Universiteit Utrecht

ProRail

An Agent-based Approach to Simulating Train Driver Behaviour

Master Thesis

MSc. Technical Artificial Intelligence
Information and Computing Sciences

Supervisors:
Dr. Mehdi Dastani, Utrecht University
Prof. Dr. John-Jules Meyer, Utrecht University
Ir. Jelle van Luipen, ProRail

by

Wilco Tielman
3507718

January 2015

Abstract

Railway simulations have been a useful tool within the railway industry. One of the simulators that ProRail, the Dutch infrastructure manager for the railways, uses is the micro-railway simulator FRISO. ProRail wants to find out if the validity of this simulator could be improved through adding agent based train drivers. In this thesis the development of these agents will be described. Different data sources about train driver behaviour were available and could be used to create a train driver model that could be implemented within the designed agents. Using this, an agent DLL was written in C++ to work together with FRISO. Simulations were then done in order to find out if the validity had improved with the added agents. Through comparing the resulting driving times with the previous train driver implementation and realisation data, it was concluded that the agents scored better. When looking at the driving behaviour of the agents, it was concluded that this lied closer to the realisation data then that of the FRISO train drivers. It was also noted that certain aspects of train driver behaviour were not modelled correctly by the agents and/or FRISO which resulted in deviations seen in driving times and driving behaviour. The presence of these aspects indicated that a sufficiently accurate model of train driver behaviour is required if reliable simulation results are desired.

Acknowledgements

Before everything else I would like to express my gratitude and appreciation for the help and support I have received during the formation of this thesis.

I would first like to thank Jelle, for getting me involved in the 'behind the scenes' of different ongoing projects, giving me more insight into the research that is done at ProRail and the workings of the train-network, train driving, and other involved aspects. I also want to thank him for pointing me in the direction of people who could help me further with different parts of my project.

I would like to thank Mehdi, for encouraging me to think about every detail and about the foundations of my reasoning during our talks. Also, my thanks goes out to Audry and Richard for sharing their experience and insights with me, for giving me a better insight into the thoughts, actions and aspects that go into driving trains. I would like to thank Emdzad, Douwe, Alexander and Jannet for showing interest in my project and sharing their thoughts and knowledge with me. I would like to thank everyone else in the department of Innovation at ProRail for making me feel welcome and involved.

Finally, I would like to thank Myrthe for proof reading this thesis and Charlotte for her support.

Contents

1 Introduction

1.1 Background

ProRail is the Dutch infrastructure manager for the railways, whose tasks include researching and improving the time tables of trains. In order to accomplish this task multiple simulators are used to look at the effects of different traffic and infrastructure measures. One of these simulators is called FRISO, a micro-railway simulator, wherein amongst other things the driving behaviour of train drivers is simulated. ProRail wanted to find out if the validity of this simulator could be improved through adding an agent based model of train driver behaviour. Agent based simulation has been noted as a powerful simulation modelling technique [4], providing a parallel between real drivers and agents [66], and has been applied to many traffic simulation systems [26, 12, 46, 86]. The project that is described in this thesis was done in order to answer this question about a possible improvement in validity. The validity in this case denotes the predictive value of the simulations done within FRISO. ProRail is further interested in a more valid train driver implementation in order to do future research into the effects of adaptations to security measures and/or train operation developments. In this thesis the process of developing and implementing this train driver agent will be lined out, from modelling to experimentation.

1.2 Problem

In this project a train driver agent was to be developed in order to find out the effect of this implementation on the predictive value of the simulations done within FRISO. The research problem that can be formulated from this goal was:

- How can you add train driver behaviour to a micro-level simulator (FRISO), using an Agent based approach?

This question could be divided into the following sub-questions:

- How can you model train driver behaviour from data?
- How can you implement train driver behaviour within an Agent?
- How can you implement agents into a micro-level simulator (FRISO)?

After this, the implementation could be used to look into the resulting train driver behaviour and the predictive value of the simulations. In order to answer these questions, different methodologies were used, as presented within the next section.

1.3 Methodology

In order to get a better insight into the different aspects involved with answering the posed questions, a literature study will be done involving: simulation, modelling of human operators, agents and machine learning. A closer look will also be taken into the different aspects involved when driving a train through looking at relevant railway literature and interviews with two train drivers. The first sub-question will be answered through processing and filtering the available data, after which the resulting information will be used to create a decision making model of a train driver with the help of machine learning methods. Using this, the second sub-question will be answered through designing an agent setup that can incorporate this decision making model and work together with FRISO. Following this, the agent setup will be implemented within FRISO. Finally, this implementation will be used to perform a number of experiments aimed at looking into the resulting train driver behaviour and the predictive value of the simulations.

1.4 Outline

This thesis will be structured as follows: First, some background information will be given about the processes involved with the driving of a train. This will be followed by a brief overview of the research that has been done in the following relevant subjects: simulation, modelling human operators, agents and machine learning. The third chapter will focus on the describing the processing that was done on the available data about train driving, the observations that were made and will present the model that was created with the help of machine learning methods. The next chapter will go into presenting the agent design. Chapter five will go over the implementation of this agent model within FRISO. In Chapter six the experiments and their results will be presented. Finally, the conclusion of the findings and suggestions for future research will be presented in Chapter seven.

2 Literature review & background information

In this chapter I will first give some background information about train driving, in order to make the reader more comfortable with concepts and terms used later in this thesis. After this I will look into several research fields that will play an important role within this thesis, namely railway simulation, the modelling of human operators, software agents, machine learning and statistical methods used. Within these fields I will focus on aspects and research done that is of relevance to modelling and implementing a train driver software agent.

2.1 An overview of Train driving

In this section I will outline some important concepts that have to do with driving a train, to give a clearer view of these concepts that will come back in the rest of this thesis. Overall, driving a train can be described as a driving task where vigilance and concentration are of high importance. It is a visually guided task with auditory cues to help the train driver. I will elaborate a bit on the main aspects and concepts used in the task of driving a train that influence or limit the driving behaviour. I will skip over a lot of rules and procedures that are of little relevance to this study, more information can be found in [76].

2.1.1 Signalling

Signalling is the term used to describe the main way that instructions and information about the permitted driving limits are conveyed. This signalling is done mainly through signals such as coloured light signals, and fixed signs.

Signals are placed between sections of track and usually come in three aspects, showing either a green, yellow, or red colour. Each of these express one

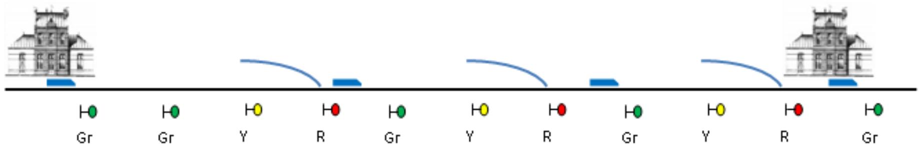


Figure 2.1: *A schematic overview of the signalling system between two stations. The blue blocks indicate the positions of the trains. The blue curves indicate the braking curve to a stop before the red signals. The signals below indicate the signal relations, where red means that there is a train within the next section of tracks. Yellow indicates that the next signal will be red and green indicates that the train can drive according to the speed limit.*

main concept. Green signifies that the train driver can safely pass that signal. A yellow signal indicates that the train driver should limit the speed of the train in the section after the yellow signal and prepare to stop. A red signal indicates that the train driver needs to stop in front of that signal. The order in which signal colours are determined is done through signal relations. A simple example would be that a yellow signal always comes in front of a red signal, as seen in Figure 2.1.

The signal relations are specified between sets of signals, this means that when a train driver sees a certain signal aspect, he can often know which signal aspect to expect afterwards.

A higher level of meaning to the signals, besides the speed restrictions, is that they generally indicate which track in front of the train is free to ride upon or is occupied/not reserved, and subsequently how close it is to the end of the trains route. In certain situations the signals are linked to the speed limits of track sections themselves, to enforce that the train does not go into that section too fast. A common example of this is a section of tracks with a number of switches. This reserving of blocks of track is either done automatically through a system called Automatische Rijweginstelling/Automatic route setting (ARI), or through the actions of train traffic controllers. ARI sets the routes of trains based on a schedule at certain triggers. Due to the set way ARI works, some freight train drivers use this to influence the system in order to minimize the amount of times a train needs to brake for a signal. Train traffic controllers are mainly responsible for dealing with situations that are beyond ARI's reach, such as communicating with the train driver when the train has

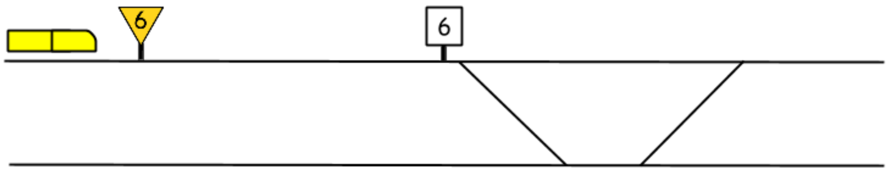


Figure 2.2: *A schematic overview of a limiting speed sign indicating that the train needs to slow down to 60km/h. After that a maximum speed sign can be seen, indicating that the 60km/h speed limit has to be reached by that point. This due to the switches that are positioned behind.*

stopped for an unplanned red signal, or changing the routes in unexpected situations. An unplanned red signal denotes a red signal that the train driver encounters that is not positioned at a planned stop.

Signs are used to convey either the maximum local speed, called snelheids borden/speed signs, or to give information about the track. One example of this is that the train is approaching a section where it needs to lower its pantograph.

Speed signs come in three variations. The first one signifies that the train driver is allowed to speed up after passing it. The second one signifies that the train driver needs to limit its speed, and the final ones indicates that the previously instructed speed limit is supposed to be reached here, see Figure 2.2 for an example. This means that unlike speed signs for cars, the designated speed is not to be crossed at the next speed sign, not at the initial one. This is due to the fact that trains take a lot more time to accelerate or slow down than cars.

2.1.2 Speed limits & ATP

The speed limits are indicated with a single digit which, times ten, indicates the speed. An example of this is an accelerating sign with 14, which means the train is allowed to accelerate to 140km/h. For a complete overview of railway signs see: [77].

The current local speed limit is dictated by the combination of signs and signals. With the lower one taking precedence. One example of this can be seen in Figure 2.3, where a train driver is approaching a yellow 8 signal, with next to it a slowdown sign of 6. The driver needs to limit the trains speed to 60km/h, but he also needs to make sure the train has reached 80km/h before the next signal and 60km/h before the next maximum speed sign of 6. This interplay of speed limits means that the train driver needs to estimate when it will have reached a certain speed and at which point.

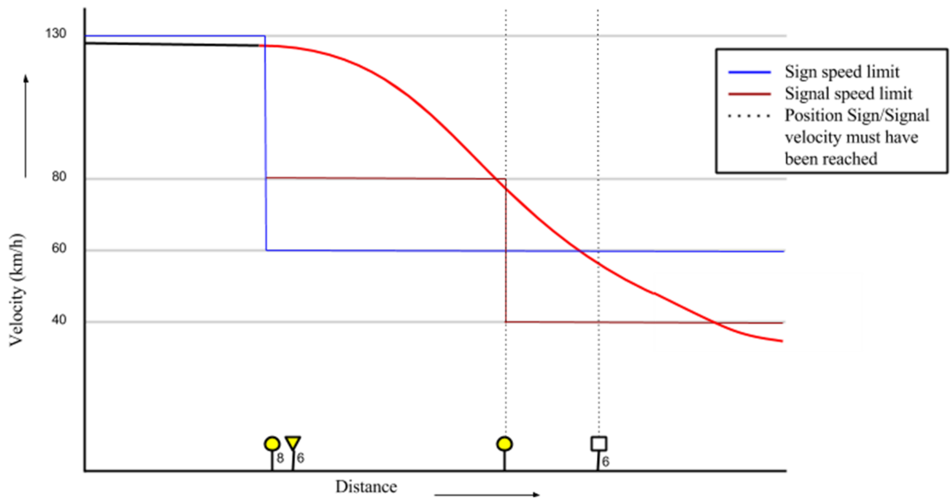


Figure 2.3: *An example of the relationship between the speed limits imposed by signs and signals. On the x axes the distance the train has travelled is indicated. On the y axes the velocity is indicated. The black and red curve represents the velocity of the train. After the train passes the yellow 8 signal with the speed sign 6 next to it, the train needs to limit its velocity to 80km/h before passing the next yellow signal. It also needs to ensure that the 60km/h speed limit has been reached before passing the maximum speed sign. The points at which these velocities have to be reached are represented with vertical dotted lines, for respectively the 80km/h and 60km/h speed limits.*

It should be noted that a train driver does not necessarily need to maintain the speed limit on the free-track sections in order to arrive on time. The free-track sections here denote the usually lengthy sections of tracks between two stops where no switches are present. The largest Dutch railway operator within the Netherlands, the Nederlandse Spoorwegen (NS), has a set of guidelines for train drivers in order to drive in an timely and energy efficient manner. This advised method is called *Universal Economical driving Idea* (UEI), which is a simple rule set that advises the train driver, based on the planned driving time and the local speed limit, at which point in time or at which velocity the train driver is to start coasting. Coasting denotes the act of giving neither traction nor applying the brakes.

Automatic Train Protection (ATP), is the main safety system train drivers need to deal with when driving the train. The ATP dictates the maximum local speeds, described above, with the following five ordinal values: 40, 60, 80, 130 and 140. The ATP indicates on the dashboard of the train what the current local speed is and reinforces this. It must be noted that this enforcement in actuality lies a bit higher than the indicated speed with approximately 5km/h, to deal with the little inaccuracies that come with determining the actual speed of the train.

Given that the ATP only uses five speed limit values, it means that certain speeds are not specifically reinforced by the ATP system, like 100 and 120km/h, at these points the ATP simply indicates the next value, in these cases 130. If the train driver exceeds the ATP indicated speed, there will be an auditory cue to make the train driver aware he needs to slow down, if the train driver does not slow down the ATP system forces the train to a complete stop with an automated braking system. The train driver can prevent this through braking with the minimal required braking amount called the braking criteria, which is dependent on the rolling stock. The *rolling stock* denotes the composition of the train, such as the kind of locomotive. An example of the ATP in use can be describes as follows: A train passes a yellow signal from a section where it was allowed to go 60km/h, within 2 seconds after the passing of the yellow signal the maximum speed indication on the ATP will switch to 40, after which it will give an auditory cue of this. If the train driver does not start braking with the braking criteria within 2 seconds the train will automatically be forced to a complete stop.

Combining these systems results in an environment where the train driver is forced by rules and safety systems to act within certain bounds. This means that certain parts of the behaviour of a train driver are very reactionary and dictated by distinct perceptual cues.

2.2 Literature review

In this section of this chapter I will look into some of the research subjects which are related to this thesis and give a brief overview of relevant work. For all of the subjects that will follow, I must note that this is not a complete overview of each field, but instead aims to give the reader sufficient background information on the relevant topics and the current state of the field in question. I will start this chapter by looking into railway simulation and FRISO, after which I will give a brief overview of modelling human operators. I will follow that by looking into agents and finish this chapter by giving some background information of the different machine learning and statistical methods that will be used.

2.2.1 Railway Simulation & FRISO

Simulation has been a valuable tool over the past 30 years for giving more insight into real world systems, like railway systems [34]. One of the main reasons for this is that simulations can give more insight into a phenomena due to the possible abstractions on different levels, which can make it easier for policy makers to intuitively understand a complex problem. Another main benefit from simulations is that they can help you see the effects of policies which are difficult to predict and costly to implement in real life [66].

Kreuger et al [51] identified different key situations in which simulations can be used to help decision-making within the railway industry. These are:

- *Asset based simulation*, for managing railway assets like locomotives and infrastructure.
- *Train operation simulations*, for insight into dynamic train operations, like braking distances.
- *Line and terminal capacity simulations*, for analysing capacity and performance of the railway network, which can help with finding bottlenecks.
- *Traffic/service simulations*, for modelling entire railway networks and operations, to help with testing 'what-if' scenarios, like with the testing of time tables.
- *Rail/non-rail interface simulations*, for getting insight into non-rail activities such as terminal design.

Because of the benefits simulations can provide within these areas, a number of railway simulators have been developed.

An early example of a railway simulator is STRESI [75]. STRESI is a microscopic simulator, which in the case of railway simulation is class of simulator that simulates the railway and signalling system down to the level of the individual trains. This to give an accurate representation of the dynamics of the train and signalling systems working in concert. STRESI together with OpenTrack [64] and RailSys [69], which are also microscopic simulators, are examples of simulators used to look at the effects of initial delays on track occupation and the resulting delays on other trains [16].

SIMONE [61] is a macroscopic simulator, which is a class of simulator that does not use the signalling system, block length, and other infrastructure related procedures. Instead a macroscopic simulator models aspects of the system in a more general way, using concepts such as average driving times and the density of traffic on sections of railway in order to research aspects of large areas of railway.

ProRail has its own set of simulators which it uses for a range of purposes. One of the most notable usages being the testing of time tables for robustness. FRISO is one of these simulators. It is a discrete event based microscopic simulator, which is developed under Enterprise Dynamics, with the purpose of analysing traffic and infrastructure measures [60, 18].

Within the field of traffic simulation, there has been a rising trend in the use of Agent Based Simulation, see [57, 12] for an overview and [28, 26, 14, 66, 1] for some examples. A main reason for this, as mentioned by [18] and [12], is that the agent computing paradigm is suitable for the development of large scale dynamic distributed systems. This suitability fits well with railway simulation, which usually encompasses a large geographical scale with dynamic distributed systems, like trains and signalling aspects.

2.2.2 Modelling human operators

The modelling of human operators is a field of study that has been approached from many different directions with many different goals in mind [55], from modelling combat pilots for military purposes [47], to help with gaining a better understanding of the workload of a driver [38], to controlling a model car [81].

One approach within the field of modelling human operators is a more cognitive and descriptive approach. Focussing more on the underlying processes that go on in the operators mind, from sensory input to arousal states and motor control. One example purpose of such models is human performance modelling, as seen in Man-Machine Integration Design and Analysis System (MIDAS) [14]. The usefulness of human performance models comes from their potential to reveal system vulnerabilities and where human-system errors can occur [35]. This approach has also been successfully used within the railway domain [38]. See [55, 14, 84] for an overview and some examples.

For a more mathematical approach, [24] lines out some different possible approaches. These approaches are more functional and are often aimed at creating a model in order to predict the actions of the operator. One example of this is using a rule based fuzzy logic system to model human operators [24, 89, 33]. Once a valid model is obtained that can predict actions of a human operator, you can use it for different purposes, such as designing a warning system to prevent dangerous situations, to test the effect of new implementations to see the effects on the operator, and many more.

When talking about driver models, [58] argues that *"We are heading for an intelligent, knowledge and rule based model of the driver that will be capable of dealing with a wide variety of realistic, complex situations."* He also argues for a hierarchical cognitive control structure, for which he states that *"The generalized problem solving task of the driver-qua road user-may be further divided in three levels of skills and control: strategical (planning), tactical (maneuvering), and operational (control) respectively"*.

Within the context of this thesis, decision making is an important part of modelling a human operator, such as a train driver. Decision making is also a field with a wide scope on its own, ranging from the descriptive to the functional. I will not go into descriptive methods, which are more aimed at understanding how people make decisions, seeing that this is not in-line with the topic of this thesis, but for an overview and examples see [53, 2].

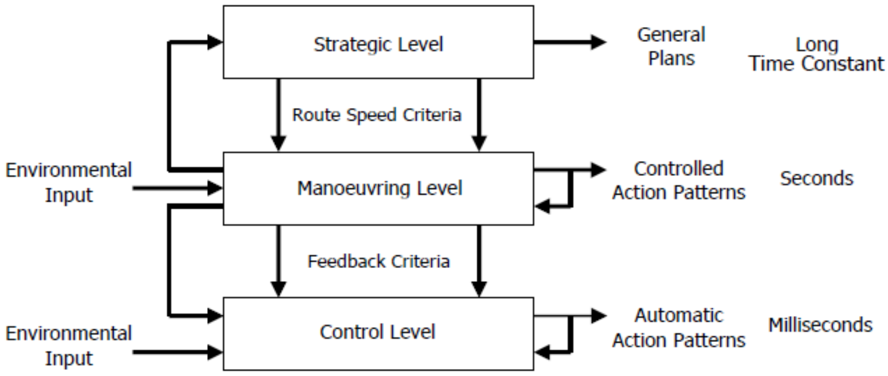


Figure 2.4: *A hierarchical structure of a driver [58].*

As mentioned by [19], decision making models are sometimes used to:

1. Help the decision makers
2. Represent the decision maker
3. Replace the decision maker

Decision making methods meant as guidelines or aides like [30, 13, 3], that are aimed at people who need to make decisions fit within the first category. Another common example is the act of making a list with pros and cons. Decision making theories and models like [20, 10], that aim to be capable of making decisions like the decision makers fit within the second and third category. One example of this within the railway industry can be found in [38], where they model a train driver in order to predict the workload, performance time and errors under different conditions.

2.2.3 Agents

There has been a wide use of the term Agent within literature, with no definitive single definition [29]. Commonly the term is used to describe an autonomous entity within an environment it can execute actions in, often noting interactivity, goal driven, reactive and planning as defining elements. Within this thesis I will use the definition of [73], which states that an agent is *"anything that can be viewed as perceiving its environment through sensors and* *acting upon that environment through actuators".*

Within the field of agent based modelling (ABM), the belief-desire-intention (BDI) [32, 7, 71] architecture has been widely used. The BDI paradigm states a reasoning approach for rational agents. Rational meaning that the agent does actions to further its goals based on what information it has about the environment. Within this architecture, beliefs are an internal representation of information about the environment the agent finds itself in. Desires are states that the agent wants to bring about, and intentions are actions that are parts of a plan. Using these concepts of belief, desire and intention agents perform actions, based on plans, to further their goals. Examples of implementations can be found in [49, 42]. For the development and design of these BDI agents, specific methodologies have been developed, one notable example being [49]. With this methodology the authors aim to give a framework for analysing and building complex multi-agent systems, based on the BDI paradigm.

Within the field of distributed artificial intelligence (DAI), Multi agent Systems (MAS) is the sub-discipline where multiple autonomous agents interact with each other within an environment [80]. [46] notes that MAS can be defined as a loosely coupled network of problem solvers that work together to solve problems that are beyond the capabilities of the individuals. The crucial aspect being here the interaction between the agents, which brings with it aspects like communication, norms and roles [5]. [80] notes that this interaction can happen also with 'non-communicating' agents, through sensing and reasoning about the actions of other agents. Advantages for MAS have been mentioned by numerous authors. [80] notes scalability, parallelism and modularity to be among those. [46] refers to MAS as offering powerful representational tools, techniques and metaphors for the way people conceptualize and implement many types of software. For an overview of the field of ABM and MAS, and its issues see [46, 88, 6, 78].

Within the field of simulation, agent based simulation has been noted by [4] to be a powerful simulation modelling technique, with benefits ranging from cost effective [26], to providing a clear parallel between real drivers and agent drivers [66]. [4] mentions four areas of application for ABM within simulation, namely flow simulation, organization simulation, market simulation and diffusion simulation. Traffic simulation falls under flow simulation.

Another benefit of ABM within the field of traffic simulation is that it captures emergent phenomena [4, 26]. For example [66], where they look at the traffic produced by agents as an emergent behaviour and show the viability of a multi-agent based simulation of unorganized traffic. This viability is also noted by [46], when they look at the application of agent based systems on transportation systems. For a review of applications and examples within traffic simulation see [12, 86].

Within the field of MAS, there has been the recent development of practical multi-agent programming languages [5]. These languages can be categorized as being a declarative agent-oriented language, an imperative agent-oriented language or an hybrid approach.

- *Declarative* meaning that there is a strict focus on logic and being formal in nature. One example being DALI [15].
- *Imperative* agent-oriented languages have more in common with, and are often built upon, normal imperative languages like Java. One example being JACK [41].
- *Hybrid* approaches define themselves by combining the possibility to use an declarative approach using logic, while also giving the possibility for programming imperatively.

One example of a hybrid approach is 2APL (A Practical Agent Programming Language) [17], which is a BDI agent-oriented programming language that aims at providing programming constructs to facilitate the implementation of agent concepts and abstractions. 2APL uses a declarative approach for representing and reasoning about an agent's beliefs and goals, while using an imperative approach for the creation of plans and for the agent's interface to the environment. For an overview and examples of MAP languages see [5, 40, 17]. [5] notes that there is still much work to be done within the field of multi-agent programming languages and name a number of major challenges, dealing with:

- Debugging tools
- Integration of agent tools into existing IDEs
- The separation of MAS platforms from agent platforms
- The dissemination of the MAS programming paradigm

2.2.4 Machine learning & Statistical methods

Machine learning can be defined as the study of computational methods for the discovery and learning of patterns and other regularities from data [62]. [9] notes that within the field of machine learning, you can distinguish three primary research foci:

1. Task-Oriented studies: For the development and study of learning systems aimed at improving their performance within a set of tasks.

2. Cognitive simulation: For simulation and exploration into human learning processes.

3. Theoretical analysis: For the study of learning methods and algorithms independent of domain. Like [21]

Besides distinguishing between different research foci, you can also distinguish different approaches within the field of machine learning, namely a *symbolic* approach or a *statistical* approach [31]. Symbolic approaches use learning techniques more in line with the learning of symbolic descriptions, such as rules and trees, while statistical approaches apply techniques that lie closer to statistics, such as support vector machines and Bayesian classifiers. Within the field of machine learning, it is also important to note the distinction between *supervised*, *unsupervised* and *reinforcement* learning. Supervised learning meaning that the data set can be seen as a set of examples, with the correct input and output available. Some general issues with supervised learning are that data processing, data preparation, feature selection and the selection of an appropriate algorithm play an important part [50]. Some of these issues are addressed within research reviews, see [50, 66]. Unsupervised learning means that there is no specific correct value available, so instead of linking the correct input and output, it tries to find and learn structures. Data clustering [45] is an example of this. Data clustering is the process of grouping together data points based on common characteristics.

Reinforcement learning refers to learning approaches where the algorithm gets a reinforcement from the environment after each action [48]. An example of this would be an agent learning how to perform a certain action through trial and error, adjusting its action in such a way as to maximize the utility of that action's result on the environment. One downside to reinforcement learning is that in most of the cases reinforcement learning methods do not

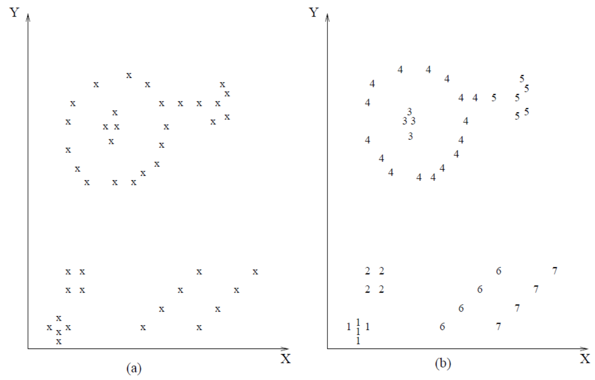


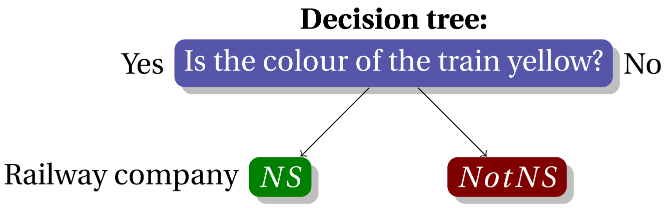
Figure 2.5: *A simple example of data clustering, where crosses within (a) are categorised within 7 different categories within (b) [45].*

scale well to larger problems.

One important part to note about learning knowledge, is the way this knowledge is represented. [9] notes a number of different types of representations of knowledge. Amongst these are parameters in algebraic expressions and decision trees.

One way to model and learn a decision tree is in the form of a classification tree. Classification trees can be seen as a tree that can classify instances based on feature values [50]. Advantages of this approach are that the resulting decision trees are easy to interpret [23] and are fast to classify input [74].

Classification trees van be learned/built through certain algorithms. One example of this is the c4.5 decision tree generator algorithm [68]. C4.5 is noted by [22] to be one of the most commonly used decision tree classifiers within the machine learning and data mining communities. A main disadvantage of classification trees is that they are easily over fitted. Over fitting means that the learned model, in this case a decision tree, fits the training data too much, which means that it loses its accuracy in classifying entries that do not come from the training set. Algorithms like c4.5 try to combat this weakness through methods like pruning the decision tree, in order to remove the branches that do not add enough distinction value for the classification purpose in question. For a more extensive overview of how decision tree classifiers work and some examples see [67, 74].



As mentioned earlier, using parameters in algebraic expressions is one way to represent knowledge within a system. One example of this can be found in [66], where they assign a value to a variable within the system around a mean value, which was observed from data, within the limits of a distribution. The distribution is thus used to predict the frequency of occurrence of certain parameter values. Some distributions have been designed specifically to facilitate this fitting [70].

Besides fitting distributions to data, fitting a curve to data is a method that can also be used for predictive purposes. One of the ways to do this is through using regression analysis. Regression analysis is a method to fit a function to a plot of points [72]. One possible application for this is that you can use the resulting function for predicting values. Within regression analysis, there are two models which are often used, namely *linear regression* and *non-linear regression*. As the name implies, linear regression fits the data points to a linear function. It does this through trying to minimize the distance of the points to the curve in question. Non-linear regression does not limit the fitting to a linear function, but can be done to any selected equation [63]. See [54, 36] for some examples.

It is noted by [80] that machine learning techniques are of much interest within the application of MAS, due to their inherent complexity. One example of this is [39], where the authors propose machine learning as a tool for the construction of agents. More specifically, they use machine learning methods to teach an agent different high level aircraft manoeuvres by example, to relieve the need of manually coding in these manoeuvres.

2.2.5 Conclusion

Within this chapter I have given an overview of the subjects simulation, human operator modelling, agents and machine learning, focusing on the aspects of each of those fields which are in close relation to the research question: How can you add train driver behaviour to a micro-level simulator (FRISO), using an Agent based approach. In the following chapters I will go through using these findings in different steps, starting with describing the data and formulating a train driver model with the help of it.

3 Empirical research

3.1 Introduction

In the previous chapter I have given a brief overview of, amongst others, classification trees and statistical fitting methods. Within this chapter I will use these methods to create an implementable model of the decision making process of a train driver. I will not attempt to create a general model of a train driver, as can be seen in [38] and Figure 3.1. Instead I will focus on using the data available to learn and fit a decision making model which focuses on a functional reasoning-act cycle. The main reason for not making a general model of a train driver that includes human capabilities, is due to the large amount of time this would require and the unavailability of relevant data. Like data that contains information about the way train drivers perceive, remember and focus on the different aspects that come in to play when driving a train. I will first give some background information about the data used, followed by a description of the data processing that was done. After this I will give an overview of the methodology used for further modelling a train drivers behaviour with the acquired information. I will close this chapter by formalizing these findings through presenting an overview of the decision making model.

3.2 Data

Data plays an important role within simulation [65], often being a crucial component of the input and output for simulations. In this section I will focus on the use of data for the creational purpose, to create a component of the simulation. In order to create a decision making model of train driving behaviour I mainly used two data sets for exploration, training and fitting purposes. One being data from the ProRail simulator MATRICS, the other being from GPS data gathered from trains. I will start with describing the two data sets and their aspects.

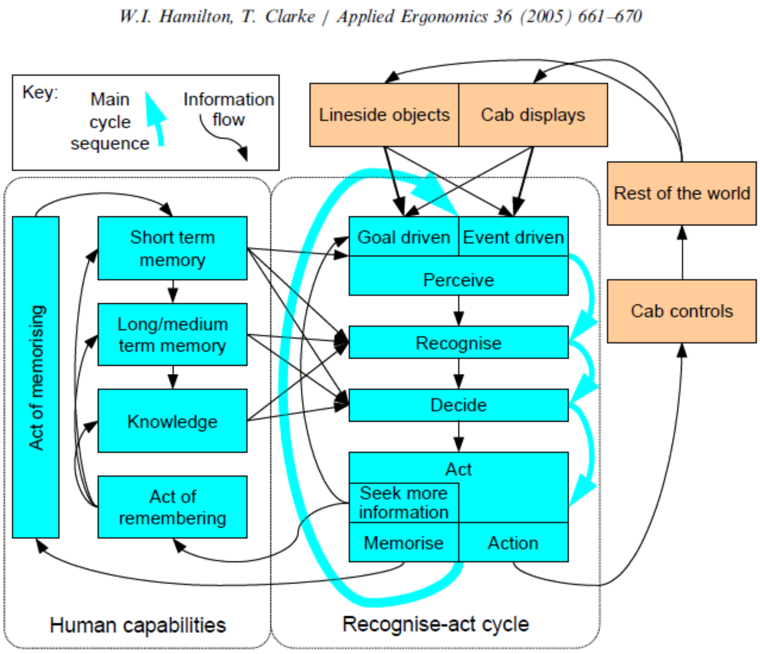


Figure 3.1: *A schematic overview of the Human capabilities and the recognise-act cycle in the CTA model.* [38]

3.2.1 MATRICS data

MATRICS is a railway simulator used by ProRail for research purposes. Within this simulator, the operator can be put into the role of a train driver, where they need to drive the train within the simulation environment and follow the time table. The data used within this thesis from MATRICS comes from two different projects, in both cases the trains within the simulator were operated by real train drivers. In both projects the operator was sitting behind a control panel that represented the main controls present within a train cab. The operator could see a representation of the cab displays and environment on a projection screen in front.

Besides the setup of the simulations it is important to note the subject of these projects, this in order to take into account what the acquired data can and cannot be used for. One of the projects, named DSSU, was done in a situation characterised by the distances between the signals. These distances were shortened and the signal relations were adjusted in such a way that the train driver would brake over multiple blocks, encountering a yellow number signal aspect followed by a normal yellow signal more often than before. This in order to decrease the amount of unnecessary braking that can occur when the train in front has been delayed, but is starting to speed up again at the same time the first train sees a yellow signal. This situation is often found at stations where trains are leaving while the next train is already arriving. The scenarios used within this project often thus had delayed trains in front of the operator in order to test the safety of this situation.

The other project was about a device and system called RouteLint. RouteLint is designed to give train drivers an overview of the seven sections of tracks ahead of the train, in order to see which sections are occupied by which trains. This devise is aimed at giving train drivers more information and to help with pro-active behaviour. Under normal circumstances information about track occupation that is further away than two or three sections of tracks would be too far away for the train driver to recognise from the visible signals. This project was aimed at getting a better insight in the effects of RouteLint, especially on the safety for using this system when approaching a red signal. Because of this the scenarios that were driven by the train drivers often included delayed trains, this so that the train driver could more often see trains in front of them with RouteLint. Due to this device influencing the way train drivers drive and make decisions only the control logs from these experiments were used. Within these control experiments no RouteLint was used or active.

MATRICS produces logs within an XML format for each time a train is driven. The main data that comes from these logs is in the form of train states, which contain the current position of the train, the speed, acceleration, time and information about the upcoming signals at that moment. Besides the train states, information is also provided for the ATB functionality and the departure and arrival times. One sample of a log file is seen in Figure 3.2.

