CHALMERS

On-board Collection and Storage of Road Data and Implementation of Predictive Transmission Fuel-saving Functions

Master of Science Thesis

Henrik Andersson

Johan Axelsson

Department of Signals and Systems
Division of Mechatronics
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden, 2009 Report No. EX068/2009

Efficient On-board Collection and Storage of Road Data and Implementation of Predictive Transmission Fuel-saving Functions

HENRIK ANDERSSON
JOHAN AXELSSON

August 2009

Examiner: Jonas Sjöberg

Department of Signals and Systems
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Cover: [The truck model and the I-shift gearbox used for testing in the thesis.
Source: www.volvo.com]

Göteborg, Sweden August 2009

Abstract

Fuel-saving functions are of great concern to truck makers today. A desired function for achieving this is the ability to access preview data about the upcoming road. This feature can be useful for a lot of purposes in many areas. Beside fuel-savings, one can for example achieve a better driving comfort and a more intelligent cruise control. Preview data can be read from a digital map, but since digital road maps containing height information are commercially rare and do not cover all roads, this thesis has looked at another solution, namely to develop a self-learning system to provide the truck with road information without an initial database.

This thesis has the focus to develop a method to create a digital road map independent of whether road maps exist or not. This is done by using the truck's built-in inclination sensor and a GPS for positioning. Since the available memory is highly limited in existing ECU's, only the road where the developed functions have the biggest effect are saved and hence partial maps are created.

To test and evaluate the developed system, a previously developed function that uses preview data has been implemented. The function is active shortly before crests and in the following downhill slope and can save approximately 1.5 percent of fuel in addition to existing fuel-saving functions.

Since the memory is limited, it has been investigated how to represent a road with respect to minimising the data needed to be stored. The simulation results show that it is sufficient to use 8 points to characterise a hill, and then perform a clamped spline interpolation for restoring the data to create a functional preview. With this coding of height information, the assigned amount of memory allows information about up to 1500 hills or about 5000 km of road to be stored. The self-learning algorithm can be implemented in existing hardware without any additional equipment.

Acknowledgements

We want to thank Volvo Technology and Volvo Powertrain for the opportunity to realise our master thesis at the transmission functions group within an important and interesting area of scope.

A special thank to our supervisors at the department, Johan Bjernetun and Anders Eriksson, for all support regardless of their busy schedules.

We also want to send our thanks to Charlotte Holmén for all the help and feedback of the documentation.

Lastly, we want to thank FH111 for a comfortable environment and for letting us upload numerous different programs without frequent complaints.

Abbreviations

CAN	Controller Area Network
GPS	Global Positioning System
NVRAM	Non-Volatile Random Access Memory
TECU	Transmission Electronic Control Unit
S-N	Spartanburg – Newberry
B-L	Borås – Landvetter
L-B	Landvetter – Borås
G-L	Grenoble – Lyon
L-G	Lyon – Grenoble

Contents

1 Introduction

This chapter first gives the context of the thesis and explains some basic concepts. Then it describes previous work in the area of scope and gives the motivation and objectives for the thesis. Last, some definitions and the outline of the report are given.

1.1 Context

Within logistics, as within most industries, cutting costs is an important issue. The costs can be lowered in many ways; one of which is to reduce vehicle fuel consumption, a measure which is also desirable from an environmental point of view.

The thesis is aimed at development of software in the gearbox of a truck to enable more advanced fuel-saving functions. Lowering the fuel consumption is a big issue among truck manufacturers today and the methods for achieving the desired results can differ. To lower the consumption the manufacturers can make changes to the hardware of the truck, such as engine, chassis and gearbox. But since vehicles today are largely controlled by microcomputers, much work can be done by just making changes in the software of the computers, a measure that is both easier and cheaper than modifying the hardware.

1.2 Key Concepts

This section gives an introduction to the key concepts of this thesis, motivating why preview is a desired function and how on-board map data acquisition can be done.

1.2.1 Preview-Based Functions

A vehicle with an automatic gearbox has always had the downside of having a worse fuel economy than a vehicle with a manual gearbox driven by an experienced driver. The reason for this is that the driver is able to adapt his driving to the road ahead and thus avoid unnecessary or misplaced shifts. The automatic gearbox on the other hand has no knowledge of the road ahead and have to shift based on the current scenario. However, this situation is about to change with the concept of preview-based functions.

Preview-based functions for vehicles are functions that are fed with information about the road ahead and thus can increase the performance, for example lowering the fuel-consumption by optimising the shifting sequence in the case of an automatic gear box.

The area concerning look-ahead methods has been of great importance, from a safety and fuel-efficiency point of view. As preview information, most of the research concerning preview information uses digital 3D maps in combination with a positioning system [5]. The methods used need access to close-to-perfect preview data of the road ahead. This means that the vehicles used have been provided with additional hardware that handles the 3D maps and performs the calculations for optimal speed control [5]. Adding additional hardware can be problematic since it increases costs and can be hard to implement together with existing hardware.

A problem with using maps is that for large parts of the world, accurate 3D maps are not yet available. As an example, map data for the four best road classes will be available around 2013 in the European Union [17].

1.2.2 On-board Acquisition of Road Data

A way to get around the lack of map data is to let methods using look-ahead store self-recorded road data for use when travelling the same road again. There exist various ways of acquiring and storing road data, and these methods give varying accuracy of the road stored.

One method tested in a previous master thesis by Gaasendal [2] uses a GPS signal to generate road topography. The idea is based on driving on the same route several times and then averaging the results to get measurement data good enough to create an accurate road profile.

A study by Jansson et al. evaluates the GPS receiver signal in combination with a barometer and torque sensor [3]. These signals have different features; the GPS gives the altitude and position directly, the torque sensor can be used to calculate the altitude change by looking at applied torque versus change in speed and the change in air-pressure can be used to calculate the altitude directly.

In the study by Jansson et al. [3] a method is developed using sensor fusion with extended Kalman filtering. This is a robust method that is not dependent on receiving information from all sensors at all time and gives a good estimate of the road appearance if satellite coverage is not enough (which affects the GPS receiver) or if friction brakes are applied (which affects the torque sensor).

The issue concerning accurate reference data is addressed several times in the research related to on-board storing of road data. It is complicated to evaluate the accuracy of the collected data since accurate digital topography maps are not commercially available for all roads.

1.3 Developed Fuel-Saving Functions at Volvo Trucks

This section describes some fuel-saving functions developed at Volvo Trucks up to the start of this master thesis.

1.3.1 Free Rolling Function (I-Roll)

At Volvo Powertrain, fuel-saving functions have been developed that saves fuel by optimising gear selection and making use of the tucks potential energy.

The main fuel-saving function called I-Roll is currently implemented in the Transmission Electronic Control Unit (TECU), which controls the gearbox of Volvo trucks. The function can be described from the fact that it is beneficial to disengage the driveline in downhill slopes thanks to the mass of a heavy truck. Disengaging the driveline removes the braking friction of the engine and thereby reduces the fuel consumption. When maximum allowed speed is reached, the driveline is engaged again and the engine brake is used to prevent the truck from increasing further in speed. During this engine brake, no fuel is injected, which means that no fuel is used at all.

When the slope is approaching its end the speed drops below the maximum allowed speed, which allows the truck to disengage the driveline again.

The lowered friction from disengaging the driveline gives the truck an increased kinetic energy, which is used to postpone the need for throttle in the end of the slope. This delay can save fuel if used under the right circumstances. However, if the circumstances are wrong, for example if a downhill slope is very short or followed by a steep uphill slope, the use of I-Roll is not beneficial. With the use of a preview of the road, activations at the wrong time can be prevented.

I-Roll bases its calculations on the current road conditions, which means it assumes that the current downhill will continue for some time ahead. The function calculates the current acceleration of the truck to get the expected speed of the truck six seconds ahead. If the speed is within given boundaries from the set speed, I-Roll is activated, which means that the gear is put into neutral.

A development of the I-Roll function is called extended I-Roll. This function is basically the same as I-Roll, but with the added feature that the speed of the truck is allowed to fall below the set speed in the end of the slope before I-Roll is deactivated. This makes the I-Roll activation time a bit longer, which saves more fuel. By default, this lower speed limit is set to 2 km/h below set speed. Extended I-Roll can save around 1% fuel on average and will be referred to as I-Roll.

Two previous master theses at Volvo Powertrain have investigated the use of preview information to ensure correct activations of functions like I-Roll. The first thesis used digital road maps together with a GPS, and developed functions which could use preview information [11]. The second thesis [4] used recorded inclination data as preview and investigated the possibility to position the truck by the use of correlation between present and previously stored inclination data.

1.3.2 The CrestRoll Function

By the use of preview, it is not only possible to improve the I-Roll function, but also to develop new fuel-saving functions. One such function called CrestRoll was developed in a previously mentioned master thesis by Jakobsson, Johansson and Klintenberg [11]. The function was developed using digital road maps to create a preview called e-Horizon. CrestRoll is, just as the name reveals, the ability to roll over the crest and into the following slope. This is basically an early activation of I-Roll in such a way that the truck can regain its speed after the crest.

In addition to this early activation to roll over crests, CrestRoll also have the feature of requesting earlier engine brake, which is activation of engine brake before maximum allowed speed is reached. This is done as soon as it is predicted that the truck will stay above crest speed and reach the maximum allowed speed before the end of the slope.

The intelligent part in CrestRoll lies in the use of preview information and the calculations of how I-Roll and early activation of engine brake should be combined in the most fuel-efficient manner in the upcoming hill. The preview is used to calculate when the functions should be activated, for example I-Roll can be activated some time before a crest, if it is known that the speed lost in the remaining uphill slope will be regained in the following downhill slope. Engine brake on the other hand, is activated as soon as it is known that the maximum speed will be reached with I-Roll. This means for a moderate slope where the maximum allowed speed will not be reached with I-Roll, engine brake will not be activated. An example of a possible combination of the functions over a crest is seen in Figure 1.

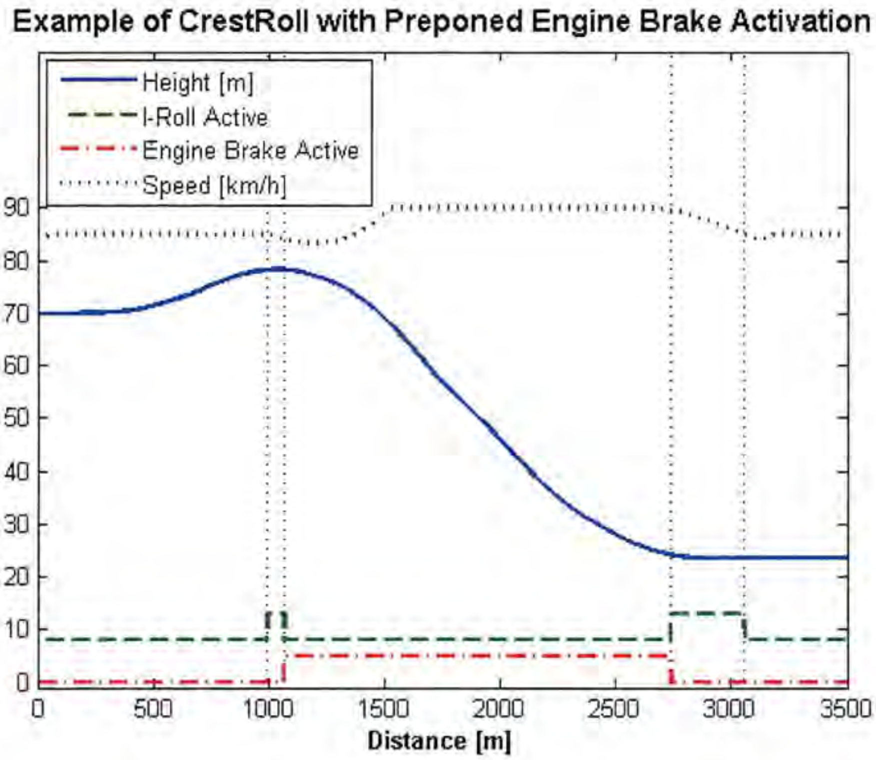

Figure 1: Example of CrestRoll over a hill. The combination of I-Roll and preponed activation of engine brake is shown at the bottom of the figure and are inactivated by default. It is easily seen that I-Roll is activated before the crest, meaning that unnessesary injection of fuel can be avoided.

1.4 Thesis Motivation

This thesis has been realised in cooperation with Volvo Technology and the transmission functions group at Volvo Powertrain and is a continuation of two previous master theses done at the same group.

The lack of available commercial road maps mentioned in Section 1.2.1 as well as needed hardware changes make it hard to implement CrestRoll in a real truck in its current form. This problem is addressed with this thesis which has the aim of developing an algorithm for recording and storing road data on-board a truck. This data can then be used by preview-based functions such as CrestRoll.

As described in Section 1.2.2, recording of road data can be done in several ways with varying accuracy. In previous work done by Larsson & Schantz [4], it was concluded that data from the inclination sensor in Volvo's I-Shift gearbox could serve as input for preview-based functions. The algorithm developed by Larsson & Schantz needed access to accurate road data and it was discovered that the memory available for storing data was a limiting factor. This fact shows the need for minimisation and compression of collected data.

1.5 Objectives

This section specifies the area of scope as well as the exclusions made for the thesis.

