 minutes; hence, there are 48 data points per day. Moreover, the database has information about the type of road (freeway, avenue, and street) and the geographic coordinates (latitude and longitude) of each node. To obtain the lacking information (distance and inclination) the NetworkX python library [50] was used.

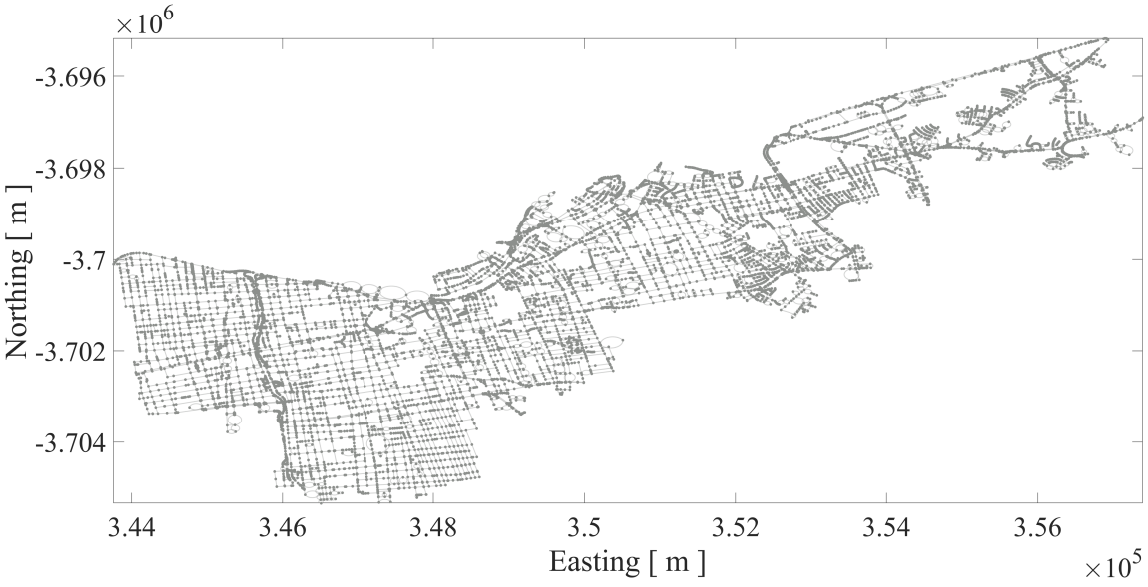


Figure 3.1: Geographic diagram of road network database.

3.1.2 Stochastic traffic model

Understanding of traffic congestion during the day is crucial because the travel times in the city are highly dependent on the state of traffic. Congestion can be classified into two categories: recurring and non-recurring [51]. Recurring congestion is the delay travelers regularly experience during known travel times (e.g., morning and evening rush hours). Non-recurring congestion delay is caused by a non-predictable or random events that disrupt traffic flow (e.g., vehicle breakdowns or crashes; road repair and inclement weather; special events that create sudden surges in demand such as the end of a sports event; and natural or human-made disasters).

The recurrent congestion is captured by a concept called Recurrent Speed Profile (RSP), which is an average speed edge for each time of the day. Another important concept in the traffic model is the Daily Speed Profile (DSP), which is a daily realization of a speed profile for one day. In the Figure 3.2 both concepts are illustrated by an example in a freeway segment; note that, most of the time, DSP is fluctuating near RSP; nevertheless, at certain times, it suffers significant decreases (e.g. between 8.46 and 10.1 hours); these drops are explained by non-recurring traffic congestion.

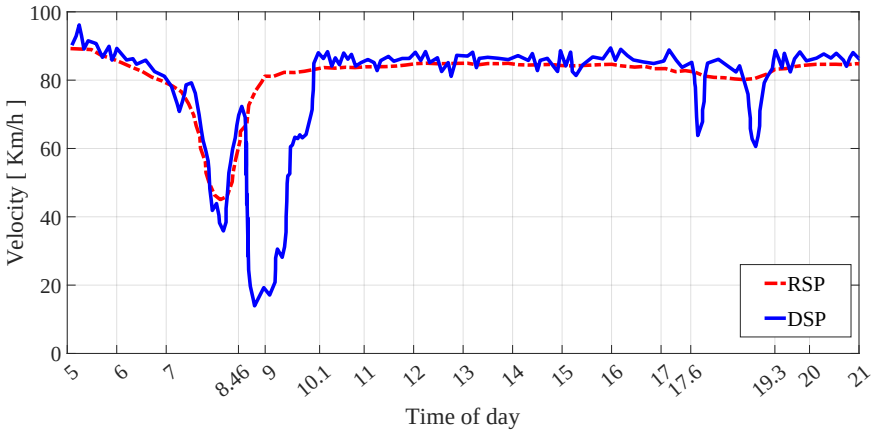


Figure 3.2: Velocity of a segment in a freeway (extracted from [52]).

Given the definition of RSP, it can be directly modeled using the database; thus, for each edge, its RSP is calculated as the edge length divided by the average travel time. However, using this procedure solely the average velocity sampled each 30 min is obtained. To calculate the RSP at any time, the unknown values are calculated by a linear interpolation between two nearest neighbors. Note that RSP has a deterministic behavior.

On the other hand, to recreate a DSP, a stochastic model is required. This model should be able to characterize the DSP fluctuation and the occurrence of non-recurring congestion.

In this work, the DSP fluctuation is modeled by the addition of a random disturbance velocity; in consequence, if the RSP of a certain edge at time t indicates that the average edge velocity is $v_{RSP}(t)$, the resultant DSP will be:

$$v_{DSP}(t) = v_{RSP}(t) + \omega(t) \tag{3.1}$$

where $\omega(t)$ is a stochastic process that is not time-dependent. Particularly, for the purposes of simulation, ω has been defined as a stochastic process, where at each time, $w(t)$ normal random variable with mean $0[km/h]$ and standard deviation $2[km/h]$. The mean zero is inspired by the fluctuation of DSP around of RSP; meanwhile, the standard deviation is calculated using the deviation of DSP with respect to RSP in the previously-reviewed example in Figure 3.2. The same non-time-variant disturbance model for all segments will be assumed. Figure 3.3 shows the same above-reviewed example, but an interval of $2 \cdot \sigma$ has been added to RSP; it depicts that the addition of a normal disturbance covered most of cases when DSP is under recurring congestion behavior; meanwhile non-recurring congestion requires alternative modeling.

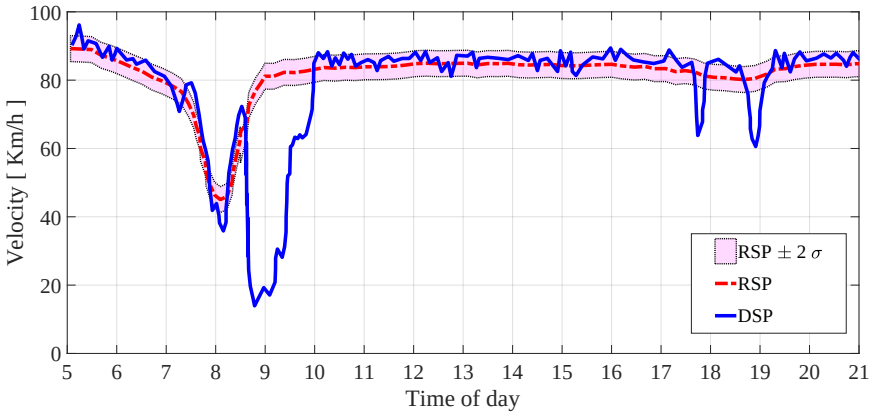


Figure 3.3: Velocity of a segment in a freeway

To model non-recurring congestion, another stochastic model is used. It should be able to statistically characterize the types of events jointly with their frequencies, duration, and impact on travel times. The data presented in [52] is used for this purpose. It has information on a high number of traffic incidents classified by type-incidents and road hierarchy. Additionally, statistics about duration and impact on the travel times are presented. A summary of the number of incidents depending on the type-incident and road hierarchy is shown in Table 3.1. Note that this database [30] classifies the road as freeways, arterials and locals. Thus, for simulation purpose, parallel relations between Freeway/Highway, Arterial/Avenue, and Local/Street are assumed; this allows the use this traffic incident data in the road network described in Subsection 3.1.1.

Incident type	Road hierarchy			Total
	Freeway	Arterial	Local	incidents
Crash	1246	2694	680	4620
Hazard	1964	680	453	4327
Stationary Vehicle	2212	4620	358	4643
Total	5422	1910	1491	13590

Table 3.1: Number of accidents for each type of road (extracted from [52])

For each column of Table 3.1, is possible to normalize by the total value; in this manner, the percentages of each type event dependent on road hierarchy are obtained, they are depicted in Table 3.2.

Type Event	Road hierarchy		
	Freeway	Arterial	Local
Crash	0.23	0.40	0.46
Hazard	0.36	0.29	0.30
Stationary Vehicle	0.41	0.31	0.24

Table 3.2: Normalized frequency of events for each type of road.

Furthermore, the statistics related to the duration of the events (mean and standard deviation) are shown in Table 3.3.

Incident type	Road hierarchy					
	Freeway		Arterial		Local	
	mean	std	mean	std	mean	std
Crash	41.30	41.10	40.80	37.70	42.50	42.80
Hazard	69.30	73.50	96.90	98.40	104.30	105.90
Stationary Vehicle	52.90	71.80	32.80	36.00	42.50	58.20

Table 3.3: Mean and standard deviation of the duration of events (extracted from [52]).

Data presented in [52] is also used to characterize the impact of a traffic incident on travel times. In [52], the impact of traffic incidents was analyzed regarding travel times for a portion of the freeway measuring 14.1 [km] long. The extra time spent due to traffic incidents on the freeway segments was denominated Extra Buffer Time (EBT); the statistics of EBT mean values are shown in the second column of Table 3.4 for each type of incident. Note that these values summarize the EBT for the total portion of freeway, but they do not take into consideration the specific impact in the affected segments. To estimate the impact of a traffic incident in a determined freeway segment, it is assumed that the portion of the freeway is divided into 15 segments with the same length. Also, it is assumed that, on average, the incidents have the same extension (equal to two segments). Under these assumptions, the EBT must be explained by a change in the travel times in the two affected segments. For instance, in the case of crash incidents the mean EBT is $7.3[min]$ (see second column Table 3.4). This must be explained by an extra travel time in two affected freeway segments; then, each segment has an extra travel time of $3.6[min]$. Given that under recurrent congestion, the segment travel time was $0.8[min]$, the occurrence of crash incidents implies an increase of segment travel time by a factor of 5.56. This last number is called Incident Impact Factor by Segment (IIFS) in the present work and, it is calculated using the same procedure for the other types of incidents, as seen on the third column of Table 3.4.

type	Mean EBT [min]	IIFS
Crash	7.30	5.56
hazard	1.96	2.23
vehicle	2.30	2.44

Table 3.4: EBT route and Normalization Factor edge (adapted from [52]).

For simulation model, it is assumed that every type of incident has the same IIFS for all types of road.

At this point, all data needed for modelling the relative frequency of traffic incidents (see Table 3.2), the duration of incidents (see Table 3.3), and the impact of traffic incident on travel time segment (see Table 3.4) has been presented. Therefore, the stochastic traffic model can now be introduced.

The proposed traffic model considers that the traffic with a time period t_u. In this work it is fixed as $t_u = 2[min]$. This is value is inspired by known routing software, Garmin, which updates every two minutes to check traffic situations [53]. As a result, the traffic state model evolves as a discrete time process. Furthermore, it is known that travel times depend on the current state of traffic due to the presence of non-recurring event generates travel delays. Particularly, given the employed data for traffic incident modeling, the traffic state only can be one of four possibilities: recurrent, crash, hazard, or stationary vehicle. Then, considering that the traffic evolution process should be in discrete time and with finite states, a Discrete Markov Chain model with four states is proposed:

- $\mathcal{T}_1 \rightarrow$ Recurrent traffic
- $\mathcal{T}_2 \rightarrow$ Crash
- $\mathcal{T}_3 \rightarrow$ Hazard
- $\mathcal{T}_4 \rightarrow$ Stationary Vehicle

Consequently, at time t each edge has a state of traffic $s(t) \in \{\mathcal{T}_{\mathrm{i}}\}_{\mathrm{i}=1}^{4}$ whose evolves according to a Markov Chain model. Under the recurrent traffic state ($\mathcal{T}_1$) the state evolution is independent of edge neighborhood, and the realizations of edge velocity are given by Equation 3.1. On the other hand, for the others states $s(t) \in \{\mathcal{T}_{\mathrm{i}}\}_{\mathrm{i}=2}^{4}$, the state evolution is not independent on the neighborhood. This is because in a real situation, when a traffic incident occurs, it usually propagates to near edges. In this regard, different models have been proposed to model this phenomena [54–56]. Particularly, in this work, a simplified model of cascading behavior [56] is adopted. This model considers that when a traffic incident occurs on an edge $e = (v_x, v_y)$, it affects the edge neighborhood of e, $N(e)$, defined as:

$$N(e) := \{(u, v) \in E | v = v_x \; or \; u = v_y\} \tag{3.2}$$

As a result, the cascade state of traffic evolves according to the state of e, thus the edges of $N(e)$ are dominated by the behavior of e. Furthermore, under these states $s \in \{\mathcal{T}_{\mathrm{i}}\}_{\mathrm{i}=2}^{4}$, the realizations of edge velocity are calculated as follows:

$$v_{DSP}(t) = \gamma(s(t)) \cdot v_{RSP}(t) + \omega(t) \tag{3.3}$$

where $\gamma(s(t))$ is obtained from Table 3.4 depending on the current traffic state $s(t)$; basically, the extra time due to a traffic incident can be converted into a reduction on the average velocity.