The data available from the MATRICS logs can function as a source of information about how the train drivers deal with and react to signals. It was less practical to look at signs, seeing that the information about the sings are not stored within the log files themselves. The MATRICS log files come from experiments where a lot of hindering situations have taken place, meaning that there were very few occasions where the train driver was able to drive according to the schedule and without encountering an unplanned yellow and red signal. In total 956 log files (8.6GB) were used, which were driven by over 65 train drivers, in 7 train series in 29 different scenarios.

```xml
<LogItem>
  <TrainState>
    <SystemTime>14:04:33.477</SystemTime>
    <Time>66,477</Time>
    <Acceleration>-0,026</Acceleration>
    <Coasting>-0,026</Coasting>
    <Speed>34,142</Speed>
    <Position>536,669</Position>
    <DistanceToNextRedSignal>5228,487</DistanceToNextRedSignal>
    <DistanceToNextSignal>438,378</DistanceToNextSignal>
    <PreviousATBSpeed>40</PreviousATBSpeed>
    <CurrentATBSpeed>130</CurrentATBSpeed>
    <NextSignal>
      <SignalState>green</SignalState>
    </NextSignal>
    <NextSignal>
      <SignalState>green</SignalState>
    </NextSignal>
    <NextSignal>
      <SignalState>yellow</SignalState>
      <SignalNumber>6</SignalNumber>
    </NextSignal>
    <NextSignal>
      <SignalState>yellow</SignalState>
    </NextSignal>
  </TrainState>
</LogItem>
<LogItem>
  <Throttle>
    <SystemTime>14:04:33.923</SystemTime>
    <Time>66,923</Time>
    <ThrottleValue>1,000</ThrottleValue>
    <Speed>34,100</Speed>
    <Position>541,039</Position>
  </Throttle>
</LogItem>
```


Figure 3.2: *Sample of a MATRICS log.*

One notable limitation of the MATRICS log files had to do with the acceleration and braking behaviour. Within MATRICS, a number of aspects that influence these behaviours were missing:

- The simulation does not take into account the possibility of the wheels skidding. *In real trains the train driver cannot always give full throttle due to the wheels not having enough friction on the railway to produce momentum, instead they will start to skid.*
- The simulation is missing movement information, such as the movement and forces experienced when a train is braking.

Due to these reasons, the precise acceleration and braking behaviour will likely be not as accurate as in a real train.

3.2.2 GPS data

The GPS data comes from a project where a GPS tracker was attached to an intercity train in order to accurately log its position, velocity and the distance to the next red signal. Besides this other information was logged like the current time, the number of satellites, the acceleration and a number of GPS accuracy measures. The GPS data sets come from an experiment where ProRail tested an application that warns a train driver if he is approaching a red signal and is not braking hard enough. The position, velocity and acceleration information were logged every second, while the distance to the next red signal was logged four times a second once the train got within a certain distance to this red signal.

The usefulness of these logs comes from the facts that:

1. There is a large amount of log files.
2. The trains were logged while travelling over large distances.

The main downsides to this data set were that it needed to be filtered for inaccuracies, and that in order to get information about stops, delay, stations, etc, it needed to be linked to other sources of information. One example being the linking of the time table to the logged arrival and departure times in order to find out the deviation to the planned times. Another downside was that it only logged the position of the next red signal, meaning that there was no indication available of the location of the speed signs and other signals.

Given that the GPS data gives a better reflection of driving behaviour in non-obstructed situations and likely gives a better representation of the acceleration and deceleration behaviour, they were used mainly for acquiring information about these aspects. In turn, the MATRICS data gave a wide array of information about the signalling environment, and were thus used mainly for acquiring information about these aspects. The MATRICS data was also used in cases where information was otherwise unavailable within the GPS data.

3.3 Processing & preparation

Within the field of machine learning data-processing and preparation play an important role. A visual representation of this process can be seen in Figure 3.3. Before the data can be used for learning and fitting purposes it needs to go through this process. Within this project, the data that was available needed to be pre-processed in a number of ways in order to make it usable for machine learning and fitting.

The eventual goal of these processes was to get insight and information that could be used to model and formalize the decision making process of a train driver. For this purpose I wrote two log processing algorithms in Java, one for MATRICS and one for GPS, in order to gather and add the desired information. Within this section I will go over the data processing that was done for the MATRICS and the GPS files.

3.3.1 MATRICS processing

For the MATRICS files, the first two steps were pre-selecting the logs that could be used. This was done through the following two steps:

1. Filtering out the logs where RouteLint was used.
2. Filtering out logs where impactful notes were made about during the projects, either due to simulation errors or remarks by train drivers.

Once there was a usable set of logs the directly available information needed to be extracted from these. Most of the time the files were ordered based on time, there were however exceptions, namely when two events happened

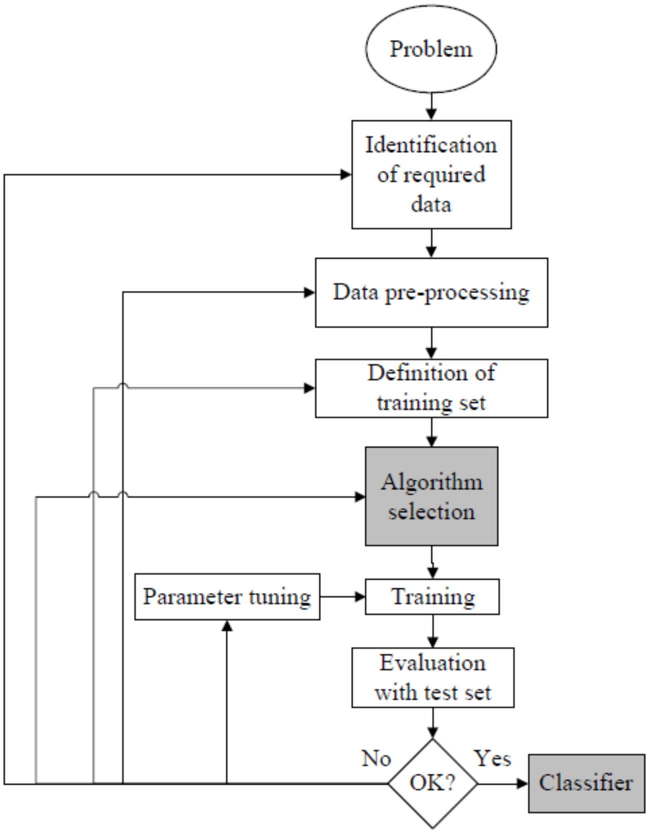


Figure 3.3: *The process of supervised machine learning.* [50]

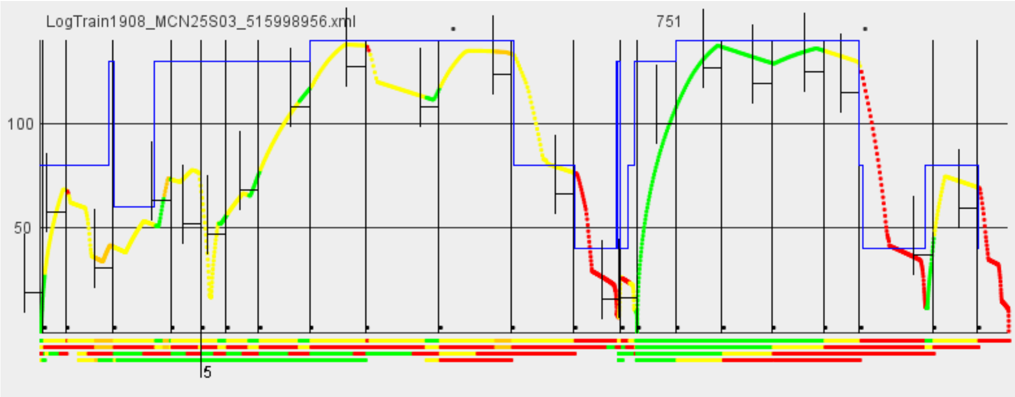


Figure 3.4: *A speed-distance diagram of a scenario within MATRICS. On the x-axes the distance is represented, with the starting point of zero, on the left. On the y-axes the velocity of the train is indicated in km/h. Underneath the x-axes the current signal aspects of the upcoming signals are represented, with the first one being on top. The blue lines display the allowed ATP speed at that point. The vertical black lines represent the position of the signals, with the H shape indicating the estimated visible distance of that signal. The speed-distance points are filled in with the colours that denote the colours of the current signal aspects, namely green with a green signal, yellow with a yellow signal and red with a red signal. Gold being the exception that indicates a yellow signal with a number below. The downwards extended black line indicates the fifth signal encountered by the train driver.*

soon after each other where one event contains more information to print than the other. This makes it necessary to sort the entries within the data set if you want to look at driving behaviour over time. The extraction was done through the following automated processing steps:

1. Read through the log files, sort and store the train state information.

2. Add to the train states information gained from the other entries (braking, traction, distance to arrival, current delay, signal information, ATP information).

Once this information was gathered it could be displayed visually on screen, as can be seen in Figure 3.4.

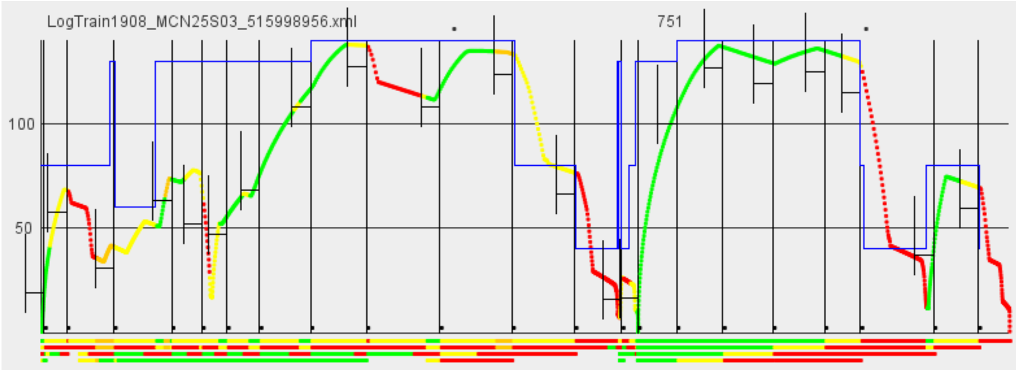


Figure 3.5: *A speed-distance diagram of a driven scenario within MATRICS. The content of this graph is the same as in Figure 4, with the exception of the colours of the speed-distance points. Here they denoting the current visible or expected signal aspect, rather than the current signal aspect.*

Looking at this information it was still unclear to see why train drivers performed certain actions, namely actions which did not seem to fit with the goal of driving in a safe and timely manner. One example visible in Figure 3.4 is the hard braking manoeuvre done after the fifth signal. To expand on this information and to give a better intuitive picture of why train drivers do certain actions information was added about the visibility of signals and signal relations. This in order to better represent the knowledge the train driver has about the visible and expected signals.

At this stage it was possible to see relations between the action that was undertaken by the train driver and the reasons for this action. These reasons stemmed largely from the rules and regulations that are in place. One example of this is visible in Figure 3.5 after the fifth signal. Here the train driver is braking hard due to the expected red signal, once he comes into the viewing distance of the next signal and sees that this is not currently red he stops braking and starts to accelerate again. It is now also visible that within this figure all braking manoeuvres are done either when approaching a red signal, when approaching a planned stop, or when the current velocity of the train is higher than the allowed velocity by the ATP.

Given the observations acquired through these visualisations a number of situational distinctions were made. These distinctions were meant to capture the overall reasons for the vast majority of actions done by train drivers. These

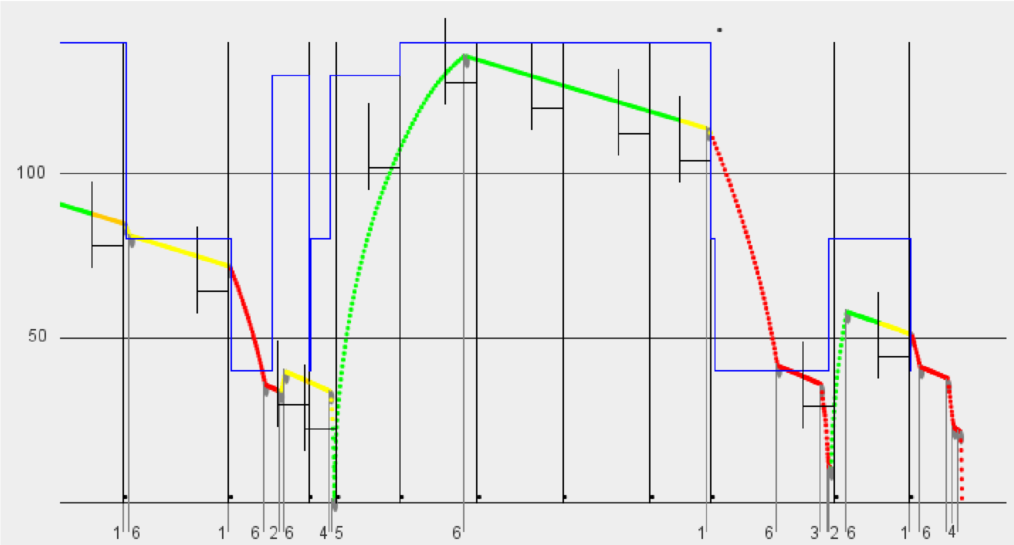


Figure 3.6: *Similar to Figure 3.5, the x-axes indicates the distance travelled, the y-axes indicates the velocity of the train. The colours indicate what the next viewable/expected signal aspect is. The grey vertical lines indicate when the train driver did an action. Below those lines numbers are added indicating to which situation this action belongs.*

were the following *situations*:

1. Approaching a signal or sign that indicates a lower allowed velocity than the expected velocity at the signal or sign in question.
2. Approaching a signal or sign that indicates a higher allowed velocity than the expected velocity at the signal or sign in question.
3. Approaching a red signal.
4. Approaching a stop.
5. Departure.
6. Significant speed has been reached.

Within Figure 3.6 you can see the decisions linked to the different situations.

The reason for distinguishing between situation 2 and 5 is that choice and timing did not play a role in 5. The timing part of a train departure will be done through FRISO, so once it indicates that it is time for departure, the train driver model should accelerate. The distinction between situations 3 and 4 comes from two main reasons. The first coming from the fact that in order to drive in a safe and timely fashion, a train driver will approach an unplanned red signal differently than a planned one. Wanting to not stop unnecessarily in front of an unplanned red signal, while for a planned one, wanting to stop in a safe, comfortable and timely fashion. Situation 4 was distinguished further from situation 3 through the presence of a scout sign, which indicates that the next station is at the end of the braking curve of the train. Actions that could not be directly linked to situations 1-5, such as braking from 140km/h towards 120km/h after having just completed an acceleration action from a departure while the maximum speed is still 140km/h, were grouped within situation 6. Actions such as these were deemed illogical, seeing that they were not forced through rules or safety systems, could not be linked to any signal or sign and do not contribute to driving in a safe and timely fashion.

These distinctions were laid out for a staff member from ProRail, who agreed with these distinctions while noting some important more detailed points not to forget. These points had to do with the course of the actions in question and will come back later in this thesis. After these situational distinctions were made, they could then be translated into the events the FRISO simulator would give for the agent to reply to. A more in depth look into the events and interaction the agent will have with FRISO can be found in Chapter 4, about the agent design.

With the event based structure of FRISO and these situations in mind, more specific information was gathered from the MATRICS logs in order to give a better overview of the decisions that were made by train drivers within these situations. This was done through first collection information from the decisions made by train drivers within MATRICS, followed by adding hypothetical FRISO events that could trigger actions through a change of *situation*. These hypothetical events were then linked to available decisions that were made in MATRICS. Thus giving information about, if an event took place in FRISO, what decisions did real train drivers make. This automated process can be summarized in the following steps:

1. Add information about each decision made within MATRICS:
    - Store the kind of situation the train driver was in.

- Store aspects of that situation and the following decision.

2. Add information about the situation where events would take place within FRISO:
   - Link the available decisions that happen around these events.
   - Note when no action took place.
3. Create logs for each event case and write the information to new log files.

A specification of the information acquired for each event can be found in Appendix 1.

3.3.2 GPS processing

In order to get information about the way train drivers accelerate, decelerate and act in normal not obstructed situations the GPS files first needed to be pre-processed. Compared to the MATRICS files they required more filtering and less processing, this due to GPS data being less reliable than software logs, and GPS coordinates giving less information about the situation the train than is available within a MATRICS log file.

The initial steps of this automated process were:

1. Filter out logs where no signal distances at all were logged.
2. Filter out the non-GPS entries.
3. Filter out the GPS entries where the data became unreliable. *This occurred either because not enough satellites were available*(<4), *or when the speed became too low* $(<1m/s)$.
4. Filter out unusual entries. *An entry was considered unusual when the speed difference between two entries that are one second apart was* $> 1m/s$. *If it was larger than* $1m/s$, *it would mean that the train was accelerating unnaturally fast.*
5. Smooth the acceleration due to high initial variation.

Step five was taken due to the high variation within the acceleration measurements of the GPS data. This variation made it impossible to use the acceleration for informative purposes when looking at the differences between individual successive entries. For this reason the average was taken for four entries around each entry. Through doing this, it was possible to use the difference in acceleration found between successive entries in order to determine an acceleration increase or decrease visible within the velocity and distance data.

After having acquired this information, meta information was added in order to get a better insight into the state a train finds itself in. This information was then used to create logs that could be used for machine learning and fitting.

- Add information:
  1. Distance travelled since the last entry.
  2. If the train has stopped.
  3. If it is a stop at a planned station.
  4. The distance to the next stop.
  5. The delay based on the time table of the NS.
  6. If the train is braking or accelerating.
  7. If the train is cruising or coasting.
- Make logs for collecting the desired information.

For an overview of the information gathered for each case, see Appendix 2.

3.3.2.1 Difficulties

It proved an unreasonable process to extrapolate the traction and brake lever position from the GPS data. This due to a combination of factors, most notably the variation in the rolling stock used, combined with real life effects which also influence the braking, such as the weather conditions, bends, slopes and train composition. There was also a slight error in the distance to the next red signal due to the fact that the system was using a GPS distance calculation between the position of the signal and the train, which resulted in it not taking into account corners. This difference turned out to be small enough to not have any significant impact.

The GPS files also include a more frequent logging entry under the name 'Bewaking', they were used to more accurately give information about the train once it is approaching a red signal. Entries of this kind were logged four times a second once the train got within a certain distance of a red signal.

These entries proved difficult to use correctly, due to the rounding of the positional data. This rounding of the latitude and longitude meant that the location was less accurate than the normal GPS entries. These entries were thus not possible to use for positional information, and in turn speed and acceleration information.

3.3.3 Conclusion

Processing the data files, exploring them, refining them, and looking at the results took a significant amount of time. Through it, together with informal interviews with two train drivers, a better insight was gained into the different aspects of driving a train and how I could use the data available to create a model. One notable observation made when looking over the different data sets that were available was that often a good description of the data were missing. Only through asking specific questions information about the accuracy of certain attributes was acquired. On top of that, often only the aspects that were relevant to the research project in question were logged. This while other aspects and attributes could have easily been logged as well. Simply logging all available data that is not derivable from the already present attributes could significantly improve the usefulness of the data ProRail gathers outside of the specific projects. Within the next section I will describe how I used this information to learn and fit aspects of this model.

3.4 Learning methods & results

In the previous sections the data and the data processing steps were described. Once the desired data is acquired the next step is to use this to help creating an implementable decision making model of a train driver within FRISO. With the availability of the GPS and MATRICS data sets the way is opened to use supervised learning techniques for this purpose. Within this section I will go through the process of fitting distributions, functions and learning classification trees in order to describe certain aspects of this decision making model. The learning of classification trees is aimed to create a starting point for a decision tree, while the fitting of functions and distributions is a more mathematical approach aimed to help creating a more variable predictive model. The earlier described set of situations will serve as a starting point for this process.

Action:

- Throttle percentage
- Braking percentage
- Onset position
- Final velocity

Figure 3.7: *Overview of an action.*

3.4.1 Classification trees

After having distinguished different situations which can serve as reasons for doing actions, there is still the need for the modelled train driver to decide which action it will perform. Within this section I will go into acquiring a selection process that can select which actions a train driver can perform given the current situation and state of the environment.

Within this setting actions can be described as consisting of four parts, namely what kind of action it is, at which point on the tracks the action starts, what the course of the action looks like and at which point the action stops. One example of an action that contains these four aspects would be that the train driver decides to break at the next signal, with the braking lever at the position that fulfils the braking criteria, until the train is going 40 km/h.

The problem of selecting which action a train driver will do lends itself to be described as a classification problem, where given a certain *situation* and the state of the environment one kind of action needs to be selected amongst a set of possible actions. Not taking into account a train drivers responsibilities outside the immediate driving task, the set of possible actions a train driver is allowed to perform can be described as:

1. Give throttle with a certain amount.
2. Brake with a certain amount.
3. Coast.

I will from now on refer to the point at which such an action selection needs to take place/has taken place as a decision point. The precise moments at which decision points take place within the simulation will be described in Chapter 4. For this selection problem, I will use a classification tree learning method on the data acquired from the MATRICS logs. The output of this classification tree learning method will function as the basis for a decision tree.

Before the learning algorithm could be applied, the learning sets (training data) needed to be created. These learning sets were based on the data sets acquired from processing the MATRICS log files. The reasons for not being able to use the processed data sets directly was twofold. One reason being that certain attributes contained information that could not be known at the decision point, this posterior knowledge of the situation at hand thus needed to be removed. The other reason was due to the way certain values were logged. One example being the logging of the decision points when approaching a planned stop. For these decision points a *final velocity* attribute was logged, which indicated until which velocity the train slowed down if a braking action started. If no braking action was started, this attribute acquired the value of -1. This attribute could thus serve as a predictor whether or not a train started to brake. An overview of the attributes that were used and how the situations were categorized can be found in Appendix 3. The resulting data sets contained information about the environment at the moment a train driver passed a decision point.

The creation of the classification trees was done through the learning algorithm named J48, which is based on the C4.5 algorithm [68], which is usable through the software application WEKA [37]. The main reasons for using this algorithm was due to its easy availability and the mention by [22] that it *"has become a de facto community standard against which every new algorithm is judged"*, thus indicating that the results acquired with this algorithm can likely be compared more easily with others. For all learning instances 10 fold cross-validation was used in order to distinguish between training and test sets. It also served to make the resulting trees more general, in the sense that the resulting trees would perform more similarly on an unknown data set like the ones it could encounter during simulations.

The decision trees were learnt for the situations 1 to 4, as described in section 3.3.1. Situation 5 was not included due to the train always having to depart, thus there not being a choice whether or not to depart. Situation 6 was not included seeing that this event corresponded with the onset of an action that could not be linked to a specific change in the environment besides the velocity of the train.

An overview of the resulting trees and error rates for situation 1 to 4 can be seen in Tables 3.1 to 3.4.

Approaching Signal/Sign Lower Speed: [Situation 1]		
Number of leaves	71	
Size of the tree	132	
Correctly Classified Instances	3705	93.5842 %
Incorrectly Classified Instances	254	6.4158 %
Total Number of Instances	3959	

Table 3.1: *Overview of the classification tree learned within Situation 1.*

Approaching Signal/Sign Higher Speed: [Situation 2]		
Number of leaves	433	
Size of the tree	789	
Correctly Classified Instances	8188	87.1899 %
Incorrectly Classified Instances	1203	12.8101 %
Total Number of Instances	9391	

Table 3.2: *Overview of the classification tree learned within Situation 2.*

Approaching Red Signal: [Situation 3]		
Number of leaves	9	
Size of the tree	14	
Correctly Classified Instances	507	93.8889 %
Incorrectly Classified Instances	33	6.1111 %
Total Number of Instances	540	

Table 3.3: *Overview of the classification tree learned within Situation 3.*

Approaching Planned Stop: [Situation 4]		
Number of leaves	9	
Size of the tree	14	
Correctly Classified Instances	3009	97.3786 %
Incorrectly Classified Instances	81	2.6214 %
Total Number of Instances	3090	

Table 3.4: *Overview of the classification tree learned within Situation 4.*

The resulting classification trees included leaves with little added value. This also made them quite large, especially in the case of situation 2. Looking closer at the content of these decision trees, even though most rules made sense, there were at times illogical conditions such as the following section of a rule within the tree learned from situation 2:

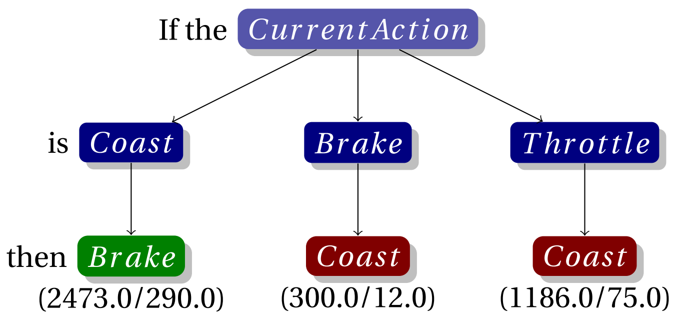
$$\textit{If } throttle = true \wedge Delay > 202.055 \wedge Dpa > 2029.6038 \rightarrow \textit{Start coasting.}$$

Where Dpa indicates the distance to the previous action in meters.

This rule states that if the train is currently accelerating, the current delay of the train is larger than 202.055 seconds, and the distance to the previous action done is larger than 2029.6038 meters, the train driver is to start coasting.

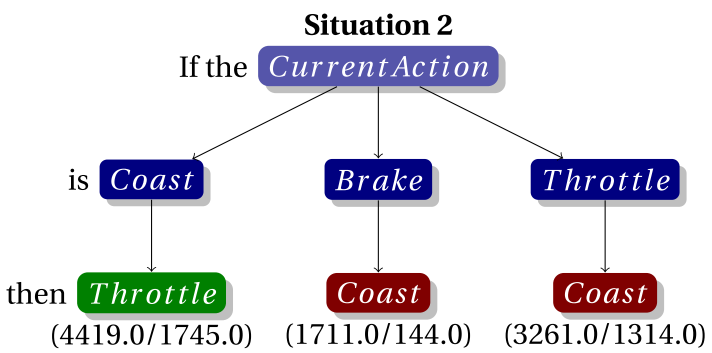
I concluded from this that the trees were overfitting. To combat this a stricter pruning value was used but resulted in similar results for the larger trees. One possible reason for this is that the MATRICS data comes heavily from delayed situations, thus resulting in a train driver that drives as if there is a delayed train in front. One observation that was possible was that the first node of the trees all used the same attribute, namely the *current action* of the train. The *current action* could have one of the following values: Giving throttle, braking or coasting. Given this information, classification trees were made using only this attribute. The resulting classification trees and their error rates can be seen in Tables 3.5 to 3.8. Within these classification trees, the values below the leaf nodes indicate the total amount of classified instances and the amount of wrongly classified instances. Within the tables there are also confusion matrices. A confusion matrix is a table that can represent the classification accuracy of a classification method. The meaning of the content of the cells within a confusion matrix can be seen in Table 3.9.

Situation 1



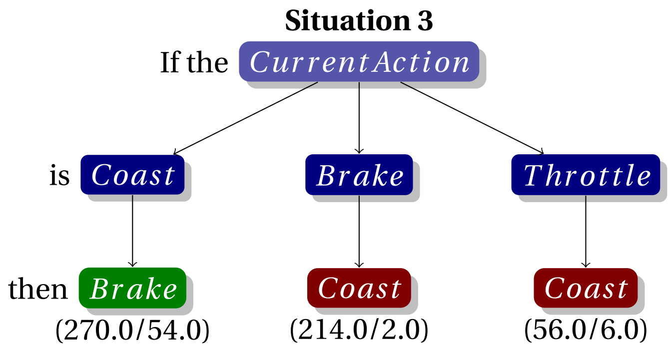
Approaching Signal/Sign Lower Speed: [Situation 1]				
Correctly Classified Instances	3582	90.4774 %		
Incorrectly Classified Instances	377	9.5226 %		
Total Number of Instances	3959			
Confusion Matrix:				
Predicted outcome:				
Braking	Coasting	Throttle		
2183	59	0	Braking	Actual value:
0	1399	0	Coasting	
290	28	0	Throttle	

Table 3.5: *Overview of the classification tree learned within Situation 1.*



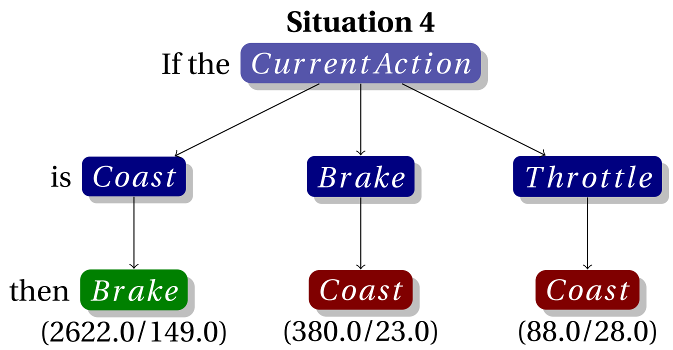
Approaching Signal/Sign Higher Speed: [Situation 2]					
Correctly Classified Instances	6188	65.8929 %			
Incorrectly Classified Instances	3203	34.1071 %			
Total Number of Instances	9391				
Confusion Matrix:					
Predicted outcome:					
Throttle	Nothing	Coasting	Braking		
2674	0	258	0	Throttle	Actual value:
1110	0	1187	0	Nothing	
0	0	3514	0	Coasting	
635	0	13	0	Braking	

Table 3.6: *Overview of the classification tree learned within Situation 2.*



Approaching Red Signal: [Situation 3]				
Correctly Classified Instances	478	88.5185 %		
Incorrectly Classified Instances	62	11.4815 %		
Total Number of Instances	540			
Confusion Matrix:				
Predicted outcome:				
Braking	Coasting	Throttle		
216	5	0	Braking	Actual value:
0	262	0	Coasting	
54	3	0	Throttle	

Table 3.7: *Overview of the classification tree learned within Situation 3.*



Approaching Planned Stop: [Situation 4]		
Correctly Classified Instances	2890	93.5275 %
Incorrectly Classified Instances	200	6.4725 %
Total Number of Instances	3090	

Confusion Matrix:				
Predicted outcome:				
Braking	Coasting	Throttle		Actual value:
2473	28	0	Braking	
0	417	0	Coasting	
149	23	0	Throttle	

Table 3.8: *Overview of the classification tree learned within Situation 4.*

Predicted value:				
Class A	Class B	Class C		Actual values:
As classified as A	As classified as B	As entries classified as C	Class A	
Bs classified as A	Bs classified as B	Bs entries classified as C	Class B	
Cs classified as A	Cs classified as B	Cs entries classified as C	Class C	

Table 3.9: *Overview of a confusion matrix.*

Within Table 3.6 for situation 2 it can be seen that the act of doing nothing, so not changing the current action, was also incorporated. The main reason for this was that the act of accelerating within this situation is not enforced through any safety systems. This is unlike the other situations, where in the vast majority of cases the act of braking will be enforced through the ATP safety system. This, combined with not being able to give throttle and brake at the same time, also explains why the current action is a good predictor in situations 1, 3 and 4. The only enforcement that a train driver has for accelerating is the goal to arrive on time, which leaves the train driver with more flexibility to not act within this situation.

Within these tables it is visible that only Table 3.6 suffers from a significant performance hit if only the current action is taken into account. From the confusion matrix in Table 3.6, you can see that the majority of cases that are classified wrongly occur in one of the following two cases: The train driver is coasting, and decides to do nothing, thus continuing to coast. The train driver is giving throttle and decides to do nothing, thus continuing to give throttle.

These cases occur mainly within the first two parts of a ride between two stations, namely during the initial acceleration after the departure and the driving behaviour during a non-obstructed free track section. It was observed that the initial acceleration curve usually continues until a velocity around the speed limit is reached. The main exceptions to this are:

- When a delayed train is in front and causes a yellow signal to be seen before a desired velocity could have been reached.
- When the next stop is close enough to require the train to start braking towards a planned stop before a desired velocity around the speed limit could have been reached.

This observation was kept in mind when determining whether a new action should be taken or not. A more in depth look into the driving behaviour during the free-track sections will be presented later in this thesis, in the section Behaviour variations.

Conclusion

Given the high prediction rates of the classification trees that were based on the *current action* attribute for situations 1, 3 and 4, it was concluded that this could serve as the initial attribute of a decision tree to select which actions can be done. These actions are in line with the regulations in place, seeing that for situations 3 and 4 the train driver will eventually need to brake in order to stop. For situation 1 the train driver will have to slow down to the required velocity and within situation 2 they will either coast or accelerate.

If I did not use current action as the sole predictor the classification trees, like for situation 2, tended to overfit and introduce nonsensical rules. Two possible explanations for this are:

1. The data came from situations where a lot of trains were delayed, thus increasing the exposure to unplanned red signals, causing rules that reflect the driving behaviour under severe delays instead of normal driving behaviour.
2. The concepts that the learning algorithm tried to capture were too general for it to work well given the present attributes.

In order to compensate for the relatively low accurate prediction rate of situation 2, the final model will take into account the observations made about the initial acceleration curve and a closer look will be given to the free-track sections within section Behaviour variations.

Given the predication rates of situations 1, 2 and 4, respectively 90.4774%, 88.5185% and 93.5275%, it was concluded that the actions done by train drivers that do not fall within this classification will be seen as exceptions to normal train driver behaviour. A part of these wrongly classified instances could be attributed the mislabelling of the situation from the train drivers perspective, likely due to the use of an estimated viewing distance. The other part of these instances, the exceptions, will only partially be modelled and noted at the end of this chapter.

3.4.2 Fitting

3.4.2.1 Introduction

Within the last section the initial decision trees were acquired. With these a possible action can be selected. Given the kind of action a train driver will do, the questions that are still remaining are:

1. At which point does an action start?
2. How does the course of an action look like?
3. At which point does an action stop?

Before going into how I approached these questions I will aim to clarify these three points briefly. The point at which an action starts will be used in the sense of a position in front of the train where the onset of this action will take place. The course of an action will be used in the sense of denoting a specific traction or brake-lever position. The point at which an action stops will be used in the sense of a specific velocity that has been reached.

In order to answer the three above mentioned questions I looked through the available data to see which parts were possible to model through either fitting the data to a distribution or function. The main reason for using this approach was due to the numerical nature of these values. A function was used in the cases where a clear relation between attributes was visible, a distribution was used in the cases where this was not. Not every necessary positional or course specification was acquired through the use of fitting methods. This due to either simulation restrains, time restrains, or unavailable data. In these cases I implemented a default value, if possible based on domain knowledge, if not a value was chosen arbitrarily. Within this section I will go into the cases where I did use the fitting methods, starting with the distributions, and briefly mention the most notable cases where I did not use fitting methods.

The fitting of the distributions was done through one of two methods. One method used a Generalized Reduced Gradient(GRG) algorithm [52], available within Microsoft Excel [59]. This GRG algorithm is aimed at optimizing non-linear problems. In this case, it tries to find the values for the parameters of a distribution that minimises the Kolmogorov-Smirnov statistic [56]. This Kolmogorov-Smirnov statistic is a goodness of fit measure, which gives a value that describes the maximum distance between two sets of values. The idea is then that if you minimize this value, the maximum distance between the two sets of values is minimized, thus that the resulting sets are closer to each other. Within this thesis I will refer to this value as the goodness of fit measure, where a lower value is better. The mathematical definition of this statistic is:

$$D_n = \sup_x |F_n(x) - F(x)|$$

Where $F_n(x)$ is the empirical distribution function and $F(x)$ a cumulative distribution function.

The other distribution method was done through the maximum likelihood method implemented in the *fitdistr* function found in the software R [82] package MASS [85]. The type of method used was selected depending on the characteristics of the data, with the *fitdistr* method giving better results if there were fewer values, or if there were missing values. The type of distributions that was fitted towards was selected based on the shape of the observed distribution.

The functions were created through using the linear regression method available within [59]. In order to indicate the accuracy of these functions I will mention the coefficient of determination, the R^2 value. The closer this value is to 1, the higher the correlation is between the line and the data. I will go through the situational distinctions one by one and give an overview of this.

Before the fitting methods could be applied, learning sets needed to be made. These sets were altered versions of the data sets acquired through the processing of the MATRICS and GPS log files. They were altered in two ways:

1. Certain filters were applied.

2. If required the data was shifted before fitting.

With shifting I am referring to moving the value-range of a distribution, mirroring the distribution, or to expand or contract the value range of a distribution. The main reason for shifting the data, was to help the fitting methods. These actions had the effect of at times increasing the goodness of fit measure due to making it easier for the algorithm to fit a distribution to the data, or were required in order for the algorithm to work. There were two main reasons for the application of filters, the first one being to filter the data to correspond with the desired situation or state. One example being the filtering of approaching a planned stop to only include cases where the train driver did a continues braking action towards a standstill, starting from a velocity between the 25 and 45km/h. The second reason for applying filters was to remove data entries that were out of range, impossible or extreme outliers.

3.4.3 Distributions & Functions

In this section I will go through the situations and give an overview of the distributions and functions acquired for the purpose of answering the questions about the onset, course and end point of an action. This to represent the variety found within the actions of real train drivers. I will not go into the specific onset, course and end point of each possible action, instead I will focus on the actions where the acquired distributions and functions play a role. Starting with situation two, *Approaching a signal or sign that indicates a higher allowed velocity*.