1.5.1 Specifying Goals

The main objective with this master thesis is to develop a method to store road data in an efficient way on-board a truck. Far from all road data can be stored, since the available memory is highly limited in the TECU. The second part of the project aims to implement the fuel-saving CrestRoll function with preview in a real truck. The preview information is obtained by on-line recording of road slope data from an accelerometer in the gearbox and by vehicle positioning by use of a GPS signal.

The goals are split up into sub-goals to give a more detailed overview. All of these sub-goals must be fulfilled to have a complete working concept.

- Find the minimum amount of road data required for robust control of the developed predictive functions. This implicates an optimization between possible fuel saving and the amount of data stored.
- Be able to position the truck when approaching a previously stored hill by use of GPS.
- Develop a method that evaluates which hills that are relevant regarding CrestRoll activation and should be stored with regard to memory issues.
- Implement the data acquisition method and the previously developed fuel-saving functions in a truck using the software ASCET.
- Create a reliable and robust solution for implementation which can serve as a basis for future production.

1.5.2 Exclusions

Since the predictive fuel-saving functions are already developed, no further work will be done within this area in this thesis. However, the functions still have to be modified to be able to use the reduced data from the data acquisition method instead of full preview data.

Moreover, the stored data will only be used when the truck travels in the same direction as when the data was stored. This is because the inclination signal from the transmission differs depending on the torque requested.

1.6 Contributions

The main contribution of this thesis work is an operational implementation of a self-learning partial road topography system in a real truck, without any hardware modifications. This involves investigation of how to characterise a hill with as little data as possible. The additional focus is on implementation of a preview-based improved fuel-saving function. Also, the fuel-saving functions have been modified for improved robustness.

1.7 Definitions

To be able to follow the terminology in this report, some basic terms are described below.

The most basic definition is the one of a *hill*. A hill is defined as the road altitude between two valleys. There are three important points of all hills; start, crest and end. This basic hill with three points can be described using two distance segments together with inclinations. If the start point is known, these segments would go through the defined points. Both points and segments are referred to throughout the report; segments are used for data storage and usage while points are used for re-creating road data. The minimum number of segments stretching between the most important points can be seen in Figure 2.

Another term used is *base length*. The base length is the resolution of the inclination used for calculations, i.e. inclination is measured with a frequency of 50 Hz, but a new value is only saved each base length meter. The reason for the division into base length m segments is to avoid small variations in the inclination and to make calculations easier. The base length determines the accuracy of the positioning and detection of crests, as the base length is the smallest unit used. The value of base length has been evaluated by simulations to 100 m. This means that the inclination is recorded during 100 m and then the mean inclination of the 100 m segment is saved and used for further calculations. Hence all points used for describing the characteristics of a hill are placed with a resolution of 100 m, for example at 100, 700, and 1300 m.

Throughout the thesis, *inclination* will be used to describe hills. Inclination refers to the angle of the road, measured in percent. As an example, an uphill slope that rises two meters in 100 meter of road has an inclination of 0.02 or 2%.

The distance driven corresponds to the hypotenuse of a right-angled triangle with the inclination as the angle between the hypotenuse and longitudinal cathethus. For this thesis, the sides mentioned are assumed to be equal. This can be done since an inclination of 5 percent would merely give a difference of 12.5 cm at a distance of 100 m, which is negligible compared to for example the error of GPS positioning.

Two different kinds of preview are used in the thesis, *full preview* and *partial preview*. Full preview is when all data about the upcoming road is available, for example through e-Horizon with a 3D map and GPS. Partial preview on the other hand is when preview data is only available for parts of the road, restored from stored data. This is the case when the algorithms developed in this thesis have been run, since they do not store and

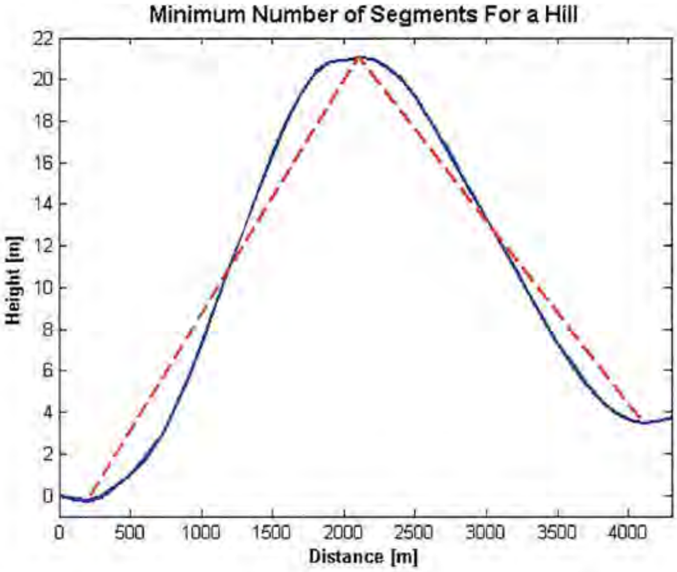

Figure 2: Segment definition; the minimum number of segments for describing a hill is two, since there are three points characterising a hill. The continuous line shows the hill and the dashed line shows the segments describing it.

keep all hills but instead choose the most important ones. The maximum length of the two previews is set to 2 km, but can of course vary for the partial preview which can end earlier than 2 km.

1.8 Outline

To be able to fulfil the objectives and reach the goals set up in Section 1.4, a lot of functions have been developed during this thesis work. The theory that is the basis for the developed functions is presented in Chapter 2. In subsequent chapters, the developed functions are presented, starting with functions for gathering and storing inclination data in Chapter 3 and continuing with functions for using the preview while driving in Chapter 4.

When all functions have been presented, results from both simulations and implementation are presented in Chapters 5-7. In Chapter 6, estimations are made which describe how much data can be stored in the available memory on-board a truck.

All results are finally resumed in Chapter 8, which also lays a basis for suggestions for future work in Chapter 9.

2 Theory

In this chapter, the theory used in the thesis is explained. Firstly, theory about interpolation spline is presented, followed by a description of how GPS works and which problems are related with this. Finally, a brief introduction to real-time systems is given. Readers familiar with these subjects can continue to Chapter 3.

2.1 Cubic Basic Splines (B-splines)

The road data has been stored as a number of points, describing the altitude change of a part of the road. When the data should be used as preview, the appearance of the saved hill has to be recreated in some way and this has been done by use of spline interpolation. The curve created is the height curve of the road, since this is a less complex curve compared with the inclination.

A continuous function is said to be a spline if it is piecewise polynomial and has a continuous first derivative. Also, for the points excluding the endpoints, the spline is of C^2 continuity [15]. The cubic spline is on the form:

$$S_i(x) = a_i + b_i x + c_i x^2 + d_i x^3 \tag{1}$$

In (1), the polynomial coefficients are a, b, c and d for each interval i. The coefficients are based on all input points, which mean that the whole spline is affected if one point is wrong. Also, the splines are made such that for each interval, the polynomial has common points with the input data. Hence, the total spline function goes through all points [14].

Furthermore, the cubic spline requires boundary conditions since there are more unknown than equations, which is explained by an example below.

Consider three cubic polynomials in the intervals

$$S_0(x) \in [x_0, x_1] \quad S_1(x) \in [x_1, x_2] \quad S_2(x) \in [x_2, x_3]$$

From (1), all polynomials have 4 unknown coefficients, which gives a total of 12 unknown. The points themselves in x_0, x_1, x_2, and x_3 generate 6 equations, since

$$\begin{aligned} S_0(x_0) &= f(x_0) \quad & S_1(x_1) &= f(x_1) \quad & S_2(x_2) &= f(x_2) \\ S_0(x_1) &= f(x_1) \quad & S_1(x_2) &= f(x_2) \quad & S_2(x_3) &= f(x_3) \end{aligned}$$

As mentioned earlier, the spline has a continuous first derivative which gives 4 additional equations

$$\partial_x S_0(x_1) = \partial_x S_1(x_1) \quad \partial_x^2 S_0(x_1) = \partial_x^2 S_1(x_1)$$
$$\partial_x S_1(x_2) = \partial_x S_2(x_2) \quad \partial_x^2 S_1(x_2) = \partial_x^2 S_2(x_2)$$

This leaves two degrees of freedom which then require two boundary conditions, which could be chosen in several different ways [14]. One way is to set the second derivative of the spline to zero at the endpoints, which would result in what is called a natural cubic spline. Another choice is to use clamped splines (also referred to as complete cubic spline interpolant), which means that the derivative of the endpoints of the spline is set to the derivative of the function in the equivalent points [15]. The boundary conditions of the latter type is described as

$$\partial_x S(x_0) = \partial_x f(x_0) \quad \partial_x S(x_3) = \partial_x f(x_3) \qquad (2)$$

Since the area of application is to approximate one hill at a time, there is a predefined closed interval where the spline must be valid. The transition between hills will be with the first derivative equal to zero (from hill definition), which is why clamped splines are most suitable. Hence, the free boundary conditions described in (2) are chosen as the derivative in start and end point, which is zero from hill definition. With this as input, the polynomial coefficients in (1) are calculated directly from the general cubic spline algorithm [14].

2.2 GPS

This section describes the functionality and properties of the GPS system. Addressed issues are errors, accuracy and update frequency.

2.2.1 Introduction

The Global Positioning System, GPS, is an American military satellite based positioning system. The system works by having 24 satellites orbiting the earth, placed in such a way that at least four satellites will always be seen from any point on earth, at any time [12].

Since this is a military system, the GPS signal originally contained a degradation of the signal, making the accuracy lower for civilians. However, this degradation was removed in year 2000 and today civilian systems have the same accuracy as military ones [12].

GPS receivers are nowadays becoming a common piece of equipment in vehicles, used for positioning of the vehicle. A GPS, in combination with a digital map, can serve as navigation systems in vehicles. When it comes to preview information, the GPS is often used in combination with 3D maps to provide a vehicle with road preview for the use of predictive vehicle applications.

2.2.2 Measurement Error of GPS Systems

When using GPS systems, there are several error sources to be aware of. In a report by Mansion [8], these are divided into system-wide and local error sources, where the signal degradation was the only system-wide error source, while there are three local error sources.

The first two local error sources are multipath and satellite geometry. Multipath means that the signal from the satellites bounces several times before getting to the receiver and thus the same signal reaches the receiver along different paths. The error from satellite geometry comes from the fact that it is harder to make a good triangulation when the satellites are close together than when they are far apart. Also, at least four satellites have to be "seen" by the receiver in order to make the measurements reliable. This is because the positioning consists of solving equations with four unknowns – longitude, latitude, height and clock offset [10].

The third local error source is the receiver used. Depending on receiver noise and time delay, the size of the receiver error can vary a lot.

2.2.3 Accuracy

In the paper by Mansion [8], it is claimed that a GPS receiver can reach an accuracy of 12-15 m in the horizontal direction, 95% of the time. These figures are in line with the ones measured by the Swedish Land survey (Lantmäteriet) [12], which states that the error is 5-15 m. The vertical accuracy is usually approximated to about 1.5 times the horizontal error, which gives an accuracy of about 20 m [12].

To get higher accuracy from GPS receivers, various techniques can be used. The idea is to have a base station with known position that measures the difference between the known position and the position given by the GPS. This difference is then sent to other, non-static GPS receivers that use the information to correct their GPS data and get a more accurate value. Using this technique, the accuracy can be as high as a couple of centimetres [8].

In many cases, a Kalman filter is used together with a model of the vehicle to get a better estimate of the position. In the study by Andersson & Fjellström [9], the GPS is coupled with a dead-reckoning system to get even better estimates, and to avoid problems with GPS signal masking etc. This technique is also used in the work by Kronander [10].

Another problem is that to get a correct measurement from the satellites, these must be in clear line of sight. This can be a problem for vehicles when driving for example in a tunnel. Different receivers handle loss of signal in different ways, and to be able to treat erroneous values correctly, the receivers' error handling must be known.

2.2.4 GPS Update Frequency

A problem when using GPS in vehicles is that the update frequency of the position often is quite low. For example, the receiver used in [9] has an update frequency of 0.5 Hz, which is far too low to be used for accurate measurements while driving. Most GPS receivers today use a frequency of 1 Hz.

Another problem with GPS update is that the GPS sends data when it detects a change and not at a certain frequency. In the study by Andersson & Fjellström [9], the time between subsequent updates varied as much as 0.7 s. There is also a time delay introduced in the receiver when processing the data, which means that the information shown on the display can be the last sample data, not the current ones [9].

2.3 Real-time Systems

A real-time system is "any information processing system which has to respond to externally generated input stimuli within a finite and specified period" [16]. This means that not only the result is important, but also the time in which it was calculated.

Real-time systems are very common in control theory, where they are often implemented as embedded systems [16]. This is the case for a lot of systems in cars and trucks and hence also for the TECU programmed in this thesis.

There are two kinds of real-time systems, hard and soft. For hard real-time systems, it is absolutely imperative that the responses occur within the required deadline. For a soft real-time system on the other hand, deadlines are important but the system will still work if a deadline is not met [16].

Normal computer systems are typically soft real-time systems. If a response is not received fast enough, it can be annoying for the user but it is not fatal. Embedded systems however are often used in safety critical applications, such as braking systems in cars, where an unmet deadline can cause much damage [16]. Those systems must be of hard real-time system type in order to be useful.

Real-time systems work by dividing the programs into several concurrent tasks that are scheduled in some way. Each task has a deadline it must meet and also a priority that shows how important the task is [16].

To make it easier for the system to keep deadlines, it is desired to keep the level of computation steady. This can be done by dividing the program into smaller parts that are run sequentially instead of one large part.