Given this Markov Chain model, the complete evolution model is defined by the initial states called $s(0)$ and the transition probability matrix [57]:

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} \tag{3.4}$$

where, m_{ij} is the transition probability to $\mathcal{T}_j$ given the current is $\mathcal{T}_i$, thus $\forall i = 1, \cdots, 4$ $\sum_{j=1}^{4} m_{ij} = 1$. Furthermore, given a state realization it is possible to characterize the expected time in this state as follows. If the current state traffic is $\mathcal{T}_n$, the time in this state t_n follows a Geometric distribution where the "success" is the sum of transition probabilities to states $\mathcal{T}_j(t)$ with $j \neq n$ i.e. $p = \sum_{j \neq n} m_{nj}$. Therefore, the mean and the variance can be computed as follows:

$$E(t_n) = \frac{1}{p} \tag{3.5}$$

$$Var(t_n) = \frac{1-p}{p^2} \tag{3.6}$$

These quantities are expressed as dimensionless unit time because these are the fractions of the update period; to convert to time in minutes they need to be multiplied by t_u.

Following the state definition, the first row of M defines the occurrence probability of each type of event, while the other rows define the duration of each event. Therefore, both frequency and duration are being modeled by this Markov Chain. In addition, please note that M will have different values depending on the type of route since the relative frequency incident and incident duration vary according to the type route.

The following structure for the matrix M is assumed:

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & 0 & 0 \\ m_{31} & 0 & m_{33} & 0 \\ m_{41} & 0 & 0 & m_{44} \end{bmatrix} \tag{3.7}$$

The underlying idea is that when an incident occurs, the traffic can only evolve to the recurrent state.

In the proposed model, the first row of M is built using information presented in [58] and [52]. In [58], the authors presented a procedure to estimate the car crash probability for street segments using a database. They model the crash probability as a Bernoulli experiment and they conclude that the crash probability in a certain segment $p \in [0, \theta \cdot 10^{-4}],\ \theta = 0, ..., 9$. Considering this condition, for simulation purpose, the present work is assumes that the crash probability must be inside of this interval. To achieve this goal, a procedure is designed; in the following paragraphs, it will be illustrated for the case local road, the other cases are analogous. The first step is to fix a scaling factor $\xi \in [0, \theta \cdot 10^{-4}],\ \theta = 0, ..., 9$, it defines the probability of incidents; therefore, $m_{11} = 1 - \xi$. This probability must be the sum of crash, hazard and stationary vehicle probabilities; hence, $\xi = \sum_{j=2}^{4} m_{1j}$. Nevertheless, the relative frequency of type incidents is known for the local road (see column 4 on Table 3.2); in consequence, the individual incident probability for each type of incident is calculated as follows:

$$m_{12} = 0.46 \cdot \xi \tag{3.8}$$
$$m_{13} = 0.30 \cdot \xi \tag{3.9}$$
$$m_{14} = 0.24 \cdot \xi \tag{3.10}$$

The other rows are defined considering the mean duration of each type of event (see Table 3.3) and (3.5). The procedure will be shown in the case of crash incident in local roads, the other cases are analogous. In this case the mean incident duration is $42,50$ [min] (see column 6 on Table 3.3); then, this duration is counted based on discrete scale of the model (it is divided by t_u), resulting in $42,5/2 \approx 21$. Afterwards, the Equation 3.5 is used to define p_{21} such that the $E(t_2)$ must be equal to 21, it is shown in the following:

$$E(t_1) = 21 \tag{3.11}$$
$$= \frac{1}{p} \tag{3.12}$$

As a result, $p = 0.05$. Therefore,

$$M_{local}(2,:) = [0.05 \quad 0.95 \quad 0 \quad 0] \tag{3.13}$$

Finally, applying the same procedure to other incidents for local roads, the resultant transition matrix is the following:

$$M_{local} = \begin{bmatrix} 1-\xi & 0.46 \cdot \xi & 0.30 \cdot \xi & 0.24 \cdot \xi \\ 0.05 & 0.95 & 0 & 0 \\ 0.02 & 0 & 0.98 & 0 \\ 0.05 & 0 & 0 & 0.95 \end{bmatrix} \tag{3.14}$$

By construction, the procedure presented above achieves equality between the mean duration of traffic incidents and the mean duration of traffic incidents in data; this fact is illustrated in the case of crash incident in Local roads, see in (3.11)-(3.13). On the other hand, the standard deviation of the duration of traffic incidents is fixed when the mean is fixed (square root of (3.6)). Notably, in the case of local roads, the standard deviation of the duration of incidents is calculated for the proposed model, and it is contrasted database again; the results are shown on Table 3.5. They show that for the three type of incidents the model can also characterize the standard deviation with a low error, being in the case of crash and hazard less than 8%, while in the case of stationary vehicle equal to 30%.

Type	SD database [min]	SD model [min]	Error [%]
Crash	42.8	38.9	8
Hazard	105.9	98.9	6
Stationary vehicle	58.2	38.9	33

Table 3.5: Standard Deviation (SD) of incident duration for proposed model and database for local roads.

Finally, for simulation purposes, the initial traffic states $s(0)$ are sampled from $\pi = M(1,:)$, where $M(1,:)$ is the first row of M.

3.2 EV energy consumption model

As reviewed in Chapter 1, a detailed energy consumption model is critical for EV routing purposes. Consequently, factors such as elevation, velocity, acceleration, and EV specifications must be considered. Taking this into consideration, the EV energy consumption model (EV-ECM) presented in [59] is chosen because it provides a detailed description of how to obtain the relevant parameters. This is key because it allows an easier real-word implementation of the consumption model. In addition, this consumption model can be easily tuned using public available EV data (using the procedure described in [59]) without the need for field data collection. Even though this consumption model is simple (it does not consider slippage nor sinkage in the wheels), it allows to properly characterize the EV consumption. Furthermore, it was tested with empirical data obtained from an EV, resulting an average error of a 5.9%.

The model is derived as follows. First, the power at the wheels is computed using the Newton's Laws:

$$\begin{aligned} P_{Wheels}(t) = v(t) \cdot \Big(& ma(t) + mg \cdot cos(\theta) \frac{C_r}{1000} (c_1 v(t) + c2) \\ & + \frac{1}{2} \rho_{air} A_f C_D v^2(t) + mg \cdot sin(\theta) \Big) \end{aligned} \tag{3.15}$$

where $v(t)$ is the EV velocity, $a(t)$ is the EV acceleration, $\theta[rad]$ is the road inclination, and the other parameters with their respective values are shown in the Table 3.6.

Parameters	Notation	Unit	Values
Vehicle weight (including driver)	m	$[kg]$	1521
	C_r		1.75
Rolling resistance parameters	c_1		4.575
	c_2		1.75
Air mass density	ρ_{air}	$[kg/m^3]$	1.2256
Frontal area of the vehicle	A_f	$[m^2]$	2.3316
Gravitational acceleration	g	$[m/s^2]$	9.8
Aerodynamic drag coefficient of the vehicle	C_D		0.28

Table 3.6: Parameters of EV-ECM ([59]).

The power delivered by the battery is given by:

$$P_{bat} = \frac{P_{Wheels}}{\eta_{Driveline} \cdot \eta_{Motor} \cdot \eta_{Bat}} \tag{3.16}$$

where $\eta_{Driveline} = 0.92$ is the driveline efficiency; η_{Motor} is the electric motor efficiency, which is equal to 0.91; and η_{Bat} is the battery efficiency, which is equal to 0.9.

Note that it has been considered that $P_{Wheels} \geq 0$. Nevertheless, when the vehicle is braking, then $P_{Wheels} \leq 0$ (denoted as P^{-}_{Wheels}), i.e. the regenerative braking works recovering energy. In this case, the model for regenerative power is given by:

$$P_{reg} = \eta_{rb} \cdot P^{-}_{Wheels} \tag{3.17}$$

where η_{rb} is the regenerative efficiency. It depends on the acceleration as follows:

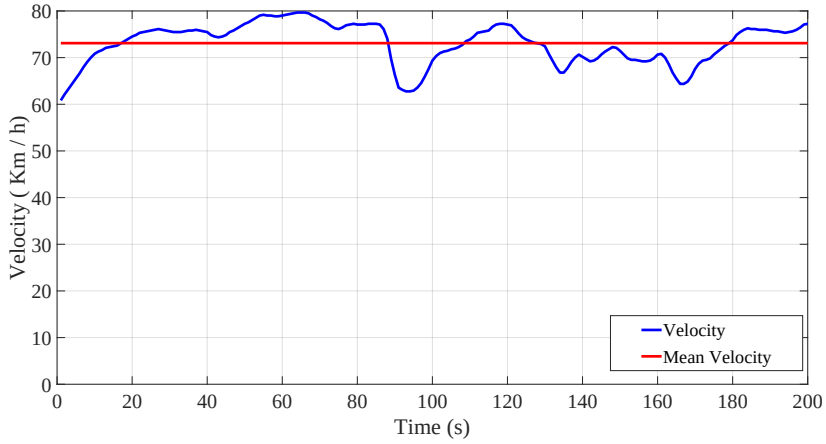
$$\eta_{rb} = \begin{cases} \left[\mathrm{e}^{\left(\frac{0.0411}{|a(t)|}\right)}\right]^{-1} & \text{if} \quad a(t) < 0 \\ 0 & \text{if} \quad a(t) \geq 0 \end{cases} \tag{3.18}$$

This EV-ECM considers that $\theta(t)$ and $v(t)$ are measured at each sample time. Consequently, to simulate the EV consumption in an edge, it is necessary to know the $\theta(t)$ and the $v(t)$ at each time. The above leads to a high computational cost. To address this issue, are assumed a constant angle and a constant velocity (equal to the mean velocity) along each edge. The angle constant is a reasonable assumption given that the edges are not too large. Nevertheless, the second assumption needs to be analyzed in more details because it directly affects the regenerative model. More specifically, assuming constant velocity, the effect of acceleration is neglected. This begs the following question: what is the impact of the proposed approximation in the precision of EV-ECM?

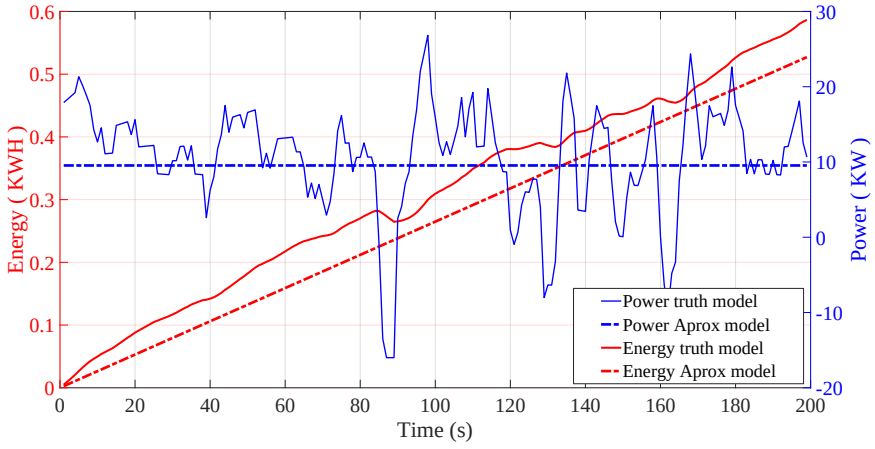
To answer this question, two examples are developed, considering two different driving cycles obtained from [60]: Highway fuel economy (HWFET) and New York (NY) driving cycles. For each one, a window time of $200(s)$ is defined; then, the power and cumulative

energy consumption are calculated considering the truth and the approximated EV-ECMs. The results are shown in Figures 3.4 and 3.5.

In the case of HWFET, the velocity varies around the average value (see Figure 3.4a); thus the assumption of constant velocity is reasonable. In this case, the approximated model has an error of -10% with respect to the truth model. In other words, the approximated model underestimates the EV energy consumption; this is explained by the neglected effect of acceleration. The underestimation of EV energy consumption is worse in the decision making context because a model that underestimates can lead a path thinking that the current energy is enough to complete it, when, in reality, and considering the estimation error, the battery will deplete its charge in the middle of the route. Note that, in this case, the neglected regenerative braking does not play an important role because the power is ≥ 0 (see Figure 3.4b) most of the time.



(a) HWFET velocity

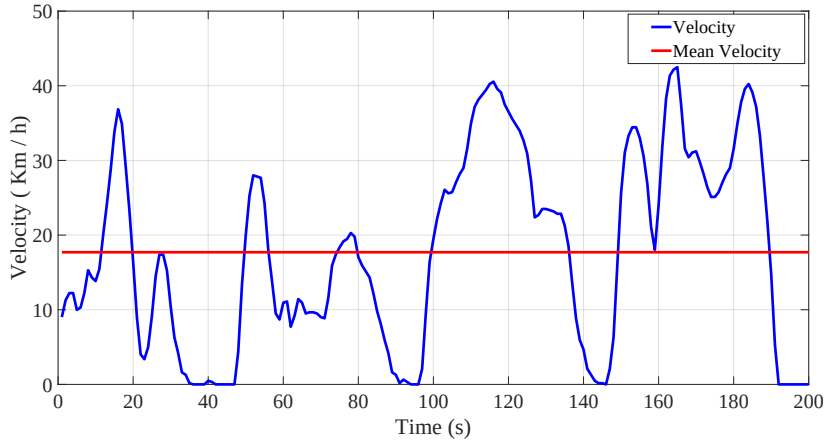


(b) HWFET power and Energy. Consumption Error= -10%

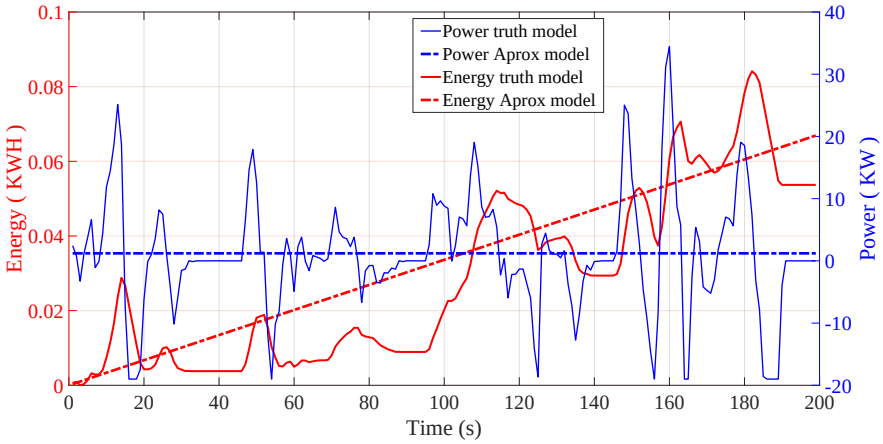
Figure 3.4: Test using HWFET driving cycle assuming inclination angle constant equal to 0.1(degree)

On the other hand, in the case of NY, the velocity is highly varying (see Figure 3.5a),

and the average velocity does not represent the real driving velocity. It leads to a significant EV energy consumption error, equal to 24%. Thus, in this case, the approximated model overestimates the EV energy consumption. This error is explained by the fact that the regenerative braking is not considered; in specific, the Figure 3.5b shows that there are several instances where the power is ≤ 0; thus the regenerative braking works and is being neglected.



