Augmenting measured GPS driving patterns by roadmap information

Stefan Landgraf

---

submitted as thesis to attain the academic degree "Dipl.-Ing."
at the

Graz University of Technology

Institute of Electrical Measurement and
Measurement Signal Processing

assessors: O.Univ.-Prof. Dipl.-Ing. Dr. techn. Georg Brasseur
Dipl.-Ing. Dr.techn. Bernhard Schweighofer

Graz, February 2011

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am …….. ……..
(Unterschrift)

Englische Fassung:

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

…… ……..
date (signature)

Abstract

Hybrid Electric Vehicles show great potential for improving fuel consumption and reducing emissions. To tap the full potential, sophisticated control strategies are needed. These strategies require a prediction of the power-demand, which depends on the elevation of the road and the speed of the vehicle. The aim of this diploma thesis is to gather this data and ascertain if it is possible to predict the necessary information from digital maps. Vehicles equipped with a Global Positioning System (GPS)-receiver were employed to gather sample data. A mapmatching algorithm was developed to bring the GPS-tracks into relation with a digital road map. This made it possible to compare the different GPS-tracks with one another and with the digital road data.

Comparison of the height data from a single GPS-track and the digital map data showed differences of up to 40 m. Several GPS-tracks were then averaged and the resulting height profile was again compared to digital map data. The resulting difference was $\sim$10 m for the selected sample route. Comparison of the speed profiles with the digital road map data showed a high correlation between the vehicle speeds and sharp bends. It was also found that the speed limits and the right of way of crossroads are important factors for the speed of the vehicle.

Kurzfassung

Hybridfahrzeuge haben großes Potential den Treibstoffverbrauch zu verbessern und Emissionen zu reduzieren. Um dieses Potential voll auszuschöpfen, sind hochentwickelte Regler-Konzepte notwendig. Diese Konzepte benötigen den zukünftigen Leistungsbedarf, der von zukünftigen Geschwindigkeits- und Höhen-Daten abhängig ist. Das Ziel dieser Diplomarbeit ist es, diese Daten zu sammeln und herauszufinden, ob es möglich ist, diese notwendigen Informationen aus digitalem Kartenmaterial zu prädizieren. Es wurden Fahr-zeuge verwendet, die mit einem Global Positioning System (GPS)-empfänger ausgerüstet waren, um die Testdaten zu sammeln. Ein Mapmatching Algorithmus wurde entwickelt, um die GPS-tracks in Relation mit den digitalen Straßenkarten zu bringen. Dies erlaubte einen Vergleich zwischen den verschiedenen GPS-tracks und den Straßendaten.

Der Vergleich der Höhendaten eines einzigen GPS-tracks mit dem digitalen Kartenmaterial zeigte Differenzen von bis zu 40 m. Mehrere GPS-tracks wurden dann gemittelt und das resultierende Profil wieder mit dem digitalen Kartenmaterial verglichen. Dies ergab eine Differenz von $\sim$10 m für die untersuchte Route. Der Vergleich von Geschwindigkeitsprofilen mit dem Kartenmaterial zeigte eine hohe Korrelation zwischen Fahrzeuggeschwindigkeiten und engen Kurven. Es wurde auch ermittelt, dass die Geschwindigkeitsbegrenzungen und die Vorrangregeln an Kreuzungen wichtige Faktoren für die Fahrzeuggeschwindigkeit sind.

Contents

Chapter 1

Introduction

Hybrid Electric Vehicles (HEVs) usually employ a combination of an Internal Combustion Engine (ICE), an electric machine and an electrochemical battery for their propulsion system. The electric machine can either act as electric engine or generator. The battery acts as a buffer for the energy in the system. An HEV demonstrates better fuel economy than conventional vehicles propelled only by an ICE. This improvement is due to the following reasons:

- HEVs can shut down their ICE completely while idling. This does not impair the drive-ability of the vehicle, as the electric engine will supply the necessary energy to get the vehicle started.
- Instead of losing kinetic energy as thermal energy when braking, an HEV can apply regenerative braking to transform part of this energy into electricity.
- The electric machine can either supply additional energy, or store excess energy in the battery. This allows the ICE to be continuously operated near its optimum efficiency point.

The gain in fuel economy depends heavily on the grade of hybridization and the driving environment. In general it can be said that HEVs perform well in urban areas with relatively low and frequently alternating velocities. Depending on these factors improvements in fuel economy, ranging from 10% to 30%, can be achieved [1]. In contrast a conventional ICE-based system will perform better than a HEV at continuous and high velocities. This is due to the fact that HEVs are heavier, as they are equipped with an additional electric machine and a larger battery. In addition idling time and braking are less likely to happen at this operating level.

To achieve a good improvement a sophisticated control system, that optimizes the energy flow in the system, is needed. Figure 1.1 shows a general energy-flow diagram of a HEV powertrain.