3 Storing Inclination Data

The first main part to be able to provide a preview is to collect road data for later use. In this chapter, the developed algorithm for recording and storing data in memory is described. First the recording procedure is defined, followed by hill characterisation and evaluation. Finally, positioning of stored data is presented.

3.1 Algorithm Overview

The CrestRoll function described in Chapter 1.3.2 requires that data about the current road is available. Since this kind of data does not exist in a truck today, it has to be gathered and stored while travelling a road. Furthermore, the information about the road requires a lot of storage space, which is very limited on-board a truck (about 32kB, see Chapter 6).

Hence, a function that would be implemented in a truck with the existing hardware configuration needs to save and use as little data as possible, but still be able to predict the appearance of the upcoming road with satisfying accuracy.

The algorithm for detecting and storing hills consists of three major parts. Firstly, it records the inclination and identifies hills in the recorded data. Secondly, it examines and evaluates the found hills and finally, if the evaluation is positive, the hill is saved in the memory. An overview of the algorithm for finding and storing hills in the memory can be seen in Figure 3.

3.2 Recording the Inclination

All Volvo trucks with an I-Shift gearbox are equipped with an accelerometer, which can be used to measure the inclination of the road. Data from the sensor can be accessed from the TECU and then used for calculations. The inclination measurements are done at 50Hz, which gives one measurement about every 40 cm when driving at 85 km/h.

When making calculations it is desired to have a constant length of the samples, since this removes the need of keeping track of sample lengths. Constant sample length is achieved by calculating the height change from the inclination measurements and then converting the calculated height change back into an inclination when base length m has been driven. This procedure also removes small variations in inclination.

The transformation from inclination to height and back to inclination is needed because of two reasons. Firstly, it is the height curve of the hill that is of interest, which makes it natural to work with height. Secondly, the inclination is much more oscillating, making an average of it over base length m too rough.

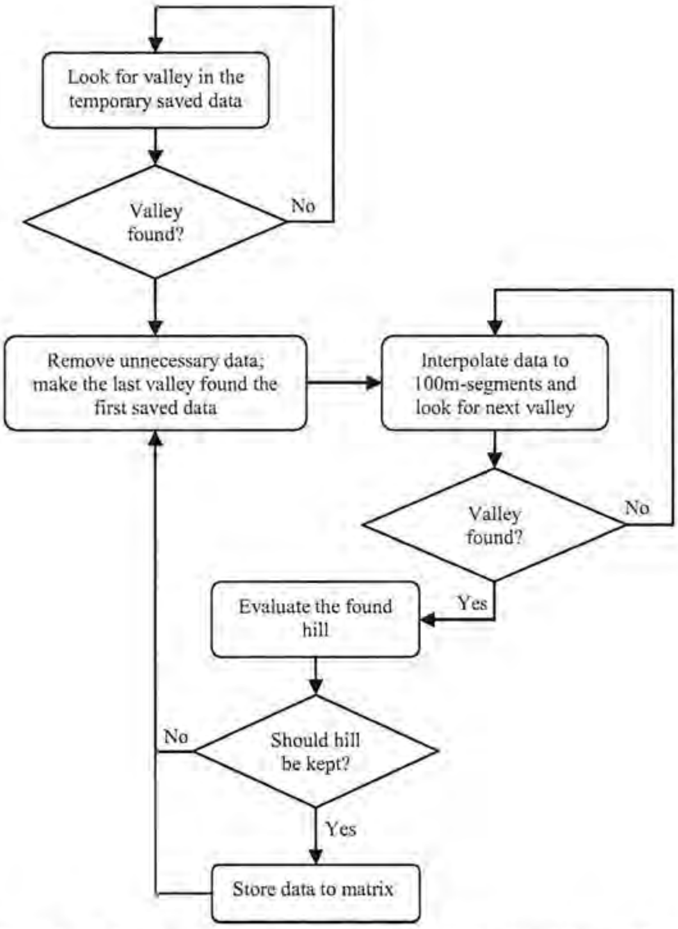

Figure 3: Flow chart for the identifying and storing hills algorithm. Hills are found in the continuously recorded inclination by looking at zero-crossings, which mark valleys and crests. All parts are described in this chapter.

3.3 Identifying Hills

A hill is defined as the road between one valley and the next valley as described in Chapter 1.7. The altitude curve of a typical hill, as well as the pertained inclination, is shown in Figure 4. Since the inclination is the derivative of the height, the start, crest and end points of a hill can be identified by looking for zero-crossings of the inclination.

When an inclination measurement has been recorded, an algorithm checks for minimum points in the data to find valleys that indicate the start of a hill. If a valley is detected, the algorithm starts recording the inclination and distance travelled and stores the GPS position of the hills starting point. The recording is done continuously until a hill-finding algorithm indicates that a full hill has been found.

When a full hill has been found, the recorded inclination and distance data for the found hill is sent through the proposed hill finding algorithm. Apart from identifying hills from the inclination data, the hill-finding algorithm also chooses which points on the hill that are important to save. It then evaluates whether the found hill should be saved or rejected, based on certain rules described in Section 3.5. If it should be saved, it is saved to memory, otherwise all the gathered data is discarded except from the last

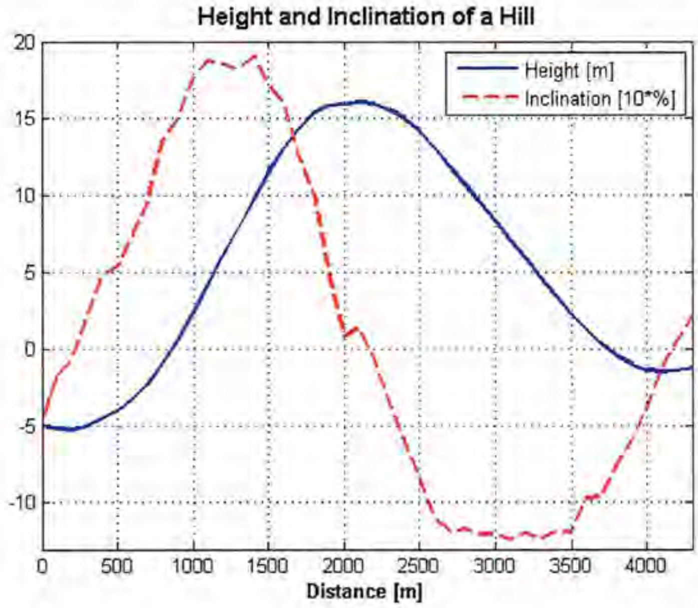

Figure 4: Example of a hill created from recorded inclination. Inclination is collected continuously and averaged each 100 m and is scaled in the figure by a factor 10 to make zero crossings more obvious.

valley, which is used as a start for the next hill. With this done, the algorithm starts looking for a new hill.

3.4 Characterising Hills

When characterising a hill and choosing which points on the hill that should be saved, it is proposed that the derivative of the inclination should be used, which is the second derivative of the height. The points should be placed where the second derivative has its maximum and minimum values, since that is where the appearance of the height curve changes the most. An example of this can be seen in Figure 5.

If the proposed algorithm does not find enough minimum and maximum points of the inclination derivative to be able to place all points, it places the remaining points on equal distances between the points that have already been found. This can be seen in the uphill slope in Figure 5 where the derivative of the inclination has no extreme value for the placed point.

Except from having the points at the right places, it is also desirable to have them spread out over the full length of the hill, to capture the overall appearance of the hill in a good way. To achieve this, a minimum distance between the points is decided upon, based on the length of the slope. This condition is the reason for the second left point in Figure 5 not being placed where the inclination derivative has a maximum point.

In total, eight points are used to identify each hill. One point in each valley, determining the start and end of a hill, one point on the crest point of the hill and five more points placed according to the rules described above. The number of points to use was decided upon after simulations, which are described further in Section 5.2.

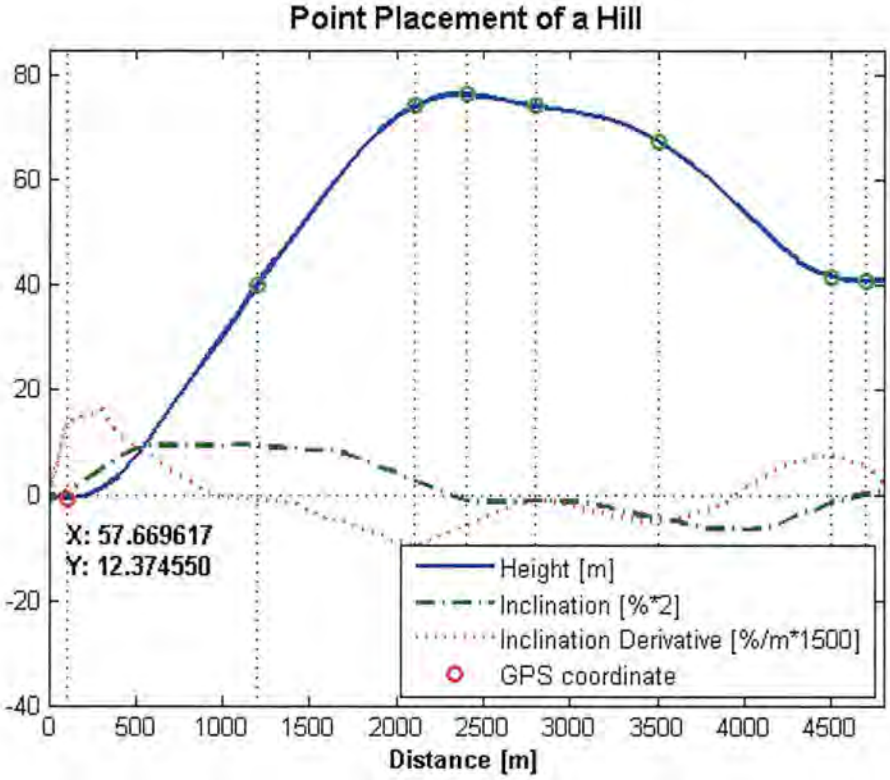

Figure 5: Point placement of a hill, mainly as a function of the inclination derivative. Guidelines are added in vertical direction at the points to show the dependence with the inclination derivative. A GPS position is bounded to the start of the hill to be able to match the preview on a later visit. The inclination is scaled by a factor 2, as well as its derivative is scaled with a factor 1500 for visibility reasons.

3.5 Evaluating Hills

This section contains the proposed method for evaluation of which parts of a road that should be stored.

It is important to notice that the criteria described in the following sections are valid only for the functions tested in this thesis, which depend heavily on the look of the downhill slopes. For other functions, the correct evaluation criteria may be something totally different.

3.5.1 The Need for Evaluation

In Section 4.1 together with Section 5, two different road approximation algorithms have been evaluated out of a fuel-saving point of view. However, since the storage space is highly limited, all hills cannot be saved. Instead, each hill must be evaluated to see if it will be useful for the fuel-saving functions and therefore should be stored. When no preview exists, the truck will use the ordinary I-Roll function. This fact makes it possible to remove smaller hills, focusing on saving the hills that are important for preview based fuel-saving functions.

3.5.2 Force Balance

A simple criterion for saving a hill is that it should be possible to use the I-Roll function in the downhill slope of the hill. This criterion can be evaluated by calculating the forces acting on the truck.

In a downhill slope, the gravitational force is acting in the forward direction of the truck, while the rolling friction and aerodynamic drag is acting in the backward direction. To be able to use I-Roll the forward forces must be larger than the backward ones, which is used as an evaluation criterion for the saving algorithm.

However, since the mass of a truck lies within an interval of approximately 6 to 60 tonnes [13], the actual mass of the truck is not always the optimal to use for evaluating whether a hill should be stored or not. The reason for this is that a heavier truck rolls in less inclination compared with a light truck. As lower limit, the mass used for evaluation is set to 40 tonnes not to exclude hills that could be used for preview when the truck has a heavier load. If the truck has a higher mass than this limit, the real mass value is used for calculations. An example of the reason for the lower limit implementation is a tractor driving without trailer, which still should be able to store hills based on a higher mass. This assumption with a lower mass limit only affect storing, which means that when the data is used as preview, the actual mass is used for calculations.

3.5.3 Hill Characteristics

To further exclude hills, different criteria for the characteristics of the hills can be used, for example length, height, maximum slope and other properties.

The fact that the I-Roll function is activated when no preview exists can be used to find suitable limits for the above mentioned properties. For example, if a downhill slope is very short, but still steep enough to activate I-Roll, I-Roll will also be activated when using the old function without preview. This means that preview information is not necessary for that hill and makes it possible to remove it. This will remove the possibility of rolling over the crest, but this loss has to be weighed against the loss in memory when saving a hill.

3.5.4 Further Criteria

In the future, more functions in the truck are expected to use preview data. This makes it important to be able to easily change the evaluation criteria, since new functions probably need other criteria than the ones used in this thesis. This has been enabled by building a separate evaluation file, which can easily be modified to add conditions for the new functions.

3.6 Identification of Available Preview Data

To be able to use the saved data as preview when driving the same road again, the GPS position of the start of the hill is saved with each dataset. This makes it possible to compare the current position of the truck with positions saved in memory. Also, since the roads can be a bit different when going in different directions, there is also a driving direction saved in each dataset. More about this can be found in Section 6.2.

The GPS positions of the truck are saved in an array while driving. Since GPS positions use very much storage space, only the last seven positions are saved, and then the array is shifted each time a new value is stored. When the start valley of a hill is found, it is marked with the corresponding GPS data among the seven saved values.