(a) NY velocity



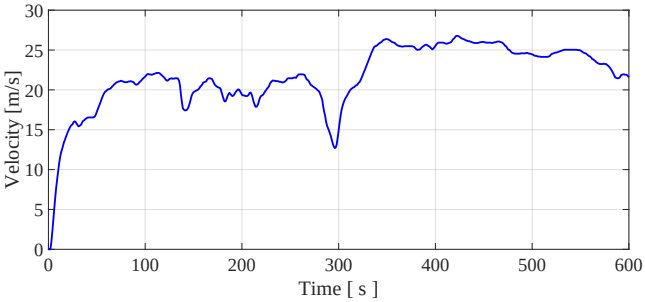
(b) NY power and Energy. Consumption Error= 24%

Figure 3.5: Test using NY driving cycle assuming inclination angle constant equal to 0.1(de$g$ree).

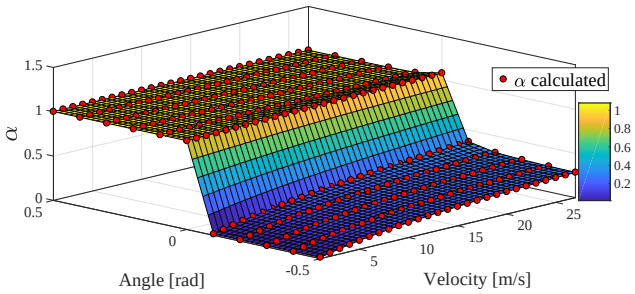
A procedure is designed to treat this discrepancy between the truth and approximated model. This procedure will allow to improve the approximated EV consumption model by a correction factor which is calculated for different mean velocities and inclinations. In other words, the EV consumption will be calculated using the approximated model obtaining E_{approx}, then this consumption will be multiplied by a correction factor α, as result the estimate consumption will be $E_{approx} \cdot \alpha$. Using this procedure, when $\alpha < 1$ the overestimation problem is corrected; meanwhile, when $\alpha > 1$ the underestimation problem is corrected. In

the following paragraph, the correction factors will be pre-computed for different pairs of mean velocities and inclinations.

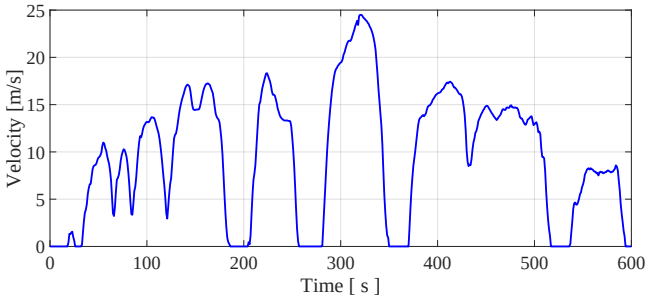
Given a road segment with a certain driving cycle whose mean velocity is v_{mean} and inclination is θ, the EV consumption is calculated using both the approximated and truth model, obtaining E_{exact} and E_{approx}, respectively. Then, the correction factor can be calculated as $\alpha(\theta, v_{mean}) = \frac{E_{exact}}{E_{approx}}$. Then, a grid of different inclination angles and mean velocities are built. For each element on the grid, the EV-consumption is calculated using both the truth and the approximated models, with these a correction factor is calculated α. To calculate E_{exact}, the velocity of the driving cycle is multiplied $v_{mean}/v_{mean-driving-cyle}$ in such a way that resultant driving cycle has a mean velocity equal to v_{mean}. The above procedure is applied for each type of road and the results are shown in the Figures 3.6b , 3.6d and 3.6f.



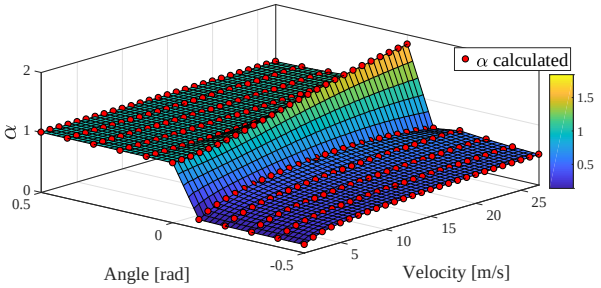
(a) HWFET driving cycle [60].



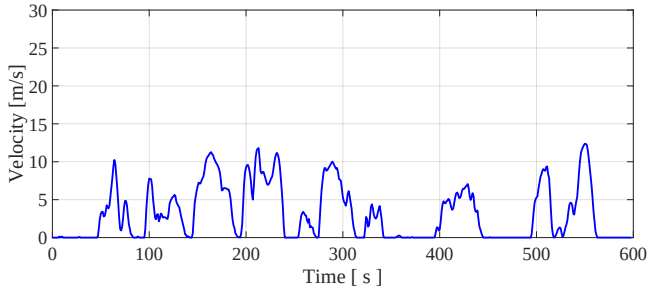
(b) α for freeway roads.



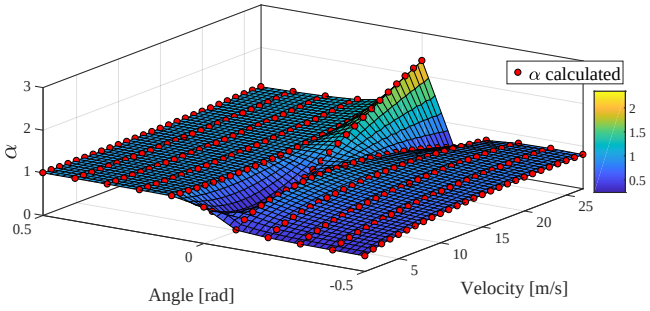
(c) SC03 driving cycle [60].



(d) α for arterial roads.



(e) NY driving cycle [60].



(f) α for local road.

Figure 3.6: α for different road types.

Chapter 4

Proposed routing strategy

As mentioned in Chapter 1, the main focus of this work is to propose a real-time strategy to solve the EV-DSSPP. The designed proposal is composed by two main parts. The first part is the design and implementation of a routing strategy that allows to find the optimal path to go from one starting point to one destination. The second part addresses the problem of how to use the previously designed routing strategy to compute en-route path updates. The complete proposal will be presented in this chapter.

The proposed routing strategy for finding the optimal path to go from one starting point to one destination is summarized in Figure 4.1. Roughly speaking, this strategy works as follows. First, it calculates the K path candidates between the starting-point to destination. Second, for each of these K paths, a prognostic algorithm is executed to characterize them probabilistically in terms of travel time and SOC performance. Finally, the optimal path is chosen according to the evaluation of a cost function. Remark that each step of this proposed routing strategy is designed considering a predefined fixed maximum computation time in order to achieve that the complete execution of the proposed routing strategy has at most a fixed and bounded computation time T_{max}. This strategy will be explained in detail in Sections 4.1- 4.3. In addition, in Section 4.4 the en-route updates strategy will be presented.

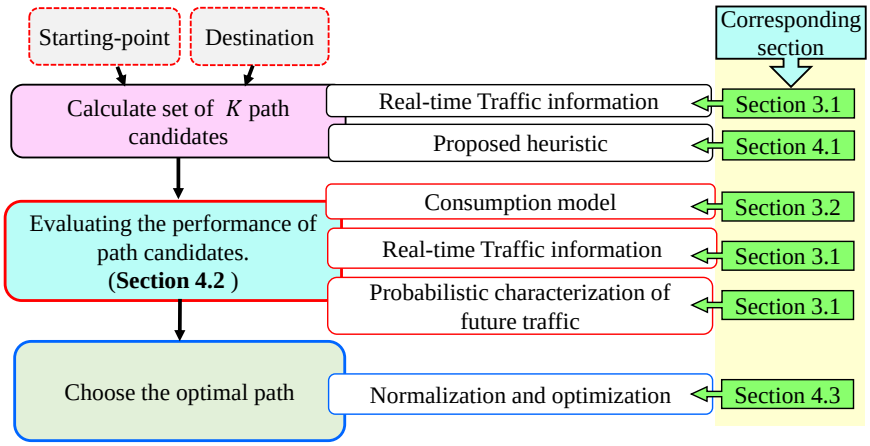


Figure 4.1: Finding the optimal path using the proposed routing strategy.

4.1 Heuristic for finding path candidates

As already presented in Figure 4.1, the proposal routing strategy works by steps. In the first step, K path candidates are calculated, and then they will be evaluated by a prognostic algorithm execution (it will be studied in the next section) in order to find the optimal path inside of this set. At this point, a reasonable question is: Why only K path candidates are calculated? The answer is related with the limited computing time because one of the main objectives of this proposal is to solve the EV-DSSPP in a real-time application and the number of candidates directly affects the computing time. In this section the method for finding path candidates will be presented.

The path candidates are selected in a deterministic and static way, considering the current expected travel time of each edge. In other words, the travel time of each edge is fixed as the current expected value, and then the candidates are calculated from this deterministic and static network. This allows to calculate the candidates faster than stochastic and/or dynamic methods saving computing time for prognostic execution, which is more demanding computationally. To calculate the candidates, the K-SPP method [40] can be used because it provides a set of ranked shortest paths. The best method (regarding the worst computational time) is the Yen's algorithm. Nevertheless, this algorithm does not achieve diversity between candidates. Diversity is critical because it allows to explore a wider zone which may allow to find better solutions. The diversity between candidates can be ensured solving the Loopless K-Shortest path problem with diversity [41]. In such a case, an additional constraint is added to K-SPP with the aim of achieving diversity between solutions. Even though the method produces good results, the implementation for real-time applications is difficult due to the high computational cost. Considering the above, a heuristic strategy to look for path candidates is proposed, it is inspired by the occurrence of a high-impact incident when one node is broken down, meaning alternative paths need to be explored.

Before introducing the heuristic, a metric to measure the diversity between two paths and between a set of paths will be introduced. The diversity can be measured in terms of the similarity, in the sense of if two path are highly similar, then the diversity between them is low, and vice verse. For two path $\mathcal{P}_x$ and $\mathcal{P}_y$ a similarity function is given by:

$$Sim_{path}(\mathcal{P}_x, \mathcal{P}_y) = \frac{L\left(S_{\mathcal{P}_x} \cap S_{\mathcal{P}_y}\right)}{\min\left\{L(\mathcal{P}_x), L(\mathcal{P}_y)\right\}} \tag{4.1}$$

where $S_{\mathcal{P}}$ is the edges set of path $\mathcal{P}$ and $L(\mathcal{P})$ is the length of path $\mathcal{P}$. Note that $0 \leq Sim(\mathcal{P}_x, \mathcal{P}_y) \leq 1$, where $Sim(\mathcal{P}_x, \mathcal{P}_y) = 0$ if $\mathcal{P}_x$ share no edges with $\mathcal{P}_y$; $Sim(\mathcal{P}_x, \mathcal{P}_y) = 1$ if the shortest path is contained in the other path; otherwise, $0 < Sim(\mathcal{P}_x, \mathcal{P}_y) < 1$. This function is selected from a set of similarity functions presented in [41].

In addition, the similarity function between one path $\mathcal{P}$ and a set of paths $\mathcal{C}$ is defined as follows:

$$Sim_{set}(\mathcal{P}, \mathcal{C}) = \max_{r \in \mathcal{C}} Sim_{path}(r, \mathcal{P}) \tag{4.2}$$

To design the heuristic, it is considered that there is a fixed time for computing purpose t^H_{max}; in this manner, the heuristic will always have at most the same execution time. t^H_{max} must be defined considering the available computing time. The proposed heuristic works with a loop; in each iteration, a possible path candidate is generated, and the list of feasible candidates $\mathcal{FC}$ is built. To generate path candidates, the heuristic removes n_{d} random-nodes from the graph, they are selected from a list of before-visited nodes Vis which is updated when a new candidate is found; then, the shortest path in the resultant graph is calculated using Dijkstra Algorithm (see Algorithm 1). If this path has a diversity respect to $\mathcal{FC}$ less than a threshold Δ, the path is added to a set of feasible candidates $\mathcal{FC}$. Otherwise, it is discarded. This procedure is repeated until the maximum computing time t^H_{max} is reached. More formally, the heuristic is presented by a pseudo-code in Algorithm 2. Lines $1-4$ guarantee that the shortest path will be in $\mathcal{C}$. Lines $5-6$ achieve that $1 \leq n_{\mathrm{d}} \leq n_{max}$. Then, in the lines $7-13$ the loop is executed until t^H_{max} is reached. Finally, in line 14, the k_c shortest paths are selected from $\mathcal{FC}$.