Situation: Approaching a signal or sign that indicates a higher allowed velocity

For this situation I will look first into the action where the modelled train driver will switch from braking to coasting. For the timing of this action I used the *Time since last signal improvement* attribute from the data gathered from MATRICS. Thus timing the onset of the switch from braking to coasting to the timing of the last signal improvement observed. This empirical distribution was then fitted to a gamma distribution, seen in table 3.10, through the use of *fitdistr*.

Source:	Number of entries:	Shape:	Scale:	Goodness of fit:
MATRICS	77	2.524498	1.368401	0.1012559

Table 3.10: *The resulting information acquired through the fitting of the Time till last signal improvement to a gamma distribution.*

The other action I will look into for this situation is arguably the most important one, namely the acceleration action. The default onset point for this action was chosen to be right in front of the train, at an arbitrary 10cm.

After having acquired the onset point for the acceleration action, the course and onset point needs to be specified. For the course of the acceleration action the position of the traction lever needed to be specified. This traction lever position will be represented as a percentage. With 100% indicating the traction lever being in the maximum position, and 0% indicating that no traction is given. The main reason for this representation, instead of using the actual numerical traction lever positions available within trains, is due to the way FRISO deals with acceleration and requires this percentage.

For determining the position of the traction lever the information acquired from the passenger train driver interview was used. This mainly because of not having access to the traction lever positions of the trains within the GPS data and the questionability of the traction lever positions acquired through MATRICS. It was however possible to observe the variation of the acceleration curves from the GPS data, this can be seen in Figure 3.8.

Within the interview with the train driver, he stated that he always accelerates with certain intervals. One example being: Accelerating to a velocity of $\approx 20 km/h$ with the traction lever in position 1. After reaching that velocity, accelerating to a velocity of $\approx 45 km/h$ with the traction lever in position 2. Etc.

In order to represent the visible variation within Figure 3.8 better, this method of acceleration described above was formalized together with three uniform distributions. Here the first distribution controls the interval between the velocities. The second one controls the gradual increase for the percentage of the traction lever. The third one controls the variation of the final velocity of this interval. These three distributions also offer the ability of adjusting the variation at will.

In order to determine the end point of the acceleration action, I aimed to answer the following question: *If a train driver decides to accelerate, to which*

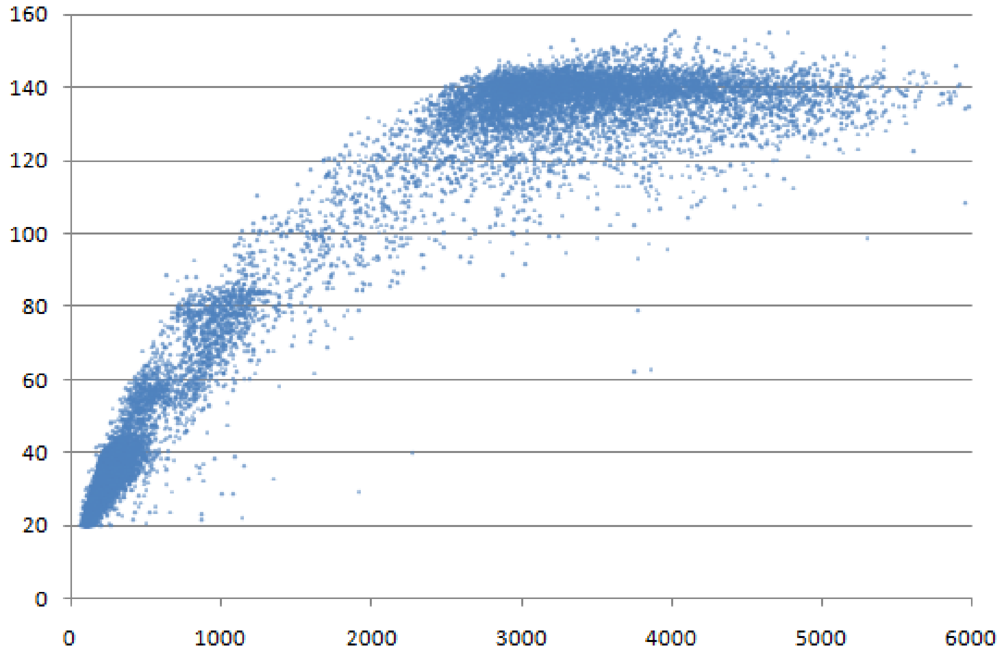


Figure 3.8: *The final velocity of departures, where the velocity difference is at least 20km/h. Giving an impression of the variation of the acceleration behaviour found between train drivers. On the x axes the traversed distance in meters since the departure is visible. On the y axes the velocity of the train is visible in km/h.*

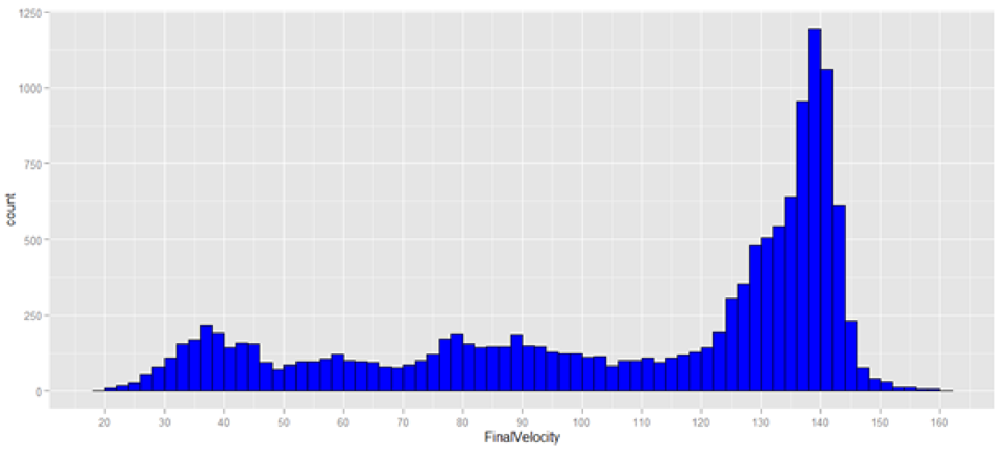


Figure 3.9: *final velocity of accelerations of at least 20km/h from the GPS data.*

velocity does he accelerate given the current maximum velocity?

For this I used the processed GPS data set named traction , this data set contained the final velocity of acceleration sections done by train drivers. In this case, an acceleration section is defined as a non-stop acceleration manoeuvre done by the train driver, where the difference between the onset and final velocity is higher than 20km/h. Within Figure 3.9, the resulting final velocities can be seen. Following this, the range of 135km/h to 145km/h was selected as a representative distribution for the selection of a final velocity given the current maximum velocity. This range was selected due to the frequency of the final velocity being around the 140km/h mark, combined with it being a velocity range that can only be reached when the maximum velocity is either 140 or 160km/ h and as such suffering from less noise. The increment of 5km/h around the speed limit of 140km/h was taken for the following reasons: When the maximum velocity is 140km/h, the maximum velocity the train can accelerate towards without triggering the ATP safety system is around the 145km/h. Thus any train that goes above has a maximum velocity that is over 140km/h. For the same reasons, any train that goes above the 135km/h must have a speed limit of at least 140km/h.

The fitting of this distribution was done on a normal distribution through the use of the GRG algorithm. For this the final velocity acquired from the data was first shifted towards the 0 km/h in order for the algorithm to work properly. This shifting meant that the lowest value possible within the data, 135km/h, would correspond with 0km/h, while the highest possible value found, 145km/h, would correspond to 10km/h. After the fitting process, the resulting normal distribution, specified in table 3.11 and visible in Figure 3.10, could be used to get an offset velocity. This offset velocity once added to the speed limit, and then minus 5km/h, will give a final velocity for an acceleration curve.

Source:	Number of entries:	Mean:	Standard deviation:	Goodness of fit:
GPS	4334	4.477670043	2.412290644	0.03162978

Table 3.11: *The resulting information acquired through the fitting of the final velocity to a normal distribution.*

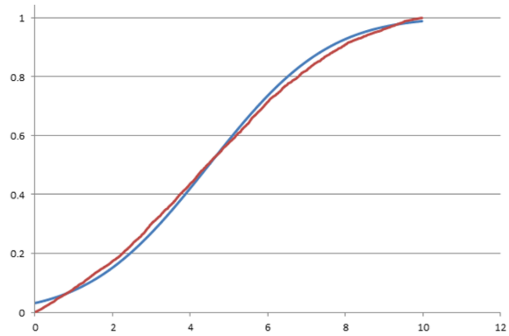


Figure 3.10: *The resulting accumulative normal distribution specified in table 3.11 is visible in blue and the observed cumulative distribution can be seen in red.*

Situation: Approaching a signal or sign that indicates a lower allowed velocity

The next situation I will look at is situation 1: *Approaching a signal or sign that indicates a lower allowed velocity.* For this situation I will look into the braking action, starting with acquiring an onset point.

For this I used the timing information acquired from MATRICS, the *Time to relevant signal passage.* The reason for using this measure instead of another, like the distance to the relevant signal passage, is because the ATP system enforces a velocity based on the timing of the speed limit change. It thus being more likely that a train driver estimates when to start braking based on the time relative to this speed limit change at the sign or signal, rather than the position of the train. The empirical distribution of the Time to relevant signal passage was first shifted after being fitted to a gamma distribution through the use of *fitdistr*. The resulting distribution can be seen in table 3.12.

Source:	Number of entries:	Shape:	Scale:	Goodness of fit:
MATRICS	1650	4.442802	0.8251686	0.06394251

Table 3.12: *The resulting information acquired through the fitting of the Time to relevant signal passage to a gamma distribution.*

After this there are still two question left, what does the course of the action look like and when does the action stop. Unlike situation two, these two are closely related due to the rules that impose that the train has to reach a certain velocity before a certain point. I will split these rules into two general cases: One being that the train driver needs to limit its velocity after passing the yellow signal and prepare to stop for the next red signal. The other being all the other cases where the train driver only needs to limit its speed to a certain velocity before reaching a specific signal or sign.

For the first case the train driver is modelled to start braking with the minimal braking amount, to just fulfil the braking criteria, after which the rest of the braking manoeuvre will be done through the situation *Approaching a red signal*.

For the second case the train needs to limit its speed to a certain velocity. This velocity was acquired through fitting a shifted empirical distribution of this velocity to a gamma distribution. The empirical distribution came from GPS data, where trains had decelerated to a velocity between the 70 and 90 km/h. This range was selected for similar reasons as the range for the final velocity of situation 2. One reason being that the frequency of braking manoeuvres that ended between these velocities was high. The other reason being that if a train driver brakes to a velocity between the 75 and 85 km/h, it was likely due to a sign or signal forcing this specific action. The reason for choosing a range between 70 km/h and 90 km/h was due to the gamma distribution fitting better if it had a larger range to fit to. The final velocity within the implementation described in Chapter 4, limited the range of this distribution to be between 5km/h more or less than the speed limit. The resulting gamma distribution acquired through the GRG algorithm can be seen in table 3.13.

Source:	Number of entries:	Shape:	Scale:	Goodness of fit:
GPS	4806	3.746256503	0.131907088	0.044319667

Table 3.13: *The resulting information acquired through the fitting of the Time to relevant signal passage to a gamma distribution.*

Once a final velocity has been acquired, the last question to answer is what the course of the braking manoeuvre will look like. This part was modelled through the assumption that train drivers prefer to brake as little as possible. Given this assumption, the point where a train driver aims to stop its braking manoeuvre will be just in front of the signal or sign that requires the speed limit to have been reached. Given that FRISO uses a linear model for deceleration, the required position of the braking lever can be calculated given the onset point, a final velocity and a stopping point. These calculations can be seen in Equations 3.1 and 3.2.

$$requiredAcceleration = \frac{finalVelocity - initialVelocity}{\frac{distance}{\frac{initialVelocity+finalVelocity}{2}}} \tag{3.1}$$

Where the $requiredAcceleration$ is the acceleration needed to reach the $finalVelocity$ given the $initialVelocity$ and the $distance$.

$$brakeSetting = \frac{requiredAcceleration * 100}{maximumDecceleration} \tag{3.2}$$

Where the $brakeSetting$ is the position of the braking lever in a percentage. The $requiredAcceleration$ is the necessary deceleration and the $maximumDecceleration$ is the maximum negative acceleration possible of this train type.

Once the required braking setting has been acquired the modelled train driver will select the maximum between the required braking lever position and the minimum braking amount. Thus making sure that the train will reach the required velocity in time, while also making sure that the modelled train driver is following the braking criteria. This is done in order to follow the assumption of train drivers wanting to brake as few as possible.

Braking

Before going into situation three and four I will line out how the braking to a stop action has been modelled. The final approach to a planned stop or a red signal was separated into two separate braking manoeuvres. This was done to represent the braking behaviour visible in Figure 3.11. Within these manoeuvres it is visible that there are three areas, in grey, where the train driver often does not brake but instead coasts until he is closer to the stopping position. These three areas correspond with common speed limitations found just before a red signal, namely those of 80, 60 and 40km/h. The speed limit of 40km/h is always present before a red signal, while the presence of the 80 and 60km/h speed restrictions depend on the present signals and signs. Situation one already deals with the braking actions towards the 80 and 60km/h speed limitations, and the onset of the braking manoeuvre towards the 40km/h speed limit. The part that is missing is thus the course and endpoint of the braking

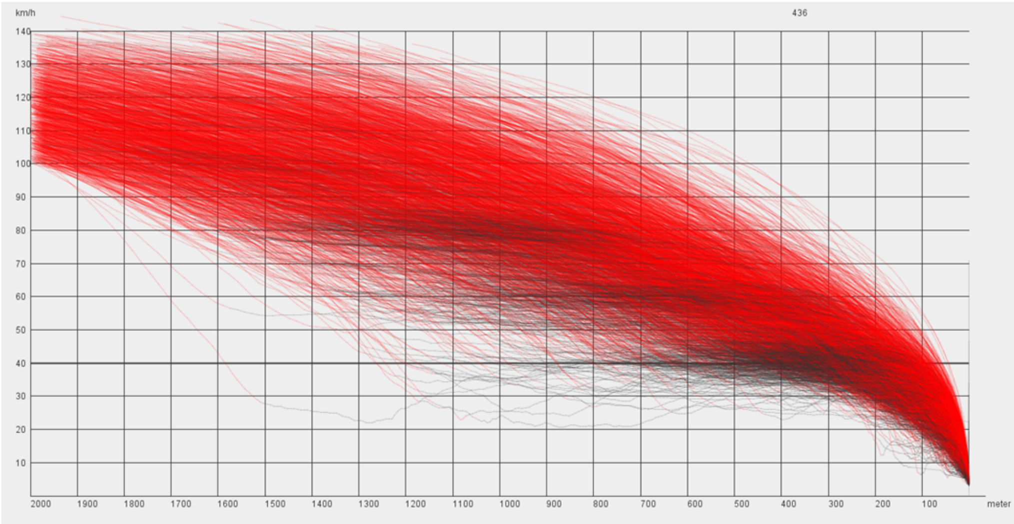


Figure 3.11: *Braking manoeuvres when approaching a planned stop. On the x-axis the distance to the stopping point. On the y-axes the velocity in km/h. Each line represents the braking manoeuvre of an individual train, with the red coloured parts indicating when the train is braking, and the grey coloured parts indicating when the train is not braking, as in: when it is accelerating, cruising or coasting.*

action towards the 40km/h speed limit, and the braking manoeuvre to a stop. The only times a speed limit of 40km/h is missing before a stop is when the train arrives on a green signal, which usually only happens on stations on the free track sections.

In order to come to a model that can describe these braking manoeuvres for different types rolling stock and variation in braking behaviour an appropriate way to model these actions needed to be selected. An approach using classification trees, to determine whether to brake or not to brake, would result in a model that is difficult to use when trying to plan the onset and endpoint of an action and would have resulted in a reasonably uniform braking behaviour. On top of that was a positional requirement that FRISO imposed on the stopping position for planned stops. This meant that it would be easier to use a method that could reason back, given the distance to the stopping point, how a braking manoeuvre would look like. Therefore I chose to combine different distributions, functions and equations of motion to describe these two braking manoeuvres.

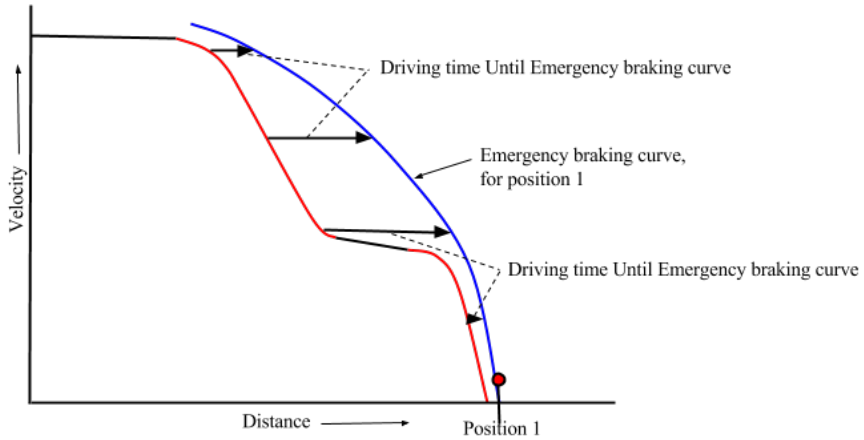


Figure 3.12: *Visual representation of the Driving time Until Emergency braking curve.*

Situation: Approaching a stop: Braking to a 40 km/h speed limit

The next situation I will go into is situation three: *Approaching a stop*. As mentioned before, the braking manoeuvre to a stop is done in two braking actions. Starting with the braking action towards the speed limit of 40km/h. For this braking action an onset point will be set right in front of the train except if the train is under the maximum allowed velocity and the initial *Driving time Until Emergency braking curve* (DUE) is deemed too high, namely above 60 seconds. The DUE refers to the driving time until the train reaches the position of the emergency brake curve, which refers to the maximum braking curve a train can do in order to stop before a specific position. This concept is shown visually in Figure 3.12.

Normally this position refers to the position of a red signal. In this case the position refers to the planned stopping position of the train next to the platform. This due to the possibility of the train arriving on a green signal. If the initial DUE is too high, a new initial DUE will be selected from a distribution, which is the same distribution where this limit of 60 seconds comes from. A selection of a new initial DUE means that the modelled train driver will wait with starting the braking manoeuvre until the train is at the position where the DUE is the same as the selected initial DUE.

The main reason for using this check on the initial DUE, is due to the scout sign. The scout sign is a sign that indicates that a station is ahead and is at

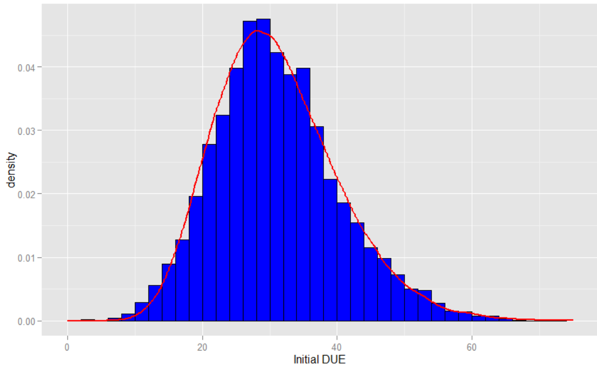


Figure 3.13: *In blue you can see a density histogram of the Initial DUE found within the GPS data. In red the fitted gamma function aimed at representing the data.*

the braking distance. Thus indicating to the train driver that he needs to start braking around that sign. The location of this sign assumes that the trains velocity is close to the speed limit, if the train is significantly below this, the onset point of the scout sign would be too early. Thus this check on the initial DUE ensures that the train does not start braking unnaturally early.

The distribution on the initial DUE was acquired through fitting the empirical distribution of the initial DUE found within the GPS data, to a gamma distribution with the GRG algorithm, which can be seen in table 3.14 and Figure 3.13. The main reason for using the initial DUE as a reference point for the onset of a braking action, is due to the distribution being independent of the initial velocity. This can be seen in Figure 3.14, where there is no observable directional influence between the two attributes. This means that regardless of the random initial DUE drawn from a distribution, it will not clash with the current velocity of the train.

Source:	Number of entries:	Shape:	Scale:	Goodness of fit:
GPS	5032	11.5139083698441	2.69911282219247	0.010799605

Table 3.14: *The resulting information acquired through the fitting of the Initial DUE to a gamma distribution.*

Once the onset position has been acquired, a final velocity to brake towards needs to be selected. Due to the observed range and frequency of braking manoeuvres towards the speed limit of 40km/h, the final velocity will fall within

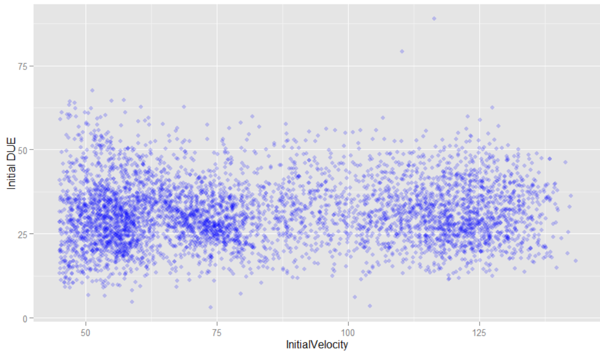


Figure 3.14: *On the x-axes you can see the initial velocity of a braking action towards a stop. On the y-axes you can see the initial DUE of this onset point. Visible are three more densely clustered areas where a more frequent occurrence of a specific initial velocity has taken place. These correspond with the speed limits of 60, 80 and 130-140. No correlation is visible between the initial velocity influencing the distribution of the initial DUE.*

the range of 25 to 45km/h. The distribution of these velocities was acquired from an adjusted version of the Final velocity attribute found within the processed GPS log files for this situation. The main reason for using the final velocity attribute and not the final position, or the final DUE, was because there seemed to be no correlation between the initial velocity and the final velocity. This can be seen in Figure 3.15, where there is no directional relation visible between the initial and final velocity. Thus indicating that regardless of the initial velocity, a randomly drawn final velocity from the fitted distribution would be a valid final velocity. The resulting gamma distribution acquired through the GRG algorithm can be found in table 3.15.

Source:	Number of entries:	Shape:	Scale:	Goodness of fit:
GPS	4985	1.887042698	4.278322934	0.045303461

Table 3.15: *The resulting information acquired through the fitting of the Final velocity to a gamma distribution.*

Once the onset point and the final velocity are determined, the only thing left to find is the course of this action. Looking at the data available from the processed GPS data, it could be seen that there is a correlation between the final DUE and the distance to the stopping point at the end of the braking action.

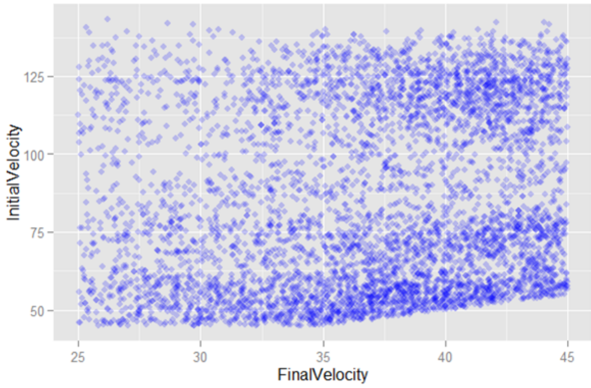


Figure 3.15: *On the x-axes you can see the final velocity of a braking action towards a stop. On the y-axes you can see the initial velocity of this braking action. The edge visible on the bottom right is due to the minimal velocity difference set for the selected braking actions, namely that of 10km/h. The more dense bandwidths across the x-axes are due to the frequency of occurrence of different braking actions, showing similarities with existing speed limits.*

This distance to the stopping point can be used to determine the position of the end of the braking action, and in turn the course of this action. The correlation between the final DUE and this distance measure can be seen in Figure 3.16. In order to formalize this correlation, I used linear regression to come to equation 3.3, with R^2 at 0.8767.

$$Distance to stopping point = 10.811 * FinalDUE + 47.09 \tag{3.3}$$

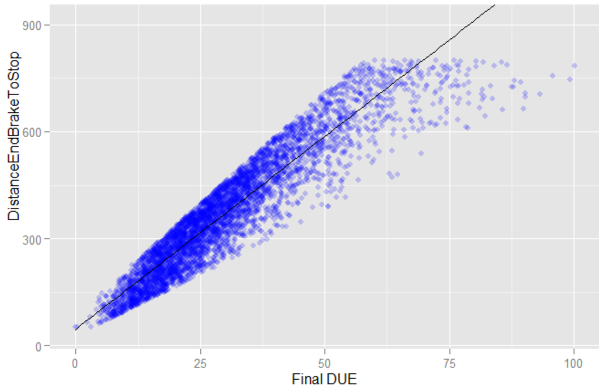


Figure 3.16: *On the x-axes you can see the final DUE of a braking action towards a stop. On the y-axes you can see the distance to the stopping point of the end of this braking action. The black line indicates the fitted function.*

In order to get the final DUE I fitted a shifted distribution to the *differences found between the initial and final DUE*. The reason for choosing the difference in DUE to get the final DUE was that the difference in DUE did not have a clear correlation with any other attributes that can be known at the onset of the action. The relation between the initial DUE and the difference in DUE can be seen in Figure 3.17. The resulting gamma distribution acquired through the GRG algorithm can be seen in table 3.16.

Source:	Number of entries:	Shape:	Scale:	Goodness of fit:
GPS	4985	9.989854391	2.894064959	0.030581005

Table 3.16: *The resulting information acquired through the fitting of the difference in DUE between the initial and final DUE of a braking curve to a gamma distribution.*

After having acquired the difference in DUE, the final DUE can be calculated through subtracting the difference from the initial, as seen in equation 3.4.

$$FinalDUE = InitialDUE - DifferenceDUE \tag{3.4}$$

This can then in turn be used to calculate the distance using Equation 3.3. Using the distributed final velocity, and the just calculated end position of the braking manoeuvre the final DUE can be re-calculated. This then gives you a similar relation between the final DUE and the position of the end of the braking curve, as seen in Figure 3.18.

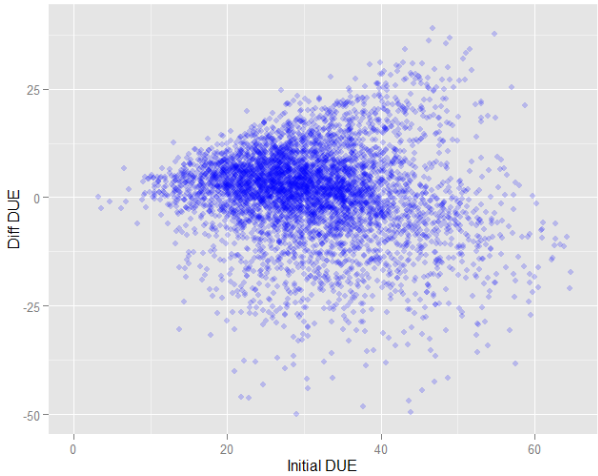


Figure 3.17: *On the x-axes you can see the initial DUE of a braking action towards a stop. On the y-axes you can see the difference in the DUE. The sharp border seen on top of this cluster is due to the final DUE not going below 0. There is a slight relation visible between the initial and difference in DUE.*

Situation: Approaching a stop. Braking to a stop

Once the first braking manoeuvre has been completed and a velocity between the 25-45km/h has been reached, the second braking curve still needs to be specified. Unlike previous actions, only the onset point needs to be formalized. This due to the stopping position already being specified as a specific position and velocity, namely the stopping position dictated by FRISO and a velocity of 0m/s. With this stopping point and the onset point, the course of the action can be acquired through equation 3.1 and 3.2.

In order to acquire an onset position the current velocity, acceleration and distance to the stopping point were used in combination with an equation. These data points were acquired from the onset point of braking manoeuvres to a stop from the GPS data. The equation that was used expressed the correlation found between the onset velocity and the onset distance to the stopping point. This equation was acquired through using linear regression on the processed GPS data, and can be seen in Equation 3.5. The R^2 of this equation is 0.4959, which low value was likely caused due to the high variation as visible in 3.19. The main reason for using the current velocity, acceleration and the

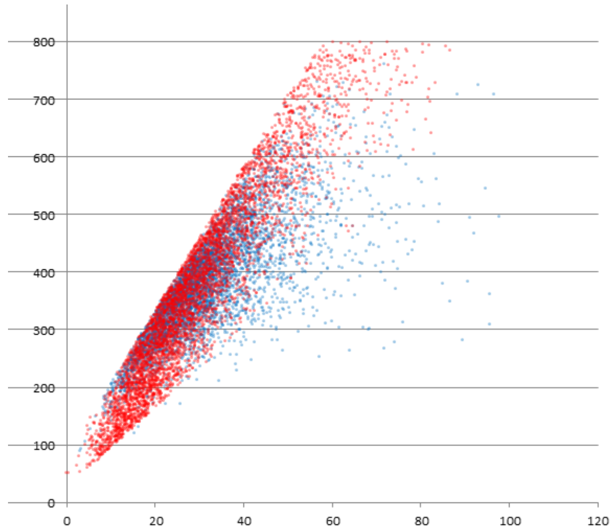


Figure 3.18: *On the x-axes you can see the final DUE of a braking action towards a stop. On the y-axes you can see the distance to the stopping point at the end of this braking action. The red points indicate observed entries from GPS data. The blue points indicate a set of simulated entries acquired through the distributions and functions described above.*

distance was to ensure that the braking manoeuvre fits with the current circumstances, which would not be the case with a single distribution over the position.

$$Distancetothestoppingpoint = 6.7139 * Onsetvelocity - 104.51 \tag{3.5}$$

The missing value within this equation is now the onset velocity. The onset velocity of the braking curve can also be described through combining an equation of motion, if you reason from the current state of the train and calculate the onset of the braking action that lies ahead. When combining Equation 3.6 with the just presented Equation 3.5 a resulting equation can be seen in Equation 3.7. Note that the x value within Equation 3.7 is the distance from the current train position to the stopping position, and thus through subtracting the distance acquired through 3.5, get the distance between the current position of the train and the onset of the braking curve. Here v_f is this the onset velocity of the braking curve.

$$v_f = \sqrt{v_c^2 + 2a_c d} \tag{3.6}$$

Where v_f is the final velocity, v_c is the current velocity, a_c is the constant acceleration and d is the displacement.

$$v_f = \sqrt{v_c^2 + 2a_c(x - (6.7139 v_f - 104.51)} \tag{3.7}$$

Where v_f is the final velocity, v_c is the current velocity, a_c is the constant acceleration and x is the current distance to the stopping point.

Equation 3.7 can be re-written in order to get the v_f, that indicates the onset point of the braking action, from equation 3.5 to the left side of the equal sign. This results in equation 3.8 [87].

$$v_f = \frac{\sqrt{4,507,645,321 a_c^2 + 200,000,000 ax + 20,902,000,000 a + 100,000,000 v_c^2} - 67,139 a}{10,000} \tag{3.8}$$

Where v_f is the final velocity, v_c is the current velocity, a_c is the constant acceleration and x is the current distance to the stopping point.

With equation 3.8, we can now calculate the onset velocity of the final braking manoeuvre, which in turn can be used to calculate the position of this action through equation 3.5.

Equation 3.5 has the problem of not having any variation between the relation of these attributes, also visible within its R^2 value, resulting in a linearity in the results. In order to represent the variation found in the braking behaviour of real train drivers, a distribution was used to offset the resulting distance. This distribution was acquired through fitting a normal distribution to the difference in velocities found between the GPS data and equation 3.5. This distribution was acquired through *fitdistr* and can be found in table 3.17. For implementation purposes, two equations were created manually in order to prevent the resulting distance to be offset by too much. These two equations and function 5 can be seen in relation to the data in Figure 3.19.

Source:	Number of entries:	Mean:	Standard deviation:	Goodness of fit:
GPS	5442	0.004782357	34.74332178	0.04009126

Table 3.17: *The resulting information acquired through the fitting of the difference in DUE between the initial and final DUE of a braking curve to a gamma distribution.*

After adding that offset distance we have the position where the train should start braking, given the current velocity, acceleration and distance to stopping point. The relation between the onset DUE and the distance to the stopping point, and between the onset velocity and the distance to the stopping point, has been kept similar through the use of this method. This can be seen in Figure 3.20 and 3.21.

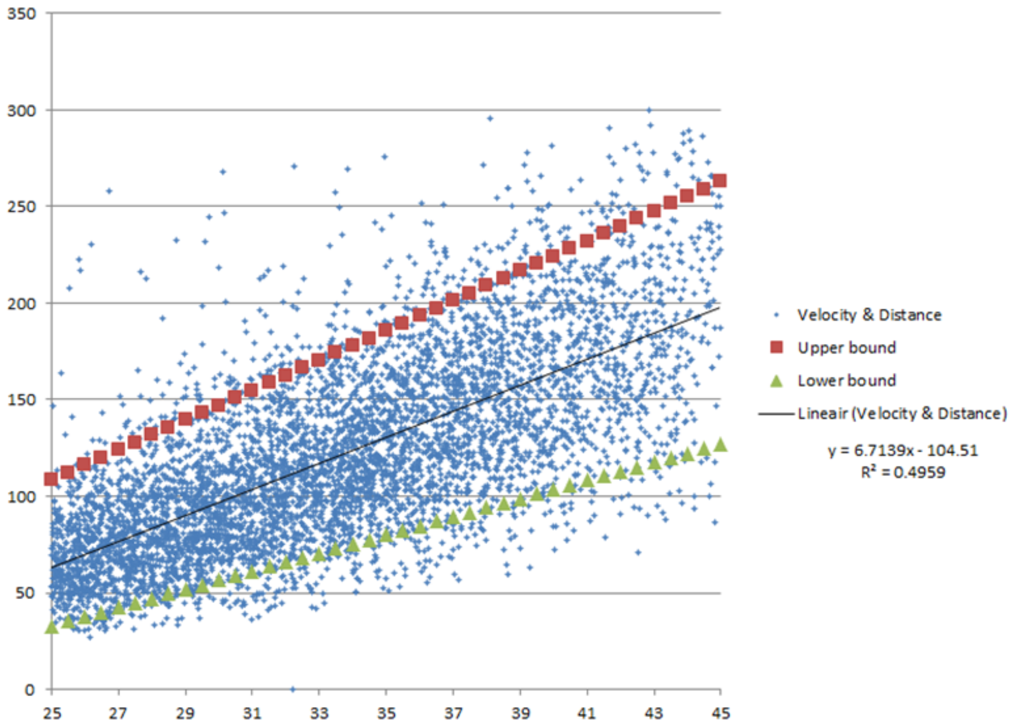


Figure 3.19: *On the x-axes you can see the initial velocity of a braking action towards a stop. On the y-axes you can see the distance to the stopping point at the onset of this action.*

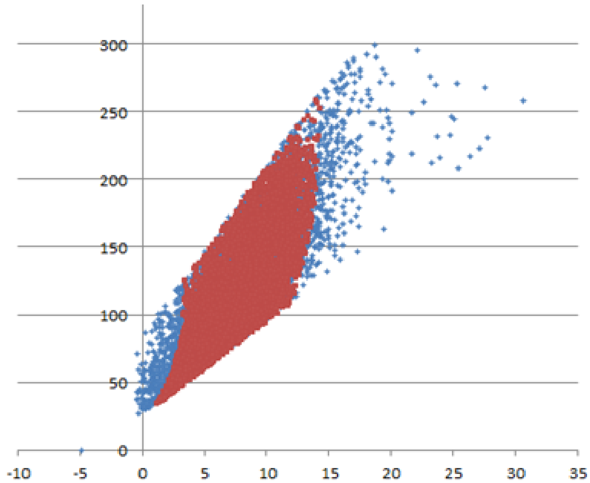


Figure 3.20: *On the x-axes you can see the initial DUE of a braking action towards a stop. On the y-axes you can see the distance to the stopping point at the onset of this action. The data points acquired from the GPS data can be seen in blue. Simulated data points, which were acquired through the distributions and functions described above, can be seen in red. The sharp borders between the red and blue dots is due to the Upper and Lower bound.*

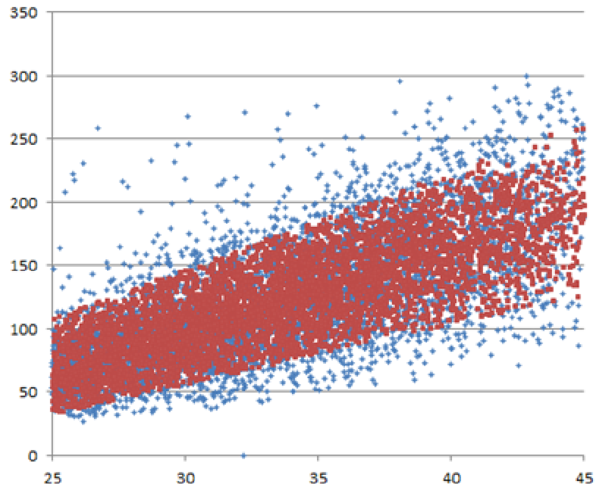


Figure 3.21: *On the x-axes you can see the initial velocity of a braking action towards a stop. On the y-axes you can see the distance to the stopping point at the onset of this action. The data points acquired from the GPS data can be seen in blue. Simulated data points, which were acquired through the distributions and functions described above, can be seen in red.*

3.4.3.1 Situation: Approaching a red signal

For approaching a red signal, the same type of distributions and equations were used as for the previous situation, situation 3, with the exception of the initial DUE check. This check was not deemed necessary since situation 4 does not use the scout sign. The resulting distributions, corresponding to the use of those of situation 3, can be seen in table 3.18. The resulting equations can be seen in equations 3.9-3.11.