4 Using Stored Data as Preview

Once the data is stored in memory, it should be used as preview the next time the truck drives on the same road. In this chapter, different methods tested for hill restoration from stored data is presented, as well as creation of the preview. Lastly, the time estimation function is described, which is used for decisions concerning activation/deactivation of CrestRoll.

4.1 Techniques for Restoring Hills from Inclination Data

This section evaluates two different techniques for restoring the characteristics of a hill; linear approximation and spline interpolation.

4.1.1 Linear Approximation

The linear approximation is basically the stored data itself, keeping the stored inclination during the length of the segment. When the segments are not very long, this technique gives quite a good approximation, but as the length of the segments increases, the result gets worse. A comparison between linear approximation and spline interpolation can be seen in Figure 6. In the figure, only five points are used, which gives significant overshoots and undershoots in the linear case. The benefit of linear approximation, though, is that it requires no calculations to approximate the road.

4.1.2 Spline Interpolation

The spline interpolations are done by creating a height curve from stored data and then translating the spline curve to inclination as described in Section 2.1. Simulations indicate that the inclination itself is not possible to restore by the use of splines, since it has a more complicated curvature.

The boundary conditions for the spline are set to the derivative of the height at the endpoints, which from hill definition is zero. The same condition is used for the crest to not allow any overshoot. To have the smooth top at the correct place is of high importance, since it affects the calculations of when to activate CrestRoll. The result of a spline interpolation can be seen in Figure 6, which is visually a much better approximation than in the linear case.

Another desirable feature of the cubic splines is that the curve passes through all given points, which simplifies calculations when storing data. Even though clamped cubic

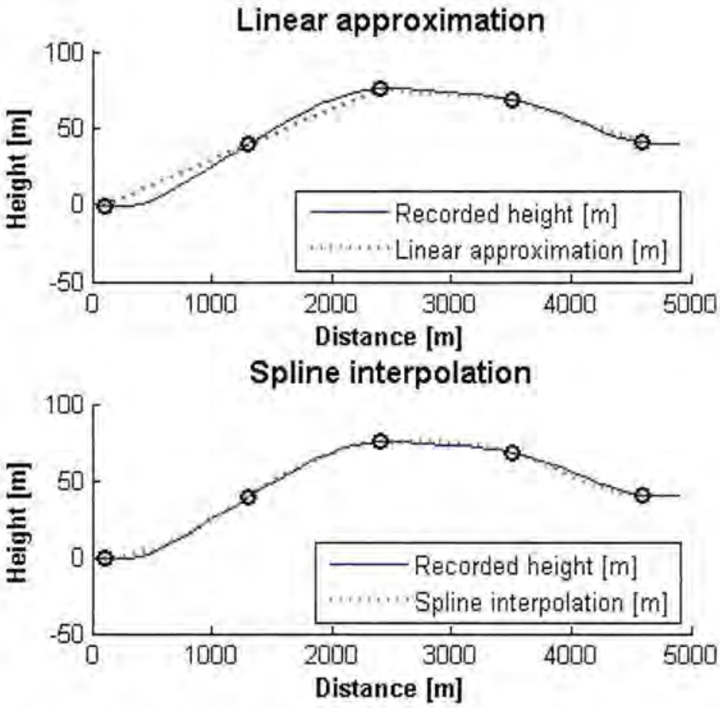

Figure 6: Comparison between spline interpolation and linear approximation of a hill. To clearly illustrate the difference between the methods, only five points are used. The linear case clearly shows a worse approximation compared with the spline case. The spline also uses the information of zero inclination in valleys and at crests, which prevents overshooting.

splines require more calculations than the linear case, it still has an advantage compared with for example Bézier curves; the cubic spline passes through the collected data points, which requires less calculations [15].

4.2 Looking for Preview Information

During driving, a GPS gives the position and a direction of the truck about every second, which is compared with the data previously stored in memory. If a match is found, the matching dataset is loaded into memory and a spline interpolation is performed. Data for the recreated hill is then available to the preview creator described in Section 4.3.

As long as the preview information continues, the truck will use it and when the preview ends, the truck will go back to using conventional I-Roll.

4.3 Creating the Preview

The resulting height curve from the spline interpolation is evaluated in several points, which then gives a higher resolution compared with the saved data. This high resolution data is converted into a preview matrix as described in Table 1.

Table 1: Description of the preview; *distToSegment* is the distance to the start of the segment, *lengthOfSegment* together with *segmentInclination* is the segment information. The number of rows of the matrix is decided from the preview horizon of 2 km.

$$\begin{bmatrix} distToSegment\#1 & lengthOfSegment\#1 & segmentInclination\#1 \\ distToSegment\#2 & lengthOfSegment\#2 & segmentInclination\#2 \\ \vdots & \vdots & \vdots \end{bmatrix}$$

This matrix is sent into a function that was developed in a previous thesis work [11] that calculates and predicts the speed of the truck based on current values of truck information (such as speed, acceleration, inclination etc.) and the preview of the upcoming road; see Section 4.4 for more information. The result is used to determine whether I-Roll or engine brake should be activated.

The preview horizon is set to 2 km, which means that the truck has a “look ahead” horizon of 2 km that can be used for controlling its behaviour.

A problem occurs when the preview ends and no more information is available, which is solved by adding a last segment with zero inclination and the same length as the preview horizon.

4.4 The Time Estimation Function

The function Time Estimation was developed in an earlier master thesis [11] and is only described briefly.

The time estimation algorithm uses the information from the preview matrix and information about the truck properties to calculate the time until a certain speed will be reached, depending on which function is activated. The times calculated are time in I-Roll until maximum allowed speed is reached, time in I-Roll until minimum allowed speed is reached, time in I-Roll until set speed is reached, time in engine brake until maximum allowed speed is reached and time in engine brake until minimum allowed speed is reached.

The different speeds that should be allowed is set by the driver, but as default the maximum allowed speed is 5 km/h above set speed, the minimum allowed speed is 2 km/h below set speed. There is also a minimum allowed speed over a crest, which is used to allow the truck to start I-Rolling before the crest of a hill, if it is known that the speed lost during the remaining uphill slope is regained in the following downhill slope. This speed is set to 5 km/h below set speed by default.

When simulating the functions, it was noticed that the time estimation algorithm had to be altered a bit to compensate for the fact that the recreated road data differ from the real road appearance. The differences, however small, can cause unwanted activations and deactivations of functions at bad times. This can be partly prevented by having margins when calculating.

5 Simulation Results and Evaluation

In this chapter, the results from the simulations are presented and the conclusions about parameters are explained. The main goals with the simulations are to see that all functions work as expected and to find out the best values for different parameters.

First, the parameter values for data storage are examined, followed by the fuel-savings made on the test routes. Lastly, results of how the evaluation criteria described in Section 3.5 affects the fuel-savings are presented.

5.1 Prerequisites for Simulations

Since the goal is to develop a concept that should be able to run in existing hardware, some restrictions have to be considered for the product, such as robustness and drivability of the truck. Moreover, the memory restrictions affect the quality of the preview, which in turn requires a more robust modelling of conditions for fuel-saving functions. As a result, the models developed earlier are modified in order to fulfil these goals.

All simulations are performed in Simulink with a Volvo simulation tool called Global Simulation platform, GSP (formerly known as Vsim+), which contains road and truck data for different routes and truck models. Log files containing real truck signals from different routes have been used to test the functions in situations as close to reality as possible.

Parameters used for simulation are a set speed of 85 km/h, maximum allowed speed of 90 km/h, minimum allowed speed of 83 km/h. However, when rolling over a crest, the lower speed limit is set to 80 km/h.

The roads that have mainly been used for simulations are Landvetter – Borås 40 km (Sweden), Lyon – Grenoble 80 km (France) and Spartanburg – Newberry 40 km (USA).

5.2 Number of Points Needed to Characterise a Hill

One important parameter examined is the number of points used to store a hill. This value is important both for providing valid input to functions and for storing reasons, since the number of points are directly related to the amount of memory required per hill. Since fuel-savings are approximately constant for more than 5 points, the choice of number of points to use is done with regard to curve fitting ability.

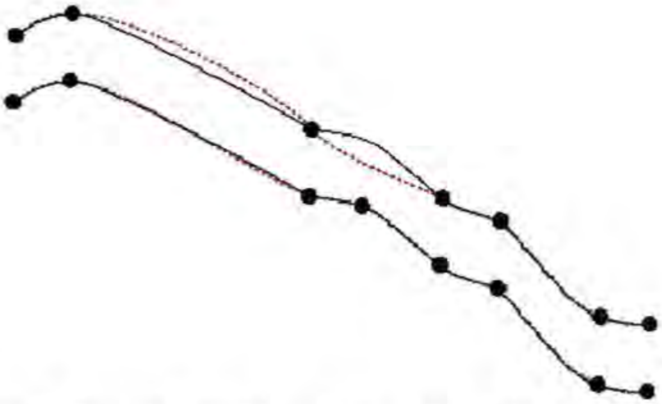

Figure 7: The figure illustrates possible problems when using too few points for spline interpolation. This example shows the spline for 7 compared with 8 points for a hill on the road Borås-Landvetter.

The visual evaluation of approximation ability shows that 8 points are enough to capture the characteristics of almost all hills. This can handle a hill containing two plateaus, see Figure 7. The hill shown in the figure has one of the most complex characteristics found during simulations of almost 800 km of road.

Furthermore, the spline fit ability has been investigated as a function of number of points and evaluated per base length of 100 m (see Section 5.3). The test is made to investigate how an increase in points affects the spline fit. As can be seen in Figure 8, the spline curve fit has a significant change for few points and approaches 100 % when the number of points goes towards infinity, a result which agrees well with intuition.

5.3 Base Length for Data Collection

In addition to the number of points, the base length has been evaluated by simulations.

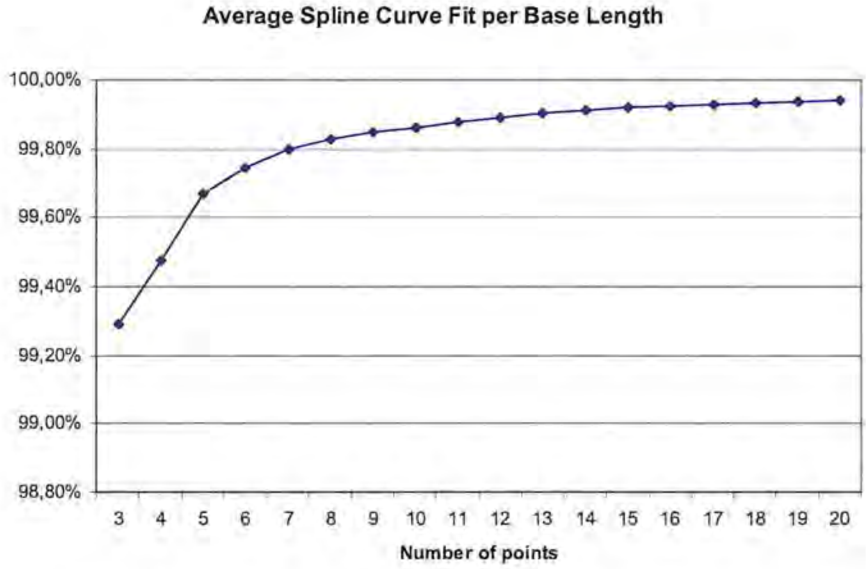

Figure 8: Average curve fit evaluated per base length. The figure shows a larger relative change when using only a few points. As expected, the curve fitting increases with the number of points.

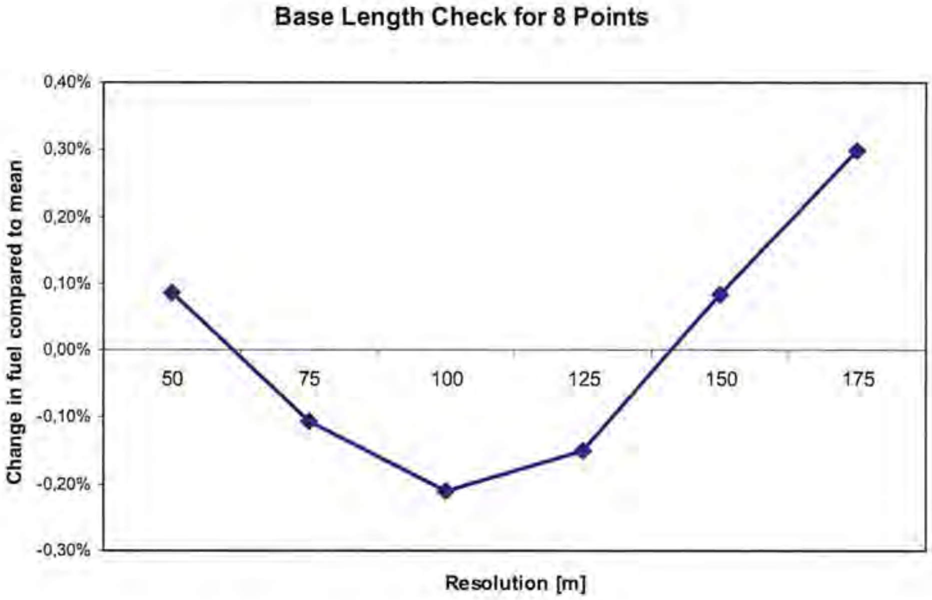

Figure 9: Base length check. Results are relative change in fuel consumption with respect to mean fuel-savings for all tested base lengths. As can be seen, the largest fuel-savings are obtained with a base length of 100 m. Too short base length captures high frequent changes in the inclination, while a too long base length gives inaccurate positioning of characteristic points.