Algorithm 2 *Heuristic for finding path candidates*

Input : Graph G, Starting-point s, Destination d, Required number of candidates path k_c, Similarity function $S(\cdot,\cdot)$, Similarity threshold Δ, Factor of exploration τ, Maximum number of removed nodes n_{max}, Maximum time for computing t^H_{max}
Output: Path candidates $\mathcal{C}$

Initiation

1: $t \leftarrow 0$ ▷ Execution clock initiation
2: $\mathcal{P} \leftarrow \mathrm{ShortestPath}(G, s, \mathrm{d})$ ▷ Shortest path from s to d in the graph G
3: $\mathcal{FC} \leftarrow \mathcal{P}$ ▷ Add $\mathcal{P}$ to the of feasible candidates
4: $Vis \leftarrow \{\mathcal{P}\}$ ▷ Add nodes of $\mathcal{P}$ to the set of visited nodes
5: $n_{\mathrm{d}} \leftarrow \min\{\lceil |\mathcal{P}| \cdot \tau \rceil, n_{max}\}$ ▷ Number of removed-random nodes
6: $n_{\mathrm{d}} \leftarrow \max\{n_{\mathrm{d}}, 1\}$

Loop

7: **while** $t < t^H_{max}$ **do**
8: $r \leftarrow UniformSample(Vis, n_{\mathrm{d}})$ ▷ Remove n_{d} random nodes from Vis
9: $G' \leftarrow G - r$
10: $\mathcal{P} \leftarrow \mathrm{ShortestPath}(G', s, \mathrm{d}, \mathcal{S})$
11: **if** $Sim_{set}(\mathcal{P}, \mathcal{C}) \leq \Delta$ **then** ▷ Diversity check
12: $\mathcal{FC} \leftarrow \mathcal{FC} \cup \mathcal{P}$
13: $Vis \leftarrow Vis \cup \mathcal{P}$

Finalization

14: $\mathcal{C} \leftarrow KShortestsPath(\mathcal{FC}, k_c)$ ▷ Choose k_c-shortests path from $\mathcal{FC}$
15: **return** $\mathcal{C}$

In the present work, Δ is fixed as 0.7 , and t^H_{max} is fixed as $3(s)$. On the other hand, the parameters τ and n_{max} are designed as follows. First, two arrays with possible values for τ and n_{max} are built, then all combinations between them are tested. Given a possible value τ^* and n^*_{max}, n_{rep} random instances are generated. In each instance, two nodes are randomly chosen, and Algorithm 2 is executed to look for k path candidates considering the respective value of τ^* and n^*_{max}. In certain cases, the heuristic would not work due to the current values of τ^* and n^*_{max}. Thus, variable $n_k(\tau^*, n^*_{max})$ is introduced to counts how many times the

k candidates are not calculated in the n_{rep} instances. This variable $n_k(\cdot,\cdot)$ represents the ability to achieve the goal of k candidates. In addition, variable $z(\tau^*, n^*_{max})$ is introduced to quantify the mean of the percentage difference between the shortest path and the other $k-1$ candidates. This variable $z(\tau^*, n^*_{max})$ represents the ability to find candidates with similar performance of the shortest path. Finally, to evaluate the performance of (τ^*, n^*_{max}) the following function is defined:

$$f(\tau^*, n^*_{max}) = \frac{z(\tau^*, n^*_{max})}{n_{rep} - n_k(\tau^*, n^*_{max})} + \frac{n_k(\tau^*, n^*_{max})}{n_{rep}} \tag{4.3}$$

This function is a sum of average ability to find candidates close to the shortest path (first summand) and to find the required number of candidates (second summand); consequently, the objective will be to minimize this function.

This procedure was implemented considering $n_{rep} = 500$ and $k = 5$; the result is shown in the Figure 4.2. It shows that for certain values the function is smaller, and, in particular, the minimum is reached in $n_{max} = 4$ and $\tau = 0.03$.

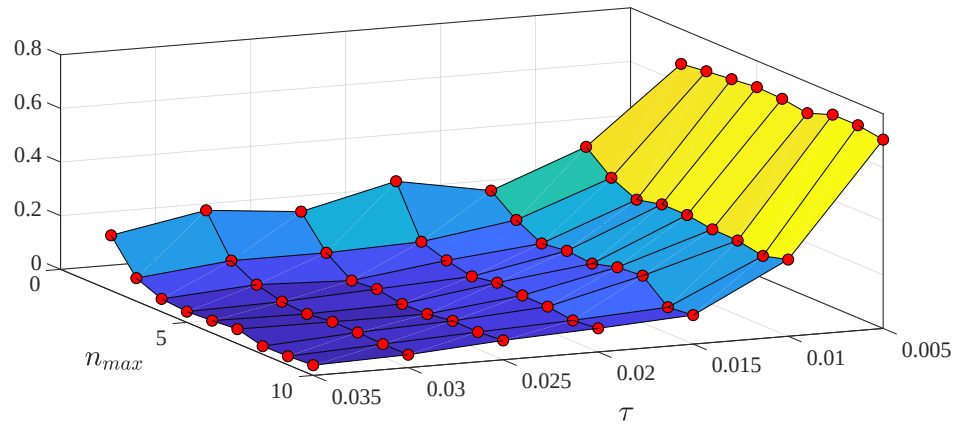


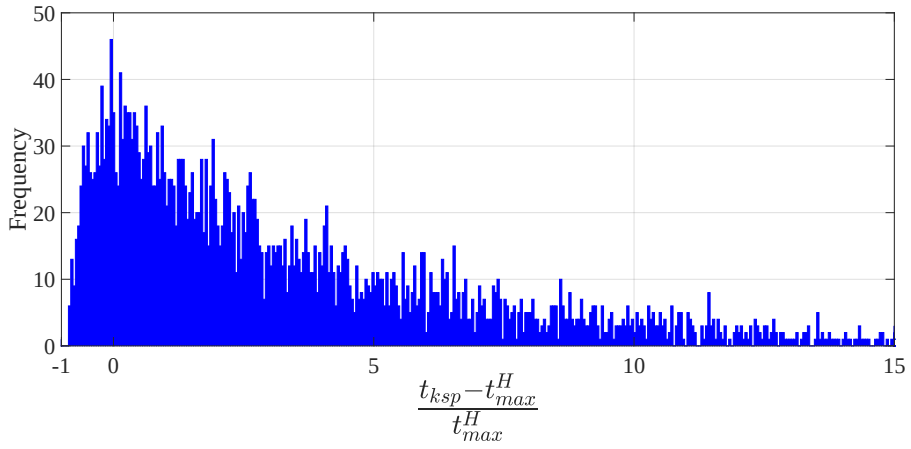
Figure 4.2: Tuning heuristic.

Having tuned the parameters of the heuristic, it is compared with Yen's algorithm. For this purpose, 3500 random instances are generated; in each instance the following steps are applied:

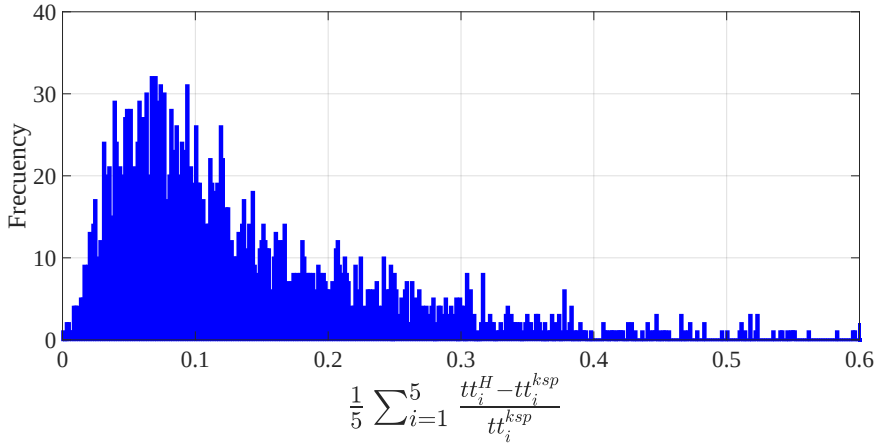
1. Two nodes are randomly selected (the first node is the Starting-point,s, and the second is the Destination, d).
2. 5 path candidates to go from s to d are calculated using Yen's algorithm. The expected travel times for each candidate TT_i^{ksp} $i = 1, ..., 5$ and the computing time are saved.
3. 5 path candidates to go from s to d are calculated using proposed heuristic implemented with the tuned parameters. The expected travel times for each candidate TT_i^H $i = 1, ..., 5$ and the computing time are saved.

Note that given that proposed heuristic has a diversity restriction, the resulting candidate sets are not necessary equal to the candidates resultant using K-SPP; even more, using the

heuristic might produce an extra expected travel time due to other paths are explored to accomplish the diversity restriction. The comparison of both algorithms is made in terms of computing time and average extra expected travel time (see Figure 4.3). The difference of computing time between both algorithm is shown in Figure 4.3a. This Figure shows that in most cases, the heuristic is faster than Yen's algorithm. Furthermore, given the design of heuristic, it can be $3[s]$ slower in the worst cases. The average expected travel time using proposed heuristic is shown in Figure 4.3b. This figure depicts that, using proposed heuristic, most of cases there are an extra expected travel, particularly the mean value is 0.14. Summarizing, proposed heuristic allows to reduce considerably the computing time and to accomplish a set of candidates with diversity; although, the resulting candidates may have higher expected travel time.



(a) Extra computational time of Yen's algorithm with respect to the proposed heuristic. Where t_{ksp} is the execution time of Yen's algorithm and t^H_{max} is the execution time of the proposed heuristic; it is fixed and equal to $3[s]$.



(b) Average extra expected travel time using proposed heuristic with respect to Yen's algorithm. Where tt_{i}^{ksp} is the travel time of path i obtained using Yen's, whereas tt_{i}^{H} is the travel time of path i obtained using proposed heuristic.

Figure 4.3: Performance comparison between Yen's Algorithm and proposed heuristic.

4.2 Performance evaluation of path candidates

Having calculated a set of K path candidates, now the objective will be to propose a method to evaluate the path performances in advance (i.e. before to travel the paths). In other words, given a specific path $(v_0, \cdots, v_n)$, the objective will be estimate its performance before travelling. In the problem under study, this performance is measured in terms of the travel time and EV energy consumption. Mainly, the latter quantity for the case of EV is better explained for the variation of State of Charge (SOC) of the EV-battery because its interpretation is self-contained; it is defined as the percentage of available energy that can be delivered by the battery respect to the maximum capacity, thus battery parameters are considered in its definition. In consequence, for prior path performance evaluation, a characterization of the future evolution of travel time x_1 and SOC x_2 along the path are needed.

The future evolution of x_1 and x_2 are stochastic because it depends on the traffic state realizations. Therefore, the aim is to characterize their PDFs based on all available information at the current time. This goal is achieved by the prognostic technique presented in Section 2.4. Prognostic based on PF allows to compute predictions of linear and nonlinear systems, being considered the state-of-art tool for model-based prognostic [49]. In addition, this algorithm has been used for SOC prediction in Lithium Ion batteries with promising results in [46, 61]. It also offers a real-time execution, thus being well-suitable for a real-time decision making approach. Next, a proper model will be formulated to guarantee the structure where the Prognostic based on PF algorithm can be applied.

First, a dynamic is characterized; this is achieved by a discrete state-space formulation that is discretized by the nodes of the path i.e. $x_{\mathrm{i}}(k)$ denotes the state x_{i} when the EV is in node v_k.

In addition, the following auxiliary variables are introduced:

- $t(k) := tt(v_k, v_{k+1})$,
- $E(k) := E_T(v_k, v_{k+1})$,

where $tt(v_k, v_{k+1})$ is travel time of the edge (v_k, v_{k+1}) and $E_T(v_k, v_{k+1})$ is the EV-consumption energy at edge (v_k, v_{k+1}) (which can be found using model presented in section 3.2, therefore, elevation distance, velocity are being considered). Note that both $t(k)$ and $E(k)$ are random variables because they depend on the traffic realization (see section 3.1.2).

Now, the dynamic equation for the evolution of travel time along the path is:

$$x_1(k+1) = x_1(k) + t(k), \tag{4.4}$$

where $x_1(k)$ is the accumulated travel time to node v_k. Basically, it states that the accumulated travel time at node v_{k+1} is equal to accumulated travel time in v_k plus the travel time between v_k and v_{k+1}.

The second equation models the SOC evolution along the path as follows:

$$x_2(k+1) = x_2(k) - \frac{E(k)}{E_c} \tag{4.5}$$

where $x_2(k)$ is the SOC to node v_k and E_c is the maximum energy that can be delivered by the battery when it is fully charged. This model is based on [46], but energy consumption is calculated in a different manner for each step. While the energy consumption is modeled by electrical principles in [46], here it is modeled by mechanical principles.

Having a dynamic model for travel and SOC ((4.4) and (4.5), respectively), Prognostic based on PF (see section 2.4) is used for prognostic computation. In the implementation a key point is the characterization of future system inputs. In this case, the future inputs correspond to the possible future realizations of the state of traffic. In this work, the prognostic implementation assumes that the Discrete Markov Chain model that describes the future evolution of the state of traffic is known (i.e. the transition matrix M and the initial condition $\mathcal{T}(0)$ are known, see Subsection 3.1.2). Considering the above, each particle will explore future traffic realizations and the path performance will be evaluated.

Finally, an extra notation is introduced to be used later. Given a specific path $\mathcal{P}$, $x_1(EP)$ represents the cumulative travel time at the end of the path; meanwhile, $x_2(EP)$ represents the SOC at the end of the path. In consequence, given a path $\mathcal{P}$, the prognostic goal will be estimate the PDF of $x_1(EP)$ and $x_2(EP)$ in advance (before departure), proving valuable information which will be employed for decision making purposes.