Source:	Use:	Type:	Number of entries:	Shape:	Scale:	Goodness of fit:
GPS	Final Velocity	gamma	296	2.317	7.025	0.022
GPS	Difference in DUE	gamma	273	13.697	5.764	0.0270
Source:	Use:	Type:	Number of entries:	Mean:	Std dev:	Goodness of fit:
GPS	Distance offset	normal	112	0.009	202.501	0.114

Table 3.18: *The resulting information acquired through the fitting of the different distributions for Approaching a red signal*

$$Distancetostoppingpoint = 7.3813 * FinalDUE + 159.94 \tag{3.9}$$

equivalent to the use of equation 3, with an R2 of 0.55.

$$Distancetothestoppingpoint = 13.177 * Onsetvelocity - 102.06 \tag{3.10}$$

equivalent to the use of equation 5, with an R2 of 0.8489.

$$v_f = \frac{\sqrt{173{,}633{,}329a_c^2 + 200{,}000{,}000ax + 20{,}412{,}000{,}000a + 100{,}000{,}000v_c^2} - 13{,}177a}{10{,}000} \tag{3.11}$$

Where v_f is the final velocity, v_c is the current velocity, a_c is the constant acceleration and x is the current distance to the stopping point.

3.4.4 Behaviour variations

After having formalized the different aspects within the possible actions within the different situations, one main question that still needs to be answered is: *If the modelled train driver is coasting or giving throttle within situation 2, will it do an action or will it do nothing?* Within the classification trees it was visible that the current action attribute was not always a very good indicator of whether to do an action within this situation. In this section GPS data will be used to look at the free-track behaviour shown by train drivers, in order to answer this question.

A number of examples of the driving behaviour shown by train drivers can be found in Figures .1 to .4 in Appendix 4. From these the following three behaviours were categorized:

1. Cruise.
2. Coast.
3. Alternating coasting & traction.

Given this classification and the observation that these were often combined by train drivers in free track sections, like in Figure .3, three ways of combining them will be used.

The first way followed the guidelines of the *Universal Economical driving Idea* (UEI) method. This method is taught by the NS to train drivers with the aim of minimizing the energy usage while still driving on time. The rules of the UEI method can be summarised as advising the train driver to speed up to a certain velocity (usually the speed limit), after which the train driver is advised to start coasting at a certain time after departure, based in the driving time between the stations. This often results in a driving behaviour where large sections of coasting are present, similar to (2) within Figure 3.22.

The second way combined the cruising and coasting behaviours, each specified through a percentage indicating with the distance of the free track section where this behaviour will be shown. Starting with cruising, followed by coasting. The percentage of this distance that would be used to cruise or coast was acquired through the distribution of these percentages visible within the GPS data. A normal distribution was chosen for this purpose. It must be noted that it was difficult to automatically assess accurately whether or not a train was coasting or cruising, seeing that the acceleration data fluctuated significantly even after smoothing. On top of that was the observation that there were at times seemingly rapid switches between coasting and cruising,

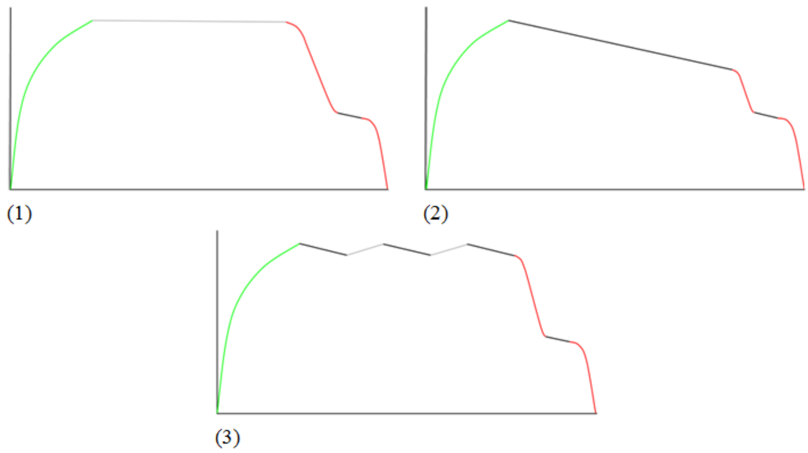


Figure 3.22: *An overview of the three categorized free track driving behaviours based on GPS data. On the x-axes the distance travelled is represented and on the y-axes the velocity of the train. In (1) the train driver cruises in-between the departure, indicated in green, and arrival sections, indicated in red. In (2) the train driver coasts in-between the departure and arrival sections. In (3) the train driver alternates between coasting and accelerating between the departure and arrival sections.*

which made the automatic categorization process more difficult. This could be not only due to the train drivers actions, but also due to things such as wind, curves in the track or slopes.

The third way used an alternation between traction and coasting, similair to (3) within Figure 3.22. The velocity differences between these actions was based on GPS data. Within the train driver model, the frequency of occurrence for each of these three free track behaviours could be specified with a percentage.

3.5 Overview decision making model

From the findings within the previous sections, the main part of the decision making model was made. The input of this model would be the *situation* where the train finds itself in combined with the beliefs about the environment. The term belief used here refers to the same term as explained in Chapter 2. Depending on the *situation* value a small decision tree would be traversed. One example of this can be seen in Figure 3.23. These decision trees were based on the classification trees specified earlier within this chapter. Within Figure 3.24 the overall model can be seen. Within this model, based on the situation and current action, an action kind will be selected. This selection is indicated in with blue arrows between the *current action* condition and the allowed action. Besides the classifications acquired through the classification trees, the decision trees also hold the exception cases which are indicated with red arrows. These exception cases are only triggered in the following circumstances:

- **Situation 1:** Approaching signal/sign lower speed:
  - $Braking \rightarrow Braking$: If new information indicates that the train is not decelerating hard enough, the agent will adjust the braking lever position.
- **Situation 2:** Approaching signal/sign higher speed
  - $Traction \rightarrow Traction$: If the agent is trying to maintain the velocity of the train, through cruising, but the current acceleration is above 0, it will lower the position of the traction lever.
- **Situation 3 & 4:** Approaching a planned stop or a red signal

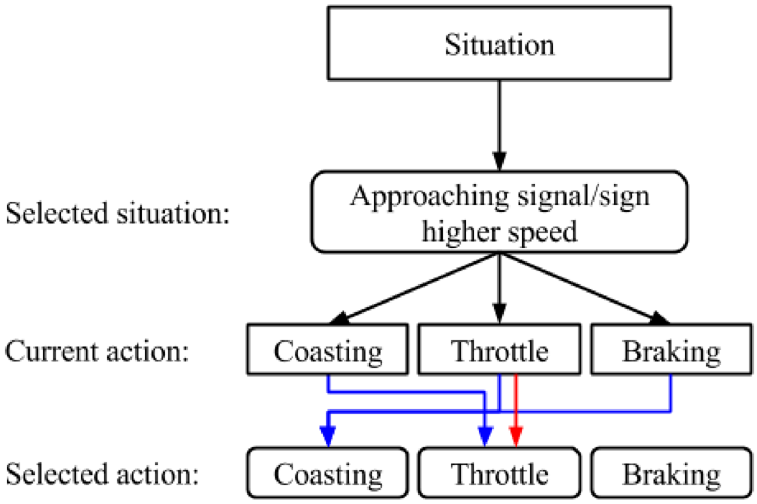


Figure 3.23: *A decision tree example, where the input is the situation Approaching signal/sign higher speed. The blue arrows indicate the allowed actions acquired through the classification trees. The red arrow indicates an exception case*

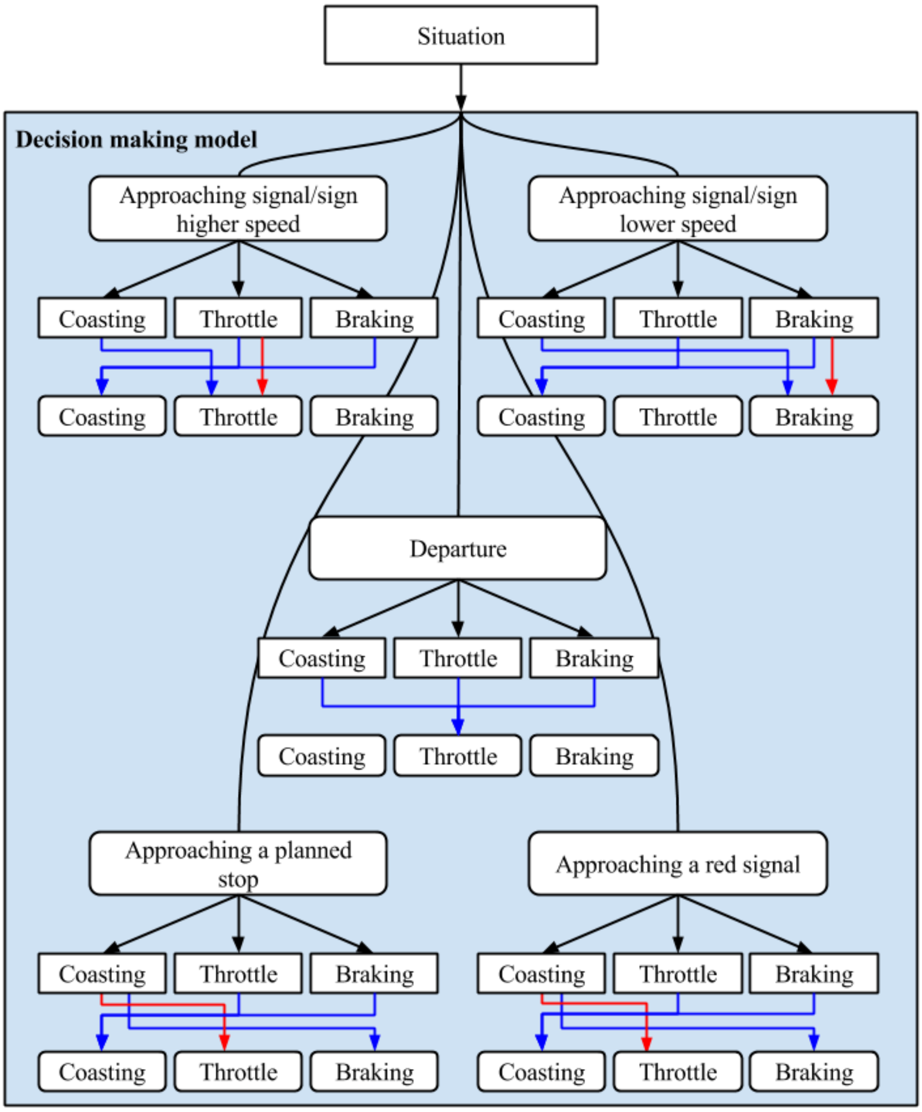


Figure 3.24: *Within this figure an overview the modelled decision making process can be seen. As main input the current situation the modelled train driver finds itself in, followed by a categorization of the current action. After this, the arrows point to the possible actions that can be done given the current action.*

– $Coast \rightarrow Traction$: If you are still far away (>300-500m) from the stopping point, and your velocity is low (<3-10m/s) the agent can accelerate.

It is important to note that the specified possible action of the decision tree will not always be selected. This is mainly due to the fact that the classification trees were learnt mostly based around the actions that were performed. Within the simulation, the agent will receive events also at times that no action is deemed necessary. One example is in the situation that the train is approaching a planned stop and the agent is already braking. If the agent has already started the braking manoeuvre because of the planned stop, it will not switch to coasting, but will continue to finish this braking action. Thus not doing any action. If the braking action was not done because of the approach of a planned stop, the agent will stop that braking manoeuvre and start to coast. Due to FRISO indicating immediately that the coasting action has been executed, the agent will now find itself in the situation of approaching a planned stop while coasting, and can plan the braking action accordingly.

Within situation 2 the selection process was also influenced by the observations made. So will the agent not interrupt the traction actions done during the initial acceleration after a departure until a velocity around the speed limit has been reached. On top of that, the agent will use the three free-track behaviours that were presented in the last section within the action selection process of situation 2.

If the agent does decide to do an action, the four attributes for this action will be selected. The onset position, the braking & traction lever position and the final velocity will be determined. This is done if possible through the use of the distributions, functions and values specified within this chapter.

3.6 Conclusion

Within this chapter an overview is given of the data and processing done within this project, followed by the machine learning and statistical fitting methods used in order to create the structure of a train driver decision making model. This is to answer the question of how to model train driver behaviour from data. This model was also created with the goal of being implementable within an agent while being able to represent the variations found within the driving behaviour of real train drivers. In the following chapter I will go into using the findings of this chapter to help with the design of the software agent.

4 Agent design

In the previous chapter I have given an overview of the data processing and decision making models. In this chapter I will go into the agent setup that will be used as the basis of the agent implementation of the train driver model within FRISO. I will first give an outline of the setup with FRISO, after which I will go into the design choices made and finish this chapter by giving an overview of the agent model.

4.1 Setup with FRISO, input & output

The agent design was done in different stages. At first the interaction with the environment through FRISO was specified by formalizing the input and output that would go between the agent and FRISO. Secondly the internal structure of the agent was determined. The internal structure here refers to the overall process of getting from input to output. Within this section I will give an overview of the input and output characteristics.

In order for the agents to work within FRISO the input and output characteristics had to be specified to work in concert with FRISO. Before specifying these characteristics, it was important to determine the way the agents would be implemented to work with FRISO. It was decided that the agent model would be programmed within a dynamic linked library (DLL). The main reasons for choosing this rather than implementing it directly into FRISO were:

1. Reusability by ProRail.
2. Expandability.
3. The programming language within FRISO, 4DScript, not being a generally well-known language.

On top of that, it was deemed beneficial that the programming language for the DLL would be C++, which is a generally well known programming language. C++ is also an unmanaged programming language, unlike 4DScript, which means that the code does not have to be compiled at runtime, which offers a speed benefit.

For the transition of information between the DLL and FRISO there were two options:

1. Continuous access.
2. Event based information packages.

Continuous access would mean that the agent would need to monitor the required information within FRISO in order to act and send information to FRISO at the appropriate times. This was deemed unnecessary and unpractical, due to it requiring a larger change to the workings of FRISO , it impacting performance if it was done continuously and the possibility of doing it within events on FRISOs side.

After determining that the transmission of information would be done through events, the link between FRISO and the agents could be defined. In the simulation each agent is to model a different train driver that controls an individual train, thus requiring one agent per train. Seeing that the DLL was linked to FRISO once at the start of each simulation run, all the agents had to be situated within this DLL. In order to facilitate the communication between FRISO and the agents, a single agent hub was defined to serve as a conduit for these messages. These messages came in two main categories:

1. Setup information.
2. Messages to and from agents.

Setup information messages either contained the settings of the current scenario, the time table for all the trains or a declaration that a train has been placed within the model. A more detailed overview of the contents of these messages can be found in Appendix 5.

Messages to the agents have the function to convey perceptual information about the environment for the agent. This information needed to be sent to the agent each time that something significant changed within the environment. Something was deemed a significant change in one of the following cases:

1. **A timetable control area (TCA) was passed, signifying that the train passed a scheduled point on its timetable:** *A train driver can normally see its timetable and determines where he is on the route based on the environment. The time indicated on these points serves as an indication whether or not the train driver is still on schedule.*

2. **The train is allowed to depart:** *Normally a train driver gets a visual signal in the train cab when all doors have closed and the train is allowed to depart, or if the departure assistant indicates the train can depart. After one of these, the train is allowed to depart only if the signal in front of the train allows this departure, as in, that the first signal is not red. Within FRISO this procedure is combined into one message to indicate the train is allowed to depart.*

3. **The stopping signal has been determined:** *The stopping signal is the signal where the train is supposed to stop at for a planned stop. Normally this signal is not used for determining the stopping position, instead a blue number sign positioned next to a platform is used to determine this. These are however not present within FRISO. Due to the variable nature of the stopping platform, this information is sent to the agent during the simulation in the form of a stopping signal.*

4. **A signal aspect of a signal in front of the train has improved:** *FRISO keeps the agent informed of the one to three signals in front of the train. If for one of these the signal aspect improves this is conveyed through this message. Normally this is a change that the train driver can visually observe.*

5. **ATP improved:** *Although not strictly necessary due to the supportive nature of the ATP system, FRISO will send a signal if the ATP improves due to a signal aspect improvement.*

6. **The train has passed a signal:** *This message indicates that the head of the train has passed a signal.*

7. **The train has passed a speed sign:** *This message indicates that the head of the train has passed a speed sign.*

8. **The train has passed a switch:** *This message indicates that the head of the train has passed a switch. This is relevant due to the rule that a train driver is not allowed to start accelerating for a speed sign or signal before*

passing it, if the train has not yet fully passed all the switches between the train and the speed sign or signal.

As mentioned within the previous chapter, the default onset position of an acceleration action is set at an arbitrary 10cm in front of the train. One exception to this can be seen within the 8th case mentioned above. The reason for both the default onset position and the exception case when switches are involved, are due to the signal improvement rule [76]. This rule specifies that when a train driver sees a signal improve its aspect to one that allows a higher velocity, he needs to maintain the current speed limit until the train has fully passed the signal. However, the train driver is allowed to start accelerating directly if he:

- Sees the signal during the day with good sight.
- Is not driving on sight. *A train driver drives on sight only in rare cases, after passing a signal where there is a possibility that the train needs to be able to stop at any moment. Within FRISO this only occurs if the simulation scenario is explicitly set to ask for such an occurrence.*
- The ATP has also improved. *This is relevant for when the signal aspect improves when the signal is still out of sight.*
- All switches have been passed.
- The speed signs allow a higher velocity.

Assuming that the simulations will be situated during the day with good sight, it thus becomes the exception that the train driver is not allowed to speed up directly after observing a signal improvement.

In order to limit the size of the messages, not all available information will be sent within each message, instead only significant information is sent within each message. In most of the cases this entails the aspects of the environment which have changed.

The only input missing now is the confirmation that an action has been completed. In the last chapter actions were defined as starting at a certain position, having a certain course and ending at a certain velocity. If an agent has given such an action for a train, and it is completed, FRISO will send a conformation that the desired velocity has been reached:

1. **The desired velocity has been reached:** *Here FRISO informs the agent that the previously given action has been completed and the desired velocity has been reached.*

If the agent wants to request information about the environment, specifically the state of the train such as the velocity and acceleration, it can do this through a *position action*. This then serves as a subscription to a perception event. This *position action* entails that FRISO will send information to the agent once the train has reached a specific position:

2. **The desired position has been reached:** *Here FRISO informs the agent that the position of the previously given sensory update request has been reached.*

A full list of the attributes that are contained in each input message can be found in Appendix 7.

Output messages sent from the agent to FRISO serve to convey the action that the agent wants the train to do, or that the agent wants to subscribe to a perception event. The action messages contain the following information:

1. Position break lever.

2. Position traction lever.

3. Desired end speed.

4. Position start speed assignment.

It is assumed that the agent will never combine the break and traction lever. The *position action* contains only one attribute of information, namely a desired position. The agent has a maximum of one action and one position action that it can do at a time, with newer actions always taking precedence. This means that if an action or position action are sent and there is already an action planned or being executed at that time, that action will be stopped and replaced by the new action. The agent can also choose to do no new action. Figure 4.1 shows an example of the message flow between FRISO, the agent hub and one agent.

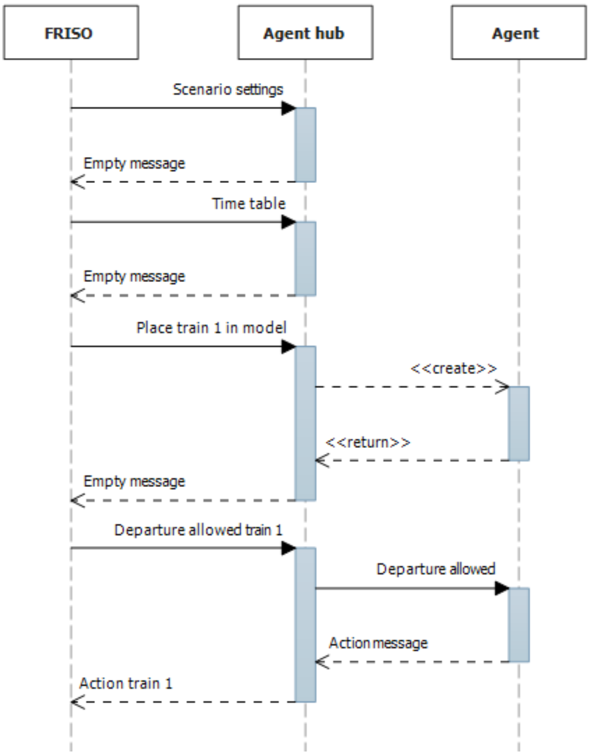


Figure 4.1: *A schematic overview of the message passing between FRISO, the agent hub and the agent. The empty messages serve as a confirmation that a message has been processed.*

4.2 Agent hub

Given that the DLL was to serve as a single entity from FRISO's point of view, giving instructions to the trains when FRISO gives updates about their situation, the DLL itself would need to serve as the base for all the agents. As mentioned before, the agent hub serves to facilitate this process by passing on the messages to the correct agents. Within this section I will give a more in depth view of the agent hub.

The agent hubs main function is to pass the messages to the correct agents. This is done through keeping a record of which train ID corresponds with which agent. Through this using this record an incoming message designated to belong to a specific train can then be forwarded to the corresponding agent. This agent, after having made a decision of the actions it will do, will send a reply back to the agent hub through the same channel as the incoming message. This message is then passed back to FRISO through the same channel as the initial incoming message. The term 'same channel' denotes that the process will wait for a reply after sending a message.

Besides passing messages, the agent hub stores the settings of the current scenario and stores, creates and deletes agents when needed. Agents are created when the agent hub receives the message that a train has been placed within the model. Using the information within this message, combined with the relevant information within the time table and settings, an agent is initialized. An overview of these functions can be seen in Figure 4.2. Agents are deleted by the agent hub through checking after each message that indicates a TCA was passed for a specific train, if all the points on the time table of that agent have been passed. If this is the case, the agent is deleted.

4.3 Agent design motivation

Before giving an overview of the agents, I will outline the design choices. To reiterate: this software agent needs to be able to drive a train within FRISO according to the train driver model. The goal of this agent was also to improve the predictions done through simulations on the expected effects of adaptations within the infrastructure, time table and innovative measures within safety and control.

Two main limitations and interactions played a role within the designing process of the internal structure of the agents that fulfil these goals. One be-

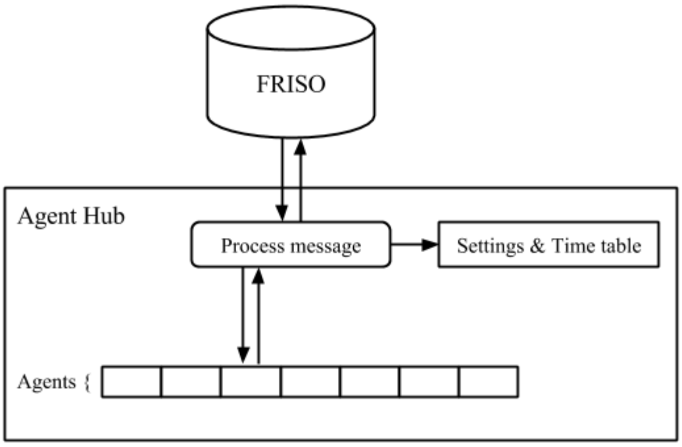


Figure 4.2: *A schematic diagram of the agent hub*

ing the limitations coming forth from the implementation of the agent within a software DLL that interacts with FRISO. The second coming from the railway domain and the train driver decision model.

Given that the agent should be able to drive the train in accordance to the train driver decision model, the capabilities of this agent should be:

1. Acquire sensory input of the environment.
2. Process and determine what the acquired input information signifies.
3. Use this acquired information to apply the decision rules to come to an action.
4. Respond to FRISO with the resulting actions.

The first and last capability have already been discussed at the start of this chapter. For the act of processing the input information I will first go into the internal representation of this information. For this knowledge representation the concept of belief was used, this concept was briefly outlined as part of the BDI paradigm within Chapter 2. Paradigms like this, that use intentional explanations, lean itself to describe complex systems in an intuitive way [88]. Within this project, the concept of belief was used with the aim of representing that the agent holds its own internal state of the environment, which does not necessarily need to correspond with the actual environment. The aspect of believing that the environment is in a certain state while it is actually in a different state often plays a visible role within the actions of train drivers. This is especially so when a train driver is driving towards a signal they expect to be of a certain signal aspect, while the signal aspect has improved before it being visible. This can be seen in Figures 3.4 and 3.5 of Chapter 3.

Often a logic based approach is used when representing and reasoning about internal notions such as beliefs [88, 71]. However, due to the requirement to implement the agent within a DLL written in C++ a different approach was taken. C++ is an imperative object oriented programming language, meaning that a symbolic representation through logic is not readily available. This means that the deliberation that is necessary when dealing with a logic based representation cannot natively be done within C++. For this reason, any necessary 'deliberation' processes about the beliefs will be implemented as functions. One example of this is determining the expected signal after signal X. Say signal X is a normal yellow signal, the expected signal to come after this is a red signal. A logic based representation of this could be:

$$expectedSignal(signal(yellow), red)$$

And the functional version could be:

Result: Returns the expected colour of the signal after *signal*
Input: *signal*
if *signal.colour* == *yellow* **then**
  Return red
end

Algorithm 1: Example function: *ExpectedSignal*

The decision was made to take a reactive approach for the action selection process, instead of using a planning system such as STRIPS [27]. The main reason for this was that planning the actions in relation to each other can only be useful if the modelled train driver knows the route very well and can estimate when he will meet certain signals and speed signs. He could then reason about these beliefs and use this knowledge of the tracks to plan his actions.

One example of this can be described as follows: The train departs and accelerates towards 130km/h on a section where 140km/h is allowed, this because the train driver knows that a signal about halfway to the next stop will be on yellow due to a scheduled crossing train. He knows that if he accelerates to 140km/h he will have to brake for that yellow signal, but if he just accelerates towards 130km/h the yellow signal will improve to green just before he reaches it. Thus ensuring that he does not need to brake meaning that he saves energy and maintains a higher velocity after that signal which will make up, or even gain, for the time lost by initially accelerating towards 130km/h instead of 140km/h.

This whole situation assumes an intimate knowledge of the track in question, the time tables and the locations of the nearby trains. Modelling all of this was beyond the scope of this project and not always possible with the current version of FRISO. As such, the other two parts of the BDI paradigm, desire and intention, which are often used for agents to plan a set of actions to achieve a goal will not be explicitly used. Instead, the intentions and desires that come with driving a train will be considered intrinsic within the reactive approach.

One example of a such a reactive approach can be seen in PENGI from Arge and Chapman [11]. [88] describes this approach as following from the idea that *"most decisions are routine, and can be coded in a low level structure, which only needs periodic updating perhaps to handle new kinds of problems"*. Which is a description that fits with the act of driving a train within this simulation setting, taking decisions which are of a narrow scope, dictated by the rules and goals of driving a train.

The resulting low level structure can thus be described as a large set of input- output conditions, or as a rule-set. The goals, desires and intentions can be said to be intrinsic into the structure and content of the rule set. Implicitly containing the goals of driving in a safe, timely and energy efficient manner.

Such a reactive method was also described by Brooks [8], where different layers of behaviour worked together, with in general lower levels representing more less abstract behaviours, and higher levels more abstract behaviours. Such a layering structure can be found in Figure 4.3. A similar approach was taken in the agent design in cases where a clear action, or no action was required.

The resulting agent design is similar in structure as a model based reflex agent [73], with the train drivers goals becoming intrinsic in the course of processing perceptions and selecting an appropriate action.

Due to the use of general situations within the decision making model, future research into the effects of adaptations to security measures and/or train operation developments could be done in two ways. This is assuming that the core functionality of the railway network does not change. The core functionality refers to main concepts within the railway network, such as the presence

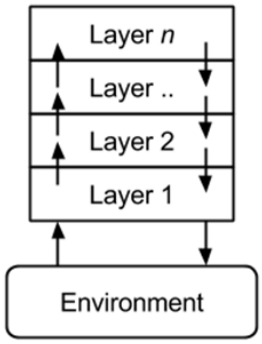


Figure 4.3: *A vertical layering*

of signals, signs and that train drivers do not choose which tracks to ride on. If the adaptation has to do with the decision making process itself of the train driver, the low level rules for each situation could be changed to exhibit the desired behaviour, which makes it possible to simulate a hypothesised end result. The use of this would be to see the effects of this adaptation if it were to bring about the desired behaviour. One example of this is giving an indication of an 'advised speed' to the train driver to follow. For this, only a new *free track behaviour* would need to be defined.

The other case would be where the adaptation does not influence the decision making process directly, but indirectly. For this case it would hold as well, that ProRail could simulate the effects of this hypothetical situation if it is assumed that train drivers will still drive in a similar way as the train driver model. One example of this is shortening the distances between the signals.

It is not possible to simulate what will happen if an adaptation is implemented, it is only possible to see the effects of an adaptation if it is assumed that adaptation is thought to have a specific effect. One example of this could be described as follows: Given the observed variation in acceleration behaviour, the question is asked if a support system of some kind that would reduce this variation, would also lower the resulting variation in driving times by amount X. It could then be simulated what the effects could be of this reduced variation before setting up a project to bring such a change about.

4.4 Agent setup

In the previous section I have given an overview of the overall agent design. Within this section I will formalize this to give the agent model.

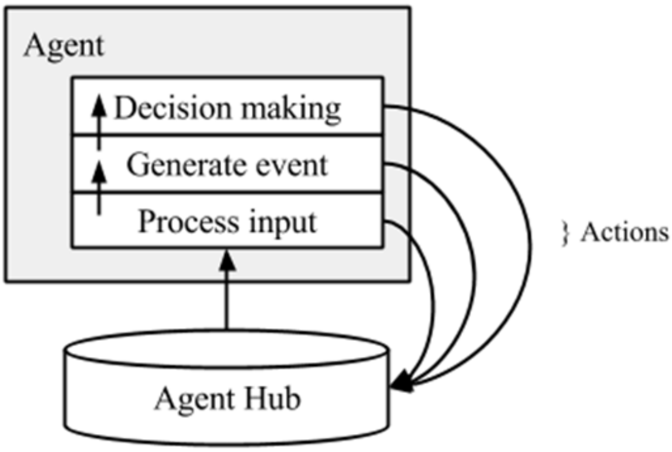


Figure 4.4: *A general overview of the agent structure and the processing flow.*

Putting together the concepts of a layered architecture, the capabilities required from the agent, and the event based concept from 2APL [17], which uses events to carry information about changes that might make an agent act. The general structure could be represented as seen in Figure 4.4.

A more detailed overview that also incorporates the aspects of beliefs, can be found in Figure 4.5. This overview presents the different components within the agent that will help facilitate the three tasks that can be seen in Figure 4.4. These components are:

1. **Agent communicator:** *The main functions of this component is to process the input and format the output.*

2. **Event processing:** *The main function of this component is to see if the environment has changed in such a way to warrant a possible action to be undertaken.*

3. **Decision making:** *The main function of this component is to apply the rules from the decision making model to see if an action needs to be undertaken given the current situation, and if so, to create such an action.*

4. **Belief Base:** *This component serves as a storage place for the information required by the agent in order to function. This includes the information and beliefs about the current environment.*

These components interact with each other in the following way:

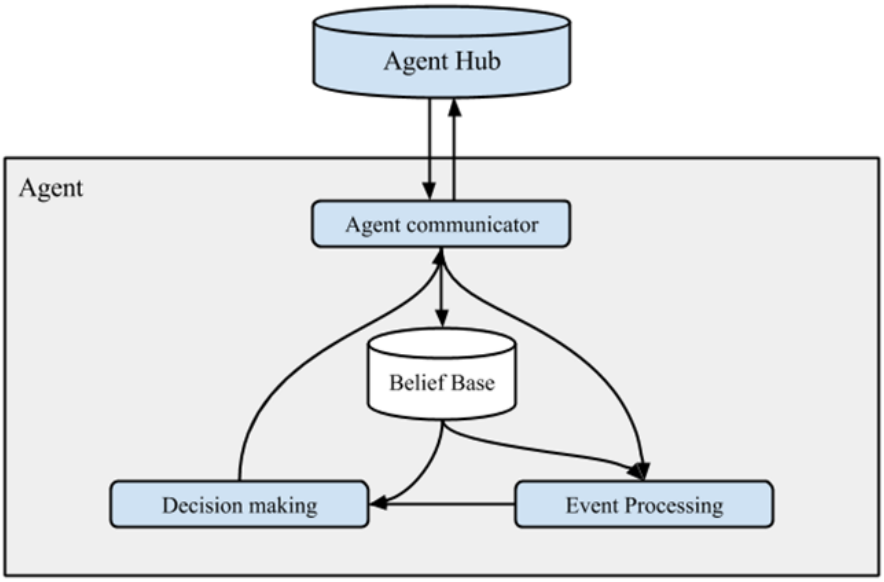


Figure 4.5: *An overview of the components within the Agent.*

1. **Agent communicator:**
   - **Input & Processing:** *Once a message has been received and processed, the resulting information is incorporated within the belief base. After this the message type is passed to the Event processing component. This component can also receive actions from other components.*
   - **Output:** *If this component receives an action from a component, it can format this into a message which then can be send to the Agent Hub.*
2. **Event Processing:**
   - **Input & Processing:** *Given a message type, certain conditions are checked to see if the environment has changed significantly in a way that could possibly warrant an action. One example being: If the agent gets the message that a signal aspect has improved, a check is made if this signal aspect visible for the agent, if this is not visible and would not give an ATP signal, it is deemed as not a significant change.*
   - **Output:** *If the environmental change is deemed significant, the current situation (as described in Chapter 3) of the train is determined*

and passed on to the Decision making component. If a change is deemed not significant, an empty action is send back to the Agent communicator component.

3. **Decision making:**
   - **Input & Processing:** *Given a situation, the rules from the decision making model are used to come to a specific action. It is also possible that no action will be done, in which case an empty action will be returned.*
   - **Output:** *The output actions are passed to the Agent communicator component.*

For both the Agent communicator and the Event processing component, there area handful or rare cases where they can create a default action themselves and then pass it to the output of the Agent communicator. One example of such a default action is the action to change nothing when the agent receives a message that a stop signal has been determined. The reason for always returning a non-action is because the determining of a stop signal can never influence anything within the observable environment of the agent. A schematic overview of these components can be found in Appendix 8.