The term "Base length" is described in Section 1.7 and can briefly be described as the resolution (or sampling length) that is used for calculations.

The evaluation is done with respect to actual fuel savings compared to the mean saving for each road. The base length has been altered for the different roads and the fuel results are compared with the corresponding mean savings for the same base length. In other words, if fuel savings were independent of base length, the change would be 0 % and if the fuel economy is better than the mean consumption, the change would be negative. The results have been weighed together into the result displayed in Figure 9. As can be seen, a base length of 100 m gives a reduced fuel consumption of 0.2 percent compared with the mean savings.

The base length affects the filtering of the measured inclination signal, since it indirectly decides how many samples that should be used to take the mean. Too short base length gives a noisy calculated mean inclination, while a too long one gives inaccurate positioning of characteristic points for the hill and thereby misplaced activation/deactivation of functions when driving on a hill. The evaluation of base length can be seen in Figure 9.

5.4 Curve Approximation

For very long hills with complex height curvature, the approximation differs a lot from the appearance of the real road. These hills are often of high interest and needs to be captured in a satisfying way. One way to handle this is to temporarily allow the functions to double the number of points used for hill identification.

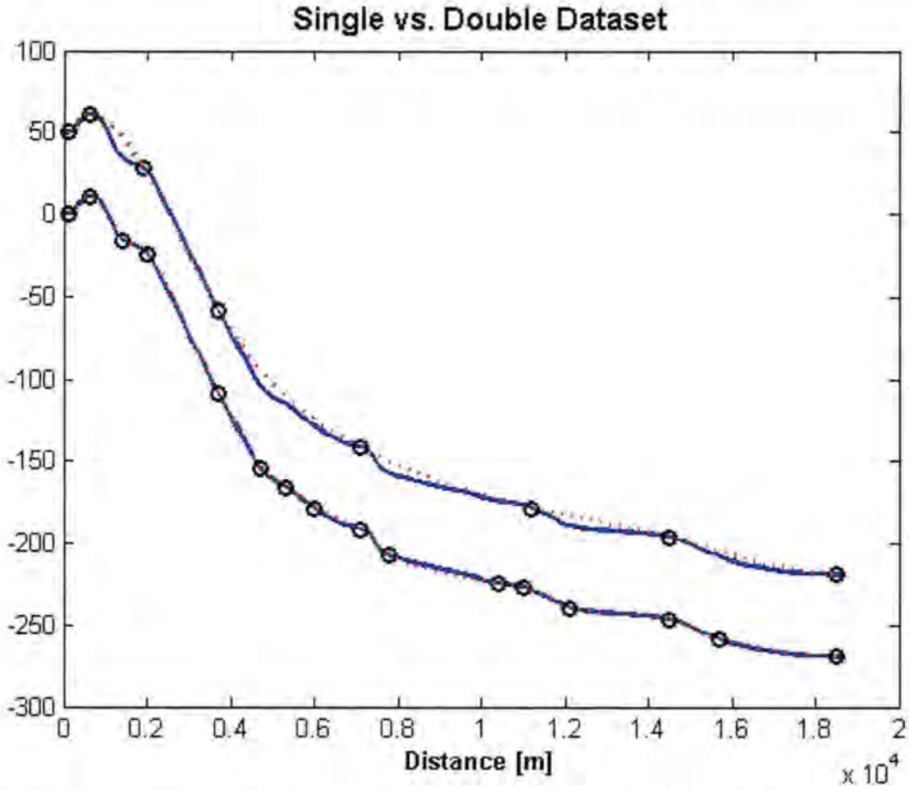

Figure 10: Comparison between single and double dataset use for a hill of almost 20 kilometres. With 8 points, the varying characteristics of a long hill can be hard to capture. In that case, a double dataset can be used, enabling hill characterisation with 16 points.

The doubling comes from the fact that the number of columns in the storing matrix is fixed. Hence, a stored data set cannot be extended with only a few points, which means that the only way to use more than 8 points is to double the number of points to 16. By the use of double data, the hill would take up two positions in memory. However, the type of hill that requires a double dataset is very limited. During simulations on more than 800 km of road, only one slope of this kind was found.

Even when double data sets are used, complex hill characteristics can be difficult to restore with splines. To limit the freedom of the spline interpolation, a constant derivative for a short distance can be assumed. This assumption gives the possibility to add guide points at a distance delta on both sides of calculated points. These guide points make the spline fit the calculated points with the correct derivative. However, guide points show upon a worse approximation when only 8 points are used and a better approximation when 16 points are used. Based on these results, it was decided that no implementation of guide points would be done in truck, since it would only improve extremely exceptional cases.

The hill together with single and double dataset approximation can be seen in Figure 10. The single dataset (upper curve) has an overshoot after the top which gives inaccurate time calculations for function activations. The curvature later on is also poorly approximated, but since the inclination is low no function activations will be made anyway. The double dataset captures the hill characteristics with a satisfying accuracy.

5.5 Fuel Savings

Since the purpose with development of partial preview is to implement fuel-saving functions, this has been the main issue to investigate in simulations. The outcome from a fuel-saving point of view is good results in general when using more than 5 points, since the current configuration of CrestRoll is quite robust to approximation errors. Simulations have shown that it is possible to achieve better fuel consumption when the partial preview is a spline interpolation instead of a linear approximation. A comparison between the average savings for spline and linear approximations can be seen in Figure 11, where the spline interpolation is on average 0.2 percent unit better than the linear approximation. As can be seen in the figure, the fuel saving is not affected to any great extent for the tested number of points, indicating that the result is not that sensitive to how the hill is modelled. The reference which simulations are compared with is a simulation with 5 km/h overspeed allowed.

An example of robustness is a minimum allowed time for certain functions to be activated, which allows the approximation to differ a bit from the real road. The minimum time active results in that if a function is activated too early, it must remain active to compensate for a preview which is not perfect. However, in bad approximations, this measure does not prevent the truck from having an undesirable speed profile (unwanted speed variations), since the function could be activated at the wrong time. On the other hand, fuel savings can still be achieved, even with a bad choice of activation time. Hence, more points do not necessarily mean better fuel saving, but better approximation of the road. If the number of points approaches infinity, fuel savings would approach full preview results.

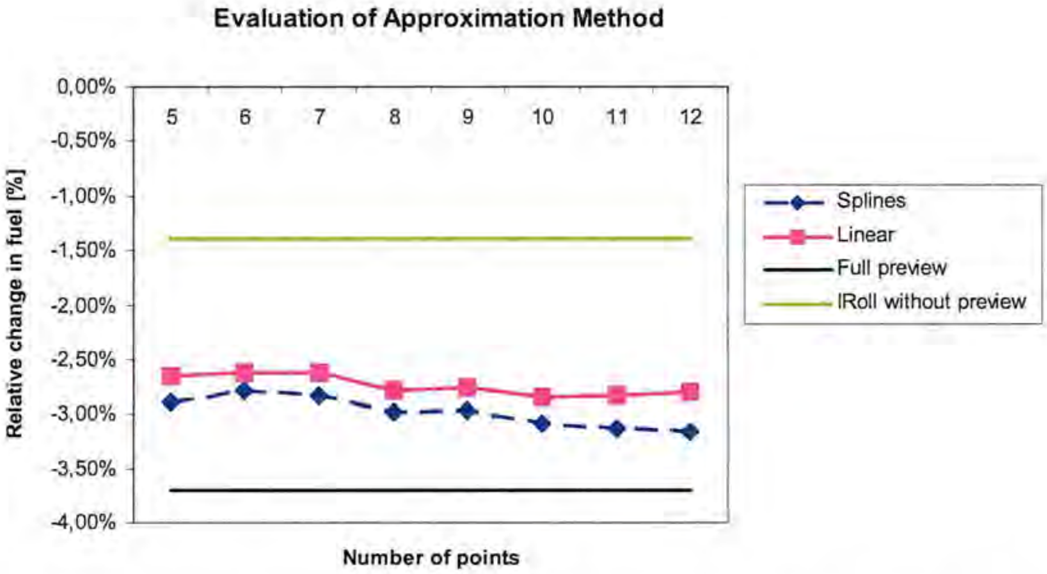

Figure 11: Average fuel savings compared to no function active as a function of number of points. The function evaluated is CrestRoll together with preponed engine brake. The roads used for this evaluation are Borås – Landvetter, Lyon – Grenoble and Spartanburg – Newberry with corresponding return trips.

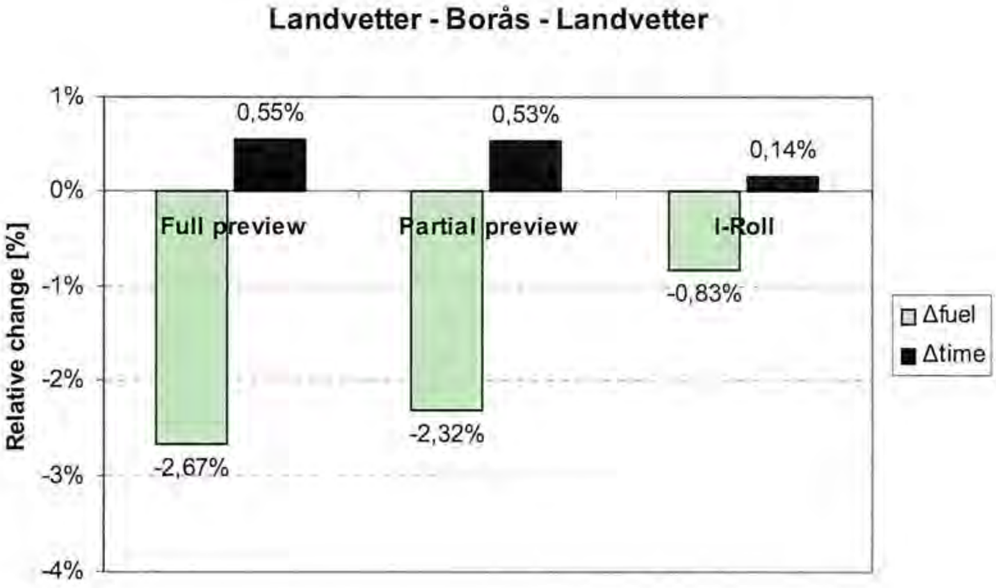

Figure 12: Average relative change of fuel and time for CrestRoll with preponed engine brake on the tested road Landvetter - Borås. CrestRoll is used for full and partial preview and as a reference, I-Roll without preview is also shown in the figure. A full summary of each road can be found in Appendix 1.

The number of points chosen for the implementation of partial preview is 8, based on both fuel-saving and curve fitting results. A previous result presented in Section 5.1 also shows that 8 points can capture the characteristics of almost all hills. A comparison in time and fuel between the partial and full preview together with I-Roll for the road Landvetter – Borås can be seen in Figure 12. Full preview has a more accurate data compared with the partial preview, which means that it activates CrestRoll earlier, giving larger fuel-savings.

Simulations of relative change in fuel and time consumption with CrestRoll, together with I-Roll as reference, have also been made for the roads Lyon – Grenoble as well as Spartanburg – Newberry. The simulation results are shown in Figure 13 as an average for the simulated roads. The variation in time between partial and full preview in Figure 13 comes from the fact that partial preview use a higher crest speed to be robust and avoid unwanted deactivations. This means a later activation of CrestRoll for the partial preview with the result that less fuel is saved. Moreover, one of the additional roads used contains many hills, why the difference in time is more obvious since there are a lot of function activations. This result can be compared to Figure 12, where the road used has less activations of CrestRoll and thereby no essential difference in travel time.

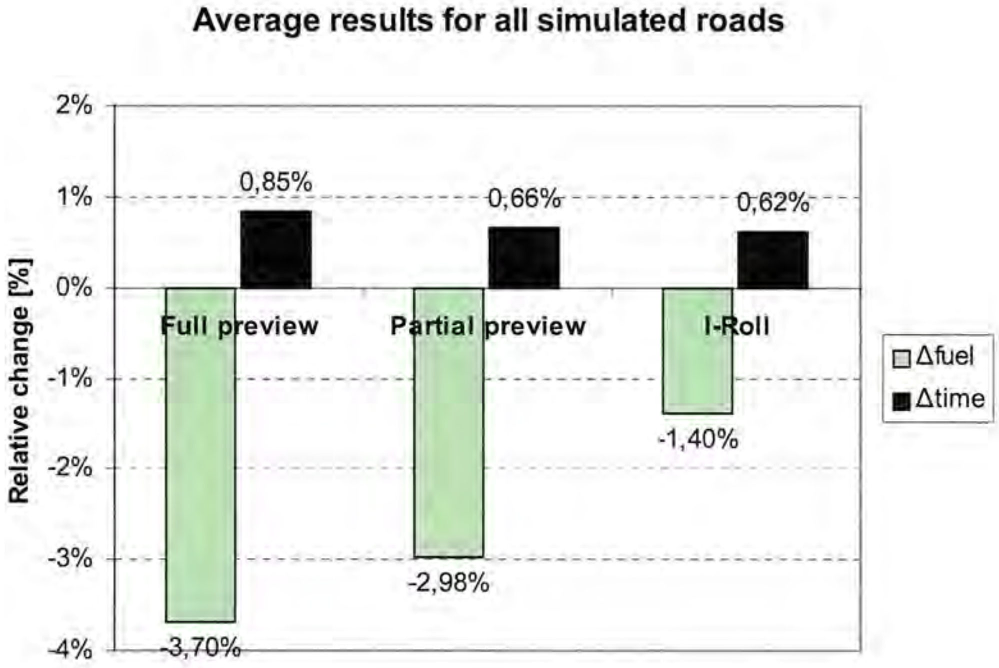

Figure 13: Average relative change in fuel and time consumption for the roads Landvetter – Borås, Lyon – Grenoble and Spartanburg – Newberry. Partial preview saves approximately 1.5 percentage points additional fuel compared with the regular I-Roll function while still maintaining the average travel time.