4.3 Prognostic based decision making approach for finding the optimal path

As mentioned previously, the primary goal of this work is to solve the EV-DSSPP in real-time with a strategy based on a PDM approach. To guarantee that the strategy may be applied in real-time the decision maker algorithm is designed considering that its total execution time must be less or equal to T_{max}. In particular, in the present work T_{max} has been fixed equal to $23[s]$. Now, the proposal strategy for finding the optimal path from one starting-point to one ending-point will be presented.

The proposed PDM based routing strategy works as follows. In the first step a set of K candidates $\mathcal{C}$ are generated using Algorithm 2, hence t^H_{max} are spent. Afterwards, the prognostic is computed for a subset of n_c candidates, $\mathcal{C}_{n_c} \subseteq \mathcal{C}$. Here, n_c is defined considering the available computational time for prognostic computation $t^P_{max} = T_{max} - t^H_{max}$. Given the implemented code, its execution time of a path performance prognostic depends linearly of the number of edges of the path, and in the worst case, the computational time as function of number of path edges is $0.007 \cdot n_\mathrm{e}$ (relation found using a 1000 prognostic path simulations), where $n_\mathrm{e} = |P|$ is the number of edges of the path. Finally, using Prognostic based on PF, a PDF estimation is obtained for $x_1(EP)$ and $x_2(EP)$ for each path candidate in $\mathcal{C}_{n_c}$; basically, this step is an application of the method for path evaluation described above in Section 4.2. At this point, the expected value of $x_1(EP)$ and $x_2(EP)$ is taken, denoted as ($\mathbb{E}\{x_1(EP)\}$

and $\mathbb{E}\{x_2(EP)\}$) , respectively. In addition, given that $x_1(EP)$ and $x_2(EP)$ are in different scale, the following normalization is applied:

$$z_{\mathrm{i}}^{norm} = \frac{z_{\mathrm{i}} - z_{\mathrm{i}}^{min}}{z_{\mathrm{i}}^{max} - z_{\mathrm{i}}^{min}} \tag{4.6}$$

where the extreme values z_{i}^{min} and z_{i}^{max} are calculated from the expected value of $x_1(EP)$ and $x_2(EP)$ in path candidates in $\mathcal{C}_{n_c}$. As a result of this normalization, $\overline{\mathbb{E}\{x_1(EP)\}}$ and $\overline{\mathbb{E}\{x_2(EP)\}}$ are obtained; they are in the interval $[0, 1]$. Later, the following objective function is proposed:

$$J'(path) = \omega_1 \cdot \overline{\mathbb{E}\{x_1(EP)\}} - \omega_2 \cdot \overline{\mathbb{E}\{x_2(EP)\}} \tag{4.7}$$

where the parameters $\omega_1 \geq 0$ and $\omega_2 \geq 0$ assign the relative importance of the travel time and SOC in the path perform evaluation. These parameters are defined considering:

$$\omega_1 + \omega_2 = 1 \; \omega_1 \geq 0, \; \omega_2 \geq 0 \tag{4.8}$$

Finally, the optimization problem is the following:

$$path^* = \arg\min \left\{ (J(path_{\mathrm{i}})) : \; \mathrm{i} = 1, \ldots, n_c \right\}, \tag{4.9}$$

which is solved trivially in the set of candidates. Note that the minus sign in (4.7) appears due to the optimization objective of maximizing the SOC. Depending on the values of ω_1 and ω_2 different optimization objectives can be searched. If $\omega_1 = 0$, the only objective will be to maximize the SOC, on the other hand if $\omega_2 = 0$ the only objective will be to minimize the travel time. Otherwise, an equilibrium between low travel time and high SOC will be found.

The procedure for normalization and path selection is illustrated by an example. Let be $n_c = 6$, meaning that the prognostic is computed for 6 path candidates. Then, the expected values for $x_1(EP)$ and $x_2(EP)$ is calculated (see Table 4.1 columns 2-3); afterwards, these values are normalized according to (4.6) (see Table 4.1 columns 4-5). Finally, the best path can be selected under three different objectives. In the case of minimum TT ($\omega_2 = 0$) the selected path is candidate 1; in the case of maximum SOC ($\omega_1 = 0$) the selected path is candidate 5; and finally, in the case of an equilibrium between minimum TT and maximum SOC ($\omega_1 = \omega_2 = 0.5$) the selected path is candidate 3.

#Path	raw		normalize		Comments
	TT (s)	SOC (%)	TT	SOC	
1	**1672.3**	**0.6146**	**0**	**1**	**Min-TT**
2	1716.9	0.6534	0.3508	0.4458	*
3	**1744.2**	**0.6816**	**0.5659**	**0.0428**	**Equilibrium**
4	1779.6	0.6500	0.8447	0.4945	*
5	**1794.6**	**0.6846**	**0.9629**	**0**	**Max-SOC**
6	1799.3	0.6430	**1**	0.5937	*

Table 4.1: Example of normalization and path selection procedure.

This procedure for search and selection of path candidates is always conducted when the expected shortest travel time t_{sh} between the current node and the destination is greater than $180[s]$. Otherwise directly the expected shortest travel time path is selected as the best route. The underlying idea is that when the expected shortest path has a small travel time, the candidate search does not make sense because the set of possibilities is small. Moreover, if the diversity is imposed, the paths explored will be too long.

Algorithm 3 present a pseudo-code that summarizes the proposal PDM approach. Note that it begins once $\mathcal{FC}$ has been calculated. Line 1 is clock initiation; Line 2 check the shortest path $t_{sh} \leq 180[s]$ in the case of true run the PDM; otherwise, it returns the path candidate number one P_1. In the case of PDM, the loop for lines $3-6$ is executed by calculating the prognostic for each candidate, until all have been calculated or until the time is higher than the limit time t^P_{max}. The line 6 calculates the time that the algorithm will accumulate if the next path prognostic is computed. In the case of PDM, Line 7 calculates the optimal path according to (4.9).

Algorithm 3 *Optimal Path*

Input : Path candidates $\mathcal{FC} = \{\mathcal{P}_1, ..., \mathcal{P}_6\}$, Available computational time t^P_{max}, Number of particles N_p, Expected TT of the shortest path t_{sh}
Output: Optimal Path from s to d ,$\mathcal{P}^*_{s\to \mathrm{d}}$

Initialization
1: $t \leftarrow 0$ ▷ Execution clock initiation
2: **if** $t_P \leq 180$ **then**
3: **while** $t \leq t^P_{max}$ **do**
4: $k \leftarrow k + 1$
5: $\mathcal{X}_k \leftarrow$ prognostic of $\mathcal{P}+_k$ with N_p particles
6: $t \leftarrow t + 0.007 \cdot |P_{k+1}|$ ▷ add extra time to next path prognostic
7: $\mathcal{P}^*_{s\to \mathrm{d}} \leftarrow$ Optimization$(\mathcal{X})$ ▷ using (4.9)
8: **return** $\mathcal{P}^*_{s\to \mathrm{d}}$
9: **else**
10: **return** $\mathcal{P}^*_{s\to \mathrm{d}} \leftarrow$ Expected Shortest Path

The proposed strategy for finding the optimal path to go from one starting point to one destination that has been just presented will be called PDM-Shortest Path Algorithm in this

work. This algorithm is graphically summarized in Figure 4.4. It shows sequentially each step with the corresponding computing time. Additionally, it depicts the building blocks needed at each step with the corresponding section number that appear in the present work. Remark that given that each step of this proposed routing strategy was designed considering predefined fixed maximum computation times, so the complete execution of this strategy has at most a fixed and bounded computation time $T_{max} = t^H_{max} + t^P_{max}$.

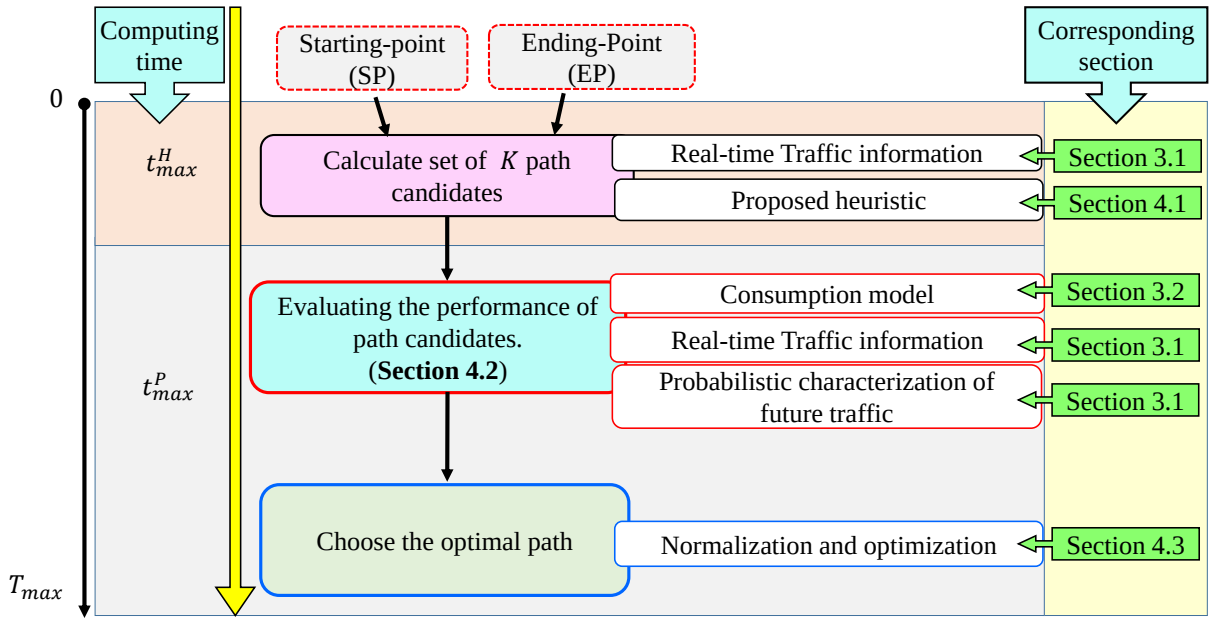


Figure 4.4: PDM-Shortest Path Algorithm. This figure shows sequentially a execution of the proposed strategy for finding the optimal path to go from one starting point to one destination.

4.4 Computing en-route path updates

PDM-Shortest Path Algorithm was presented in the previous section. This algorithm works finding the optimal path from one starting point to one ending-point. Now, this algorithm will be employed to compute en-route path updates during a travel. The main motivation to re-compute the optimal path en-route is that the availability of new traffic information because it may enable to find a better path. For instance, let suppose a driver who is driving an EV along of a certain path, if he or she knows in advance the occurrence of a non-recurring event in the current path, the driver can take an alternative path in order to avoid the non-recurring congestion caused by this event. Note that this work supposes that there is a system that periodically provides information about the current traffic states. The time period between two consecutive traffic information updates is assumed as fixed an equal to t_u (see Section 3.1). The proposed strategy for computing en-route path updates while completing a route is summarized in Figure 4.5. This strategy will be explained in the next paragraphs.

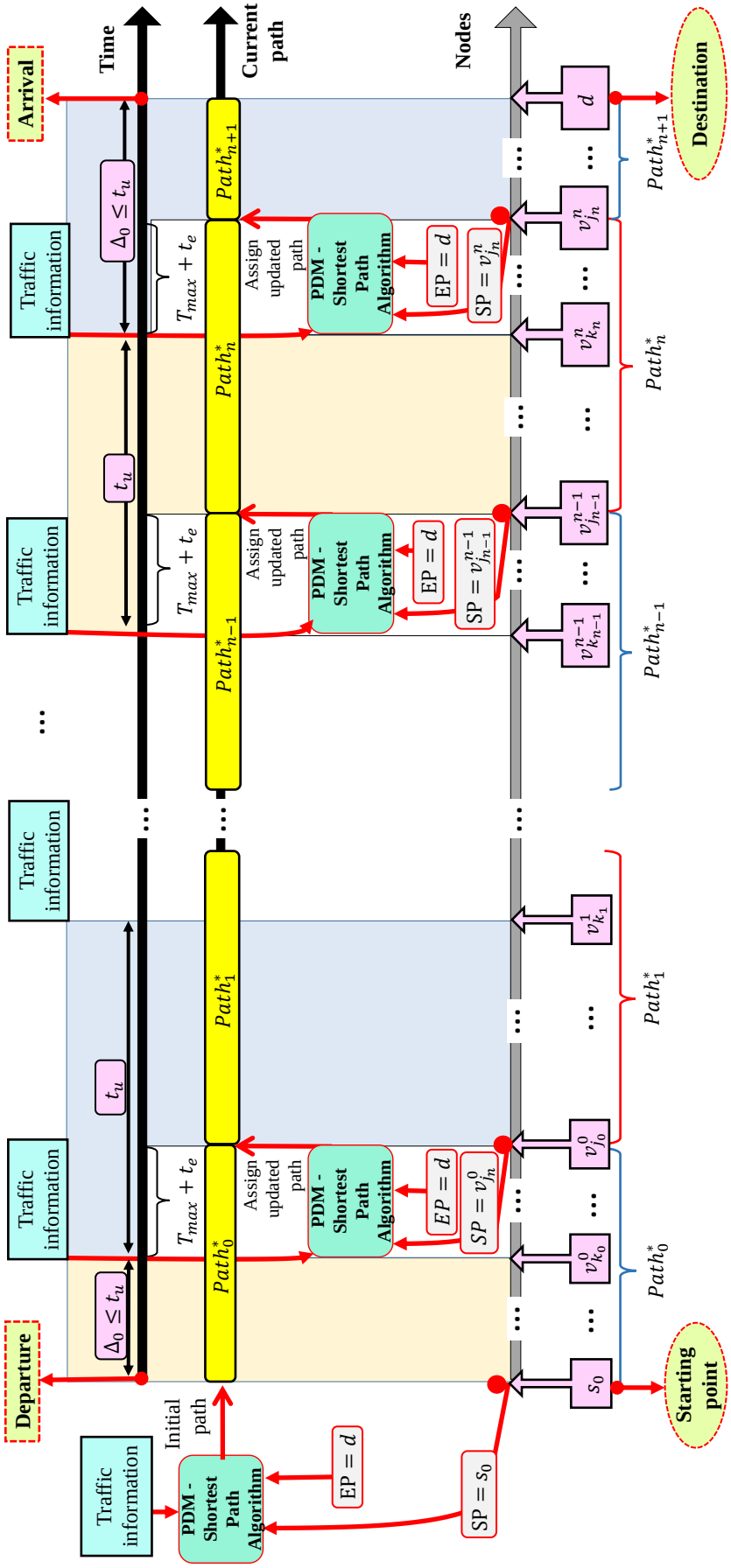


Figure 4.5: Computing en-route path updates for one travel.