The belief base, which is not considered an interactive component, is made up of various variables and data structures used to represent concepts within the agents mind. On top of this are a number of functions that can use the input from the Agent communicator to determine a variety of concepts. One of these functions determines the expected aspect of a signal that is currently out of sight. These functions fall under the processing of the input gathered from the messages. The beliefs that are acquired can then be used by the other components.

4.5 conclusion

Within this chapter the findings of Chapter 3 are combined with concepts often found within agent literature, in order to come to an agent model that can be implemented to work with FRISO. In the following chapter I will use this model and the findings from Chapter 3 to implement a DLL that can be used with FRISO.

5 Implementation

In the previous chapter the agent setup was formalized. The next step is to use this setup and the findings of the previous chapters to implement a train driving software agent. In this chapter I will go over the implementation of the agent setup within the DLL that will be linked to FRISO. I will first give an overview of the implemented components, followed by an overview of the assumptions that were made during this implementation process. I will conclude this chapter by giving an overview of the difficulties that were encountered during the implementation process and present the final implemented structure.

5.1 Components

In this section I will go over the implementation of components that were described in the previous chapter, starting with the agent hub. Due to the DLL being written in C++, an object oriented approach was taken when implementing the different components. At the end of this section an overview of these components can be seen in a class diagram in Figure 5.1.

5.1.1 Agent hub

Besides the functionalities discussed in the previous chapter, such as storing instances of the agents, the implemented agent hub stores a number of other objects. Most notably, the agent hub stores the objects that represent the following agent components:

- Agent communicator
- Event processing
- Decision making

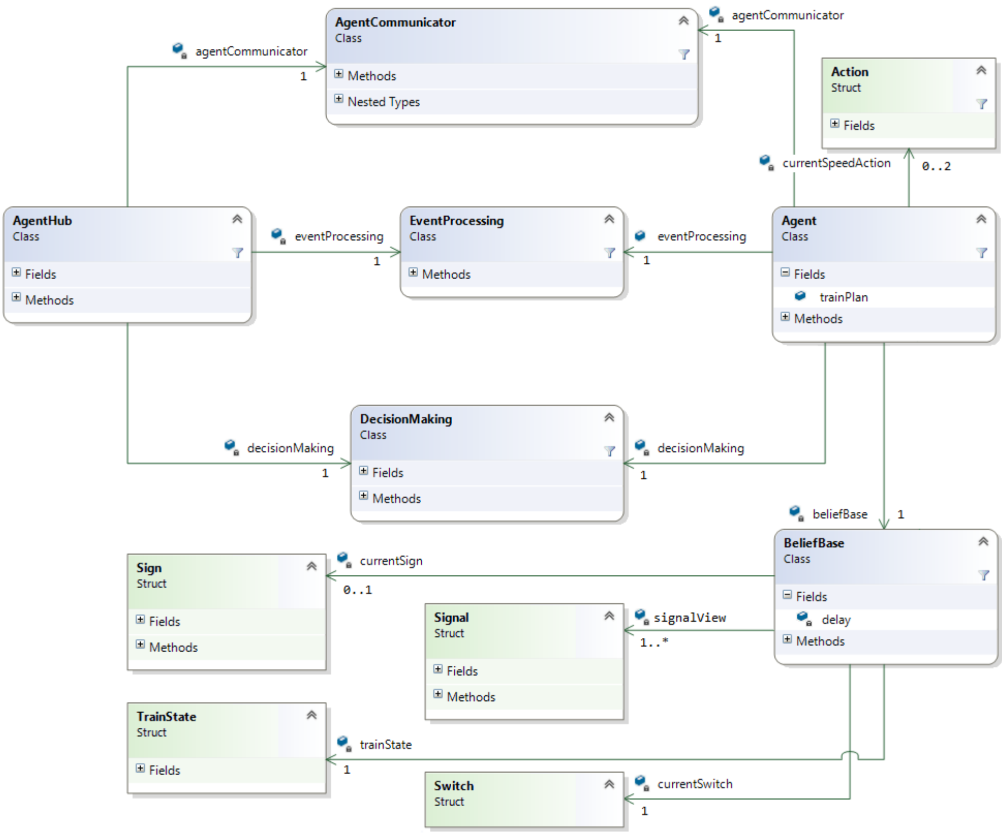


Figure 5.1: *A collapsed class diagram of the DLL, containing the agent hub, the different components and the agent. A more detailed class diagram can be found within Appendix 9*

The main reason for storing these three components within the agent hub, rather than the agent itself, is due to the possibility within C++ to pass a reference to these objects to each agent. Through passing a reference and only storing the components within the agent hub a performance benefit is gained by not having to create and delete identical components each time an agent is created or deleted. Passing just a reference to these objects means that each agent will be able to use these components directly even though they are actually located within the agent hub. The fact of only having one instance of each component is not a problem since they do not need to be unique for each agent. The lack of need for uniqueness does not hold for the belief base component, because it stores the beliefs for that agent.

The agent hub further contains two extra components, namely a unique random generator and a logging component. The unique random generator serves the purpose of generating pseudo random numbers for the entire DLL, this ensures that simulations can be repeated identically by using the same seed. This random generator is mainly used during the creation of actions, where the distributions presented in Chapter 3 are used to select values such as the final velocity for an acceleration action. The logging component can be used to log information about the DLL and the individual agents. Within the implementation, the agent hub will pass the received messages from FRISO onto the agent communicator, regardless of content.

5.1.2 Agent communicator

The agent communicator object contains all the message processing functionalities detailed within the previous chapter. Due to the agent communicator being stored within the agent hub it will also do the message processing task for the agent hub. The messages that FRISO sends to the agent hub are contained within an XML format. After receiving a message the agent communicator will extract the relevant information, depending on the message type, and either store it within the agent hub or pas the acquired information on to the designated agent. If a message requires the updating of the internal state of an agent, functions within this agent are called to update specific parts of its internal state and belief base. These functions update: the train state, signals, signs, switch, train plan and realization time (delay). After the information is passed to an agent, the agent communicator will notify the event processing component about the message type that has been received and processed.

5.1.3 Event processing

As stated in Chapter 4, the main task of the event processing component was to see if the environment has changed in such a way to warrant a possible action given the message type and the information available within the agent. In order to facilitate this process, a number of functions are contained in the event processing component. The main function within this component determines the current situation the agent is in. During this process the maximum allowed velocities of the signs and signals are also analysed. This is not only to determine whether or not, for example, the agent is approaching a sign or signal that indicates a lower allowed velocity, but also to select the significant sign/signal. The significant sign/signal refers to the sign/signal the agent needs to react to. There are five distinctions made between possible combinations of signs and signals.

1. Sign on the signal
2. Sign, with further on a signal
3. Signal, with further on a sign
4. A single sign
5. A single signal

A sign is considered to be on the signal if it is placed within 25m of the signal. This is according to the specifications train drivers need to follow. If a speed sign and signal are placed closely next to each other, the speed limit indicated on the sign is set to start at the place of the signal. A sign/signal is considered to be placed further on, if it is currently visible but further away than 25m of the visible sign/signal that comes before.

One important aspect to note about the selection process of the significant sign/signal, is that it is possible for the agent to have multiple signs and signals within viewing distance. Due to this a selection needs to be made. The sign in question here always refers to the first visible sign. The signal refers to the closest visible limiting signal, with a default of the first signal if no limiting signal is present. A limiting signal here denotes a signal that restricts the velocity of the train in any way, such as a yellow or red signal.

The reason for making the five distinctions based on possible sign and signal combinations, is that it needs to be determined which sign or signal the agent needs to react to first. For example: within the situation where the agent is approaching a signal/sign that indicates a lower allowed velocity, it is currently not specified which visible signal or sign gets the priority. This priority can be deduced based on the order and the meaning of the signals and signs in question.

One example of this, as first seen in Chapter 2 and now in Figure 5.2, is the case where the agent is approaching a yellow signal 8, with next to it a speed sign with a 6, while going 130km/h. This means that starting from the signal position the train needs to start braking towards 60, while making sure that the 80km/h will be reached before passing the next yellow signal and making sure that the 60km/h will be reached before the next maximum speed sign. Within the implemented agent this is ensured through first doing the braking action for the most limiting indicator, while during the braking curve it is checked that this braking action does not violate the requirements of the least limiting indicator. If it is observed that the current braking action will violate the requirements of the least limiting indicator, the braking action is adjusted accordingly.

5.1.4 Decision making

Within the decision making component, the decision making model has been implemented. As in the overview presented in Chapter 4, an initial decision tree is traversed based on the situation given by the Event processing component. After this, based on the current action, a section of code is entered that selects and creates an action. All traction actions use the traction lever position method as described in section 3.4.3. The braking actions that are created take into account the significant sign/signal that was identified in the event processing object.

One notable aspect of these braking actions is that it is assumed that the train drivers know the distances between the signs and signals. This assumption is founded in the fact that train drivers have to have knowledge about a route if they are to be allowed to drive it. For the agent, this means that if available, it will use the distance information acquired from FRISO to set the braking lever appropriately.

The onset of the braking action towards a stop was initially set to be at the observation point of the scout sign (verkenbord), which indicates that there is a station ahead at the end of the braking curve ot the train. It turned out however that this could cause unnatural behaviour in sections where the stops

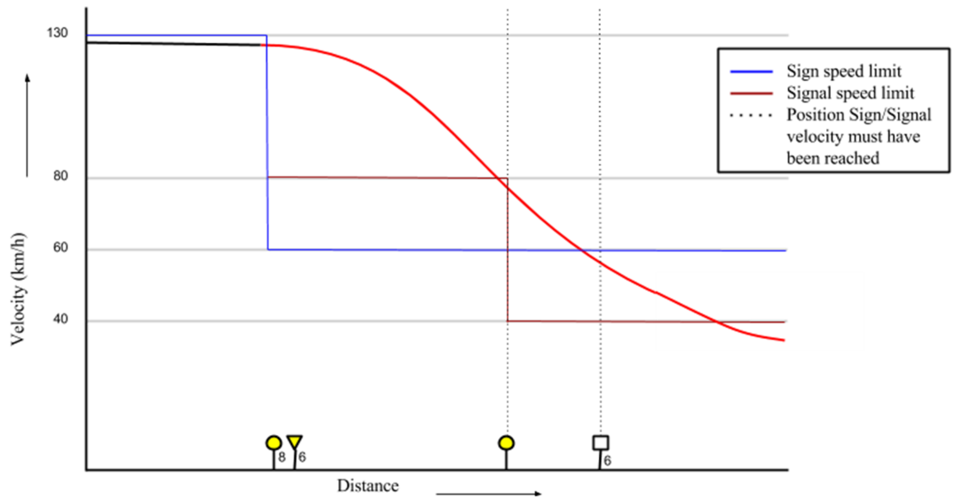


Figure 5.2: *An example of the relationship between the speed limits imposed by signs and signals. On the x axes the distance the train has travelled is indicated. On the y axes the velocity is indicated. The black and red curve represents the velocity of the train. After the train passes the yellow 8 signal with the speed sign 6 next to it, the train needs to limit its velocity to 80km/h before passing the next yellow signal. It also needs to ensure that the 60km/h speed limit has been reached before passing the maximum speed sign. The points at which these velocities have to be reached are represented with vertical dotted lines, for respectively the 80km/h and 60km/h speed limits.*

are close to each other. The reason for this unnatural behaviour was due to the train not having the time to reach the speed limit before reacting to the scout sign. This meant that the train would starts its braking too soon, seeing that the positioning of the scout sign assumes a velocity close to the speed limit, even if a check was made on the initial DUE. The distribution of initial DUE was thus becoming skewed to a higher degree than initially observed. To counter this, an offset was determined through using the observed relation between the onset velocity and the onset distance to the stopping point from the braking actions within the GPS data. This was used then to change the point at which a train would react to the scout sign, depending on the local speed limit. The functions for this can be seen in Figure 5.3 and Equation 5.1 and 5.2. Complete linearity was prevented through the already present variation of the initial velocity due to the different free-track behaviours. This resulted in a more varied and more accurate onset point for braking action towards a stop. The fitted distributions were also stored within this decision making object. These distributions included the required transformation information, to undo the shifting that was done when fitting the distributions, and were reinforced to be within set limits of their minimum and maximum allowed value.

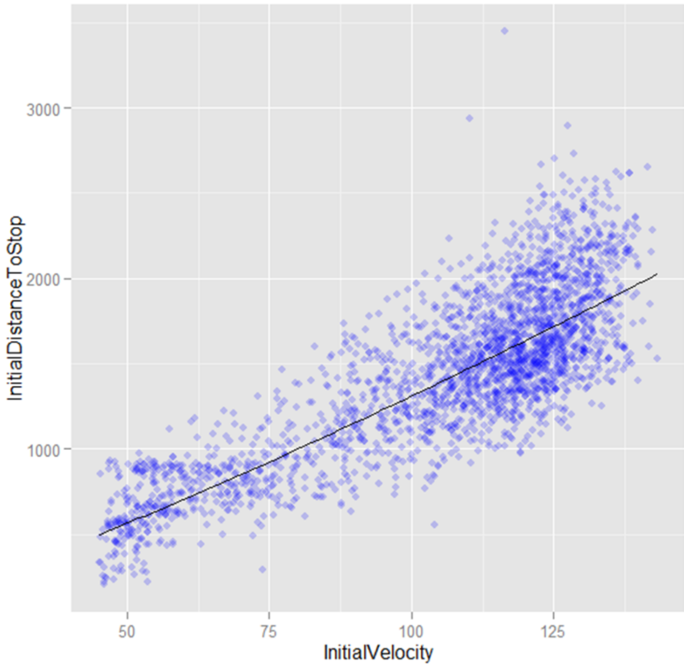


Figure 5.3: *A visual representation of the correlation between the initial velocity and the initial distance to the stopping point of the first braking action towards a planned stop. On the x axes the initial velocity can be seen. On the y axes the initial distance to the stopping point can be seen. In blue the GPS data points are visible. The black line represents the fitted function of Equation 5.1.*

$$DistanceToStop = 4.939 * InitialVelocity^{1.212} \tag{5.1}$$

The function used to determine the onset point, the distance to the stop, given the initial velocity.

$$Offsetdistance = 4.939 * speedLimit^{1.212} - ScoutSignDistance \quad (5.2)$$

The function used to determine the offset distance for the point at which an agent reacts to the scout sign.

5.1.5 Agent

The agent object serves as a connecting part of the software agent. It contains references to the other components, stores the belief base and maintains its own set of information. The functions that are present within this object mostly facilitate the updating of the belief base and the setting of the free-track behaviour. The braking criteria of a train is also determined here. Besides storing the belief base, the agent stores information about intrinsic characteristics of the train and certain parts of the environment of which beliefs are held within the belief base. Intrinsic characteristics refers to aspects of the train that the agent cannot change, such as the name of the train, the train length, the train ID, the viewing distance, maximum deceleration of the train, etc. It also stores a list of the actual signals that are in front of the agent in order to be able to update the belief base with that information when required. Lastly, the agent object also stores information used for the creation of logs.

5.1.6 Belief base

The implemented belief base component mainly serves as the data storage compartment of the agent, maintaining the information about the environment and internal state that are used for the decision making process. On top of storing information, the belief base has a number of small functions to facilitate determining the expected signals and the attributes that belong to specific kinds of signals. One example of acquiring the attributes of a signal is translating the received information 'yellow signal' to: A signal that is not blinking, that does not have a number, that indicates a maximum allowed velocity of 40km/h after passing it, while not requiring a maximum allowed velocity in order to pass it.

5.2 Assumptions

During the implementation process certain assumptions were made. These can roughly be divided into two categories: Value based or Process based. Below I have listed the most notable assumptions that were made within these two categories that haven't been mentioned yet, and the reasons why:
Value based:

- The signal view distance was set to be uniform for all train drivers at 801 meters.
  - *It was initially set at 800 meters as indicated by a staff member of ProRail. This is double the maximum minimum distance that a signal has to be visible at according to the signal specifications [44]. As in, the minimum distance a signal has to be visible at the maximum speed limit of 160km/h is set at 400m, thus indicating the maximum minimum distance. The 1 was added to prevent rounding errors.*
- The sign view distance was set to be uniform for all train drivers at 201 meters.
  - *It was set at 200 due to that being the minimal viewing distance of signs where the local speed limit is > 80km/h [43]. The 1 was added to prevent rounding errors.*
- The desired velocity, d_v, selected through a distribution when accelerating under a certain speed limit, s_l, in km/h is set to be: $s_l - 5 \le d_v \le s_l + 5$
  - *The reason for this was the assumption that the train driver cannot go faster than 5km/h above the speed limit and that the train driver would not initially stay a lot under the speed limit, unless he is driving under UEI specifications, due to the goal of arriving on time.*
- The timing for the onset of a braking action is the same for signals as for signs.
- The simulations done are set to be during the day, with good sight.
  - *This was mainly done to keep the implementation simpler, seeing that a train driver needs to follow different rules if this is not the case. Such as the rule dictating that if a train driver is not driving with good sight during the day, he is only allowed to start accelerating after passing the signal or sign that indicates a higher allowed*

velocity. On top of that was the fact that the MATRICS data used for the classification trees comes from such an environment.

Process based:

- The principles and values used to determine the actions of the agent are similar for both intercity and stop trains.
  - *This was mainly done due to the lack of information about stop trains, and the assumption that the goals and driving material of these trains are similar enough to model both within one model. For freight trains it was determined that the driving behaviour differs significantly from passenger trains, this was concluded due to the information acquired from a former freight train driver and a GPS data set of freight trains.*
- The agent always observes the signs and signals if they are within the viewing distance, as in, he cannot miss observing a sign or signal.
  - *It was deemed not desirable for this implementation that the agents would be able to make mistakes, as such the possibility for that was not included within this implementation.*

5.3 Difficulties

During the process of implementing this structure within the DLL and FRISO a number of difficulties were encountered. A number of these had to do with the way FRISO works and with the interactivity between the agents and FRISO. Within this section I will go over a number of these to give an impression of the kinds of hindrances encountered and how implementation can bring forward exceptions which are hard to predict beforehand without intimate knowledge of all the aspects involved.

- The acceleration of a train would be set to 0 if the agent is trying to accelerate above the rolling stocks maximum speed. This would result in very uniform driving behaviour, due to the agent trying to get the trains velocity closer to the speed limit.
  - *This issue was dealt with through adding an internal parameter that contains the maximum velocity of the train, such that when this acceleration value of 0 is encountered, the agent recognizes it*

as the maximum velocity of the train, and takes that as the maximum speed limit possible. Thus ensuring that there is variation in the behaviour.

- Rounding errors occurred through the message passing process between FRISO and the DLL. One example of this is the case that the agent asks FRISO for a position action at a distance that is 800m in front of the next signal, when the viewing distance is set at 800m. At a distance of 800.0001m FRISO tells the agent that the desired position has been reached, the agent then checks if it can see the next signal, which it checks though (distance ≤ viewdistance), which tells the agent that it cannot see the next signal yet. The agent then asks FRISO again for a position action at a distance of 800m in front of the next signal. FRISO immediately tells the agent again that it has reached that position, etc. Resulting in an endless loop. Such endless loops occurred also in different situations, such as for FRISO telling the agent that the desired velocity has been reached, while it is 0.0001m/s less than the desired velocity.
    - *This issue was mostly dealt with through placing +1 in cases where this occurred.*
- Due to the default onset distance for an acceleration action being at 10cm in front of the train, it was possible for the train to get stuck. This occurred when the train is approaching a red signal and the agent starts to brake towards a stop. Nearing a full stop, the agent receives a message from FRISO that the signal aspect has improved, indicating that he can now accelerate again. The agent then tells FRISO to start accelerating at a position of 10cm in front of the train, however, during that 10cm the train can reach a full stop due to coasting resistance, thus never reaching the onset position for accelerating.
    - *The possibility of this is still present in the current implementation due to the low occurrence rate. In order to ensure that this could never happen aspects within FRISO would need to be changed, such as adding a message that informs the agent any time the train reaches 0m/s.*
- The stopping position of a train gets dictated through a suggested position that is contained within the stopping signals information. The agent is allowed to stop within 100m of that stopping point, which FRISO will recognise it as a stop next to the platform. However, with short stops

it could occur that the stopping signal is far away from the head of the train, thus resulting in another signal being present within that 100m range. If the train stopped in front of that other signal, FRISO will check if the next signal is the stopping signal, which it is not, and thus determining that the train has not stopped next to the station even though it is within the 100m range of the stopping point.

  - *This issue was partially solved through the agent checking if there is a signal within that 100m range. If there is another signal, the agent will aim to stop 18m after that signal. It is set at 18m due to the errors that occur when estimating the velocities and distances for a braking curve. If the agent needs to coast for a relatively long time before the onset of the final braking manoeuvre, the simulated resistance will slow down the train, resulting in a slightly earlier stopping point. If the agent overshoots the relevant signal by too much, it is possible that the agent stops outside of the 100m range, and thus it will not be recognised as a valid stop. Due to the different rolling stock characteristics, and the inability to estimate completely accurately the coasting deceleration, this possibility is still there.*

- In certain occurrences the train needs to turn around at a station (this excludes a short stop). In this case the entirety of the train needs to be positioned between the stopping signal and the previous signal. Due to the variation in the braking behaviour it could occur that the agent stops with the end of the train just next to the previous signal, which would stop FRISO from turning the train around, while the agent would patiently wait for it to be turned around and the message that allows it to depart.
  - *This issue was solved through checking if the estimated distance between the stopping point and the previous signal would be larger than the length of the train. If this is the case, the agent will aim to stop at a position such that the end of the train has just passed the previous signal. This action is also depended on estimations, and due to the limited accuracy of estimating the coasting deceleration, it is still possible for this to go wrong.*

Other difficulties had to do with errors or bugs that were present within FRISO, such as:

- Not all maximum speed signs being present within the infrastructure model that FRISO uses. This could mean that the agent will try to decelerate for a speed sign that forces the train to slow down, but that it will not have reached the allowed velocity in the correct location due to the missing maximum speed sign.

- More straightforward bugs such as: Wrong position data for signals, which would change depending on the message type received. Trains not being removed from the model after completing their time table, and more.

- Limitations within FRISO for passing signal information. FRISO is supposed to send the agent the data of the 3 signals ahead, as long as the route has been set. Within the railway network this information is available until the next red signal. Within FRISO, at times only information of the first signal was passed through even though it is a green signal. *This issue was partially solved through re-checking the situation and action after passing a signal if there was only 1 available at the time of creating the action. One example use is in the case of approaching a yellow signal indicating a 6. This means that the train driver needs to decelerate towards 60 and needs to have reached this velocity before the next signal. The agent will assume that the distance between those two signals will be large enough to brake from the current velocity to 60km/h if starting the braking action at the first signal. After passing the yellow 6, the agent checks the current braking lever position, and sets it appropriately now that the position of the signal where the train needs to have reached 60km/h is known.*

There were also difficulties due to certain exception cases present within the infrastructure. One example of this is can be seen in Figure 5.4. Here it can be seen that after the departure the first speed sign dictates a maximum speed of 110km/h, after which there are 3 successive green signals with an 8, indicating a passing speed limit of 80km/h. This means that the train is not allowed to pass these with a velocity higher than 80km/h. After these there is a long section without any specification, where according to the rules the train driver is allowed to accelerate towards the 110km/h. There is however another maximum speed sign that dictates 80km/h, without there being a limiting speed sign before. Within the implementation this caused problems, seeing that the agent had already started to accelerate towards the 110km/h, and when seeing the 80km/h sign at the minimum view distance, had to brake harder than was possible. After consulting an employee at ProRail who looked at the infrastructure of that section, it turned out that the ATP safety system was implemented differently than normal on that section of track. The adjusted version dictated that the maximum velocity was 80km/h, even though there was a speed sign indicating 110km/h before.

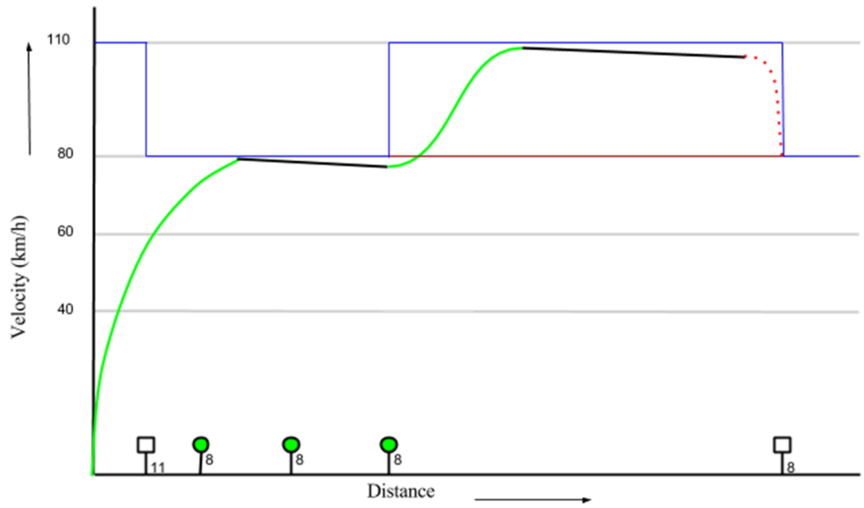


Figure 5.4: *A visual representation of an infrastructure exception. On the x axes the distance is indicated. On the y axes the velocity of the train is indicated. The green and black line represents the driving behaviour of the agent. In red the required braking manoeuvre is represented, starting at the observation of the speed signal. In blue, the maximum speed limit is indicated according to normal rules within the simulation. In brown the exception case for the ATP setting is shown, which forces the speed limit to be at 80km/h within the infrastructure.*

5.4 Conclusion

Within this chapter an overview is given about the implementation and the differences with the model described in Chapter 4. The assumptions that were made during the implementation process are also outlined, and a selection of difficulties that were encountered were discussed. The next chapter will go into using this DLL within simulations in order to assess the effects of an agent based approach to modelling train driver behaviour.

6 Experiments

6.1 Introduction

In the previous chapter the implementation of the agent setup within the DLL was presented. In this chapter the implemented DLL has been used together with FRISO to conduct a number of experiments. First the setup of these experiments will be discussed, after which the results of these experiments will be presented and discussed. Before the setup of the experiments was determined, the goal and research question were used to come to two questions around which these experiments could be set up. The goal was to find out if the predictive value of the simulations done within FRISO could be improved through a software agent implementation that models train driver behaviour. The research question that came with this goal was:

- *How can you add train driver behaviour to a micro-level simulator (FRISO), using an Agent based approach?*

This question has largely been answered throughout the previous chapters. Covering the steps from modelling train driver behaviour, to designing the agent, to implementing the agents within a DLL that can be used by FRISO. The main aspect that has not been covered yet is the result of this implementation. In order to look into this two questions were posed:

1. How well does the implemented agent, model the behaviour of real train drivers?

2. Is there an improvement in the predictive value compared to the present train driver model within FRISO, using this agent based approach?

Within the next section the experimental setup will be laid out in order to answer these questions.

6.2 Experiment setup

Before going into the setup of the experiments that were used to answer the posed questions some notable concepts will be further explained.

Currently, the overall aim of the simulations within FIRSIO is to give more insight into the results of traffic and infrastructure measures. For this it is desired that the processes within the simulation are close in likeness to reality, this to give a more accurate representation of the effects of these measures. One way to find out how close in likeness the simulations are to reality, is through looking at the realized times of the trains within the simulation and comparing these with the realized times of real trains.

After talking with several employees at ProRail it was advised to not look at the punctuality of the trains. Instead, it was suggested to look at the driving times between stations. The main reason for this was that the driving times are mainly influenced by the train drivers themselves. If you were to use the punctuality instead, the following factors would play a notable role on these times:

- **Halting times at stations:** Train drivers do not have influence on these halting times, seeing that they are dependent on the conductor for the timing of the departure, and that the waiting time at a station is influenced heavily by the amount and movement of people getting in and out of the train.
- **Previous punctuality:** If a train departs delayed at a station it is possible that the train driver is not able to compensate for that delay and thus arrives also delayed at the next station.

Another method to compare the realism of the simulations mentioned by ProRail was to look at the amount of red signals a train driver encounters during a ride. If a realistic image of the number of red signals is desired it is necessary to simulate every train within a model. This was not feasible for two reasons:

- The exceptions found within the implementation, some of which were discussed within the previous chapter, prevented that all trains would be able to drive all the time without getting stuck and interrupting the simulation.
- Freight trains were not explicitly modelled, meaning that freight trains within the model would very likely drive significantly different than can

be observed from data. Thus influencing the amount of red signals a train encounters.

If not all trains are present within a model that are present on the railway, it would likely significantly impact the amount of red signals encountered due to the knock-on effects train delays can have.

The knock-on effects refers to the effects that a delay of a single train can have upon the delays of all other trains that are present. One example of this is when a train arrives at a station 5 minutes delayed. Due to this delay, the train that was supposed to arrive afterwards, now arrives at the same time at the station as the delayed train. Due to the limited number of platforms available, and the limitations of the amount of trains that can fit on one section of track, choices need to be made which train arrives first, departs first, etc. This can subsequently delay the second train, which can delay other trains, etc. When creating the time table these knock-on effects are taken into account, and slack-time is added to counter these.

Within this process the choices that are made by traffic controllers also come into play. Traffic controllers can for example give priority to trains arriving, departing or crossing. This influences the overall flow of railway traffic. Within FRISO there are some automated methods to deal with these issues, such as:

- First come first serve: *The train that arrives first at a crucial point gets priority.*

- Set order: *If there is a conflict, the train that is supposed to arrive first gets priority, regardless of the delay.*

- According to plan: *The paths are reserved according to plan. It is assumed within this setting that no trains are ever delayed and as such no conflicts will arise.*

It was decided that the experiments would only look at one train series at a time, thus ensuring that the traffic control method no longer plays an impacting role. Thus, in order to compare the realism of the simulations numerically, the driving times of trains between two stations was taken as a measurement.

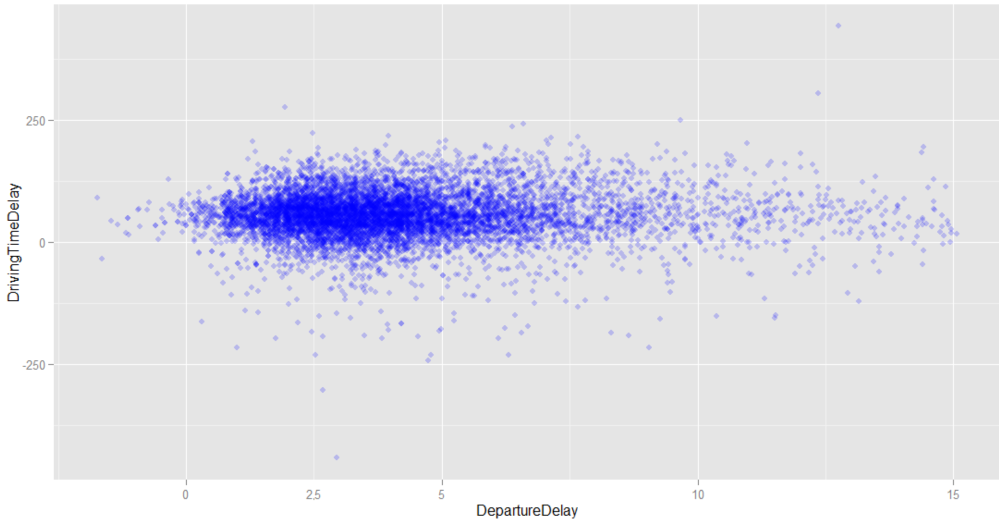


Figure 6.1: *A scatter plot indicating the relation between the delay a train has at its departure and the driving time it has to the next station compared to the planned driving time. On the x-axes the delay at departure can be seen in minutes. On the y-axes the difference between the driving time to the next station and the planned driving time can be seen in seconds. A positive value on the y-axes indicates how much faster the train went compared to the planned driving time.*

One notable point about using the driving times is that it does not take into account the possible influence of departure delays. The departure delay refers to the delay a train can have at the moment of departure, which for example can be caused by having arrived too late just before. This departure delay could influence the driving behaviour over the next ride between two stations. The train driver could for example think that he needs to catch up, and thus drive a bit faster than normally. This effect can be seen in Figure 6.1, where there is a slight visible speed-up of the trains as the departure delay increases, this however seems to reach a limit at some point around 2.5 minutes. This limit also seems to be very much dependent on the location, which likely comes from the slack-time that is available and planned for in between stations. A visualisation of this can be found in Appendix 10. The downwards deviation that can be seen as the departure delay increases from 0 is likely due to subsequent obstructions encountered from trains in front of the train. It must be noted that the planned driving time does not always indicate well the realistic fastest driving time between two stations, thus introducing noise into Figure 6.1.

It is possible within FRISO to use realized departure delays to set the initial departure timing of a train has within a model. The same goes for stopping time deviations. However, seeing that the agent currently does not model this concept in an accurate fashion, these aspects were not used.

After it was determined to use the driving times between stations as a measure of likeness, the next step was to acquire the driving times from realisation data. The main source of this data came from the performance analysis bureau (PAB) of ProRail, which stores and maintains the official realisation times of the time table. The PAB data used came from the month of February 2014. This month was chosen due to it being within the year 2014, thus using the same time table as were available within the simulation models of FRISO. A simulation model within FRISO denotes a model that is used by FRISO, where the locations, time tables, settings, simulation runs and other aspects of a simulation are specified. The month of February was also chosen because it had a relatively low amount of rain and other adverse weather effects [79]. Seeing that the models do not take into account weather effects, this was deemed desirable. Given the availability of extensive GPS data for the 1900 train series, this was also used as a reference for realized driving times.

Due to not having other trains within the model that could cause delays, the agents would drive within a situation with no hindrance. This is unlike the trains from the realisation data. In order to counter this the trains that took more than three minutes longer than the planned driving times between two stops were considered delayed and subsequently removed from the PAB data. Exceptions to this can be found in Appendix 11.

One important thing to note is the definition of the driving times. Officially a train is considered departed if it has started to move and a train is considered arrived if it has stopped moving. Due to the lack of available information for measuring this point precisely within normal operations, ProRail estimates this point using an estimated velocity and deceleration at the last measuring point before a stop, or the first after departure. This measuring point comes in the form of a weld that is located between sections. This same system is used to locate the trains position at a section on the railway. At times this meant that there was no detection possible between two stations, this resulted within the logs for a uniform arrival and/or departure times. At times impossible driving times were encountered within the PAB data, such as a case where the train drove between two stations with a planned time of 3 minutes in 7 seconds, according to the logs.

Within FRISO the arrival and departure point can be measured precisely. For the GPS data these points are identified as soon as it can be distinguished given the accuracy of the GPS measurements. In this case it meant that a train was considered stopped if it was going slower than 1m/s. Due to the use of these different methods to determine the arrival and departure times there will be some influence on the eventual resulting driving times. It is assumed that these differences will mostly be the same across the different rides. Thus, for example, if the results show that every stop there is a 5 second difference between the GPS times and the PAB times, this could be explained to a certain extend due to the different measures used. If there was a non-uniform difference, it could be a strong indicator that this is likely due to the different driving behaviour or due to a severe deviation in the estimation done.

Also note that the models within FRISO use the basic hour pattern (basis uur patroon) as their time table, which is different from the actual time table used every day. It is different in the sense that it only uses the basic hourly pattern seen throughout the day, it does not use the different times that are scheduled for, for example, nights. This could influence the driving times seen within the realisation data. So could the train have a minute more or less to drive between two stations. These differences however, only occur a few times compared to the overall rides that occur during the entire day. It was thus decided to not filter out the cases where this occurs.

Because the agents were designed with a number of adjustable parameters, discussed within the previous chapters, a selection of values for these parameters were used within the experiments. A number of different relevant parameters with respect to driving were also available for the default FRISO train drivers. These were:

1. Maximum speed. *This parameter determines the maximum velocity a train will go at.*
2. Deposit speed. *This parameter can be used to set a lower bound for selecting a maximum velocity. FRISO can use this parameter to determine a velocity between this value and the speed limit in order to arrive more on time.*
3. Braking variation. *This Boolean parameter determines whether or not*

a distributed braking action is done. Within FRISO this means that the train will pick a random number from a uniform distribution that results in a braking curve between the braking criteria and the emergency braking curve.