5.6 Evaluation of Hills to Keep

Various methods for evaluating which hills should be saved are described in Section 3.5. The evaluation parameters have been simulated to test how large part of the hills that really needs to be saved in order to still be able to achieve good fuel-saving results.

Simulation results with all evaluation functions active show that about 50 % of all found hills need to be stored to memory to still achieve good fuel results. The effect of hill evaluation can be seen in Figure 14. In general for the Figure, more fuel is saved at the cost of longer travel time when hills are removed from the preview. The reason is that when preview existed, CrestRoll was not beneficial to be activated for some small hills. However, when no preview exists, I-Roll is often activated, which still saves fuel but at the cost of longer travel time. For the road Grenoble – Lyon (and return), one of the hills removed has a small fuel-saving potential, but I-Roll will not be activated due to a small inclination when no preview exists. This gives a marginal lower fuel-saving when hills are removed, but not by any great extent. Overall, the evaluation function has no significant influence on the relative change in fuel and time.

An example of which hills to save can be seen for Grenoble – Lyon in Figure 15. Hills that are mainly uphill or moderate downhill are not saved. The save ratio is 64 %, even though it may look like more. The reason for this is that all hills require the same amount of storing, no matter if they are 100 or 10 000 meters long.

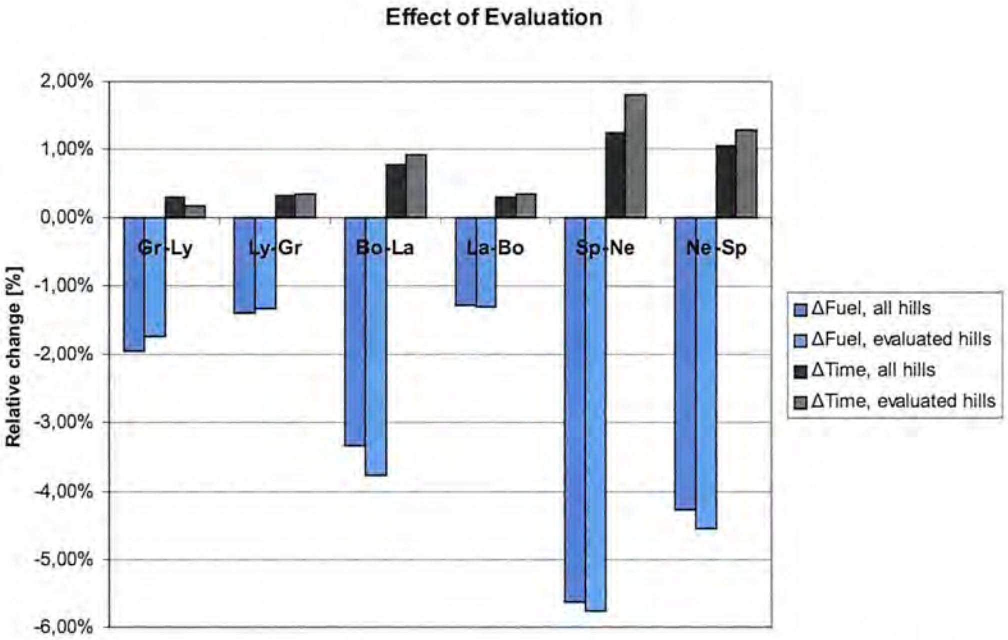

Figure 14: Effect of evaluation; hills with low fuel-saving potential are removed. About 50 % of all hills are saved, which does not affect the fuel and time results to any large extent.

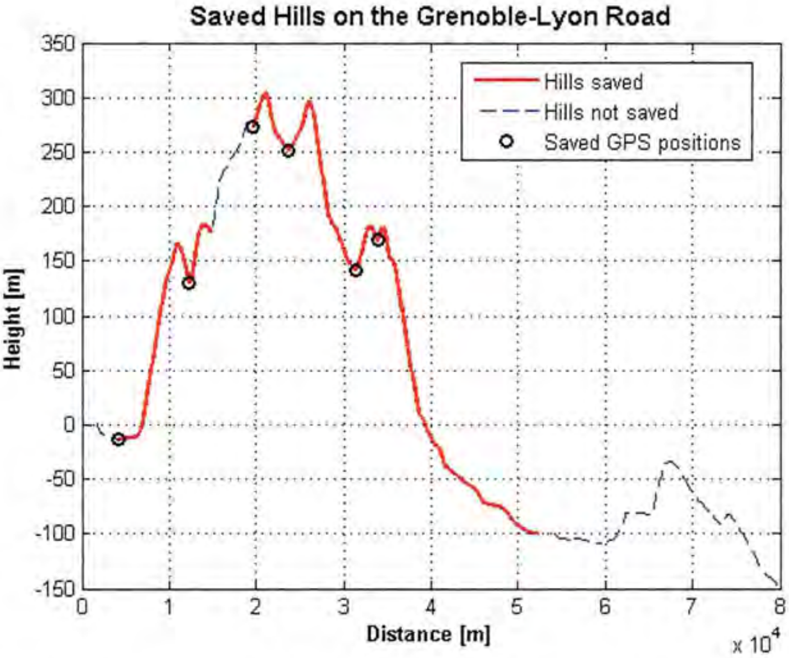

Figure 15: Saved hills on the Grenoble-Lyon road. The saved hills ratio is 64 %. Hills with short or moderate downhill slopes are discarded.

6 Memory Estimation

This chapter aims at determining how data should be stored with respect to memory limitations, as well as calculating how many hills that can be stored in the assigned memory of the TECU. This includes positioning information as well as inclination and length of segments.

6.1 Storing Data in Truck

The current design makes it impossible to read and write to the NVRAM memory while running, which requires the stored data to be in the RAM during drive and then written to the NVRAM memory when the truck is shut down. This means that 32 kB for road data will always be allocated in the RAM, since ASCET does not support dynamic arrays. In addition to this, the program requires data collection and processing, which also uses RAM. The total amount of memory assigned for preview functions in RAM is 45 kB, which, thus, makes it very important to keep memory usage at a low level.

As the storage space is very limited, all data cannot be saved. This is the main reason for choosing as few segments as possible for identifying a hill. From a storing point of view, all hills must be described by the same amount of data to enable the use of matrix layout.

Instead of saving values for the 8 points used for hill characterising, information about the seven segments between the points is saved. This lowers the amount of data and fits well together with the averaging done while recording. When saving the gathered data, a format is used that is later easy to search through for matching values. The format is a huge matrix and looks like

Table 2: Matrix layout describing all data stored after identification of a hill.

$$\begin{bmatrix} GPSpos_1 & direction_1 & \#visits_1 & continue_1 & segmLength_{1,1} & inc_{1,1} & segmLength_{1,2} & inc_{1,2} \cdots \\ GPSpos_2 & direction_2 & \#visits_2 & continue_2 & segmLength_{2,1} & inc_{2,1} & segmLength_{2,2} & inc_{2,2} \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \ \ddots \end{bmatrix}$$

where *GPSpos* is the GPS position of the starting point of the hill, *direction* is the direction of the truck when the hill was stored, *continue* is a variable to indicate if a hill is split in two, *#visits* is a counter of number of times the truck travelled the same stored route, *segmLength* is the length of a segment and *inc* is the inclination of a segment. When using seven segments, this matrix will have 16 columns and as many rows as there are saved hills.

This format of the stored data makes it easy to search through for matching positions. When doing this, only the first column has to be checked for matches, and when a match is found, the row contains all information needed. In reality, the matrix is not implemented as a matrix, but rather as several arrays, one for each column. This is due to restrictions in the software used for programming the TECU. However, this implementation also has the advantage of being able to choose different data types for the different columns and hence use different number of bits in each column to minimize storage usage.

6.2 Assigning Bits to Stored Data

The GPS position contains two 32 bit values (one for longitude and one for latitude) and decides the order of where data should be stored. Sorted data simplifies the search to find stored data. For storing which direction the truck travelled in when recording the current hill, 3 bits will be used as shown in Table 3.

Table 3: Bit description of direction parameter for stored data set.

Number	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Direction	N-NE	NE-E	E-SE	SE-S	S-SW	SW-W	W-NW	NW-N

Together with GPS position, the direction will be used to recognize that the same route is travelled as the data stored and prohibit recognition of road in wrong direction. Otherwise, another GPS position would have to be stored to determine a vector in the direction of the truck movement.

Another issue to take into account is how to choose which hills that should be removed if the memory is full. One way to show if a stored road is likely to be used as preview again is to have a counter of how many times the road has been visited. The counter is set to 3 bits, which then can reach to 8 drives on one stored route.

The distance stored for each inclination is a multiple of the evaluated base length (100 m) from Section 5.3. By the use of multiples, the stored data can be reduced significantly. If 7 bits are used, a stored segment can at the maximum be 12.7 km. This maximum distance is in direct relation to the number of segments used for a hill.

The inclination value is defined in percent from the inclination sensor. By multiplying the inclination by 10 and using only one decimal number, it can be stored using 8 bits, which results in a range of $-12.8 \leq \alpha \leq 12.7$ % for the road inclination α. This is a sufficient range with respect to simulation results, but roads with steeper slopes exist. However, this is no problem since the functions which use inclination are saturated for the given range, i.e. uses engine brake when inclination is less than approximately -3 percent. Positive inclinations would not cause a problem, since CrestRoll is not active in steep uphill slopes.

In addition to this data, a flag of one bit is needed as an indicator whether the hill is divided into two continuous curves or not. The reason for this is given earlier in Section 5.4. This bit use could be avoided by not assigning any GPS position to the second dataset of the hill. However, this would complicate the sorting of preview data as well as one bit would be unused, since the smallest data type is 1 byte (8 bits) in ASCET. This means that everything that is to be stored has to be in 8 bit multiples. Since the bits of direction, counter and the continuous flag adds up to 7, these will be merged using bitwise writing leaving one unused bit. The number of bits used for each segment length is 7, which gives one unused bit for each segment. To use this loss in an efficient way, the 7 bits of the seventh segment length are distributed among the unused bits. To visualize this operation, the result has been summarized in Table 4.

Table 4: Bit distribution of the seventh segment length among available bits.

Variable	$s7_7$	direction	Count	flag	$s7_6$	s1	$s7_5$	s2	$s7_4$	s3	$s7_3$	s4	$s7_2$	s5	$s7_1$	s6
Bits	1	3	3	1	1	7	1	7	1	7	1	7	1	7	1	7

The final data storing "matrix" with bit assignment is described in Table 5 below. To simplify the table, only the first bit for the distributed seventh segment is shown.

Table 5: Bit assignment for storage of all required data.

Variable	$GPS_{X,1}$	$GPS_{Y,1}$	$s7_1$	$direction_1$	$count_1$	$flag_1$	segm. $Length_{1,1}$	$Inc_{1,1}$	...	segm. $Length_{6,1}$	$Inc_{7,1}$
# bits	32	32	1	3	4	1	$7 \cdot 8 + 6 \cdot 8 = 104$				
# bytes	4	4	1				13				

This choice of bits results in a total of 22 bytes per stored hill. In relation to available memory, this means at the maximum 1489 hills can be stored. However, the length of road that can be stored is hard to determine since two hills are not alike.

For the estimation of how long total route data that can be saved, an average of 4 hills per 10 km has been used. This average is based upon the recorded routes available for the project. The total road distance that could be saved (if all hills are considered and GPS would still be stored) is then 3722 kilometres. With the use of the developed evaluation function for hills, more than 5000 km road coverage is possible. This can be compared to if a full preview would be stored (with one inclination per base length of 100 meters); 2409 km road would fit in the assigned memory for storing to NVRAM (if all hills are considered). This approximately corresponds to 963 hills. Moreover, full preview storing would require dynamic arrays for storage, since the length of hills vary.

6.3 Effect of Reducing Bits

The number of bits used for storing data has a large influence on the amount of hills that can be stored. The available memory gives the number of maximum stored hills as described in (3) below.

$$numOfHills = \frac{assigned\ NVRAM}{numOfBytesPerHill} = \frac{32 \cdot 1024}{9 + 2x - floor\left(\frac{x}{7}\right)} \qquad (3)$$

Where x is the number of segments used. In (3), all values are calculated as bytes. If 8 segments would be used, the number of stored hills would be 1365, which then would mean a loss of more than 8 % in stored hills compared with when using 7 segments. Moreover, the choice of 8 segments would give one unused bit per stored hill, which would be inefficient storing.

The same problem with inefficient storing occurs if the number of segments would be less than 7, since the last segment cannot be distributed among a total of 6 bits.

7 Implementation

This chapter considers the implementation of the developed functions in a truck, taking hardware limitations into account.

7.1 Setup in Truck

The setup in the test truck consisted of a computer running monitoring software for CAN and the TECU software. The Controller Area Network (CAN) is a network used for communication between different devices in a truck, for example the engine and the gearbox. The computer was connected to the CAN-bus used by the TECU and to a CAN-bus coming from an external GPS-receiver.