The proposed strategy for computing en-route path updates (see Figure 4.5) works applying sequentially the PDM-Shortest Path Algorithm. More specifically, every time that new traffic information is available, the current path is updated using the PDM-Shortest Path Algorithm. In this execution it is key to note that the starting-point(SP) should be an ahead node (a node of the current path that has not yet been visited by the EV) in the current path, which guarantees that the travel time from the current position to this node be long enough to achieve the complete execution of this routing strategy before arriving to the starting point. Therefore, this travel time has to be at least T_{max}, which is the upper bound for execution time of the PDM-Shortest Path Algorithm. However, if due to variance the EV arrives to the starting-point before the computations of the PDM-Shortest Path Algorithm are finished, the EV will stop in SP to wait while the updated path is being computed. In this work the optimal path obtained in the $n-$th update is denoted as follows:

$$path_n^* = (v_0^n, v_1^n, ..., v_{f_n}^n, d) \tag{4.10}$$

where v_0^n is the starting-point and d is the ending-point. Note that $f_n + 1$ is the number of nodes of $path_n^*$.

Figure 4.5 shows the operation of the proposed strategy for computing en-route path updates during complete travel from starting point to end point. Firstly, note that at the beginning of this travel the starting point (SP) is s_0 and the ending-point (EP) is d. Before departure, PDM-Shortest Path Algorithm is executed for finding the optimal path from s_0 to d using the available traffic information at that time. As a result, $path_0^* = (v_0^0, v_1^0, ..., v_{f_0}^0, d)$ is obtained. This path is followed until new traffic information is available, in which case the path is updated as follows. Let us assume that we are following the route $path_n^*$, and we want to find the $n+1$-th update $path_{n+1}^*$. An ahead node $v_{j_n}^n$ in the current path $path_n^*$ (i.e. $v_{j_n}^n$ is a node of the current path that has not yet been visited by the EV when the new traffic information arrives) is selected as starting point and PDM-Shortest Path Algorithm is again executed to obtain the updated path $path_{n+1}^*$, which will be followed once the EV has arrived to node $v_{j_n}^n$. As mentioned previously, the idea is that PDM-Shortest Path Algorithm is executed while the EV is traveling from the current position until $v_{j_n}^n$, so when $v_{j_n}^n$ is reached, the updated path will be ready to follow it. Consequently, $v_{j_n}^n$ is chosen such that it is the first ahead node in $path_n^*$ that will be reached with a travel time (from the current position along of $path_n^*$) large enough to guarantee the complete execution of proposed routing strategy. That is with a travel time greater or equal than $T_{max} + t_e$, where t_e is an extra time that is added as a safety margin to ensure that the updated path will be ready at node $v_{j_n}^n$. The method of selection of the ahead node $v_{j_n}^n$ will explained in the next paragraph. The strategy for path update just described is repeated every time that new traffic information arrived until the destination point is reached; this procedure is illustrated in Figure 4.5.

The ahead node $v_{j_n}^n$ in the current path $path_n^*$ is chosen by a procedure based on iterative execution of travel time prognostic, which is shown in Algorithm 4. This algorithm iteratively estimates the predicted travel time PDF to go from the current node $v_{k_n}^n$ to an ahead node $v_{k_n+q}^n$, $q = 1, 2, ...$ (in the current path $path_n^*$) in order to determine which ahead node $v_{j_n}^n$, $j_n > k_n$, achieves the travel time requirement for a complete execution of the proposed routing strategy. More specifically, this algorithm works as follows. It is initialized from the current node $v_{k_n}^n$ with an initial population of particles equals to zero (Line 3), which

represent the predicted PDF of the cumulative travel times starting from $v^n_{k_n}$. Then, the loop starts (Line 4). In each iteration, the travel time prognostic is computed for each ahead node $v^n_{k_n+q},\ q = 1, 2, ...$ (in increasing order), resulting a set of particles $\left\{x^i(q)\right\}_{i=1}^{N_p}$ (Line 7). Afterwards, the fastest particle of $\left\{x^i(q)\right\}_{i=1}^{N_p}$ is calculated, that is the particle with the minimum cumulative travel time to reach to node v^n_{k+q} (Line 8). This procedure is repeated until the particle with the minimum cumulative travel time be greater or equal than $T_{max}+t_{\mathrm{e}}$, in which case the corresponding node is returned as $v^n_{j_n}$, or until the ending point be reached, in which case the ending point is returned as $v^n_{j_n}$ (Line 4). Note that the current position $v^n_{k_n}$ is defined as the current node, if the EV is exactly at one node; otherwise, the current position $v^n_{k_n}$ is defined as the nearest ahead node in the current path $path^*_n$. In this work $t_{\mathrm{e}} = 2[s]$.

Algorithm 4 *Ahead node*

Input : Current path $path^*_n = (v^n_0, ..., v^n_{f_n}, d)$
Output: Ahead node $v^n_{j_n}$

Initialization
1: $\Upsilon \leftarrow 0$
2: $q \leftarrow 0$
3: $\{x^i(0)\}_{i=1}^{N_p} \leftarrow \{0, ..., 0\}$
4: **while** $\Upsilon \geq T_{max} + t_{\mathrm{e}}$ or $v = d$ **do**
5: $\quad u \leftarrow v^n_{k_n+q}$
6: $\quad v \leftarrow v^n_{k_n+q+1}$
7: $\quad \left\{x^i(q)\right\}_{i=1}^{N_p} \leftarrow$ Prognostic travel time from u to v starting with $\{x^i(q-1)\}_{i=1}^{N_p}$
8: $\quad \Upsilon \leftarrow \min_{i=1,...,N_p} x^{\mathrm{i}}(q)$
9: $\quad q \leftarrow q + 1$
10: $v^n_{j_n} \leftarrow v$
11: **return** $v^n_{j_n}$

At this point, it is relevant to discuss the importance of the computation times in the work setting of Figure 4.5. In this setting, every time that the new traffic information is available, the current path must be updated quickly to avoid possible near congestion. In this work the traffic information is updated every $t_u = 120[s]$ and the computation time of the proposed routing strategy is $T_{MAX} = 23[s]$. Note that T_{MAX} should be small compared with t_u to avoid near congestion; otherwise, the EV will continue in the same path for a long time, even tough new traffic information is available.

4.5 Observations related with the proposed PDM based strategy to solve the EV-DSSPP.

First of all, note that PDM-Shortest Path Algorithm was designed considering a bounded computing time denoted as T_{max}. In consequence, the optimal path is obtained in a time

less or equal to T_{max}, which allows to quickly update the current path incorporating the new available traffic information. This fact is a goal that contrasts with the reviewed work related to DSSPP in Section 2.2.2, where most works addressed the DSSPP by MDP formulation and solved it by a dynamic programming strategy, because their computation times for an execution could not be set. Even more, some works that reported their computing time [4, 14] evidenced dimensionality problems for large networks, which complicates real-time implementation.

Additionally, in the present work the EV consumption model is detailed step by step with its corresponding parameters. The core of this model (dynamic equation obtained by mechanical principles and parameters) was based on a real-word consumption model presented in [59], where the procedure for finding the model parameters is presented. The above contrasts with the reviewed works in Section 2.2.2 because they were mainly focused on designing of strategies to solve the DSSPP, whereas the consumption was not studied in depth.

Chapter 5

Simulation Analysis

The proposed PDM approach for the EV-DSSPP has been studied by a simulation. The simulations were performed on a desktop computer, with an Intel Xeon 3.5$[GHz]$ processor and 32$[GB]$ of RAM. The parameters employed in the simulations are summarized in Table 5.1.

Parameter	Unit	Value
$SOC_{initial}$		0.9
ξ		$6 \cdot 10^{-4}$
N_p		100
T_{max}	$[s]$	23
t^H_{max}	$[s]$	3
t^P_{max}	$[s]$	20
Δ		0.7
n_{max}		4
τ		0.03
K		6

Table 5.1: Simulation parameters.

The simulation considers one travel starting at the node n_s and finishing at node n_{d} (see geographic location in Figure 5.1a). The distance between both nodes was approximately 12$[km]$, and he travel started at 7 AM. The simulations were performed considering 3 different optimization objectives (see (4.9)); thus, the proposed strategy is executed with 3 configurations as summarized in Table 5.2.

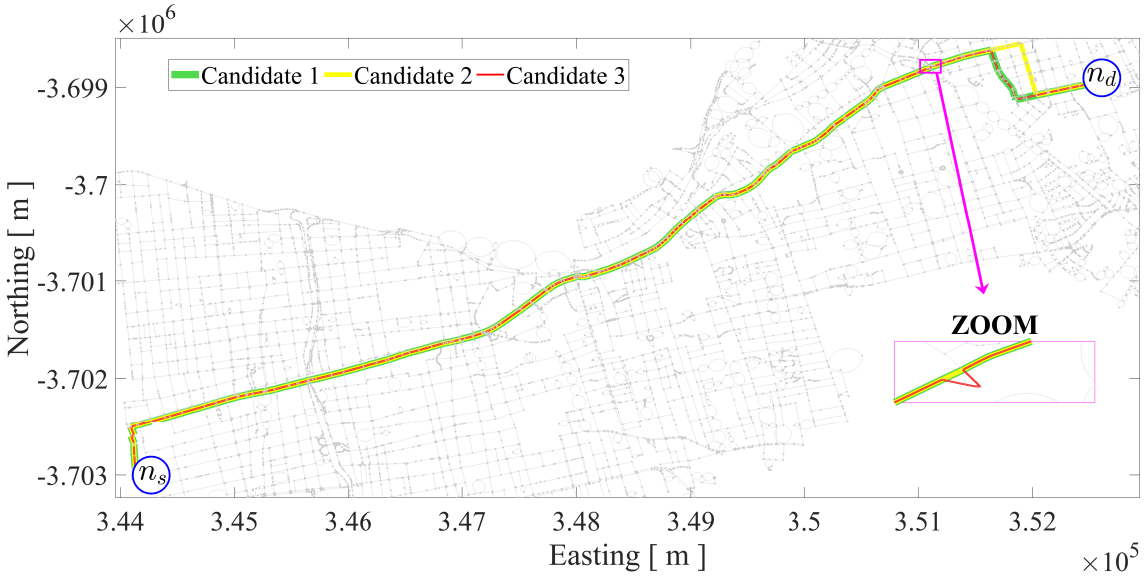
Configurations	Parameters of objective function	Objective	Notation
Config. 1	$\omega_1 = 1 \quad \omega_2 = 0$	Minimize travel time	Min TT
Config. 2	$\omega_1 = 0 \quad \omega_2 = 1$	Maximize SOC	Max SOC
Config. 3	$\omega_1 = 0.5 \quad \omega_2 = 0.5$	Compromise between: Min TT and Max SOC	Compromise

Table 5.2: Configurations.

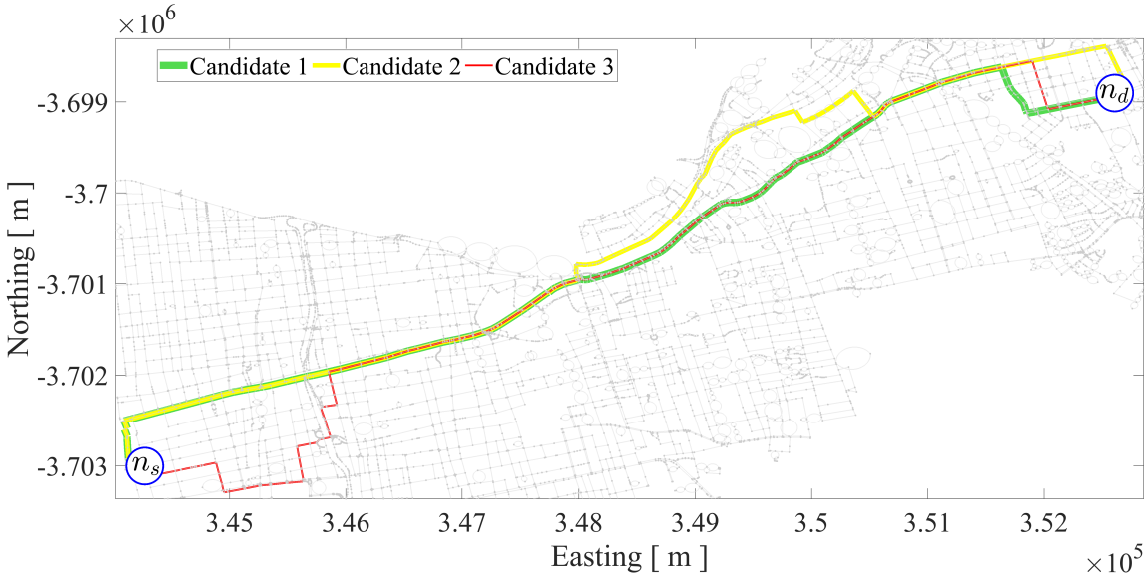
Before studying each case in depth, the proposed heuristic for selection of path candidates is analyzed means of an example. For this purpose, a set of three initial path candidates obtained using the proposed heuristic and the K-SPP were computed, they are shown in Figure 5.1. The first observation is that in the case of K-SPP (see Figure 5.1a), only two path candidates are easily visible; the third is imperceptible because it is highly similar to others, and only it can be seen by zoom. On the other hand, the three candidates obtained by the proposed heuristic (see Figure 5.1b) are clearly distinguished. Moreover, they explore different zones since they must respect the similarity restriction.