4. Acceleration deviation. *This parameter can be used to change the default acceleration curve by multiplying the acceleration by a value. On default, this acceleration is set at the maximum possible acceleration.*

FRISO also has the option to change the braking criteria, which on default is set to $-0.33 m/s^2$. Note however that this cannot be set uniquely for the rolling stock that is used, it is set for all trains that are present within the simulation model. For this reason, the default setting will be used normally, with one experiment using the appropriate setting for the specific train series that will be driven. This in order to look at the effect of this setting, while also taking into account that normally when a simulation will be done within FRISO, it is not possible to set this parameter uniquely for each train present within a simulation model. In order to answer the other question about how well the implemented agent models the behaviour of real train drivers, the driving behaviour of the agents found within the previously lined out experiments were compared with the driving behaviour from the GPS data. This could be done for the train series where GPS data was available, namely the 1900 series. The main way of comparing the driving behaviour was done through comparing the speed-distance diagrams. The reason for using these diagrams was to show the different aspects and effects of the driving behaviour locally.

The train series that were selected for these experiments were selected in such a way as to ensure that they did not have the same station order in common. In Table 6.1 and 6.2, an overview is given of the train series used within the experiments:

Train series	Train type	Number of stations within models	Number of agent setting variations	Number of FRISO setting variations
1900	Intercity	7	8	6
4300	Sprinter	15	1	1
12700	Intercity	1	1	1
700	Intercity	5	1	1
2600	Intercity	6	1	1
3600	Intercity	2	1	1
5600	Sprinter	4	1	1

Table 6.1: *Overview of the experiments that were done.*

Train type	Number of stations
Intercity	21
Sprinter	19

Table 6.2: *Overview of the amount of stations per train type.*

In Table 6.3, an overview is given of the amount of train rides that were simulated and observed. These amounts denote the lowest number of trains and the highest number of trains that were observed for a ride between two stations. The numbers for the agents are the result of doing simulations of 600 minutes 7-8 times.

Train series:	Agents lowest count	Agent highest count	Realisation lowest count	Realisation highest count
1900	119	152	947	988
4300	102	140	159	1078
12700	72	72	431	457
700	140	160	378	434
2600	94	140	766	952
3600	133	140	965	1005
5600	133	140	976	1040

Table 6.3: *Overview of the amount of rides for the rides for each train series.*

The models that were used were models that have been used by ProRail before, which use the timetable for 2014.

6.3 Results

In this section the results of the experiments will be presented. This will be done for each of the questions posed at the start of this chapter, starting with the question aimed at looking into the predictive value of the agent implementation compared to the current implementation. The results presented here will be further discussed within the section: *Discussion*.

6.3.1 Driving times

In order to make a comparison between the agents and the FRISO train drivers, different settings were tried out and compared. The resulting experiments can be divided into two categories:

1. Default settings

2. Adjusted settings

The first category was used to compare the agents and the FRISO train driver with the realisation data across different train series. The second category of experiments were used to see the results of using different settings for the agents and FRISO.

In order to compare the driving times between two stations and their distribution, the differences between the 5th, 50th and 95th percentile and the mode were used. This was done through subtracting the numerical value found for the agents or FRISO, from the value found within the realisation data. This will be from now on referred to as the error rate. For example the error rate between the 50th percentile values can be expressed as:

$$errorRate = Median\ realisation\ driving\ time - Median\ agent\ driving\ time \tag{6.1}$$

This was calculated for each ride between two stations, after which an average absolute error rate was calculated over each of these error rates. This average absolute error rate will be denoted as E. The function for this can be seen in equation 6.2, where in this case the 50th percentile is used:

$$E = \frac{1}{n} \sum_{i=0}^{n} |mr_i - ma_i| \qquad (6.2)$$

Where mr is the median realisation driving time between two stations, and ma is the median agent driving time between two stations. n is the number of stations that has been travelled to. E is the average absolute error rate for the median driving time between stations.

6.3.1.1 Default settings

The overall results of first category of experiments, where the agents and FRISO trains were driven with their default settings, can be seen in Table 6.4. In the top rows of this table the total number of rides between two stations are divided to include only rides of train series that go in one direction. This direction selection was done based on the direction category present within the time tables of the simulation models used. These categories were H and T. In the third row, these two sets are combined to show the overall error rates of all the driving time distributions. The error rates are calculated as shown above. The last two columns show the E values for FRISO. In the second column of this table the total number of rides can be seen. A ride here denotes that one train drove from one station to the next station, of which there were on average 130 between each two stations in each direction. Due to there being no deviation with the FRISO train driver with default settings, this is only compared to the observed 50th percentile and the mode. In the bottom row the standard deviations for the different error rates are shown, for all directions. The standard deviations given an indication of the range and number of outliers that were present when comparing the driving times.

Direction	Number of rides	Agents				FRISO	
		5th P E	50th P E	95th P E	Mode E	50th P E	Mode E
H	5266	56.92	59.85	78.08	74.70	75.10	84.88
T	5101	53.03	54.06	75.36	56.60	61.55	62.80
All	10367	54.97	56.96	76.71	65.65	68.33	73.84
		std Dev	std Dev	std Dev	std Dev	std Dev	std Dev
All	10367	63.86	72.49	96.13	85.23	70.75	78.47

Table 6.4: *Overview of the error rates E for the different percentiles (P) and mode, of the different directions.*

In Figure 6.2 the driving times and their distributions are visualised of the 4300, 2600 and 1900 series, in order to give a better insight into the source of the results shown in Table 6.4. The error rates that correspond to the distributions shown in Figure 6.2 can be found in Table 6.5.

	Agents				FRISO	
Train series	5th P E	50th P E	95th P E	Mode E	50th P E	Mode E
1900	21.81	26.36	61.55	38.14	78.71	83.86
2600	46.23	64.25	109.43	71.83	96.33	99.67
4300	70.12	69.03	70.48	77.87	74.80	79.80

Table 6.5: *Overview of the error rates E for the different percentiles (P) and mode, of the different train series.*

6.3.1.2 Adjusted settings

The second category of experiments was done in order to assess the effects of the parameters available for the agents and FRISO. These were done using the same train series, namely the 1900 series. The main reason for choosing this series was that the GPS data was available to also be used to compare the driving times to the ones observed within the GPS data, and to make it possible to compare the speed-distance diagrams. Within Table 6.6 the different settings that were used for the agents are presented. The first three columns indicate the free-track behaviour settings and the percentage of rides where they were used. These correspond with the behaviours described at the end of Chapter 3 as such:

- Cruise & Coast: *Combines the cruising and coasting behaviours, each specified through a percentage indicating the percentage of the free track where this behaviour was shown. Starting with cruising, followed by coasting.*
- UEI: *Follows the guidelines as indicated through the UEI method.*
- Alternation: *Uses an alternation between traction and coasting based on velocity interval.*

The *Maintain Max* parameter indicates whether or not the agent tried to arrive on time through maintaining a velocity close to the maximum velocity if the time of the last TCA indicates a delay of more than 60 seconds.

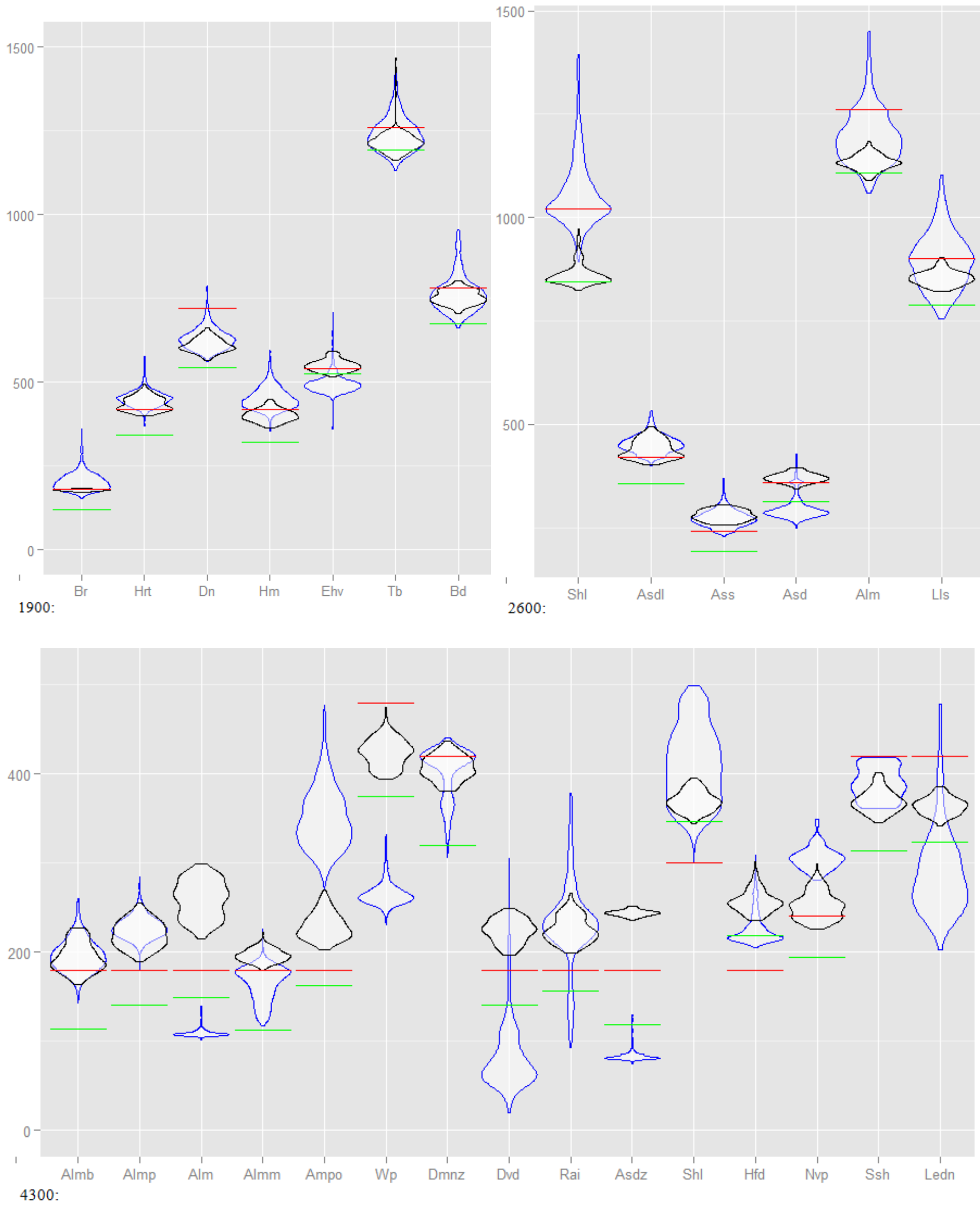


Figure 6.2: *Within this figure an overview can be seen of the driving times for the 1900, 2600 and 4300 train series for direction H. On the x-axes the arrival stations can be seen, with on the left the first arrival station. On the y-axes the driving time can be seen between the departure and the arrival station. The red line represents the planned driving time. The green line represents the driving time of FRISO. The blue violin plots represent the density of the driving times found within the realisation data. The black violin plots represent the density of the driving times for the agents.*

The *Max velocity* indicates the maximum velocity that was allowed for that train. The reason for adding this parameter, is that the ATP of the trains driven by the NS have a maximum velocity limit of 140km/h, which is of importance if the train is driving over a section of track that allows 160km/h.

The last parameter indicates the acceleration version that was used. The acceleration version refers to the setting of the traction lever if the train is to accelerate. As discussed in Chapter 3, this was done through a set of uniform distributions that control the interval and frequency of increasing the position of the traction lever depending on the velocity. Two versions of this were used, with the second one resulting in a slightly steeper acceleration curve than the first one.

Name	C & C	UEI	Alternation	Maintain Max	Max velocity	Accel version
A	55%	20%	25%	Yes	160	1
AV	100%	0%	0%	Yes	160	1
AU	0%	100%	0%	Yes	160	1
AS	0%	0%	100%	Yes	160	1
A NM	55%	20%	25%	No	160	1
AV NM	100%	0%	0%	No	160	1
A NM Ac	55%	20%	25%	No	160	2
A NM M	55%	20%	25%	No	140	1

Table 6.6: *Overview of the setting types that were used for the agents.*

Different settings were also used for FRISO, an overview of these can be seen in Table 6.7. The reason for setting the deposit speed at 140 and 110 was that 140 is the default setting and 110 is around 20% below the maximum speed, which was advised to take by an employee at ProRail. The reason for setting the acceleration deviation at 0.8 was also because it was advised by an employee at ProRail. These settings have been used before at ProRail in order to bring the driving times of FRISO closer to the realisation driving times. In the last row the FRISO DBA version is used with a braking criteria of $-0.19m/s^2$ instead of $-0.33m/s^2$, which is the same that is used for the agents.

Name	Maximum speed	Deposit speed	Braking variation	Acceleration deviation
F	140	140	False	False
F 125	125	140	False	False
F B	140	140	True	False
F A	140	140	False	0.8

F D	140	110	False	False
F DBA	140	110	True	0.8
F DBA -0.19	140	110	True	0.8

Table 6.7: *Overview of the setting types that were used for FRISO.*

The experiments were done using the same simulation model within FRISO, of the 1900 series in direction H. The error rates of these experiments with the adjusted settings can be seen within Table 6.8. The results of the GPS driving times are also added within this table, in order to compare those driving times with the times acquired from the PAB data.

Experiment	5th P E	50th P E	95th P E	Mode E
A	21.81	26.36	61.55	38.14
AV	21.40	27.00	57.38	36.00
AU	23.70	23.71	57.16	30.71
AS	18.45	27.71	63.28	38.00
A NM	24.52	26.57	46.61	31.71
AV NM	28.43	32.00	47.49	27.29
A NM Ac	27.10	24.93	40.11	31.71
A NM M	25.22	29.36	39.76	32.14
F	-	78.71	-	83.86
F 125	-	59.57	-	59.00
F B	49.09	75.00	126.30	83.43
F A	-	71.14	-	76.29
F D	-	56.43	-	55.57
F DBA	56.81	55.07	83.91	54.14
F DBA -0.19	55.08	53.57	72.93	53.71
GPS	46.97	73.17	80.44	79.00

Table 6.8: *Overview of the error rates E for the different percentiles (P) and mode, of the different settings used.*

Seeing the error rates attributed to the GPS data, the overall best scoring settings driving times were compared with different realisation data sets and with the planned driving times, which can be seen in Table 6.9. This was done to gain a better insight into the differences between the simulated driving times and the different realisation data sets and the planned driving times.

Based on these results, the settings that resulted in the overall lowest error rates for the agents and FRISO are compared within Figure 6.3 with the different realisation data. The results present within this figure came from:

- The agents with setting 'A NM Ac'
- The FRISO train drivers with setting 'F DBA'
- The GPS driving times
- The PAB driving times
- The planned driving times

Finally, in order to give a better insight into the differences found between the PAB and GPS driving times, the driving times for each ride are presented in Table 6.10. Here the PAB data serves as the 'realisation' entry when calculating the error rates. This Table most notably shows the at times large differences between the two realisation data sets.

	A NM Ac			F DBA		
Arrival station	Planned	PAB	GPS	Planned	PAB	GPS
Br:	-7	13	-	53	73	-
Hrt:	-27.5	3.5	8.5	13	44	49
Dn:	92	-2	82	49.5	-44.5	39.5
Hm:	8	34	36	40	66	68
Ehv:	-22	-71	23.5	-50	-99	-4.5
Tb:	4	-23	151.5	57	30	204.5
Bd:	-3	-28	51	-4	-29	50
50th percentile E	23.357	24.929	58.75	38.071	55.071	69.25

Table 6.9: *Overview of the error rates for the agents and FRISO with the specified settings, when using three different data sets as the 'realisation' data when calculating the error rates.*

	5th percentile	50th percentile	95th percentile	Mode
Hrt:	0.8	-5	-26.6	-6
Dn:	-75.5	-84	-88.9	-84
Hm:	-0.05	-2	3.05	-39
Ehv:	-74.75	-94.5	-114.2	-81

Tb:	-70.7	-174.5	-195.85	-184
Bd:	-60	-79	-54.05	-80

Table 6.10: *Overview of the error rates acquired when comparing the GPS data to the PAB data. Here the PAB data is used as the realisation data when calculating the differences.*

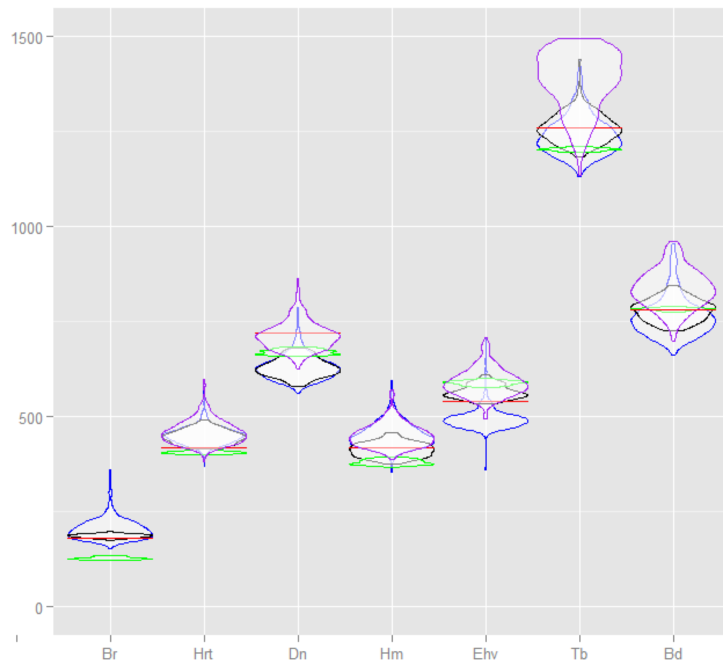


Figure 6.3: *Within this figure a comparison can be seen forthe driving times for the 1900 series. On the x-axes the arrival stations can be seen, with on the left the first arrival station. On the y axes the driving time can be seen. The red line represents the planned driving time. The green violin plot represents the density of the driving times for FRISO DBA. The blue violin plots represent the density of the driving times found within the PAB realisation data. The black violin plots represent the density of the driving times for the agents. The purple violin plots represent the density of the driving times found within the GPS realisation data.*

6.3.2 Driving behaviour

In order to answer the question about how well the agents model train driver behaviour, the speed-distance diagrams of the 1900 series were compared. The reason for using the 1900 series was that GPS data was available for that train series. In Figures 6.4 to 6.6 the speed-distance diagrams are shown for a number of sections. Before describing the figures it must be noted that the straight deceleration lines at times seen for the agents and for FRISO are due to the logging not registering anything in between, thus causing a straight instead of a curved line. The agents used the settings *A NM Ac M*, meaning that they did not try to maintain their velocity if they were delayed more than 60 seconds, used the second acceleration variant and had a maximum velocity of 140km/h. These settings were chosen based on the results of the previous experiments where different settings were used. The sections that are displayed within the figures below were chosen based on their error rates, seen in Table 6.11, and their characteristics. The section to Bd showing a steps wise braking curve within the GPS data, Ehv having the highest negative difference and Hm having the highest positive difference.

Section:	50th percentile difference
Tb to Bd	-29
Hm to Ehv	-72
Dn to Hm	19

Table 6.11: *Overview of the different error rates for the selected sections, compared to PAB data.*

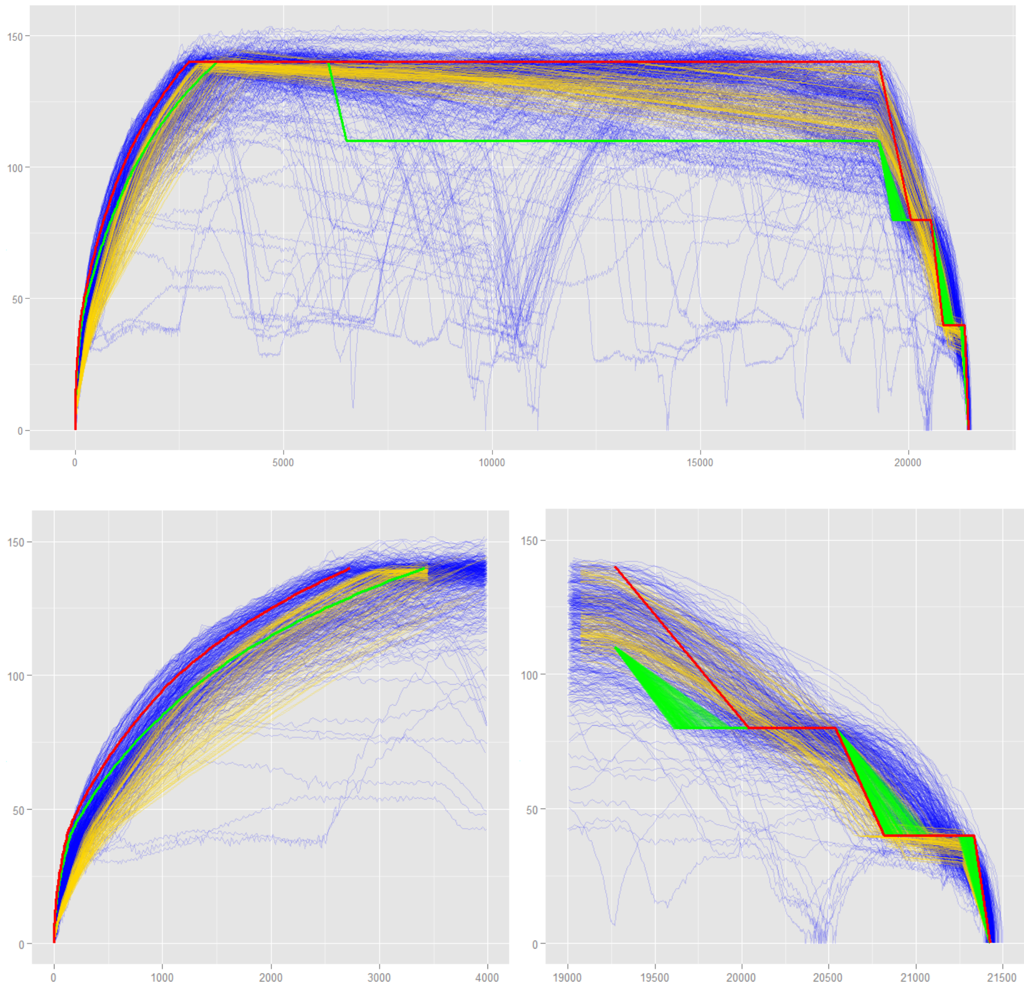


Figure 6.4: *The speed way diagram from Tilburg (Tb) to Breda (Bd). On the x-axes the distance in meters. On the y-axes the velocity in km/h. The blue lines represent the GPS data. The gold lines represent the agents driving behaviour. The red line represents the default FRISO driving behaviour and the green lines represents the FRISO DBA driving behaviour.*

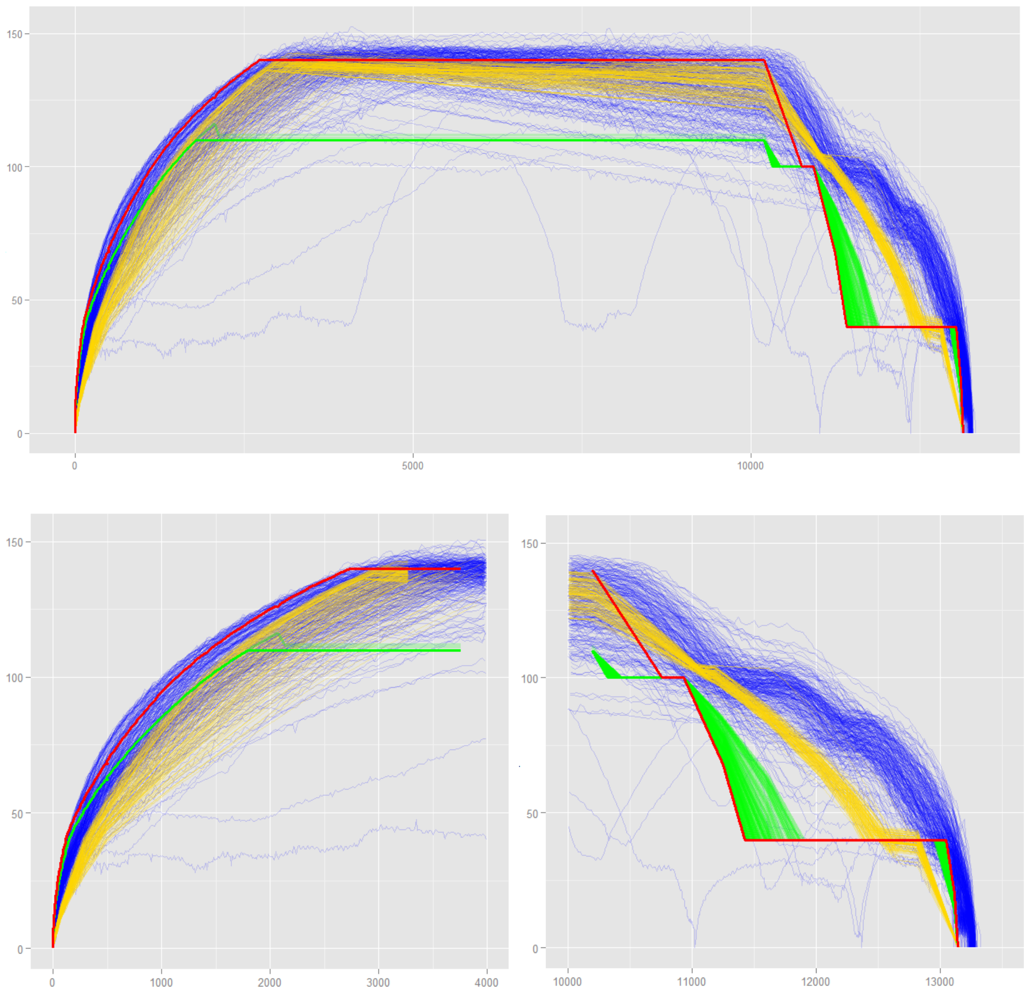


Figure 6.5: *The speed way diagram from Helmond (Hm) to Eindhoven (Ehv). On the x-axes the distance in meters. On the y-axes the velocity in km/h. The blue lines represent the GPS data. The gold lines represent the agents driving behaviour. The red line represents the default FRISO driving behaviour and the green lines represents the FRISO DBA driving behaviour.*

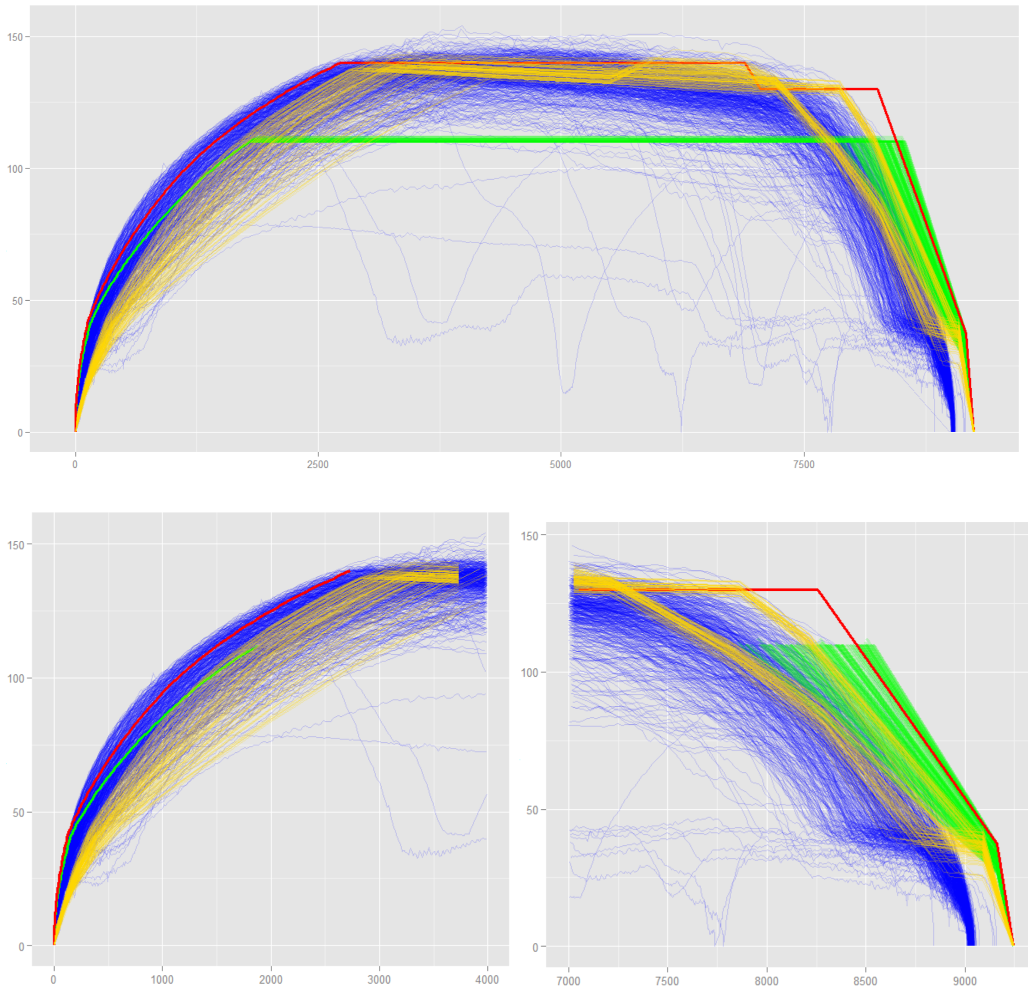


Figure 6.6: *The speed way diagram from Deurne (Dn) to Helmond (Hm). On the x-axes the distance in meters. On the y-axes the velocity in km/h. The blue lines represent the GPS data. The gold lines represent the agents driving behaviour. The red line represents the default FRISO driving behaviour and the green lines represents the FRISO DBA driving behaviour.*

6.4 Discussion

Within this section the results will further be discussed and related to the questions posed. Starting with the experiments aimed at looking into the predictive value of the agent implementation compared to the current implementation.

6.4.1 Default settings

To reiterate, one of the questions posed at the start of this chapter was:

1. Is there an improvement in the predictive value compared to the present train driver model within FRISO, using this agent based approach?

For this, experiments with default settings were used. Looking at the results within Table 6.4, it could be seen that the distribution of driving times acquired with the agents lie closer to the distribution of observed driving times of the PAB data, compared to FRISO. This for both the mean and modus comparison. This is an indication that the agents with default settings give a better predictive value compared to the present default train driver model within FRISO. It can also be seen that the standard deviations of the error rates for both FRISO and the agents are quite large. This is an indication that there are a significant amount of outliers present, where there is a large difference in driving time between the realisation data and the simulated train drivers.

In order to get a better insight into the how and why of this, a closer look was taken at three train series, namely the 1900, 2600 and 4300. Overall, it can be seen in Table 6.5 and Figure 6.2 that the agents in all three cases have a better predictive result. Notable is that the agents for the 1900 series have the best scores for E, which is likely due to the fact that they are modelled for a large part based on that train series. This is a possible indication that the information acquired from the GPS data about train driver behaviour is significantly influenced by the location. The location here refers to the tracks that are traversed by the 1900 series which could have influenced aspects such as the distribution of the final DUE of a braking action towards a stop, due to the distances between signs and signals present on these tracks.

It can also be seen that for the 5th, 50th percentile and Mode error rates that overall, the agents score better than the default FRISO train drivers. Notable is the high error rate for the 4300 series compared to the 2600 and 1900 series. This is an indication that the agent model is worse at modelling stop trains than intercity trains. One explanation for this could be that stop trains encounter significantly different situations which the agent model does not correctly take into account. Another explanation for this could be that train driver behaviour in stop trains is significantly different from that of intercity trains.

The high error rates of the agents for the 95th percentile for the 2600 and 1900 series, compared to the 50th percentile rates is likely influenced by the fact that trains that have encountered obstructions due to other trains, are still present within the realisation data that the model is being compared to. These obstructed trains do not always have a driving time delay higher than 3 minutes, which would have resulted in them not having been taken into account. This driving time delay effect can also partially be seen in the realisation data of the 4300 series at the stop of Schiphol (Shl), where there is a relatively large amount of trains arriving late with a very spread out density curve. This could also have been caused due to Schiphol having flexible arriving and departure platforms, meaning that a train will only be assigned a platform some minutes before arrival, instead of it being planned within the time table.

Another notable aspect of the driving times visible within Figure 6.2, is that FRISO nearly always drives faster than planned, with two exceptions of Shl and Hfd. Both of these exceptions are likely due to the 'early' restriction of 40km/h, which causes FRISO to brake hard and early, resulting in a longer time of driving at 40km/h than is necessary. Figures of this can be seen in Appendix 12. It is also visible within Figure 6.2 that there is a lot more deviation between the realisation and the planned times for shorter driving times. This likely has to do with the way the times are planned, seeing that these are not always based on an estimated driving time. An in depth look into planning will not be made within this thesis, but a possible simplified explanation will be given for two deviations for the 4300 series. The trains within the realisation always have a larger driving time than the planned time at Ampo, while for the next stop, Wp, the trains are able to make up for that delay due to being able to always drive faster than the planned driving time. This could be due to the planning on whole minutes that is done for small stations like Ampo which could have rounded down an initial estimated driving time. Also, it could be due to a larger amount of slack-time being desired for the larger and more busy stations like Wp.

A closer look was taken at the very large deviations between the agents and the realisation data for the 4300 series. Mainly for the arrival times for Alm, Dvd and Asdz. Within the simulations it was visible that the agents observed relatively soon after departure the next yellow signal that is positioned in front of the red signal at the next stop. This means that the agent would stop accelerating towards the speed limit as soon as it observed this signal and start coasting. This coasting then resulted in a very long coasting distance due to the distances between the signals. A likely explanation for the faster arrival times of real trains is that the train drivers keep accelerating towards the yellow signal and only starts to brake when the ATP requires the train driver to do so. Another thing to note about this behaviour is that it is likely influenced by whether or not a train driver will be on time if he were to not continue to increase the trains velocity. This because the same scenario is also found before the arrival at the station Almb, but did not cause a large deviation. This can be seen within Figure .10 in Appendix 12. These findings serve as a possible indication that stop trains encounter significantly different situations than intercity trains, which could be a cause for the agent model being worse at modelling stop trains than intercity trains.

The large deviation between the agents driving times and the realisation data found at the station of Wp, can partially be explained to the fact that the agent preferred to do a normal braking manoeuvre to a planned stop over following the signal limitations first. This behaviour can also be seen in Figure 6.5. It was assumed that the modelled braking manoeuvre to a planned stop would result in a braking curve with a later onset point than if it were to follow the signals, which would mean that if there are signals the agent would give those priority seeing that it would be too early to start a 'normal' braking manoeuvre to a planned stop. However, it turned out that this was not true and resulted in this case that the agents took longer to arrive.

For both FRISO and the agents, the early arrival compared to the realisation data at station Dmnz, could be partially explained due to the temporary increased speed limits that are present there. A plot of this can be found in Appendix 12 in Figure .11. These temporary speed limit increases would results in both the agents and FRISO accelerating unnecessarily. A likely explanation for larger driving times found within the realisation data is that train drivers do not use those short speed limit increases if it is not necessary to arrive on time, which was not the case here.

Taking all of these aspects together, it can be concluded that there are certain aspects of train driving which are not modelled correctly within the agent model and/or within FRISO. It is most notably these aspects which results in the large deviations between the realisation data and the agents and/or FRISO.

6.4.2 Adjusted settings

In order to get insight into the effects of the parameters present for both the agents and FRISO, experiments with different settings were done. Within Table 6.8 the overall results of these experiments can be seen. It can be seen that the different settings have different effects on the distributions of behaviour.

For the agents it can be seen that with NM on for the agents, the 95th percentile error rate becomes lower due to the agents no longer trying to compensate for delays. Note that the 95th percentile of the realisation data is likely influenced by trains who encountered obstacles like delayed trains in front. It can be seen that the UEI setting improved mainly the 50th and mode error rates. Likely due to this free track behaviour resulting in longer driving times, which are more in line with the realisation data.

For FRISO, it can be seen that the D setting and the maximum velocity limiting setting have the largest effects. It can also be seen that the FRISO DBA -0.19 setting scores the best when comparing it to the realisation data. This is a slight improvement over the FRISO DBA version, which is likely due to there being more trains that brake slower, causing the bandwith of braking curves from FRISO to move closer to that of the GPS data. This can be seen when comparing the braking curves in Figure 6.4 with Figure .12 in Appendix 12.