The external GPS receiver was necessary since not all trucks have a built-in GPS receiver and in those that have, the signal is not available to the TECU. For testing purposes, this was solved by using an external GPS connected to the computer and routing the signal to the TECU via the CAN-bus. In the future, all trucks will probably have a built-in GPS receiver and the signal can then be routed to the TECU.

The monitoring software provided the possibility to observe the existing variables in the TECU while driving the truck. This possibility made the error handling much easier and made it possible to study the behaviour of the algorithms more closely while driving the truck.

7.2 Computation Time

The TECU in a truck is a real-time system, which means that it is important for each task to finish within a certain time, see Section 2.3. Also, a lot of functionality is already implemented in the TECU, which makes the memory and computation time available for new functions limited.

Because of the limitations described above, two main issues for implementation have been the use of memory and the computation time. To lower the load on the TECU, the functions were rewritten when moving from MATLAB to C and care was then taken to keep the computations on a stable level. This was done both by modifying the algorithms and by dividing them into several parts, with one part running each clock cycle. This procedure let one part of the code use the same amount of time as did the whole function before.

The code division was made with respect to which parts of the code that were the most time consuming. The division was possible due to the fact that when driving a truck, the maximum speed is usually 25 m/s (90 km/h). Since the functions interpolate the inclination to segments of 100 m, it takes at least four seconds to form a segment. This gives enough time to finish the computations while enabling function distribution among a total of twelve steps.

To lower the impact on other functions, the new algorithms were also placed in a slower process and were given lower priority than the most essential tasks, for example gear related calculations.

7.3 Positioning

As mentioned earlier, a GPS signal is used to find a preview matching the trucks current position. As described in Section 2.2.3, the accuracy is about 15m in longitudinal and latitudinal direction. This means that an exact match of the position is not possible, especially not when moving. Instead, the trucks position is allowed to differ up to 50 m from the stored position and still count as a match.

The same thing goes for the direction. The direction has eight possible values, but it is calculated from the GPS values, and to have error margins, the direction is allowed to vary ±45° from the stored direction and still count as a match.

7.4 Problems with Implementation

The algorithm development has been done in MATLAB and the functions have then been tested through simulations with Simulink. To be able to run in a truck, all functions have been converted into C-code and compiled to be run in the TECU. The TECU is programmed using a software called ASCET that is based on C-code and also has a graphical coding interface that is later converted to C-code.

The change of programming language created some problems, mostly due to the fact MATLAB has a lot of features for arrays and matrices that do not exist in C. A lot of problems with implementation were also due to differences in the computers CPU used for simulations and the TECU. Those differences have mainly been about how floating point numbers are handled.

After overcoming the initial problems with data types and differences in processors, the algorithms worked as desired and all functions but two have been tested and validated in a real truck. The two functions not tested are the horizon creator and the CrestRoll function.

7.5 Overall Results from Implementation

In this section, the implementation results for all parts of the proposed self-learning algorithm are presented.

7.5.1 Storing Data

The data storing algorithm works as described in Chapter 3. All hills are characterised correctly and the memory layout works as intended.

7.5.2 Evaluation

In Figure 16 the results from running the evaluation algorithm on recorded data is seen. As the figure shows, it is the longest and steepest hills that are saved, which is natural when optimising for CrestRoll.

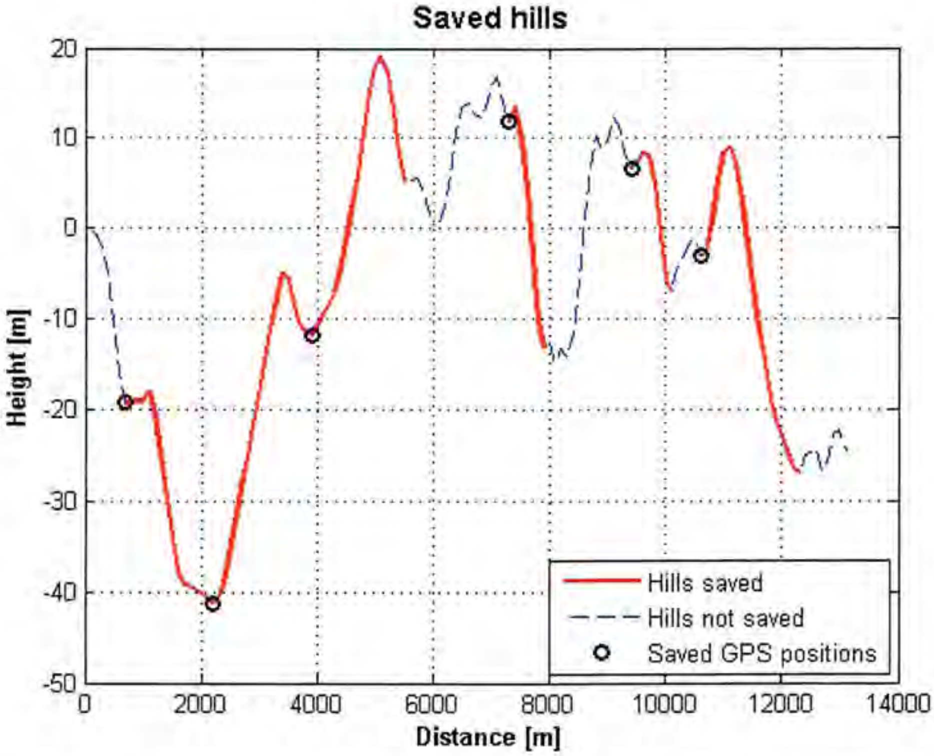

Figure 16: Test drive from Volvo Lundby to Marklandsgatan via Älvsborgsbron. For this test, 40 % of the hills were saved.

7.5.3 Matching Preview

During test runs, the preview function managed to match the trucks current position with the position of a previously saved hill. For the tested route, all hills where found, which indicates that the positioning method with GPS and direction works as wanted.

7.5.4 Creating Splines

When driving the truck and a match was found, the data was loaded from memory and converted to a spline without any problem. Even though spline interpolation can be calculation heavy, no big effect was seen in the processor load of the TECU.

7.5.5 Creating Horizon

Because of problems when implementing other functions, there was not enough time to validate the horizon creating algorithm in the truck. However, the code for the function does exist in the TECU and is supposed to work as intended. Simulations have shown that no problems should be expected with this part.

7.5.6 CrestRoll

The CrestRoll function is implemented in ASCET. Due to lack of time it has not been validated in the truck but has been seen to work well in simulations.

When running CrestRoll in a real truck, the early activation of engine brake cannot be used. This is because the functionality for controlling the engine from the TECU is not yet available.

8 Conclusions

The aim of this thesis has been to develop a self-learning system for use together with fuel-saving functions in trucks. The system shall be able to record inclination data, store it in memory and then be able to recreate the appearance of the road on a later visit. The system shall also be possible to implement in existing hardware.

Simulation results indicate that the developed algorithms manages the tasks well and that large fuel-savings can be made using preview data instead of currently existing functions.

Apart from the proposed storing and restoring algorithms, an evaluation algorithm has been developed. The algorithm evaluates which of the recorded hills that are the most beneficial to save from a CrestRoll point of view. By using this evaluation method, twice the amount of road can be covered within the same amount of memory.

All developed functions have been converted to C-code and implemented in the TECU. Most functions have also been validated by testing in a real truck.

The results from on-road tests show that it is possible to implement the developed functions in the TECU with only minor changes in existing software and no additional hardware. The self-learning functions also give an output similar to one from e-Horizon and those systems are therefore compatible with each other.

Simulations indicate fuel savings of additional 1.5 percentage points for the CrestRoll function when partial preview is used compared to the current truck configuration.

9 Future Work

This chapter presents openings that have been found during the work, but which have not been considered important enough to fit into the already tight schedule. Since the time span of the thesis work is limited to 20 weeks, a lot of things have been low prioritized in favour of the more important tasks.

9.1 Measures for Full Memory

As more data is continuously stored in memory, the memory will finally be filled. When this has occurred and new hills need to be saved, it has to be decided which of the saved hills that should be removed. In this thesis, a simple solution has been made in overwriting from position one in the memory, which could be improved in several ways. Firstly, one thing that can be noted is that many long haul trucks often drive the same route. This will lead to that many hills are travelled several times, whereas some are only visited once or twice. When this is the case, hills with one visit can easily be removed, since they are probably not a part of the ordinary route. Secondly, the GPS position of the stored data can be examined when taking current position of the truck into consideration. The data positioned furthest away from the truck is likely not used very often.

9.2 Dynamic Road Save

Since the thesis has focused on implementing only the CrestRoll function, the decision to save data from valley to valley is obvious. However, there are other functions which are active in other parts of the road, for example between the tops of two hills. These functions would benefit from using another start and endpoint of the preview.

9.3 Update of Previously Stored Data

During various travels on the same road, the inclination signal will not be exactly the same from time to time, even if it is compensated for mass changes etc. Examples of this phenomenon have been seen in the study by Larsson & Schantz [4], where it was shown that the inclination is not identical the same between travels. This study [4] also showed upon a high repeatability, but there are errors present that affect the signal. Examples of these are insufficient precision of mass estimation, GPS precision (see Section 2.2.3) and environmental issues such as wind and snow.

The algorithm developed to store data in this thesis is based upon the first time the road is travelled. To minimise erroneous stored inclinations, the data could be updated the next time the truck travels the same road. This can be implemented by the weighing the stored data with the number of times the hill has been travelled. The importance of this feature needs to be investigated further, since no obvious problem occurred during simulations.

10 References

[1]. Hellström, Erik, Ivarsson, Maria, Åslund, Jan & Nielsen, Lars (2007). Look-ahead Control for Heavy Trucks to Minimize Trip Time and Fuel Consumption. *Proceedings from the Fifth IFAC Symposium on Advances in Automotive Control.* Seascape Resort, USA.

[2]. Gaasendal Christensen, Anders (2006). *Optimization of Gearshift Strategies using Road Information.* Master's thesis, Technical University of Denmark.

[3]. Jansson, Henrik, Kozica, Ermin, Sahlholm, Per & Johansson, Karl Henrik (2006). Improved road grade estimation using sensor fusion. *Proceedings of the HYCON & CEmACS Workshop on Automotive Systems and Control, WP4c.* Lund University.

[4]. Larsson, Marcus & Schantz, Anders (2008). *Route Recognition from Recorded Inclination Data Enabling Use of Preview-Based Fuel-Saving Functions.* Master's thesis, Chalmers University of Technology.

[5]. Hellström, Erik, Fröberg, Anders & Nielsen, Lars (2006). A Real-Time Fuel-Optimal Cruise Controller for Heavy Trucks Using Road Topography Information. number 2006-01-0008 In: *SAE World Congress.* Detroit, MI, USA.

[6]. Hung, Pen -Shan & Su, Tsui -Chuan (1998). *Map-Matching Algorithm of GPS Vehicle Navigation System.* Technical report. Reng Chia University, Taichung, Taiwan.

[7]. Feijóo, C., Pérez, F. & Asensio, A. (1992). Real Time Estimation of GPS Measurement Error. In: *Vehicle Navigation and Information Systems Conference (3rd:1992).* Oslo, Norway. P. 369-375.

[8]. Manson, Paul (1997). Real-Time Applications of GPS. *Proceedings from GIS/GPS conference. Doha, Quatar.*

[9]. Andersson, David & Fjellström, Johan (2004). *Vehicle Positioning with Map Matching Using Integration of a Dead Reckoning System and GPS.* Master's thesis, Linköping University. LiTH-ISY-EX-3457-2004.

[10]. Kronander, Jon (2004). *Robust Automotive Positioning: Integration of GPS and Relative Motion Sensors.* Master's thesis, Linköping University. LiTH-ISY-EX-3578-2004.

[11]. Jakobsson, Samuel, Johansson, Niklas & Klintenberg, Oscar (2008). *Preview-Information-based Transmission Functions for Improved Fuel Economy in Heavy Trucks.* Master's thesis, Chalmers University of Technology.

[12]. Lantmäteriet (Updated 2008-12-17). *GPS och annan mätningsteknik*. (Electronic) Available: <http://www.lantmateriet.se>/Kartor/Geodesi och Gps/GPS och annan mätningsteknik. (2009-05-04).

[13]. Volvo Trucks Sweden (Updated 2009-03-07). *Datasheet Truck Tractor FM42TB1LC*. (Electronic) Available: <http://www.volvo.com/trucks/sweden-market/sv-se/trucks/Volvo-FM/productPage_FM.htm?TAB=4> (2009-05-04).

[14]. Restrepo, Juan (Updated 2003-04-12). *Cubic Splines*. (Electronic) Available: <http://www.physics.arizona.edu/~restrepo/475A/Notes/sourcea/node35.html> (2009-05-05).

[15]. Cohen, Elaine, Riesenfeld, Richard F. & Elber, Gershon (2001). *Geometric Modeling with Splines: An Introduction*. Natic, Massachusetts: A K Peters.

[16]. Årzen, Karl-Erik, El-Khoury, Jad & Törngren, Martin (2006). *Real-Time Control Systems*. Course Book, Lund Institute of Technology and The Royal Institute of Technology.

[17]. The free library (Updated 1998-10-12). *NavTech Expands Map Database Coverage of Europe and North America; Company to Add Significant New Countries*. (Electronic) Availale: <http://www.thefreelibrary.com/NavTech+Expands+Map+Database+Coverage+of+Europe+and+North+America%3B...-a053078393> (2009-08-18).