Considering the example has just reviewed, the proposed heuristic for selection of path candidates offers two advantages with respect to k-SPP. The first, is its lower computational time (as discussed in Section 4.1). The second is the ability to explore different zones. The latter is a remarkable advantage for two reasons:

- In the proposed PDM approach, the candidates are searched in a static way (because the current expected travel time is considered for this purpose). Then, each candidate is evaluated by Prognostic based on PF; at this point, the dynamic and stochasticity are incorporated into the decision making process. Considering the above, the optimum obtained statically (in the selection of candidates) is not necessarily the optimum obtained after Prognostic based on PF evaluation. Therefore, the diversity in the path candidates is essential because it enables to explore different zones; thus a better path can be obtained with the stochastic and dynamic evaluation.
- The selection of path candidates is made considering only the time requirement. Thus, the energy consumption requirement solely is taken into account in the path evaluation by prognostic algorithm. Considering the above, the diversity in the path candidates is important because it enables to explore a wide area, which may enable to find better solutions in terms of energy consumption.



(a) K-Shortest path algorithm



(b) Proposed heuristic

Figure 5.1: Comparison between proposed heuristic and K-Shortest path algorithm.

For each of the configurations under study (see Table 5.2), two methods to solve the EV-DSSPP by the proposed PDM approach were implemented:

- **Without update en-route:** This method calculates the optimal path using PDM-Shortest Path Algorithm only at the beginning of the path (before departure).
- **With update en-route:** This method updates works computing en-route path updates, considering the current traffic state. This method was presented previously in Figure 4.5.

These two methods allow the testing of initial hypotheses, which are two: (1) the real-time EV-DSSPP can be solved by a PDM approach incorporating models of consumption model, road model, historical and real-time traffic information (recurrent and non-recurrent traffic congestion); and (2) updating the paths en-route offers higher performance in terms of travel time and energy consumption than solely using the optimal solution computed before departure. The methods were tested by simulations with the same initial condition in terms of SOC and state of traffic. The output variables recorded from each simulation are summarized in Table 5.3.

Notation	Unit	Description
P_u		path with update strategy
P_{non-u}		path without update strategy
TT_u	$[s]$	TT with update strategy
TT_{non-u}	$[s]$	TT without update strategy
SOC_u		SOC with update strategy
SOC_{non-u}.		SOC without update strategy

Table 5.3: Output variables from simulations.

Given that P_{non-u} is calculated solely considering the initial traffic conditions, during its trajectory, it is possible to find traffic incidents that affect the path performance. consequently, the simulations were grouped considering the percentage of edges with typical traffic in the P_{non-u} that were found during the trajectory. This grouping allows the analysis of the performance of both strategies as a function of the Percentage of Edges with Typical Traffic (PETT) in P_{non-u}.

It can be observed the results of the simulations of **Config. 1** are shown in Figure 5.2. Firstly, to observe that, independently of the PETT in P_{non-u}, for most of the realizations, the percentage of saved travel time (using update en-route) is greater than zero. Therefore, even without traffic incidents in P_{non-u}, the path obtained with update en-route P_u offers better performance in terms of travel time. In addition, when there are a greater number of traffic incidents in P_{non-u}, the percentage of saved travel time using en-route update method is higher. These results validate the fact that the incorporation of real-time traffic information can enhance the performance of routing system because it allows to know in advance where there is an incident, and thus to make decisions to avoid the congestion.

Recall that most of the works reviewed in Section 2.2.4 led to the same conclusion, in the sense of en-route updating allows the reduction of travel times; even when using different methodologies. In particular, in the present work by a PDM approach, the same conclusion can be made, validating the fact that the proposed approach allows to incorporate in a proper way the real-time traffic information.

On the other hand, there are cases where the en-route updating has a worse performance in terms of travel time (see Figure 5.2). These occur because sometimes the algorithm makes a decision considering that the new path will be faster, but a non-recurring event occurs in the new path and the travel time is higher. In summary, en-route updating creates better performance in terms of travel time in most realizations, but not all.

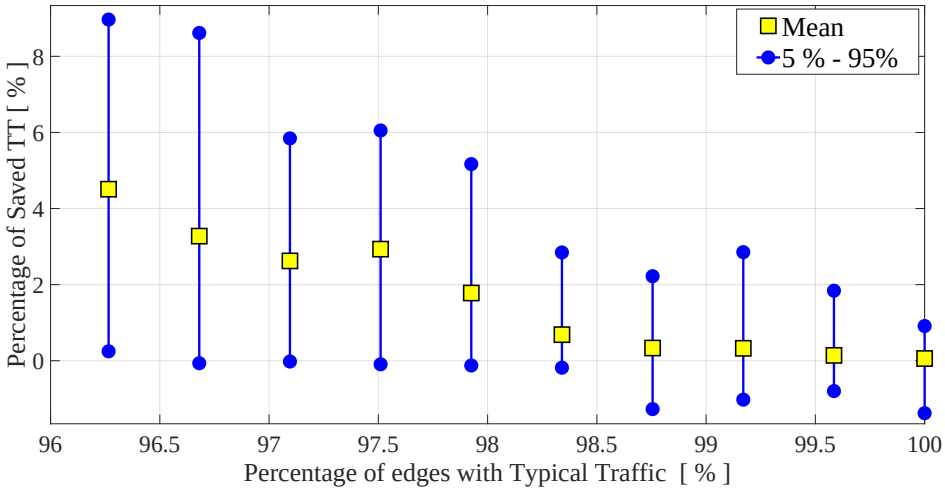


Figure 5.2: Percentage of Saved travel time (TT) using update en-route ($\frac{TT_{non-u}-TT_u}{TT_{non-u}} \cdot 100\%$).

The result of the simulations for **Config. 2** are shown in Figure 5.3. As in **Config. 1**, independently of the PETT in P_{non-u}, the percentage of saved SOC is greater than 0 for most of realizations, with an average of 4.15%. Unlike **Config. 1**, however, it is observed that the percentage of saved SOC does not depend significantly on traffic incidents. This fact is explained for the EV-ECM employed, which shows a less energy consumption for low velocities; this fact will be discussed later.

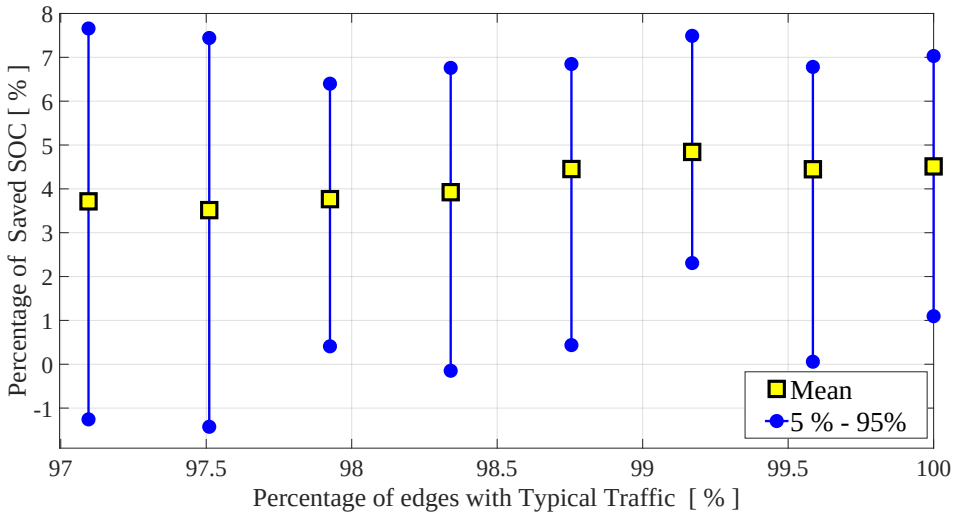


Figure 5.3: Percentage of Saved SOC using update en-route ($\frac{SOC_u-SOC_{non-u}}{SOC_{non-u}} \cdot 100\%$).

The results of the three experiment with the 3 different configurations are presented in Figure 5.4. The main observation is that there is a trade-off between the TT and SOC (with regard to mean values), in the sense that the fastest path has the highest energy consumption (lowest SOC) and vice versa. This behavior remains in evidence in the **Config. 3**, given

that, in this case, an equilibrium between SOC and TT is searched. The results show that the solution result for SOC and TT are in the middle of **Config. 1** and **Config. 2**.

In terms of percentage, on average, **Config. 1** offers a travel time 2.3% less than **Config. 2**. Nevertheless, to achieve this goal, it spends a 4.5% more energy than **Config. 2**. This compromise between travel time and EV energy consumption has been anticipated in the Chapter 1; in [13], the authors highlight the fact that for EVs the fastest path is not necessarily optimal in terms of energy consumption

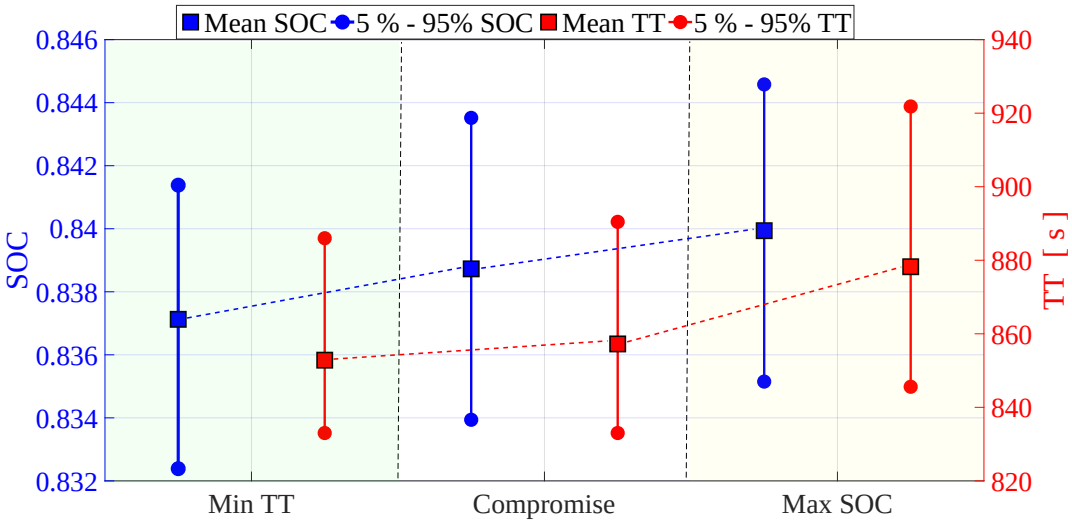


Figure 5.4: Summarize of the 3 configurations.

It is important to note that the results were obtained considering $\xi = 6 \cdot 10^{-4}$. This parameter defines the traffic incident rate. Therefore, if ξ is higher, the occurrence of lower PETT will be more frequent; on the other hand, for ξ lower, the occurrence of no incident will be more frequent. Nevertheless, the proposed algorithm has better performances in both cases; thus the analysis made here will continue being valid.

Another interesting point to highlight is the following. Given that the proposed routing strategy uses a prognostic algorithm as a tool to solve the EV-DSSPP, the same prognostic implementation can be used but for different goal. Let suppose that a driver who is interested to estimate the travel time and energy consumption of a certain path before departure, in such case the same prognostic implementation can be employed to predict the required quantities. This goal is key to avoid the range-anxiety [11].

To summarize of the results, the following points are remarked,

- The strategy that uses en-route updates offers better performance in terms of travel time and SOC than the other strategy that only calculates the best path at the beginning. Therefore, the proposed solution allows the proper incorporation of the real-time traffic information.
- There is a trade-off (with regard to mean values) between the travel time and energy consumption.

Chapter 6

Conclusions

Having completed the present work, it can be concluded that all of the initial objectives were reached. In the following paragraphs, each one will be analyzed.

First, a model for stochastic traffic simulation was designed and implemented. It incorporated both types of congestion: recurrent and non-recurrent. The recurrent congestion was modeled using a database with average travel times. Meanwhile, the non-recurring congestion was modeled using a discrete Markov Chain model. The use of the Markov Chain model allowed the incorporation of different types of traffic incidents (crashes, hazards and stationary vehicles) as discrete states of the chain. Moreover, each type of incident was well-defined in terms of its duration and impact on the edge travel time. The main advantage of this model is that factors such as location or time dependence can easily be incorporated through a variant transition matrix. In particular, in the present work, this matrix (M, as defined in Section 3.1.2) was considered to be variant depending on the road type. Nevertheless, it was considered the same matrix for all roads of the same type. In general, this assumption might not be real, due to the existence of road segments that have higher incident probability than others. Furthermore, the incident probability changes during the day because there are hours at which there are higher number of incidents than at other hours. Therefore, all of these variants may be incorporated in future works, with the aim of generating a more realistic simulation model.

An approximated model was introduced for the EV-ECM, in which factors such as inclination, velocity, acceleration, and regenerative braking were considered. The proposed model considered a fixed velocity in an edge, but corrected the consumption according to a pre-calculated factor of correction (denoted as α, see Section 3.2). The main advantage of this proposed EV-ECM is that it allows calculating faster consumption simulations, without a significant loss of precision. Note that the correction factor was calculated using different driving cycles (obtained from [60]) for each type of road; the underlying idea is to model the possible velocity profile that the EV might experience on a determined type of road. Nevertheless, for practical applications, they should be obtained from the EV under study, more specifically, from the EV driver behavior because the driving cycle on a determined road will depend on the driver. In addition, the behavior may be time-variant. All of these factors can be addressed in future works.