The GPS driving times are here also compared to the realisation data from PAB and has a large deviation to the PAB driving times compared to the agents and the FRISO train drivers. Within Table 6.10, the differences between PAB and the GPS data were further highlighted, indicating that there is non-uniform difference. Indicating that in certain places the PAB prediction is less accurate than others in a significant way, like for Tb.

The results from Table 6.9, where the driving times of the agents and FRISO were compared to those found within the GPS data. It can be seen here both the agents and FRISO score worse when comparing it to GPS driving times compared to the PAB driving times, with the agents scoring better than FRISO for the 1900 series.

In order to give a better insight into the distributions of the observations from Table 6.9, these results were plotted and compared within Figure 6.3. For the GPS data it could be seen that the distribution mostly has the same shape as the one from the PAB data, with the exception of Tb. The fact that they usually have the same shape of distribution but are at times shifted, indicates a prediction error in the way the PAB times are calculated, but one that does keep the relative driving times similar.

For the Agents it can be seen that the shape of the distributions is similar to both the PAB and GPS data. For the distributions visible for FRISO, it can be seen that they usually have the same shape. Within Figure 6.5 it can be seen that the larger driving times for Ehv visible in Figure 6.3 are likely caused due to the steep braking curve, which causes the FRISO train drivers to maintain a lower velocity than necessary, resulting in a delayed arrival time.

These findings indicate that the comparison between just the driving times does not tell the whole story about an accurate representation of driving behaviour. The findings from the GPS driving times indicate that the PAB data is not always an accurate enough measure to compare and validate simulated driving times with.

6.4.3 Driving behaviour

This section will go into the other question posed at the start of this chapter, namely:

1. How well does the implemented agent, model the behaviour of real train drivers?

The results of the experiments done for this question have been presented within a previous section. Here I will go into the different aspects that can be observed from these results, shown in Figures 6.4 to 6.6, starting with some notes about the graphs.

Within Figure 6.6 it can be seen that the stopping position within FRISO can differ from the one seen within the GPS data. In this case this is due to FRISO not having direct access to the location of the platforms, which resulted here in a wrongly indicated position of the platform. It can also be seen that the coasting lines for the agents differ from the ones seen within the GPS data, which are a bit steeper.

Starting with the acceleration curves, it can be seen that the agents more often accelerate slower than found within the GPS data. Possible reasons for this are:

1. The material used within the FRISO simulation, ICM, for the 1900 is only 1 of the types that is used on within this train series. It could be that this is a slower rolling stock than the others that are used. This could then also be seen due to FRISO accelerating with 100% traction being slower than some other trains. *This could be resolved through using different*

rolling stock within the simulation, which is currently not a dynamic feature within FRISO, as in, you can switch, but only per simulation, not within 1 simulation.

2. The distributions and intervals used to set the traction lever are set to conservatively. Thus resulting in a lower acceleration curve. *This could be changed through changing these distributions.*

On top of this it can be seen that the final velocity of the initial acceleration curve of the agents is less varied than that visible within the GPS data, indicating that train drivers also accelerate below and above the range of +-5 of the maximum velocity. Some of this deviation could also be explained due to inaccuracies within the GPS measurements. One possible reason for train drivers accelerating towards a velocity below this limit is that it is not required in order to arrive on time.

For the two different FRISO versions shown, it is visible that the initial acceleration curves are within the bandwidth of the GPS acceleration curves. There is a switch in acceleration visible around the 40km/h, possibly due to the way FRISO models acceleration curves. Notable is the fact that the acceleration curve of the default FRISO train driver, in red, is not above the ones seen within the GPS data while the default FRISO train driver accelerates with the highest possible traction lever position. One possible explanation for this is that within the GPS data different rolling stock can have been used, which could have a higher maximum acceleration curve.

Continuing the initial acceleration curve, it can be seen that the agents follow the bandwidth of the GPS data throughout the free-track section. The deviation within the agents behaviour can be seen here as well. For the default FRISO version, it can be seen that the train maintains the maximum velocity and is thus often above the majority of the observed speed-distance lines. Within Figure 6.5 and 6.6 the FRISO DBA version maintains a velocity well below the speed limit until the final braking manoeuvre. In Figure 6.4 the FRISO DBA trains first accelerate towards the speed limit, after which they brake towards a lower velocity while there is no obligation from the speed limit to do so. These behaviours are likely caused due to the fact that the D setting sets the FRISO trains to choose a velocity that would minimise the difference with the planned arrival time.

At the start of the braking manoeuvre to the stop, the agents seem to initiate it around the same position as the trains observed from the GPS data. They do however do this with less variation, as is most notable within Figures 6.5 and

6.6. For both FRISO versions it can be seen that the onset point can be late in comparison to the GPS data if the arrival is on a green signal, as in Figure 6.6. For the FRISO DBA version this is slightly better due to the variation towards more gradual braking curves.

Within the braking curves it can also be seen that the agents seem to prefer a single gradual braking curve instead of a steps wise one, as seen in Figure 6.4. This is likely due to the same reason for the large deviation from the realised driving times seen earlier for the Wp arrival discussed before. To reiterate, "*It was assumed that the modelled braking manoeuvre to a planned stop would result in a braking curve with a later onset point than if it were to follow the signals, which would mean that if there are signals the agent would give those priority seeing that it would be too early to start a 'normal' braking manoeuvre to a planned stop*". Notable is that the observed step wise braking in 6.5 was caused due to a yellow signal 4 being present within the simulation model, which is not present within the infrastructure, otherwise the train drivers observed within the GPS data would not be able to coast for sections above the 40km/h. Both the default FRISO version and the FRISO DBA version follow the steps wise braking curve as seen in Figure 6.4. With the DBA version being closer to the GPS braking curves due to the variation. In both cases these braking curves are still often below the GPS braking curves, which for the DBA version could be mitigated through using a less strict braking criteria. Within Figure 6.4 it is also visible within the GPS data that trains at times arrive on green, which is due to the action of traffic controllers, which caused more deviation within the observed behaviour seeing that the arrival situations can be different. This possibility for deviation is currently not available within FRISO. Within Figures 6.5 and 6.6 it is visible that the default FRISO and the DBA version perform less accurately due to often setting a strict braking curve. Within Figure 6.5 this results in a larger driving time due to reaching a lower velocity sooner than is required. Within Figure 6.6 this steeper braking curve results in a late onset time and a decrease in driving time due to this.

These findings indicate that overall the driving behaviour of the agents lies closer to that observed within the GPS data, with similar deviations. The FRISO versions deviate from the observed behaviour mainly in the following ways: No deviation for the default implementation, a non-realistic free-track behaviour for the FRISO DBA version, and for both versions a very strong braking curve. The notable less well performing aspects of the agents model were the more gradual acceleration curves and the assumption made about the strength of the braking manoeuvre towards a planned stop.

6.4.4 Performance

After having looked into answering the questions posed at the start of this chapter, this final section will look into the programs performance computation wise.

During the running of the experiments, it was found out that a FRISO simulation with 4 trains took around twice as long with the agents DLL activated compared to without, if ran at the maximum simulation speed. Due to this observation a performance report was made using Visual Studio 2012. Within this, it could be seen in the Call Tree that 2.18% of the running time was spent within the agents DLL as described in Chapter 5. 97.63% of the running time was spent within FRISO itself. This indicated that the encountered delay was not caused by the runtime of the DLL. Besides the percentages, it could be seen in the call tree that a significant amount of time within the DLL was taken by the processing of the XML messages itself.

A likely explanation of the simulation time taking around twice as long with the DLL enabled is the number of events FRISO has to process compared to the default implementation. It could be seen within the agent logs that there could be more than 10 messages within 1 simulation minute per train, which each could cause the scheduling or re-scheduling of events within FRISO. This could happen in the case where there were a relative quick succession of signals and signs that required the agent to adjust its previous action based on new information. Due to the simulation being able to run sped up, this could result in a significant increase in the amount of events FRISO needed to process compared to the default implementation.

This slowdown is thus the result of an implementation that uses frequent events to acquire information and do actions within the simulation environment. The runtime could be improved slightly through a more efficient use of messages, and in turn events, this would however not relieve the need for more frequent events than the current implementation. After discussing this with people at ProRail, the conclusion was that the current use of a DLL and a messaging system is the safest implementation method possible and that the decrease in performance would not impact the usefulness of the simulations.

6.5 Conclusion

Within this chapter an overview of the experiments is given, followed by their results and a discussion of these. The goal of these experiments was to answer the questions posed at the start of this chapter.

The results of the experiments showed that the default agents gave a better predictive value than the default FRISO train drivers. It was also seen within the results of the experiments done with the adjusted settings, that the FRISO DBA version scored better than the default FRSIO version. The adjusted agent version scored better than the FRISO DBA version. It must be noted again, that the agents scored significantly better on that specific train series. It was also concluded that there are certain aspects of train driving which are not modelled correctly within the agent model and/or within FRISO. It was most notably these aspects which resulted in the large deviations between the realisation data and the agents and/or FRISO.

Within the experiments that used the alternate settings, the findings indicated that a closer look at the driving behaviour was desirable and that the PAB data was not always accurate enough measure to compare and validate the simulated driving times.

When looking at the driving behaviours through comparing the speed-distance diagrams, it was observed that overall the driving behaviour of the agents lies closer to that observed within the GPS data, with more similar deviations compared to the FRISO versions. Some notable aspects that deviated between the agents driving behaviour and the GPS data was also observed.

It was also seen, that improvements can be made on the agents predictive value of the driving times through adjusting the different parameters. This was also observed when comparing default FRISO settings with adjusted ones. The fact that the agent implementation has more adjustable parameters and values adds to the possible variation in research that can be done with the FRISO simulator.

To summarize, due to the capabilities of the agents to take into account their environment and taking into account the relations between aspects like velocity and the onset point of a braking curve that is close to empirical observations they were better able to model train driver behaviour than the FRISO and FRISO DBA versions. On top of that the agents included more possibilities for adjustment due to the amount of parameters available. The performance decrease in the running time was deemed to originate from the approach taken to simulate the agents through the event based system available within FRISO.

7 Conclusion

The results and conclusions of the previous chapters can be combined to formulate answers to the main research question and its sub-questions posed at the start of this thesis. The main research question was:

- How can you add train driver behaviour to a micro-level simulator (FRISO), using an Agent based approach?

Which was divided into the following sub-questions:

- How can you model train driver behaviour from data?
- How can you implement train driver behaviour within an Agent?
- How can you implement agents into a micro-level simulator (FRISO)?

The answer to the first sub-question was acquired through processing and visualizing the available data, which revealed important aspects about the actions done when driving a train. The most notable aspects that were observed based on domain knowledge, were the importance of the expectations of the train driver related to the order of the signal aspects and a set of overall reasons for which train drivers do the majority of their actions. Using these aspects and the other information available within the data, decision trees could be learned which served to select which kind of action the modelled train driver is allowed to perform given the state of the environment.

This could then be combined with distributions and functions that specified the onset point, course and endpoint of a selected action. Combining these elements the behaviour of train drivers could be modelled from the available data.

In order to answer the second sub-question and incorporate the train driver model within an agent, an agent setup was designed. The resulting agents were similar in design to a model-based reflex agent [73] and used other concepts found within agent literature. This agent receives input information from FRISO about the environment, which is processed and stored as beliefs. Using beliefs and incorporated domain knowledge, the agent could determine whether or not this new information could warrant an action. If this was the case, the model of the train driver behaviour could then be used to select an action.

The last sub-question was aimed at the implementation of this agent setup and the integration with FRISO. This was done through programming the agent setup within a DLL in C++ that could be attached to FRISO. Within this DLL an agent hub served as the connecting component between FRISO and the individual agents through passing on the relevant inputs and outputs between the agent that is in control of the specified train. During the implementation of this DLL certain assumptions needed to be made and certain difficulties were encountered. These difficulties either had to do with exception cases, bugs or shortcomings related to the interaction with FRISO.

After answering the different sub-questions, the next step was to find out what the effects were of this implementation. The goal here was to find out if the predictive value of the simulations done within FRISO could be improved through a software agent implementation that models train driver behaviour. Looking at the results of the experiments it could be seen that when comparing both the default FRISO train drivers with the default agents to realisation data from PAB, that the agents gave a better predictive value of the driving times. A closer look was taken at the outliers that were present for both FRISO and the agents model. It was concluded that certain aspects of train driving behaviour which were not modelled correctly within FRISO and/or the agents, were the cause of these outliers.

Looking at the results of the experiments done with adjusted settings, it could be seen that improvements could be made through changing the parameters available within the agent implementation and FRISO. It was also concluded that driving times alone were not always a good indication about whether or not the driving behaviour is similar. This was concluded partially due to the differences found between the PAB realisation data and the GPS realisation data. The differences found here also reinforce the importance of knowing the accuracy of the data that is being used.

When looking at the driving behaviours shown through the speed-distance diagrams, it could be seen that there were at times differences between the infrastructure model within FRISO and the actual infrastructure causing different driving behaviour and/or driving times than were seen within the GPS data. It was also seen that differences in driving behaviour and driving times could be caused due to the way traffic control is done within FRISO. More notably, the results in the speed-distance diagrams showed that overall the varied driving behaviour of the agents were closer to the GPS realisation data than FRISO. Indicating that an agent based approach is better able to model and simulate the variation visible in train driver behaviour compared to the current FRISO implementation.

Within both the driving times and the speed-distance diagrams it could be seen that there were deviations. From these it was concluded that certain aspects of train driver behaviour were not modelled correctly by the agents and/or FRISO. Resulting in at times large deviations from the realisation data. When looking closer into these cases the cause could usually be identified. The cases that were looked at often had to do with the following aspects that influenced the driving behaviour:

- The punctuality of the driving time to the stop.
- Differences between the infrastructure present in FRISO and the actual infrastructure.
- Knowledge about the tracks, such as the distance for which a speed limit increase is present.

It was also noted that the used braking criteria and rolling stock can play a significant role on both the driving behaviour and in turn the driving times. The fact that there are outliers which are caused by the above mentioned characteristics, indicates that an approach that is not aimed at re-producing train driver behaviour that can deal with these aspects, has the disadvantage of not having a stable method to deal with outliers. Indicating that if the goal of the simulation process is to acquire results that give a reliable indication of reality, certain significant aspects of train driving behaviour, such as the ones mentioned above, need to be taken into account in a sufficiently accurate manner.

To summarize, an agent based approach to modelling train driver behaviour resulted in the ability to individually and variously represent the behaviour of the trains. This in turn resulted in the simulations done with default agents giving a better predictive value of the driving times than the current default FRISO implementation. It was concluded that driving times alone are not enough when comparing driving behaviour. Due to the agents taking into account their environment and the concepts and expectations that come with it, combined with the empirically based creation of actions, the driving behaviour of the agents lied closer to the behaviour found within the GPS data when compared with the FRISO train drivers. The driving time and driving behaviour results showed that an agent-based approach to modelling train driver behaviour is a viable approach that resulted in more valid simulation method for train driving behaviour within FRISO for ProRail.

The outliers that were found in the driving time results showed that certain aspects of train driver behaviour were not modelled correctly within the agents and/or FRISO. The nature of these aspects indicated that an approach aimed at sufficiently modelling real train driver behaviour, of which the presented agent-based train driver model is a start, is needed if the simulation results are aimed at giving a reliable indication of reality.

7.1 Future research

The findings of this thesis are of course not the endpoint of research into the topic of an agent based approach to modelling train driver behaviour. In this section I will go over some possible future work that could be done, divided into two categories. One category will go into possible improvements and research subjects related to the developed train driver agent. The other category will go into work that is less directly related to the developed train driver agent, but presents possible future research subjects which could be of interest.

7.1.1 Improvements

In the previous section three aspects were presented that form a likely explanation for the at times large deviations found within the driving times and driving behaviour. For two of these, the punctuality aspect and the knowledge of the track aspect, a more in depth look could be taken to find out the influence they have on the driving behaviour and improve the agent model of a train driver. One example of an aspect that could be involved here is the current lack of information regarding the punctuality that the agents have. A real train driver has the ability to view where the train is supposed to be at with a greater precision than is currently modelled within FRISO. In FRISO this is done only at the main time table points that indicate the stations, while a real train driver can also see the time table points in-between the main points, giving thus more detailed information about the current punctuality of the train.

It was noted before that freight trains would need to be explicitly modelled before a full simulation model could be run where freight trains are present. The modelling of freight train drivers would thus enable an improvement in the range of simulations that are possible to do with the agents. A closer look could also be taken at high speed trains and stop-trains in order to improve the agent model. From the results of the experiments it was visible that the results of the 1900 series were closer to the realisation data than other train series. This was an indication that the creation of the agent model was influenced by the location and possibly the rolling stock it was trained upon. A better look at these aspect could be taken through using GPS data from other train series and looking at the differences in the empirical distributions that were used when creating the agent train driver model. From this, it could be possible to look into a way to deal with this possible location and rolling-stock bias.

Looking at the fitted function for the approach to a red stop, it was visible that the *Distance to the stopping point* value was not very accurate. This was likely caused partially due to a low number of observations that were available within the data. Another reason could be due to the braking behaviour towards a red signal already being influenced by the previous braking behaviours. For example, a train driver likely knows when a yellow signal is an indication of an obstructed train in front of it, or if it is just the indication of the next planned stop. This could influence the way the train driver reacts and approaches the yellow signal and in turn the red signal. A more in depth look into this could be done through also taking into account the onset point of the braking manoeuvre caused by the yellow signal before.

Likely not only the *Distance to the stopping point* value could be improved upon through looking more in depth into other aspects that come into play and could influence local behaviours, such as the aspects of punctuality, knowledge about the track and previously experienced obstructions.

7.1.2 Future subjects

As mentioned in Chapter 4, a reactive approach was taken towards the agent design rather than one utilizing desires, intentions and planning. It was noted, that an approach that does plan actions in relation to each other can only be useful the train driver knows the track well. It could be interesting to look into an agent based approach that does utilize the concepts of desires and intentions for planning. For this it would be required that FRISO sends all relevant information to the agent of the expected track sections, such as the signal and sign aspects and location. The inherent variability of the reserving of these track sections would not be a problem due to the agent being able to re-plan its course of action if it observes a deviation from his expectation. The addition of information about the expected track sections would thus also be a way to model the knowledge of a track that a real train driver has. With such an implementation, it would also be possible to let the agents drive to achieve goals such as driving in a safe, timely and energy efficient manner, or other combinations thereof. Another interesting addition could be to have the agents learn from their experiences, to have them take into account often encountered situations like a train in front of a certain station that often causes obstructions.

Research has been done in teams of agents working together with other agents or humans [1, 25, 83]. It could be interesting to see if these concepts could be used for train drivers and traffic controllers. The effects could be studied of having agent train drivers work together through communicating and negotiating with each other to reach their goals [80]. It could also be possible to look into mixed human-agent teams where the agents could help decision making processes that traffic controllers need to do, such as simulating the effects of their actions or suggesting courses of action.

A more in depth study could be done into the possible differences in driving behaviour that exist between different train drivers. Interesting here could be in which aspects their behaviour is different, such as timing, reaction to delays, the influence of the rolling stock, location, etc.

Using an agent based approach, it would also be possible to look into the effects and influence that different aspects of train driving have on the overall punctuality, safety and energy use. One example being the effect of a stricter braking criteria for certain rolling stock series. For this a more accurate way to incorporate the braking criteria for different rolling stock within FRISO would also be desirable. Another improvement within FRISO here would be to use a traction and braking lever positioning method that corresponds with the rolling stock, instead of using a percentage that might not be possible to use within the actual train.

Lastly, more research could be done into using human performance modelling within simulations for assessing aspects like the workload a train driver has to deal with. One example for this can be seen in [38], where they used a cognitive task analysis approach to assess infrastructure and cab drivability. Aspects of this could also be worked into an agent based train driver model, such as the reaction times for doing actions such as moving the traction and braking lever.

Bibliography

[1] Erwin JW Abbink, David GA Mobach, Pieter J Fioole, Leo G Kroon, Eddy HT van der Heijden, and Niek JE Wijngaards. Real-time train driver rescheduling by actor-agent techniques. *Public Transport*, 2(3):249–268, 2010.

[2] Icek Ajzen. The theory of planned behavior. *Organizational behavior and human decision processes*, 50(2):179–211, 1991.

[3] Dennis Baker, Donald Bridges, Regina Hunter, Gregory Johnson, Joseph Krupa, James Murphy, and Ken Sorenson. Guidebook to decision-making methods. *Retrieved from Department of Energy, USA: http://emiweb. inel. gov/Nissmg/Guidebook_2002. pdf*, 2002.

[4] Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences of the United States of America*, 99(Suppl 3):7280–7287, 2002.

[5] Rafael H Bordini, Lars Braubach, Mehdi Dastani, Amal El Fallah-Seghrouchni, Jorge J Gomez-Sanz, Joao Leite, Gregory MP O'Hare, Alexander Pokahr, and Alessandro Ricci. A survey of programming languages and platforms for multi-agent systems. *Informatica (Slovenia)*, 30(1):33–44, 2006.

[6] Rafael H Bordini, Mehdi Dastani, and Michael Winikoff. Current issues in multi-agent systems development. In *Engineering Societies in the Agents World VII*, pages 38–61. Springer, 2007.

[7] Michael E Bratman, David J Israel, and Martha E Pollack. Plans and resource-bounded practical reasoning. *Computational intelligence*, 4(3):349–355, 1988.

[8] Rodney A Brooks. A robust layered control system for a mobile robot. *Robotics and Automation, IEEE Journal of*, 2(1):14–23, 1986.

[9] Jaime G Carbonell, Ryszard S Michalski, and Tom M Mitchell. An overview of machine learning. In *Machine learning*, pages 3–23. Springer, 1983.

[10] Ozkan Celik and Seniz Ertugrul. Predictive human operator model to be utilized as a controller using linear, neuro-fuzzy and fuzzy-arx modeling techniques. *Engineering Applications of Artificial Intelligence*, 23(4):595–603, 2010.

[11] David Chapman. Abstract reasoning as emergent from concrete activity. *Reasoning about actions and plans*, pages 411–424, 1987.

[12] Bo Chen and Harry H Cheng. A review of the applications of agent technology in traffic and transportation systems. *Intelligent Transportation Systems, IEEE Transactions on*, 11(2):485–497, 2010.

[13] Zarrin K Chua and Karen M Feigh. Pilot decision making during landing point designation. *Cognition, technology & work*, 15(3):297–311, 2013.

[14] KM Corker. Human performance simulation in the analysis of advanced air traffic management. In *Simulation Conference Proceedings, 1999 Winter*, volume 1, pages 821–828. IEEE, 1999.

[15] Stefania Costantini and Arianna Tocchio. A logic programming language for multi-agent systems. In *Logics in Artificial Intelligence*, pages 1–13. Springer, 2002.

[16] Andrea D'Ariano. *Improving real-time train dispatching: models, algorithms and applications*. Number T2008/6. Netherlands TRAIL Research School, 2008.

[17] Mehdi Dastani. 2apl: a practical agent programming language. *Autonomous agents and multi-agent systems*, 16(3):214–248, 2008.

[18] Paul Davidsson, Lawrence Henesey, Linda Ramstedt, Johanna Törnquist, and Fredrik Wernstedt. An analysis of agent-based approaches to transport logistics. *Transportation Research part C: emerging technologies*, 13(4):255–271, 2005.

[19] Robyn M Dawes and Bernard Corrigan. Linear models in decision making. *Psychological bulletin*, 81(2):95, 1974.

[20] Hussein Dia. An agent-based approach to modelling driver route choice behaviour under the influence of real-time information. *Transportation Research Part C: Emerging Technologies*, 10(5):331–349, 2002.

[21] Thomas G Dietterich and Ryszard S Michalski. A comparative review of selected methods for learning from examples. In *Machine Learning*, pages 41–81. Springer, 1983.

[22] Chris Drummond, Robert C Holte, et al. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over sampling. In *Workshop on Learning from Imbalanced Datasets II*, volume 11. Citeseer, 2003.

[23] Zied Elouedi, Khaled Mellouli, and Philippe Smets. Belief decision trees: theoretical foundations. *International Journal of Approximate Reasoning*, 28(2):91–124, 2001.

[24] Seniz Ertugrul. Predictive modeling of human operators using parametric and neuro-fuzzy models by means of computer-based identification experiment. *Engineering Applications of Artificial Intelligence*, 21(2):259–268, 2008.

[25] Xiaocong Fan and John Yen. Modeling and simulating human teamwork behaviors using intelligent agents. *Physics of life reviews*, 1(3):173–201, 2004.

[26] Karen M Feigh, Amy R Pritchett, AP Shah, SA Kalaver, A Jadhav, DM Holl, RC Bea, and AZ Gilgur. Analyzing air traffic management systems using agent-based modeling and simulation. 2005.

[27] Richard E Fikes and Nils J Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3):189–208, 1972.

[28] Klaus Fischer, Jorg P Muller, and Markus Pischel. Cooperative transportation scheduling: an application domain for dai. *Applied Artificial Intelligence*, 10(1):1–34, 1996.

[29] Stan Franklin and Art Graesser. Is it an agent, or just a program?: A taxonomy for autonomous agents. In *Intelligent agents III agent theories, architectures, and languages*, pages 21–35. Springer, 1997.

[30] János Fülöp. Introduction to decision making methods. In *BDEI-3 Workshop, Washington*, 2005.

[31] Johannes Fürnkranz, Dragan Gamberger, and Nada Lavrač. *Foundations of rule learning*. Springer, 2012.

[32] Michael P Georgeff and Amy L Lansky. Reactive reasoning and planning. In *AAAI*, volume 87, pages 677–682, 1987.

[33] Nasser Ghasem-Aghaee and TI Oren. Towards fuzzy agents with dynamic personality for human behavior simulation. In *SUMMER COMPUTER SIMULATION CONFERENCE*, pages 3–10. Society for Computer Simulation International; 1998, 2003.

[34] CJ Goodman, LK Siu, and TK Ho. A review of simulation models for railway systems. In *International Conference on Developments in Mass Transit Systems 1998*, volume 453. IEEE, 1998.

[35] Brian F Gore. Man–machine integration design and analysis system (midas) v5: Augmentations, motivations, and directions for aeronautics applications. In *Human Modelling in Assisted Transportation*, pages 43–54. Springer, 2011.

[36] Rob MP Goverde. *Punctuality of railway operations and timetable stability analysis*. Netherlands TRAIL Research School, 2005.

[37] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: and update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[38] W Ian Hamilton and Theresa Clarke. Driver performance modelling and its practical application to railway safety. *Applied Ergonomics*, 36(6):661–670, 2005.

[39] Clinton Heinze, Simon Goss, and Adrian Pearce. Plan recognition in military simulation: Incorporating machine learning with intelligent agents. In *Proceedings of IJCAI-99 Workshop on Team Behaviour and Plan Recognition*, pages 53–64, 1999.

[40] Koen V Hindriks, Frank S De Boer, Wiebe Van der Hoek, and John-Jules Ch Meyer. Agent programming in 3apl. *Autonomous Agents and Multi-Agent Systems*, 2(4):357–401, 1999.

[41] Nick Howden, Ralph Rönnquist, Andrew Hodgson, and Andrew Lucas. Jack intelligent agents-summary of an agent infrastructure. In *5th International conference on autonomous agents*, 2001.

[42] Marcus J Huber. Jam: A bdi-theoretic mobile agent architecture. In *Proceedings of the third annual conference on Autonomous Agents*, pages 236–243. ACM, 1999.

[43] ProRail AM Infrasystemen. *Ontwerpvoorschrift: Plaatsing en toepassing van borden (OVS69133-2-V002)*. ProRail, 2012.

[44] ProRail AM Infrasystemen. *Ontwerpvoorschrift: Plaatsing en toepassing van lichtseinen (OVS69133-1-V004)*. ProRail, 2012.

[45] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[46] Nicholas R Jennings, Katia Sycara, and Michael Wooldridge. A roadmap of agent research and development. *Autonomous agents and multi-agent systems*, 1(1):7–38, 1998.

[47] Randolph M Jones, John E Laird, Paul E Nielsen, Karen J Coulter, Patrick Kenny, and Frank V Koss. Automated intelligent pilots for combat flight simulation. *AI magazine*, 20(1):27, 1999.

[48] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *arXiv preprint cs/9605103*, 1996.

[49] David Kinny, Michael Georgeff, and Anand Rao. A methodology and modelling technique for systems of bdi agents. In *Agents breaking away*, pages 56–71. Springer, 1996.

[50] Sotiris B Kotsiantis, ID Zaharakis, and PE Pintelas. Supervised machine learning: A review of classification techniques, 2007.

[51] Harald Krueger, Eric Vaillancourt, Ann M Drummie, Steve J Vucko, and Joe Bekavac. Simulation within the railroad environment. In *Proceedings of the 32nd conference on Winter simulation*, pages 1191–1200. Society for Computer Simulation International, 2000.

[52] Leon S Lasdon, Allan D Waren, Arvind Jain, and Margery Ratner. Design and testing of a generalized reduced gradient code for nonlinear programming. *ACM Transactions on Mathematical Software (TOMS)*, 4(1):34–50, 1978.

[53] Raanan Lipshitz, Gary Klein, Judith Orasanu, and Eduardo Salas. Taking stock of naturalistic decision making. *Journal of Behavioral Decision Making*, 14(5):331–352, 2001.

[54] Cora JM Maas and Joop J Hox. Robustness issues in multilevel regression analysis. *Statistica Neerlandica*, 58(2):127–137, 2004.

[55] Charles C Macadam. Understanding and modeling the human driver. *Vehicle System Dynamics*, 40(1-3):101–134, 2003.

[56] Frank J Massey Jr. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78, 1951.

[57] Martijn Mes, Matthieu Van Der Heijden, and Aart Van Harten. Comparison of agent-based scheduling to look-ahead heuristics for real-time transportation problems. *European Journal of Operational Research*, 181(1):59–75, 2007.

[58] John A Michon. *A critical view of driver behavior models: what do we know, what should we do?* Springer, 1985.

[59] Microsoft. Microsoft excel 2010 [computer software], 2010.

[60] AD Middelkoop and L Loeve. Simulation of traffic management with friso. In *Computers in Railways X. The Tenth International Conference*, 2006.

[61] Dick Middelkoop and Michiel Bouwman. Simone: large scale train network simulations. In *Proceedings of the 33nd conference on Winter simulation*, pages 1042–1047. IEEE Computer Society, 2001.

[62] Tom M Mitchell. Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, 45, 1997.

[63] Harvey J Motulsky and Lennart A Ransnas. Fitting curves to data using nonlinear regression: a practical and nonmathematical review. *The FASEB journal*, 1(5):365–374, 1987.

[64] Andrew Nash and Daniel Huerlimann. Railroad simulation using opentrack. *Computers in railways IX*, pages 45–54, 2004.

[65] TI Ören. Impact of data on simulation: from early practices to federated and agent-directed simulations. In *Proceedings of EUROSIM*, pages 26–29, 2001.

[66] Praveen Paruchuri, Alok Reddy Pullalarevu, and Kamalakar Karlapalem. Multi agent simulation of unorganized traffic. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1*, pages 176–183. ACM, 2002.

[67] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[68] John Ross Quinlan. *C4. 5: programs for machine learning*, volume 1. Morgan kaufmann, 1993.

[69] Alfons Radtke and Dirk Hauptmann. Automated planning of timetables in large railway networks using a microscopic data basis and railway simulation techniques. *Publication of: WIT Press*, 2004.

[70] John S Ramberg, Edward J Dudewicz, Pandu R Tadikamalla, and Edward F Mykytka. A probability distribution and its uses in fitting data. *Technometrics*, 21(2):201–214, 1979.

[71] Anand S Rao, Michael P Georgeff, et al. Bdi agents: From theory to practice. In *ICMAS*, volume 95, pages 312–319, 1995.

[72] John Rice. *Mathematical statistics and data analysis*. Cengage Learning, 2006.

[73] Stuart Russell and Peter Norvig. A modern approach. 1995.

[74] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.

[75] Kurt Schultze. *Modell für die Asynchrone Simulation des Betriebes in Teilen des Eisenbahnnetzes*. Number 38. Aachen, 1985.

[76] NS Reizigers sectie regelgeving. *Handboek Machinist*. NS, 2006.

[77] NS Reizigers sectie Spoorwegveiligheid en Regelgeving. *Seinenboek*. NS, 2005.

[78] Yoav Shoham and Kevin Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2008.

[79] Rob Sluijter. Knmi [website]. http://www.knmi.nl/klimatologie/maand_en_seizoensoverzichten/maand/feb14.html, 2014.

[80] Peter Stone and Manuela Veloso. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8(3):345–383, 2000.

[81] M a Sugeno and M Nishida. Fuzzy control of model car. *Fuzzy sets and systems*, 16(2):103–113, 1985.

[82] R Core Team et al. R: A language and environment for statistical computing. 2012.

[83] David Traum, Jeff Rickel, Jonathan Gratch, and Stacy Marsella. Negotiation over tasks in hybrid human-agent teams for simulation-based training. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 441–448. ACM, 2003.

[84] Sherman W Tyler, Christian Neukom, Michael Logan, and Jay Shively. The midas human performance model. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 42, pages 320–324. SAGE Publications, 1998.

[85] William N Venables and Brian D Ripley. *Modern applied statistics with S*. Springer, 2002.

[86] Joachim Wahle, Ana Lúcia C Bazzan, Franziska Klügl, and Michael Schreckenberg. The impact of real-time information in a two-route scenario using agent-based simulation. *Transportation Research Part C: Emerging Technologies*, 10(5):399–417, 2002.

[87] Wolfram|Alpha. Wolfram alpha [website]. http://www.wolframalpha.com/input/?i=v+%3D+sqrt%28c%5E2%2B2*a*%28x-%286.7139*v-104.51%29%29%29+solve+for+v, 2014.

[88] Michael Wooldridge and Nicholas R Jennings. Intelligent agents: Theory and practice. *The knowledge engineering review*, 10(02):115–152, 1995.

[89] Seiji Yasunobu and Shoji Miyamoto. Automatic train operation system by predictive fuzzy control. *Industrial applications of fuzzy control*, 1(18):1–18, 1985.

Glossary

Automatic Train Protection (ATP)	*Automatische treinbeïnvloeding (ATB)*	A safety system that dictates the speed limit with the following values: 40, 60, 80, 130 and 140.
Braking criteria	*Remcriterium*	The minimum deceleration required for the ATP to register a braking manoeuvre. If the train is required to slow down, but does not meet this criteria, the train will be forced to a top.
Cab	*Cabine*	The part of the train where the train driver controls the train from.
Coast	*Uitrollen*	The situation where neither traction nor braking lever are activated, and the train is moving only based on its momentum.
Cruise	*Cruise*	The situation where the train driver positions the traction lever in a way to maintain the current velocity.
Driving time Until Emergency braking curve (DUE)	*Rijtijd Tot Snelremcurve (RTS), ook bekend als: Time to STS*	The driving time until the train reaches the position of the emergency brake curve, which refers to the maximum braking curve a train can do in order to stop before a specific position.
Free track	*Vrije baan*	The sections of tracks between two TCA's.
Rolling stock	*Railvoertuigen*	Vehicle that can move over the railway
Scout sign	*Verkenbord*	A railway sign that indicates that the next platform is at braking distance.

Signal aspect	*Sein aspect*	The indication that is given by a railway signal, such as a green 8 indication.
Time table	*Dienstregeling*	The planned times for all trains to arrive, depart, passage, etc, at specified TCA's.
Timetable control area (TCA)	*Dienstregelpunt*	A specified area or point that is involved with the time table. Serving as aiming points and indicators for the punctuality of the trains.
Pantograph	*Pantograaf*	A common type of current collector mounted atop of the train that makes contact with the overhead power lines.
Traction/brake lever	*Tractie/rem hendel*	The two levers that are present in most passenger rolling stock that control the amount of traction or braking that is applied. Each of these have a set of possible positions indicating the amount of traction or braking that is applied.
Traffic control	*Verkeersleiding*	The department of ProRail that is tasked with directing train traffic through controlling switches and signals where necessary.
Universal Economical driving Idea (UEI)	*Univerzeel Zuinig rijden Idee (UZI)*	A train driver method that is focussed on driving energy efficient through following a set of rules.