Appendix 1 – Simulation results for the main test roads

Simulation results for the main roads used in the thesis. All partial preview results are based on 8 points per hill. For comparison issues, all hills have been saved in the partial preview.

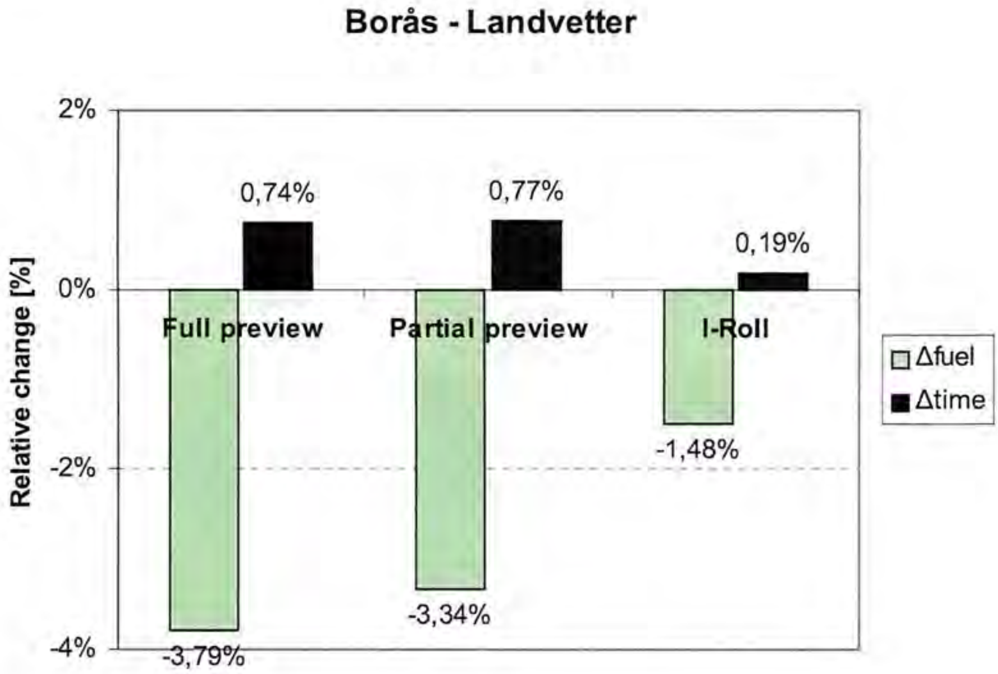

Figure 17: Simulation results for the road Borås – Landvetter (Sweden).

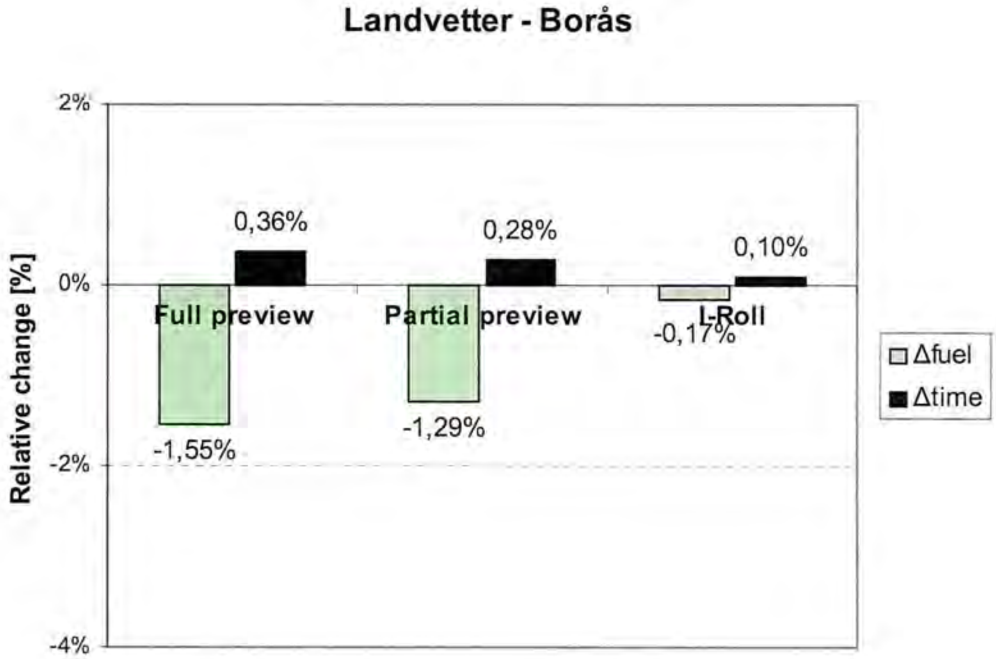

Figure 18: Simulation results for the road Landvetter – Borås (Sweden).

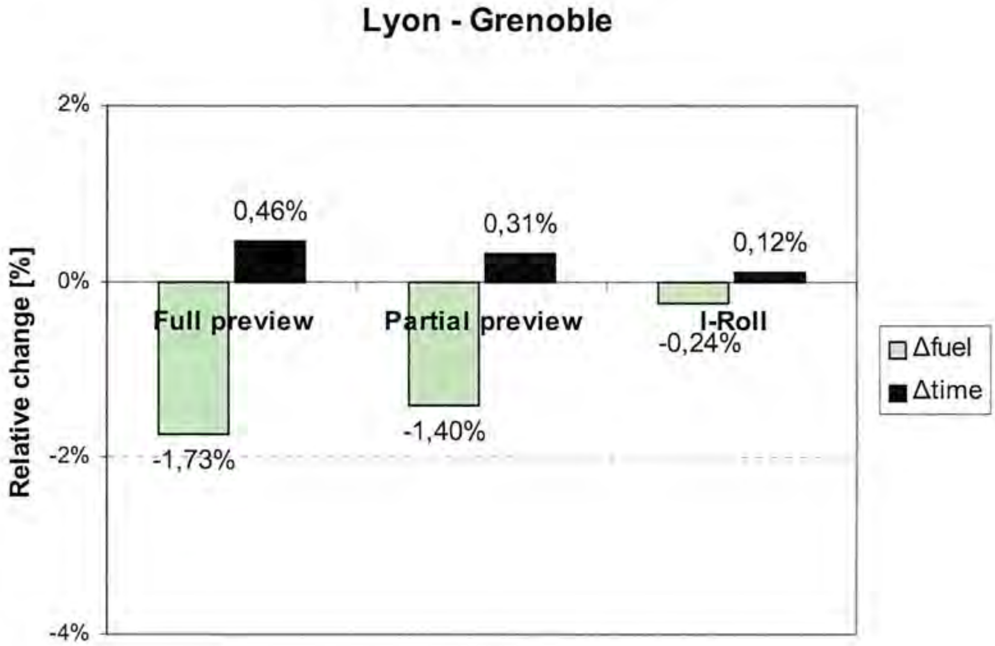

Figure 19: Simulation results for the road Lyon – Grenoble (France).

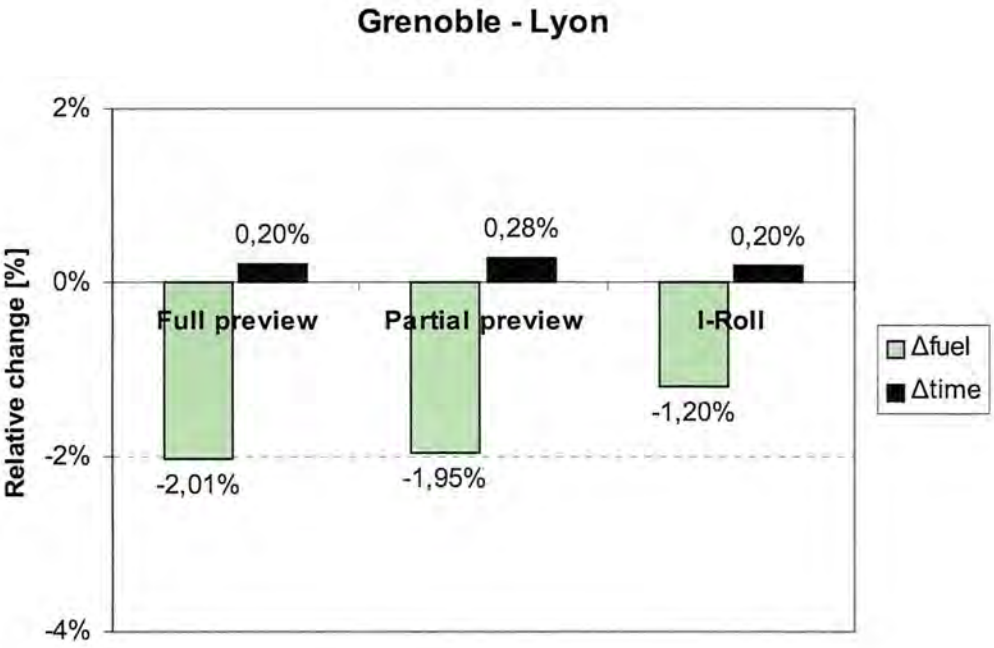

Figure 20: Simulation results for the road Grenoble – Lyon (France).

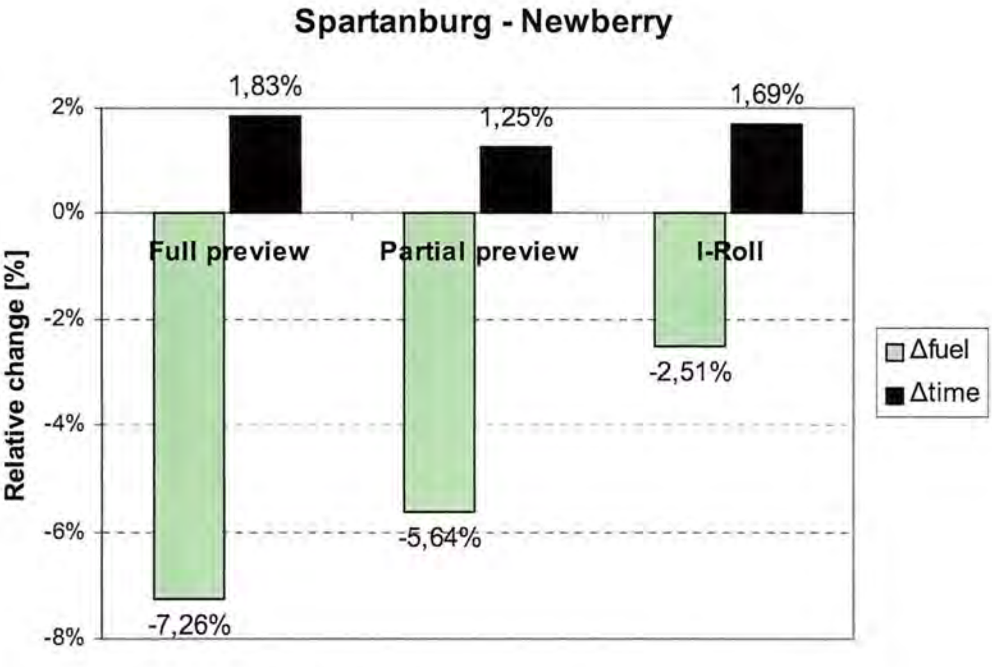

Figure 21: Simulation results for the road Spartanburg – Newberry (USA).

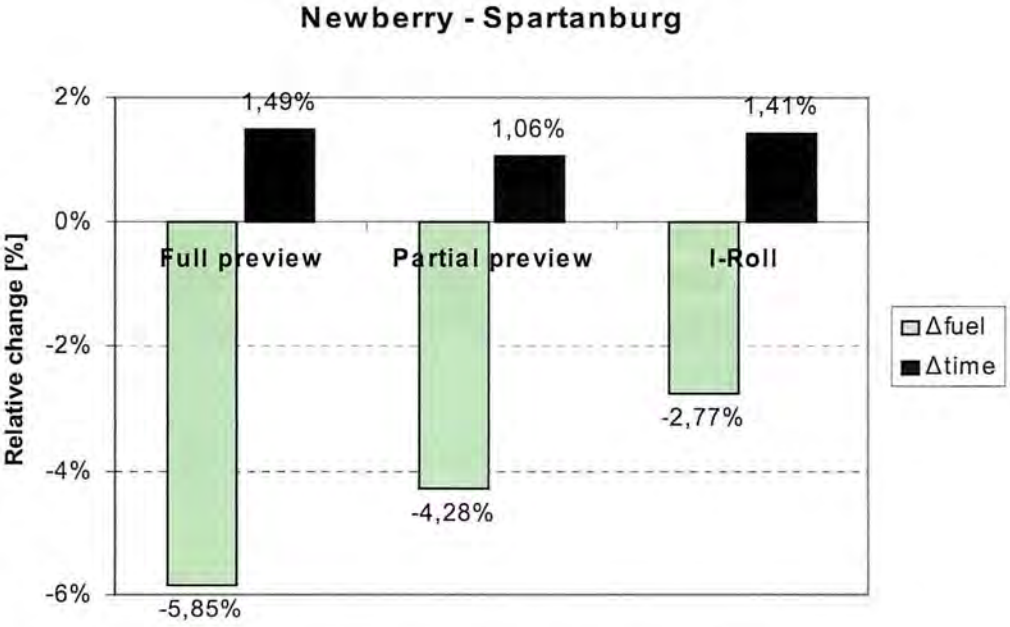

Figure 22: Simulation results for the road Newberry - Spartanburg (USA).