Additionally, a novel PDM approach for solving in real-time the EV-DSSPP was presented. It incorporates a detailed EV-ECM and traffic model. It makes the EV routing decision to consider not only the current state of traffic, but also the future traffic evolution through a prognostic algorithm. Its real-time execution is guaranteed by the design of the algorithm because each algorithm step is executed considering a predefined fixed maximum computation time. The proposed routing algorithm enabled the incorporation of real-time information in the decision making process; and thus, the en-route updates. Along these lines, two methods were tested: those with updates and those without updates en-route. The results showed that en-route update method offers better performance in terms of travel time and energy consumption; this validated the hypothesis number 2 and corroborated the reviewed works in Chapter 1.

Moreover, the results showed a trade-off (with regard to mean values) between the travel time and energy consumption because, generally, the fastest path has the highest energy consumption. Therefore, at the time of the path decision, it is essential that the driver clearly indicates his or her priority between travel time and/or energy consumption, since this election will define the resulting optimal path.

For the proposed PDM approach, the selection of path candidates was computed statically and deterministically, considering the current expected travel time of each edge. This procedure does not guarantee that the candidates have good performance in the dynamic evaluation, that is in the prognostic step. Therefore, in future works, methods that incorporate expected traffic dynamics will be explored. On the other hand, given that the decision is based on a model-based prognostic method that uses particles to propagate the future uncertainty, probabilistic and physic restrictions can be added for instance, limiting the probability of reaching specific nodes with a determined SOC or power restriction or power restriction. These ideas also will be studied in future works.

Bibliography

[1] A. Chakrabartty and S. Gupta, "Estimation of congestion cost in the city of kolkata—a case study", *Current Urban Studies*, vol. 3, no. 02, p. 95, 2015.

[2] Y. S. Chang, Y. J. Lee, and S. S. B. Choi, "Is there more traffic congestion in larger cities?-scaling analysis of the 101 largest us urban centers", *Transport Policy*, vol. 59, pp. 54–63, 2017.

[3] D. Sever, N. Dellaert, T. Van Woensel, and T. De Kok, "Dynamic shortest path problems: Hybrid routing policies considering network disruptions", *Computers & Operations Research*, vol. 40, no. 12, pp. 2852–2863, 2013.

[4] D. Sever, L. Zhao, N. Dellaert, E. Demir, T. Van Woensel, and T. De Kok, "The dynamic shortest path problem with time-dependent stochastic disruptions", *Transportation Research Part C: Emerging Technologies*, vol. 92, pp. 42–57, 2018.

[5] S. Gao and H. Huang, "Real-time traveler information for optimal adaptive routing in stochastic time-dependent networks", *Transportation Research Part C: Emerging Technologies*, vol. 21, no. 1, pp. 196–213, 2012.

[6] L. C. Casals, E. Martinez-Laserna, B. A. Garcia, and N. Nieto, "Sustainability analysis of the electric vehicle use in europe for co2 emissions reduction", *Journal of cleaner production*, vol. 127, pp. 425–437, 2016.

[7] M. Longo, N. M. Lutz, L. Daniel, D. Zaninelli, and M. Pruckner, "Towards an impact study of electric vehicles on the italian electric power system using simulation techniques", in *Research and Technologies for Society and Industry (RTSI), 2017 IEEE 3rd International Forum on*, IEEE, 2017, pp. 1–5.

[8] R. A. Daziano, "Conditional-logit bayes estimators for consumer valuation of electric vehicle driving range", *Resource and Energy Economics*, vol. 35, no. 3, pp. 429–450, 2013.

[9] A. Adepetu and S. Keshav, "The relative importance of price and driving range on electric vehicle adoption: Los angeles case study", *Transportation*, vol. 44, no. 2, pp. 353–373, 2017.

[10] L. Noel, G. Z. de Rubens, B. K. Sovacool, and J. Kester, "Fear and loathing of electric vehicles: The reactionary rhetoric of range anxiety", *Energy Research & Social Science*, vol. 48, pp. 96–107, 2019.

[11] G. De Nunzio and L. Thibault, "Energy-optimal driving range prediction for electric vehicles", in *Intelligent Vehicles Symposium (IV), 2017 IEEE*, IEEE, 2017, pp. 1608–1613.

[12] I. Tsiropoulos, D. Tarvydas, and N. Lebedeva, *Li-ion batteries for mobility and stationary storage applications Scenarios for costs and market growth*. Publications Office of the European Union, 2018.

[13] Y. Wang, J. Jiang, and T. Mu, "Context-aware and energy-driven route optimization for fully electric vehicles via crowdsourcing", *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1331–1345, 2013.

[14] E. Jafari and S. D. Boyles, "Online charging and routing of electric vehicles in stochastic time-varying networks", *Transportation Research Record: Journal of the Transportation Research Board*, no. 2667, pp. 61–70, 2017.

[15] N. Deo, *Graph Theory with Applications to Engineering and Computer Science*, ser. Prentice-Hall series in automatic computation. Prentice-Hall, 0. [Online]. Available: `http://gen.lib.rus.ec/book/index.php?md5=1ddcfb042e75592a9e682fcaab72ae08`.

[16] J. L. Gross and J. Yellen, *Graph theory and its applications*. Chapman and Hall/CRC, 2005.

[17] P. Z. Jonathan L. Gross Jay Yellen, *Handbook of Graph Theory, Second Edition*, 2nd ed., ser. Discrete Mathematics and Its Applications. Chapman and Hall/CRC, 2013.

[18] W. D. Wallis, *A beginner's guide to graph theory*. Springer Science & Business Media, 2010.

[19] L. Taccari, "Integer programming formulations for the elementary shortest path problem", *European Journal of Operational Research*, vol. 252, no. 1, pp. 122–130, 2016.

[20] E. W. Dijkstra, "A note on two problems in connexion with graphs", *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[21] T. Stoilov and K. Stoilova, "Routing algorithms in computers networks", in *International Conference on Computer Systems and Technologies*, 2005.

[22] E. Nasrabadi and S. M. Hashemi, "On solving dynamic shortest path problems", in *Proceedings of 20th International Conference/Euro Mini Conference on Continuous Optimization and Knowledge-Based Technologies (EurOPT 2008)*, 2008, pp. 48–53.

[23] K. Sung, M. G. Bell, M. Seong, and S. Park, "Shortest paths in a network with time-dependent flow speeds", *European Journal of Operational Research*, vol. 121, no. 1, pp. 32–39, 2000.

[24] R. K. Ahuja, J. B. Orlin, S. Pallottino, and M. G. Scutella, "Dynamic shortest paths minimizing travel times and costs", *Networks: An International Journal*, vol. 41, no. 4, pp. 197–205, 2003.

[25] B. Ding, J. X. Yu, and L. Qin, "Finding time-dependent shortest paths over large graphs", in *Proceedings of the 11th international conference on Extending database technology: Advances in database technology*, ACM, 2008, pp. 205–216.

[26] U. Demiryurek, F. Banaei-Kashani, and C. Shahabi, "A case for time-dependent shortest path computation in spatial networks", in *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, 2010, pp. 474–477.

[27] L. Wang, L. Yang, and Z. Gao, "The constrained shortest path problem with stochastic correlated link travel times", *European Journal of Operational Research*, vol. 255, no. 1, pp. 43–57, 2016.

[28] X. Ji, "Models and algorithm for stochastic shortest path problem", *Applied Mathematics and Computation*, vol. 170, no. 1, pp. 503–514, 2005.

[29] E. Nikolova, J. A. Kelner, M. Brand, and M. Mitzenmacher, "Stochastic shortest paths via quasi-convex maximization", in *European Symposium on Algorithms*, Springer, 2006, pp. 552–563.

[30] A. T. Hojati, L. Ferreira, S. Washington, P. Charles, and A. Shobeirinejad, "Reprint of: Modelling the impact of traffic incidents on travel time reliability", *Transportation research part C: emerging technologies*, vol. 70, pp. 86–97, 2016.

[31] S. Kim, M. E. Lewis, and C. C. White, "Optimal vehicle routing with real-time traffic information", *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 2, pp. 178–188, 2005.

[32] S. Kim, M. E. Lewis, and C. C. Whites, "State space reduction for nonstationary stochastic shortest path problems with real-time traffic information", *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 3, pp. 273–284, 2005.

[33] B. W. Thomas and C. C. White III, "The dynamic shortest path problem with anticipation", *European Journal of Operational Research*, vol. 176, no. 2, pp. 836–854, 2007.

[34] A. R. Güner, A. Murat, and R. B. Chinnam, "Dynamic routing under recurrent and non-recurrent congestion using real-time its information", *Computers & Operations Research*, vol. 39, no. 2, pp. 358–373, 2012.

[35] M. Rajabi-Bahaabadi, A. Shariat-Mohaymany, M. Babaei, and C. W. Ahn, "Multi-objective path finding in stochastic time-dependent road networks using non-dominated sorting genetic algorithm", *Expert Systems with Applications*, vol. 42, no. 12, pp. 5056–5064, 2015.

[36] U. Ritzinger, J. Puchinger, and R. F. Hartl, "A survey on dynamic and stochastic vehicle routing problems", *International Journal of Production Research*, vol. 54, no. 1, pp. 215–231, 2016.

[37] M. M. de Weerdt, S. Stein, E. H. Gerding, V. Robu, and N. R. Jennings, "Intention-aware routing of electric vehicles", *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 5, pp. 1472–1482, 2016.

[38] T. M. Sweda, I. S. Dolinskaya, and D. Klabjan, "Adaptive routing and recharging policies for electric vehicles", *Transportation Science*, vol. 51, no. 4, pp. 1326–1348, 2017.

[39] Z. Yi and P. H. Bauer, "Optimal stochastic eco-routing solutions for electric vehicles", *IEEE Transactions on Intelligent Transportation Systems*, no. 99, pp. 1–11, 2018.

[40] J. Y. Yen, "Finding the k shortest loopless paths in a network", *management Science*, vol. 17, no. 11, pp. 712–716, 1971.

[41] H. Liu, C. Jin, B. Yang, and A. Zhou, "Finding top-k shortest paths with diversity", *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 3, pp. 488–502, 2018.

[42] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking", *IEEE Transactions on signal processing*, vol. 50, no. 2, pp. 174–188, 2002.

[43] J. V. Candy, *Bayesian signal processing: classical, modern, and particle filtering methods*. John Wiley & Sons, 2016, vol. 54.

[44] M. E. Orchard and G. J. Vachtsevanos, "A particle-filtering approach for on-line fault diagnosis and failure prognosis", *Transactions of the Institute of Measurement and Control*, vol. 31, no. 3-4, pp. 221–246, 2009.

[45] C. Musso, N. Oudjane, and F. Le Gland, "Sequential monte carlo methods in practice", pp. 247–260, 2001.

[46] D. A. Pola, H. F. Navarrete, M. E. Orchard, R. S. Rabie, M. A. Cerda, B. E. Olivares, J. F. Silva, P. A. Espinoza, and A. Perez, "Particle-filtering-based discharge time prog-

nosis for lithium-ion batteries with a statistical characterization of use profiles", *IEEE Transactions on Reliability*, vol. 64, no. 2, pp. 710–720, 2015.

[47] M. Mishra, J. Odelius, A. Thaduri, A. Nissen, and M. Rantatalo, "Particle filter-based prognostic approach for railway track geometry", *Mechanical Systems and Signal Processing*, vol. 96, pp. 226–238, 2017.

[48] W. Caesarendra, G. Niu, and B.-S. Yang, "Machine condition prognosis based on sequential monte carlo method", *Expert Systems with Applications*, vol. 37, no. 3, pp. 2412–2420, 2010.

[49] M. Jouin, R. Gouriveau, D. Hissel, M.-C. Péra, and N. Zerhouni, "Particle filter-based prognostics: Review, discussion and perspectives", *Mechanical Systems and Signal Processing*, vol. 72, pp. 2–31, 2016.

[50] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using networkx", in *Proceedings of the 7th Python in Science Conference*, G. Varoquaux, T. Vaught, and J. Millman, Eds., Pasadena, CA USA, 2008, pp. 11–15.

[51] J. C. Falcocchio and H. S. Levinson, *Road traffic congestion: a concise guide*. Springer, 2015, vol. 7.

[52] A. Tavassoli Hojati, "Modelling the impact of traffic incidents on travel time reliability", PhD thesis, The University of Queensland, 2014.

[53] D. Smith, S. Djahel, and J. Murphy, "A sumo based evaluation of road incidents' impact on traffic congestion level in smart cities", in *Local Computer Networks Workshops (LCN Workshops), 2014 IEEE 39th Conference on*, IEEE, 2014, pp. 702–710.

[54] H. Xiong, A. Vahedian, X. Zhou, Y. Li, and J. Luo, "Predicting traffic congestion propagation patterns: A propagation graph approach", in *Proceedings of the 11th ACM SIGSPATIAL International Workshop on Computational Transportation Science*, ACM, 2018, pp. 60–69.

[55] H. Nguyen, W. Liu, and F. Chen, "Discovering congestion propagation patterns in spatio-temporal traffic data", *IEEE Transactions on Big Data*, vol. 3, no. 2, pp. 169–180, 2017.

[56] Y. Liang, Z. Jiang, and Y. Zheng, "Inferring traffic cascading patterns", in *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, 2017, p. 2.

[57] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition", *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[58] J. Krumm and E. Horvitz, "Risk-aware planning: Methods and case study for safer driving routes.", in *AAAI*, 2017, pp. 4708–4714.

[59] C. Fiori, K. Ahn, and H. A. Rakha, "Power-based electric vehicle energy consumption model: Model development and validation", *Applied Energy*, vol. 168, pp. 257–268, 2016.

[60] U. S. E. P. Agency, *dynamometer drive schedules*, `https://www.epa.gov/vehicle-and-fuel-emissions-testing/dynamometer-drive-schedules`, Accessed: 2010-09-30.

[61] P. A Espinoza, A. Perez, M. Orchard, H. Navarrete, and D. Pola, "A simulation engine for predicting state-of-charge and state-of-health in lithium-ion battery packs of electric vehicles", in *PHM Society European Conference*, vol. 8, Oct. 2017.