Appendix 1: MATRICS logs

In this appendix, the different logs that were created from the MATRICS data are outlined, with their attributes described.

Log: ***Approaching signal/sign lower speed:***		
Entry ID:	Unit	Description
Signal One	Signal ID	What kind of signal aspect (Colour and number) is currently in front of the train.
Signal Two	Signal ID	What kind of signal aspect (Colour and number) is after the current signal in front of the train
Current velocity	Km/h	The current velocity of the train.
Distance to relevant signal	Meter	The distance to the signal that forces the speed restriction in question. Note that this can also be negative.
ATP current velocity difference	Km/h	The difference between the current velocity and the current velocity allowed by the ATP.
ATP expected velocity difference	Km/h	The difference between the current velocity and the expected velocity allowed by the ATP after the next signal passage.
Distance to view point	Meter	The distance to the point where the signal was visible.
Current action	Action ID	What action the train is currently doing. Either traction, braking or coasting.
Time to relevant signal passage	Seconds	The time to the signal that forces the speed restriction in question. Note that this can also be negative.
Signal one blinking	Boolean	Whether signal one is blinking.
Final velocity	Km/h	If the train starts to brake at this point, until which speed the train brakes.
Final velocity compared to expected max	Km/h	If the train starts to brake at this point, how the final velocity compares to the current expected maximum velocity.

Final velocity compared to ATP	Km/h	If the train starts to brake at this point, how the final velocity compares to the maximum velocity allowed at the point the train stops braking.
Expected max	Km/h	The expected maximum allowed velocity after the next signal passage.
Final ATP	Km/h	If the train starts to brake at this point, the maximum allowed velocity by the ATP at the point where the train stops braking.
Previous action time passed	Seconds	The time that has passed since the train driver last did an action.
Previous action distance	Meter	The distance to the last action the train driver did.
Previous action kind	Action ID	The kind of action the train driver last did.
Second signal colour when passing	Colour ID	The signal aspect of the second signal when the train passes it.
Velocity at second signal	Km/h	The velocity of the train at the second signal.
ATP at second signal	Km/h	The ATP maximum allowed velocity after the second signal.
Velocity at second signal compared to max	Km/h	The velocity of the train at the second signal compared to the maximum allowed velocity after the second signal.
Braking	Traction ID	Whether the train driver decided to give throttle, to start braking, or to start coasting at this point.

Log: ***Approaching signal/sign higher speed & Departure:***		
Entry ID:	Unit	Description
Signal One	Signal ID	What kind of signal aspect (Colour and number) is currently in front of the train.
Signal Two	Signal ID	What kind of signal aspect (Colour and number) is after the current signal in front of the train
Current velocity	Km/h	The current velocity of the train.
Distance to relevant signal	Meter	The distance to the signal that forces the speed restriction in question. Note that this can also be negative.
ATP current velocity difference	Km/h	The difference between the current velocity and the current velocity allowed by the ATP.

ATP expected velocity difference	Km/h	The difference between the current velocity and the expected velocity allowed by the ATP after the next signal passage.
Current action	Action ID	What action the train is currently doing. Either traction, braking or coasting.
UZI value difference	Km/h	What the difference in velocity is between the current velocity and the UZI value.
UZI final difference	Km/h	If the train starts to give throttle at this point, what the difference in velocity is between the final velocity after speeding up and the UZI value.
Delay	Seconds	What the current delay is, timed from the last time table control area.
Velocity compared to previous coasting onset	Km/h	What the difference is between the current velocity, and the velocity at the last point where the train started coasting.
Previous action time passed	Seconds	The time that has passed since the train driver last did an action.
Time since last signal improvement	Seconds	The time that has passed between now and the last time the train driver saw a signal aspect improve.
Expected max	Km/h	The expected maximum allowed velocity after the next signal passage.
Final velocity	Km/h	If the train starts to give throttle at this point, until which speed the train accelerates.
Final velocity compared to expected max	Km/h	If the train starts to give throttle at this point, how the final velocity compares to the current expected maximum velocity.
Final velocity compared to current max	Km/h	If the train starts to give throttle at this point, how the final velocity compares to the current maximum velocity allowed.
Velocity difference	Km/h	If the train starts to give throttle at this point, how the onset and final velocity compare.
Time compared to event	Seconds	If the train does an action, how the time of that action compares to the time when the signal/sign first became visible.
Previous action distance	Meter	The distance to the last action the train driver did.
Previous action kind	Action ID	The kind of action the train driver last did.
Throttle	Traction ID	Whether the train driver decided to give throttle, to start braking, or to start coasting at this point.

Log: ***Approaching a red signal & Approaching a planned stop:***		
Entry ID:	Unit	Description
Signal One	Signal ID	What kind of signal aspect (Colour and number) is currently in front of the train.
Current velocity	Km/h	The current velocity of the train.
ATP current velocity difference	Km/h	The difference between the current velocity and the current velocity allowed by the ATP.
Current action	Action ID	What action the train is currently doing. Either traction, braking or coasting.
Situation	Situation ID	An ID to tell whether this action was done while approaching a red signal or a planned stop.
Distance to red signal / Distance to arrival	Meters	In the situation of approaching a red signal: Distance to the next red signal. In the situation of approaching a planned stop: Distance to arrival.
Driving time Until Safety braking curve	Seconds	The time until the critical point where, if the train were to break fully, it would stop at the position of the red signal.
Braking number	Integer	How many times the train has already braked while in the current situation.
Previous action kind	Action ID	The kind of action the train driver last did.
Previous action time passed	Seconds	The time that has passed since the train driver last did an action.
Previous action velocity difference	Km/h	The difference in velocity between the current point and the last time the train driver did an action.
Braking	Traction ID	Whether the train driver decided to give throttle, to start braking, or to start coasting at this point.

Appendix 2: GPS logs

In this appendix, the different logs that were created from the GPS data are outlined, with their attributes described. The following attributes were logged with the intent of using them to acquire the desired information within the desired circumstances.

Log: *Braking for a stop:*		
Entry ID:	Unit	Description
Initial velocity	m/s	The velocity of the train at the onset of the braking action.
Final velocity	m/s	The velocity of the train at the end of the braking action.
Diff velocity	m/s	The difference in velocity between the initial and final velocity.
Average acceleration	m/s^2	The average acceleration between the onset and end of the braking action.
Max deceleration	m/s^2	The maximum deceleration that was logged between the onset and end of the braking action.
Distance travelled	Meter	The distance between the onset and end of the braking action.
Initial Driving time Until Emergency braking curve	Seconds	The DUE at the onset of the braking action.
Final Driving time Until Emergency braking curve	Seconds	The DUE at the end of the braking action.
Initial distance to red	Meter	The distance to the next red signal at the onset of the braking action.
Initial distance to stop	Meter	The distance to the stopping position of the train.
Braking number	Integer	The number of times the train has already done a braking action during this approach.

Log: ***Braking to red:***		
Entry ID:	Unit	Description
Initial velocity	m/s	The velocity of the train at the onset of the braking action.
Final velocity	m/s	The velocity of the train at the end of the braking action.
Diff velocity	m/s	The difference in velocity between the initial and final velocity.
Average acceleration	m/s^2	The average acceleration between the onset and end of the braking action.
Max deceleration	m/s^2	The maximum deceleration that was logged between the onset and end of the braking action.
Distance travelled	Meter	The distance between the onset and end of the braking action.
Initial Driving time Until Emergency braking curve	Seconds	The DUE at the onset of the braking action.
Final Driving time Until Emergency braking curve	Seconds	The DUE at the end of the braking action.
Initial distance to red	Meter	The distance to the next red signal at the onset of the braking action.
Initial distance to stop	Meter	The distance to the stopping position of the train.

Log: ***Braking:***		
Entry ID:	Unit	Description
Initial velocity	m/s	The velocity of the train at the onset of the braking action.
Final velocity	m/s	The velocity of the train at the end of the braking action.
Diff velocity	m/s	The difference in velocity between the initial and final velocity.
Average acceleration	m/s^2	The average acceleration between the onset and end of the braking action.
Max deceleration	m/s^2	The maximum deceleration that was logged between the onset and end of the braking action.
Distance travelled	Meter	The distance between the onset and end of the braking action.

Log: ***Traction:***		
Entry ID:	Unit	Description
Initial velocity	m/s	The velocity of the train at the onset of the braking action.
Final velocity	m/s	The velocity of the train at the end of the braking action.
Diff velocity	m/s	The difference in velocity between the initial and final velocity.
Average acceleration	m/s^2	The average acceleration between the onset and end of the braking action.
Max acceleration	m/s^2	The maximum acceleration that was logged between the onset and end of the braking action.
Distance travelled	Meter	The distance between the onset and end of the braking action.

Log: ***Traction behaviour:***		
Entry ID:	Unit	Description
Number of traction sections	Integer	The number of traction sections that are distinguished.
Average velocity difference	m/s	The average velocity difference between the onset and end of the traction actions.
Percentage distance traction sections	%	The percentage of travelled the distance where the train has given traction.
Number of braking sections	Integer	The number of braking sections that are distinguished.
Average speed difference braking sections	m/s	The average speed difference within the braking sections.
Total speed difference braking sections	m/s	The total speed difference within the braking sections.
Number of cruise sections	Integer	The number of cruise sections that are distinguished.
Total distance cruise sections	Meter	The total distance covered by the cruise sections.
Percentage distance cruise sections	%	The percentage of the travelled distance where the train was cruising.

Average distance cruise sections	Meter	The average distance covered by a cruise section.
Number of coast sections	Integer	The number of coast sections that are distinguished.
Total distance coast sections	Meter	The total distance covered by the coast sections.
Average distance coast sections	Meter	The average distance covered by a coast section.
Average speed difference coast sections	m/s	The average speed difference within the coast sections.
Percentage distance coast sections	%	The percentage of the travelled distance where the train was coasting.
Time difference departure arrival	Seconds	The time between the departure and arrival points.
Distance difference departure arrival	Meter	The distance between the departure and arrival points.
Arrival location	String	The name of the arrival location.
Departure location	String	The name of the departure location.
Arrival time	Minute	The time of arrival.
Total logged distance	Meter	The total distance over which the sections are logged.
Arrival delay	Minute	The arrival time of this train compared to the planned time.
Departure delay	Minute	The departure time of this train compared to the planned time.
Average speed logged distance	m/s	The average velocity over the logged section.

Appendix 3: Classification logs

In this appendix an overview of the classification logs is given. The logging conditions for determining in which situation the train finds itself, is described above the tables that contain the used attributes. A description of these attributes can be found in Appendix 2 Note that the viewing distance is set at 500 meters, given that within the MATRICS version used for the data acquisition, the viewing distance was close to this number.

Logging conditions *Approaching signal/sign lower speed:*

- The train driver can see a new signal
- The colour aspect of this signal is not equal to red
- The velocity of the train will be higher than the maximum speed allowed by that signal at the time of passing that signal.

Log: ***Approaching signal/sign lower speed:***
Entry ID:
Signal One.
Signal Two.
Current velocity.
Distance to relevant signal.
ATP current velocity difference.
ATP expected velocity difference.
Distance to view point.
Current action.
Time to relevant signal passage.
Signal one blinking.
Expected max.
Previous action time passed.

Previous action distance.
Previous action kind.
Braking.

Logging conditions *Approaching signal/sign higher speed:*

- The train driver can see a new signal
- The colour aspect of this signal is green, or: The colour aspect of this signal is neither green nor red and the velocity of the train will be lower than the maximum speed allowed by that signal at the time of passing that signal.

Log: *Approaching signal/sign higher speed & Departure:*
Entry ID:
Signal One.
Signal Two.
Current velocity.
ATP current velocity difference.
ATP expected velocity difference.
Current action.
UZI value difference.
Delay.
Previous action time passed.
Time since last signal improvement.
Expected max.
Previous action distance.
Previous action kind.
Throttle.

Logging conditions *Approaching a red signal:*

- The train driver can see a new signal, that signals colour aspect is red and it is not a red signal for a planned stop.

Logging conditions *Approaching a planned stop:*

- The train is within 1000 meters of the next stopping position and there is no red signal present before the stopping position.

Log: ***Approaching a red signal & Approaching a planned stop:***
Entry ID:
Signal One.
Current velocity.
ATP current velocity difference.
Current action.
Situation.
Distance to red signal / Distance to arrival.
DUE.
Braking number.
Previous action kind.
Previous action time passed.
Previous action velocity difference.
Braking.

Appendix 4: Driving behaviours

In this appendix some examples of driving behaviour can be seen. The data used for these graphs comes from the GPS data. On the x-axes of the graphs the distance the train has travelled since the start of the logging can be seen in kilometres. On the y-axes the velocity can be seen in km/h. The colours represent the categorization of the action attributed by the processing algorithm used. With green indicating a section where significant traction was given. In red a section where the train was braking. In black a section where the train was coasting and in white a section where the train was cruising or slightly accelerating.

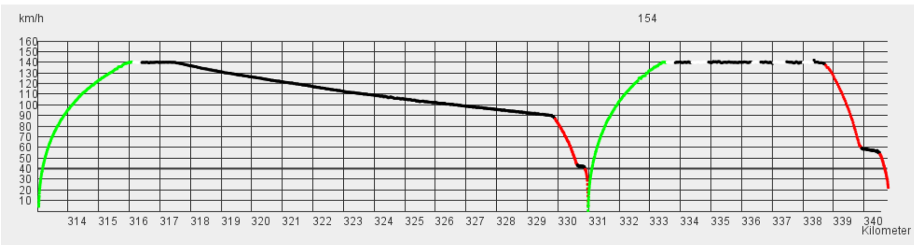


Figure .1: *Here the train driver coasts for a large percentage of the distance before its first stop. After accelerating again, he maintains the speed limit.*

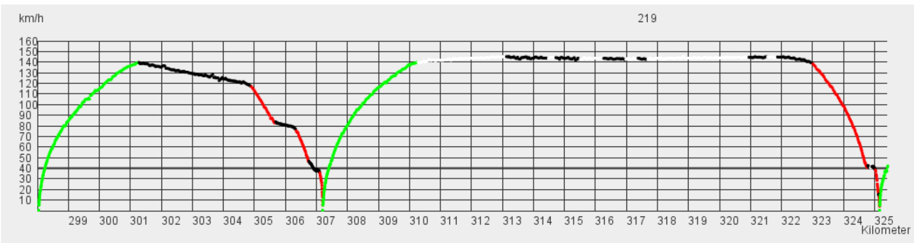


Figure .2: *Here the train driver coasts for before its first stop. After accelerating again, he maintains the speed limit.*

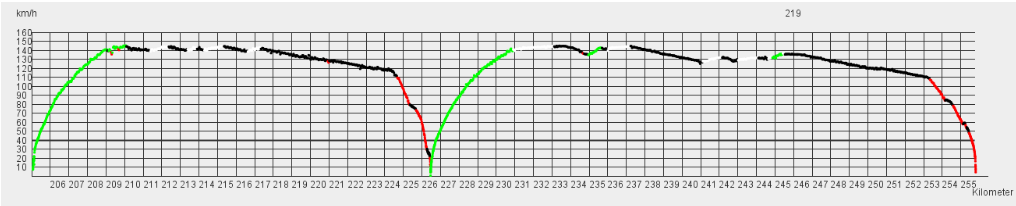


Figure .3: *Here the train driver switches between accelerating slowly and coasting around the speed limits, after which he coasts a while before the first stop. In the second section the train driver seems to switch between maintaining the trains velocity around the speed limit and coasting.*

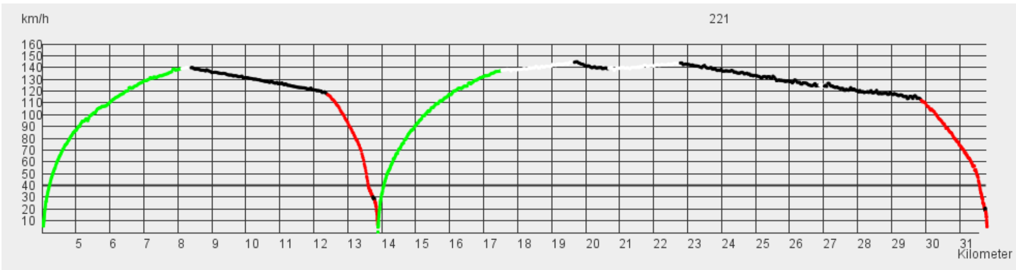


Figure .4: *Here the train driver coasts for before its first stop. After accelerating again, he maintains the maximum allowed velocity through alternating a slow acceleration and coasting, after which he coasts towards the final braking manoeuvre.*

Appendix 5: Setup messages

In this appendix a table can be seen which contains the information that can be within the setup messages sent from FRISO to the DLL. For an overview of the data types within the below mentioned attributes, see Appendix 6.

Message	Attribute	Description
Scenario Settings:	DepartsOnBetterThanYellow	Whether or not the train is allowed to depart on a yellow signal.
Time Table:	TrainList	An array of the trains that will be present within the model, with the time table of the trains included.
PlaceTrainInModel:	CurrentSimulationTime	The current simulation time.
	TrainState	The current state of the train.
	SignalList	A list of the signals in front of the train.
	SpeedSignList	A list of the speed signs in front of the train.
	Switch	The location of the last passed switch.
	MaxSpeedSignals	The speed limit according to the signals.
	MaxSpeedSpeedSigns	The speed limit according to the speed signs.

Appendix 6: Precept input data types

In this appendix an overview is given of the data types that are present within the messages between FRISO and the agents.

Simple data types:	
Attribute	**Data type**
CurrentSimulationTime	String
TrainState	TrainState
SignalList	SignalArray
SpeedSignList	SpeedSignArray
Switch	Switch
MaxSpeedSignals	Double
MaxSpeedSpeedSigns	Double
NextSignal	Signal
TrainID	Integer
TrainName	String
ReplicationNumber	Integer
RealisationType	String
ActivityType	String
TCA	String
OriginalSignal ID	Integer
OriginalSignal Name	String
DeterminedSignal	Signal
ATBSpeed	Double

Complex data types:		
Name	**Attribute**	**Data type**
TrainState	TrainID	Integer
	TrainName	String
	ReplicationNumber	Integer
	CurrentPosition	Double
	CurrentSpeed	Double
	CurrentAcceleration	Double
	PositionBrakeLever	Integer
	PositionTractionLever	Integer
Signal	SignalID	Integer
	SignalName	String
	Type	Integer
	Height	Integer
	DistanceFromStartPoint	Double
	CurrentSignalAspect	String
SpeedSign	SpeedSignID	Integer
	SpeedSignName	String
	Type	String
	DistanceFromStartPoint	Double
	Speed	Double
Switch	SwitchID	Integer
	SwitchName	String
	DistanceFromStartPoint	Double
Train	Train ID	Integer
	TrainName	String
	ReplicationNumber	Integer
	TrainPlan	PlanActivityArray
	TrainLength	Double
	Maximum Deceleration	Double
	TrainType	String
PlanActivity	ActivityType	String
	SequenceNumber	Integer
	PlannedTime	String
	ActivityTCA	String
	StopSignal	Signal

Appendix 7: Agent input messages

Below a table containing a specification of the content of the messages received from FRISO by the agent. A specification of the attributes can be found in Appendix 6.

Message	Attribute
RealisationTrainActivity	CurrentSimulationTime
	TrainID
	TrainName
	ReplicationNumber
	RealisationType
	ActivityType
	TCA
DepartureAllowed	CurrentSimulationTime
	TrainState
StopSignalDetermined	CurrentSimulationTime
	TrainState
	OriginalSignal ID
	OriginalSignal Name
	DeterminedSignal
SignalAspectImprovedOfNextSignal	CurrentSimulationTime
	TrainState
	SignalList
DesiredSpeedReached	CurrentSimulationTime
	TrainState
DesiredPositionReached	CurrentSimulationTime
	TrainState
SignalPassageFrontTrain	CurrentSimulationTime
	TrainState
	SignalList

SpeedSignPassageFrontTrain	CurrentSimulationTime
	TrainState
	SpeedSignList
SwitchPassageFrontTrain	CurrentSimulationTime
	TrainState
	Switch

Appendix 8: Agent components

In this appendix schematic overviews of the components of the agent can be seen. An overview of the decision making component can be found in Chapter 3.

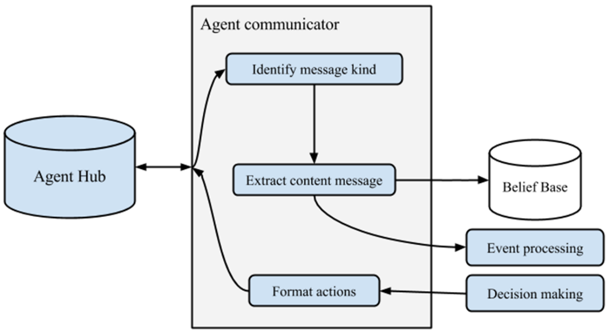


Figure .5: *A schematic overview of the workings of the Agent communicator component of the agent.*

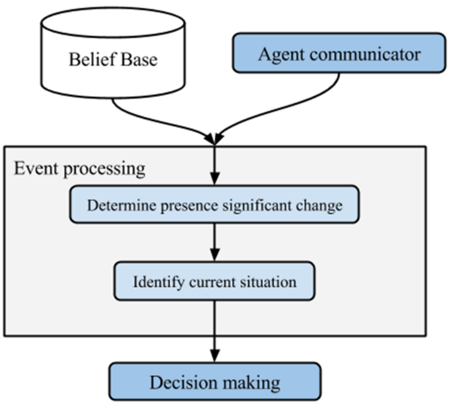


Figure .6: *A schematic overview of the workings of the Event processing component of the agent.*

Appendix 9: Class diagram

In this appendix the structure of the agents DLL can be seen through two UML class diagrams.

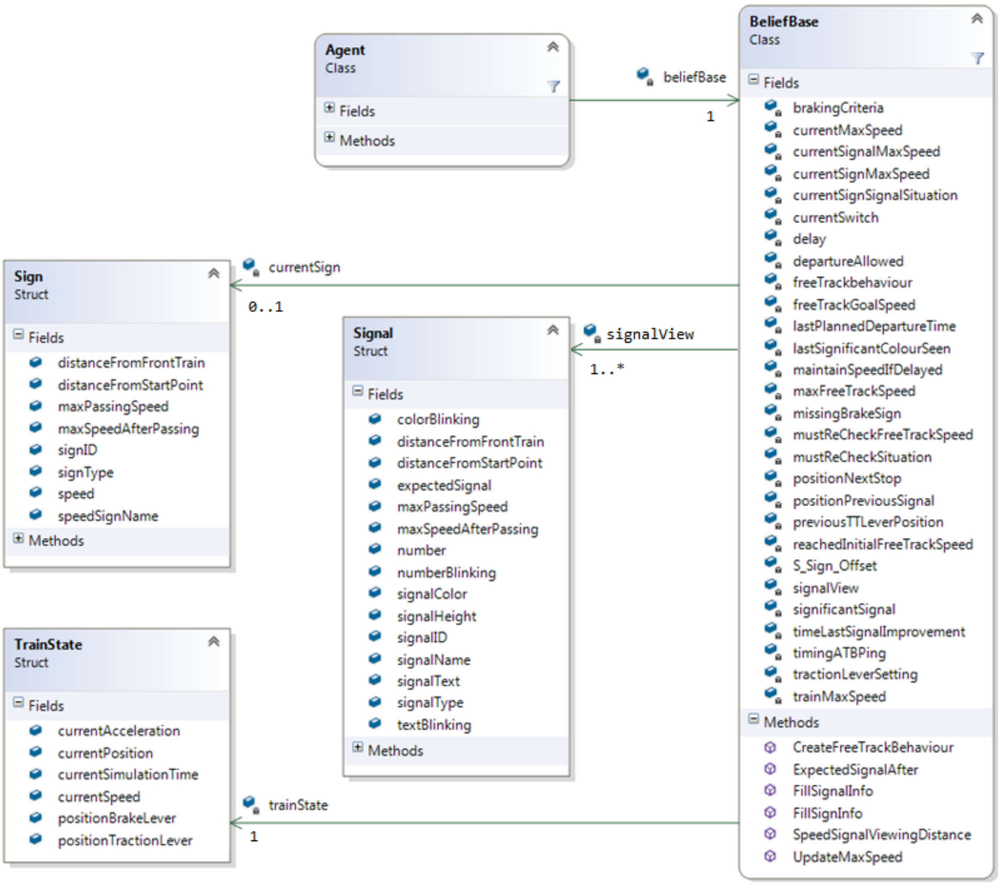


Figure .7: *A UML class diagram of the structure of a part of the DLL.*

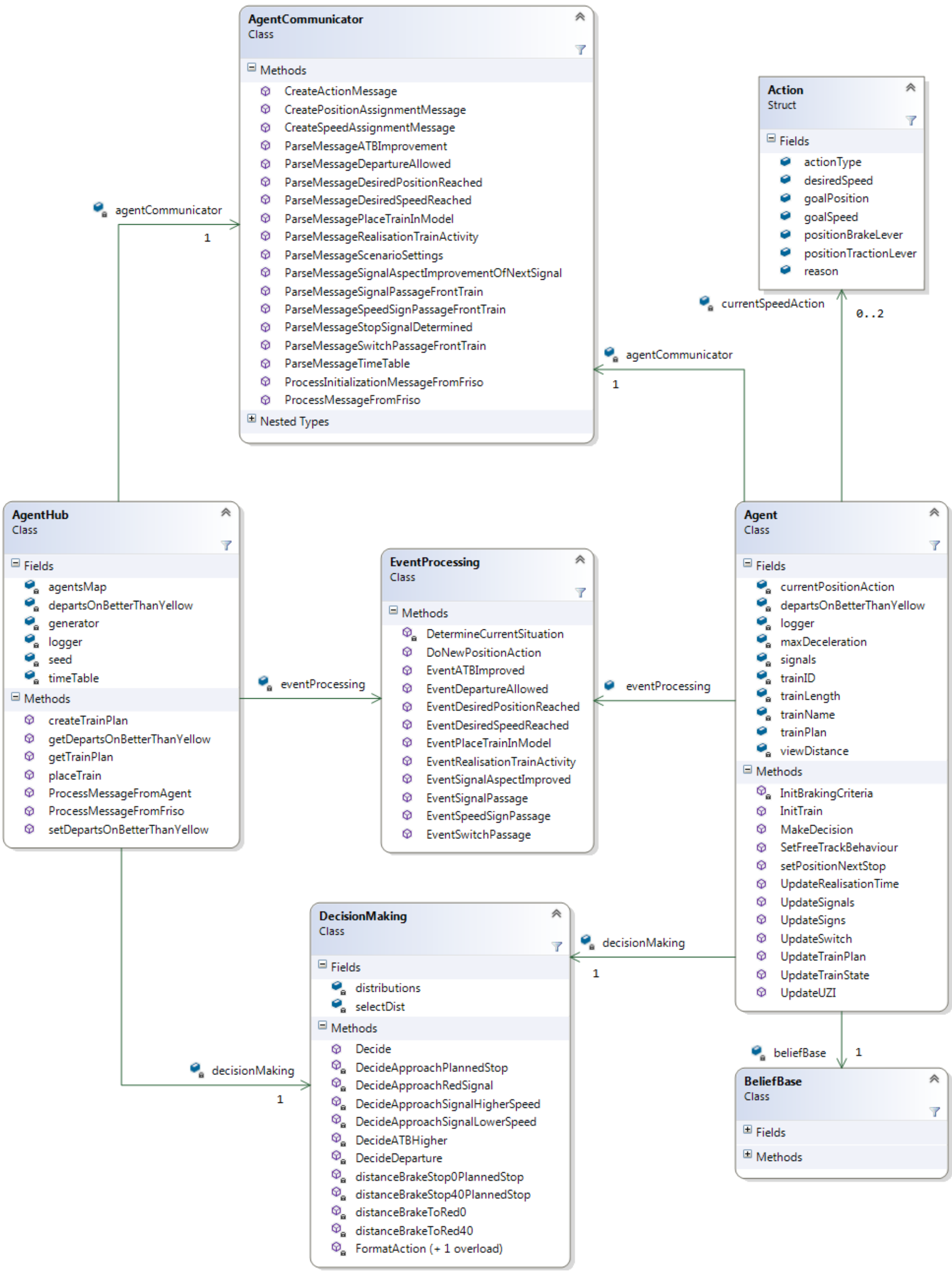


Figure .8: *A UML class diagram of the structure of a part of the DLL.*

Appendix 10: Effect of departure delays

In this appendix the effect of the arrival location is shown on the driving time delay. The driving time delay indicates the delay that is acquired between two stations compared to the planned driving time, where a positive value indicates a faster driving time then planned. Note that for simplicities sake the departure location is not taken into account here.

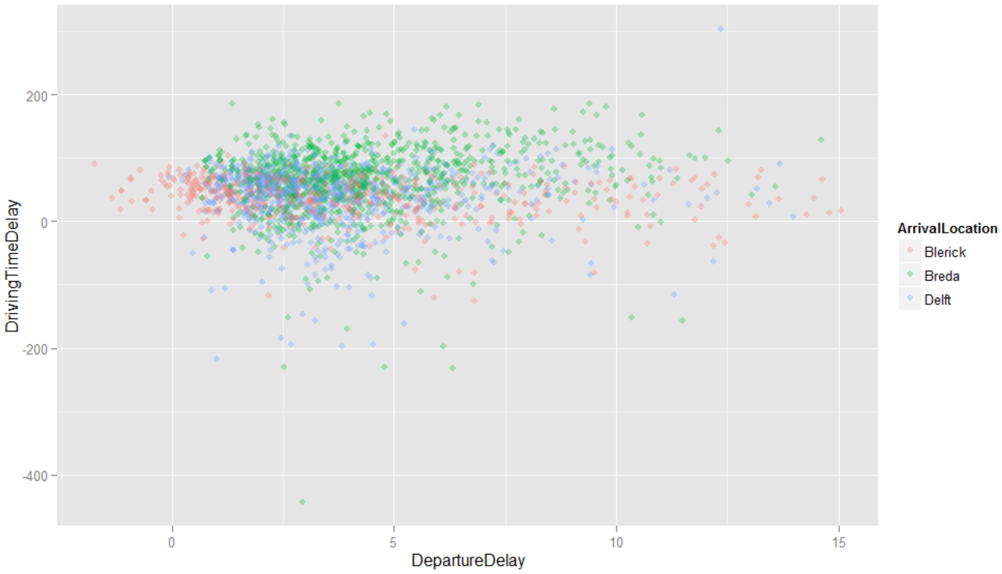


Figure .9: *A scatter plot indicating the relation between the delay a train has at its departure, the driving time it has to the next station compared to the planned driving time, and the arrival location. On the x-axes the delay at departure can be seen. On the y-axes the difference between the driving time to the next station and the planned driving time can be seen in seconds. The colour indicates the arrival location.*

Appendix 11: Delay exceptions

In this appendix the exceptions to the filtering of the PAB driving times are noted. The norm was to exclude trains which deviated from the planned time with more than 180 seconds (3 minutes).

Train series:	Direction	Exceptions
1900	T	Trains that were more than 180 seconds early were not excluded from Tb to Ehv, due to this happening quite often.
1900	T	The delay time was set at 210 seconds between Ehv and Hm due to trains frequently taking more than 180 seconds longer than the planned time.
4300	T	The delay time was set at 240 seconds between Wp and Ampo due to trains frequently taking more than 180 seconds longer than the planned time.
2600	H	The delay time was set at 380 seconds between Ledn and Shl due to trains frequently taking more than 180 seconds longer than the planned time.
2600	T	Trains that were more than 180 seconds early were not excluded from Tb to Ehv, due to this happening quite often.
2600	T	Trains that were more than 180 seconds early were not excluded from Tb to Ehv, due to this happening quite often.
5600	T	Trains that were more than 180 seconds early were not excluded from Wz to Zl, due to this happening quite often.

Table .17: *Overview of the exceptions that were made when filtering PAB driving times.*

Appendix 12: Speed-distance diagrams

In this appendix a collection of speed-distance diagrams are shown, indicating differences in the driving behaviour between FRISO and the agents implementation.

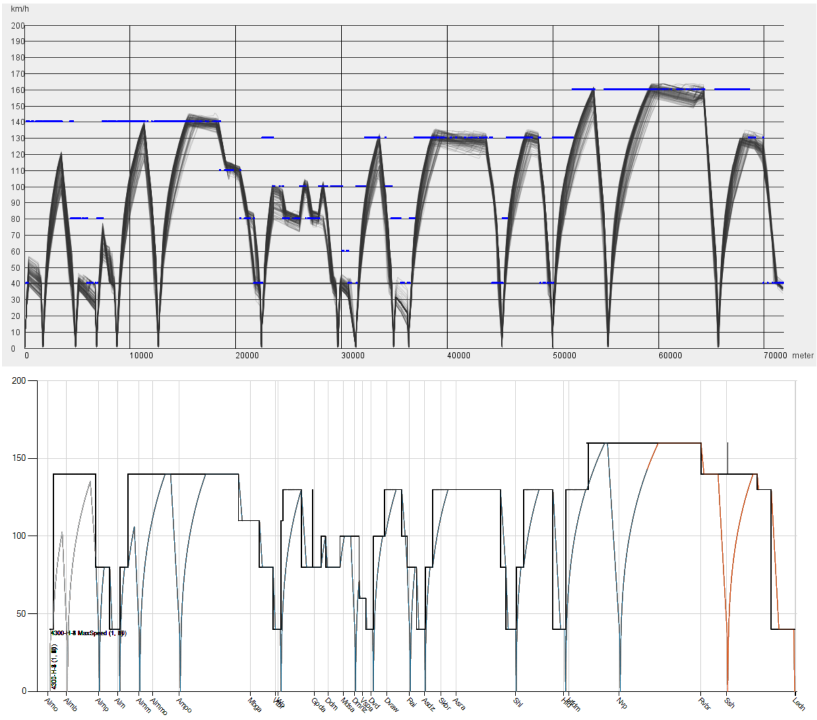


Figure .10: *Two speed-distance diagrams are shown here for the 4300 train series. In the top figure the speed-distance diagram of the agents can be seen, with default settings. In the bottom figure the speed-distance diagram can be seen for the default FRISO train driver.*

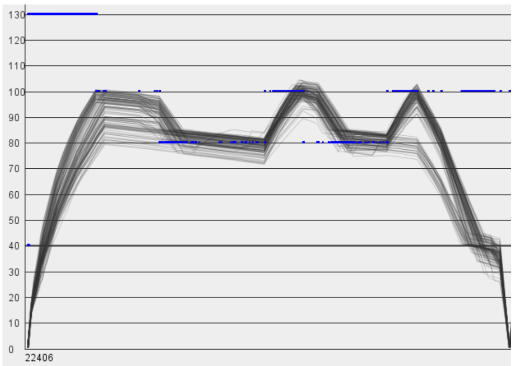


Figure .11: *A section of the top Figure .10, where the spikes in the speed limit can be seen, which resulted in an earlier arrival time than found within the realisation data*

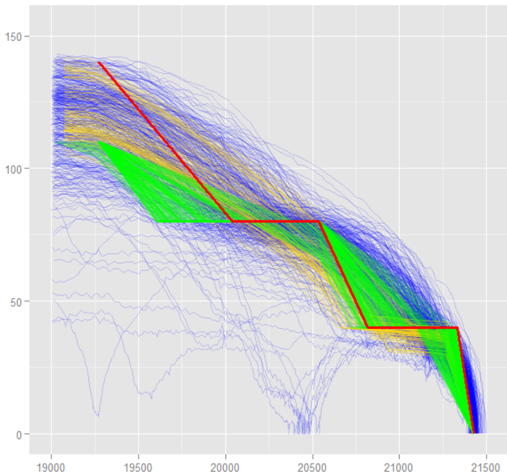


Figure .12: *The speed way diagram for the arrival at Breda (BD) from Tilburg (Tb). On the x-axes the distance in meters. On the y-axes the velocity in km/h. The blue lines represent the GPS data. The gold lines represent the agents driving behaviour. The red line represents the default FRISO driving behaviour and the green lines represents the FRISO DBA driving behaviour with a braking criteria of -0.19.*