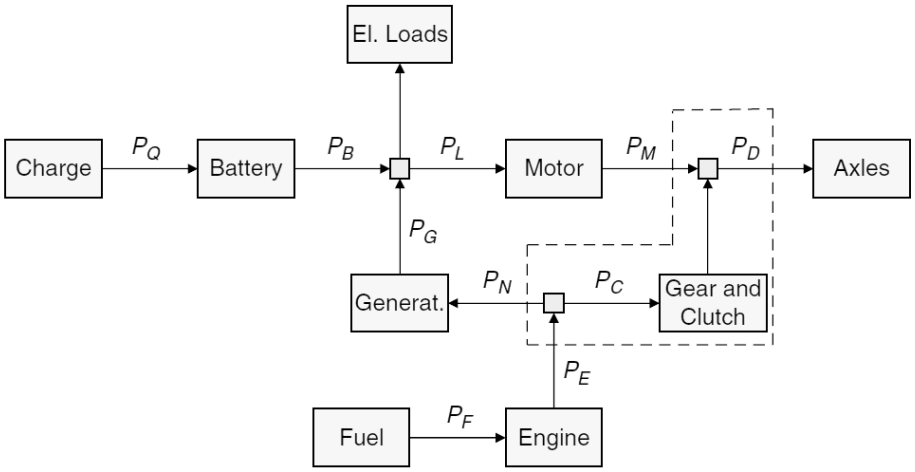

Figure 1.1: Energy flow of a general hybrid powertrain [1]. Simpler architectures can be derived by omitting some of the paths and nodes. The dashed block can be physically realized with a planetary gear set.

A critical aspect of the control system is managing of the battery's State of Charge (SOC). It is vital that the SOC of the battery does stays within certain boundaries, otherwise the battery's life cycle will be reduced. The lifetime of the battery also depends on the temperature and the number of charge and discharge cycles.

> This means that the controller must make sure that the necessary power is supplied to the vehicle, while minimizing fuel consumption and ensuring that the constraints concerning the battery are met.

The power needed by the vehicle depends on the following factors:

- Acceleration and deceleration
- Slope of the road
- Traffic congestion
- Other influences

Advanced control strategies make use of predictions on the future power-demand.

The aim of this work is to provide such data. Height- and speed-profiles will be examined. The free data from the Shuttle Radar Topography Mission (SRTM) will be used in combination with data gained from measured

Global Positioning System (GPS)-tracks, to gain and analyse the feasibility of this data. It will also investigate the prediction of future profiles based on this data. Further studies at the Institute of Electrical Measurement and Measurement Signal Processing (EMT), at Graz University of Technology, will use this data to simulate different controllers.

This diploma thesis is structured as follows:

In chapter 2 a sample of existing control strategies will be summarised. This will show how the information from driving patterns can be applied in conjunction with a controller. It also introduces several mapmatching algorithms that will be needed later.

Chapter 3 will present the different data sources used in this work. These consist of the digital road data, the Digital Elevation Model (DEM) data and the track data. It will also briefly describe the geodetic reference systems in which these datasets were available. Some of the necessary preprocessing steps will also be described.

The combination of the different geospatial datasets is an important aspect of this work. The mapmatching algorithm used will be described in detail in Chapter 4.

In chapter 5 the MATLAB functions implementing the algorithm described in the chapters 3 and 4 are given. The focus is on how the different functions and scripts are called so the result of this diploma thesis can be reproduced.

Chapter 6 will present the results of this study. Height and speed profiles will be analysed and compared with the concepts introduced in chapter 2.

Finally chapter 7 will summarise the results and provide an outlook for further work.

Chapter 2

Literature review

As mentioned in the introduction, advanced control strategies ideally use information of future driving conditions and power demand, to maximise the performance of the vehicle. This information is often referred to as driving patterns. In the following section 2.1, the concept behind driving patterns will be described. The section then will go on to provide numerous examples of how such driving patterns were applied in control strategies.

The second section 2.2 of this chapter will deal with measuring driving patterns with a Global Positioning System (GPS)-receiver. The GPS will be described in short. It then deals with how the driving pattern or tracks gained with the GPS, can be put in relation to each other and to a digital map with mapmatching algorithms.

2.1 Driving patterns and their application

Driving pattern, driving cycle and driving condition are terms often used in control strategies that take into account information about the driven route. To avoid confusion the terms are used in this work to match the following descriptions:

Ericsson described a driving pattern as follows [2]: *Driving pattern is generally defined as the speed profile of the vehicle, but can be expanded to include other parts of driving behaviour, such as gear changing. The concept of driving pattern does not normally include trip generation, choice of travelling mode or route choice.*

Driving cycles in this work refer to standardised driving patterns, which are used by the authorities to measure pollutant emissions of a vehicle. Examples for these are the New European Driving Cycle (NEDC) and the American Federal Test Procedure (FTP-75). The measurement is done by running the vehicle on a dynamometer and measuring the emissions.

A driving condition will refer to a set of similar driving patterns classified into one group. Often the classification was done to match the subjective

interpretation of the driving condition by a human. For example driving at high constant speeds over long periods, can be interpreted as driving on a highway.

Ericsson [2] has studied what properties define a driving pattern. A large set of driving patterns was gathered by driving in an urban area. 62 parameters were gathered for each driving pattern. These parameters range from average speed and acceleration, over the number of stops, to engine speed depending on the selected gear. This large set of parameters was then reduced to 16 independent driving pattern factors.

The next step was to determine which parameters had a significant relation to fuel consumption and emissions. The 16 factors were used as input variables of two emission models. From these models the dependency, between the driving pattern factors on one side and fuel consumption and emission produced on the other side, was gleaned. Nine of these factors were found to have a high degree of influence on fuel consumption and emissions. See table 2.1 for the list of these factors.

The results of Ericsson's work show the importance of driving patterns for the control strategy. As fuel consumption and emissions produced are dependant on the driving pattern, a controller that takes driving patterns into account will perform better than a controller that does not.

Lin et al. [3] have done research on an adaptive power management control strategy for a diesel engine truck with parallel hybridization. They formulate the controller as an optimal control program. The output of the controller is the power-split between the diesel engine and electric machine. The cost function is defined over a driving pattern of length N as follows:

$$J = \sum_{i=0}^{N-1} [fuel(i) + \mu * NOx(i) + v * PM(i)] + \alpha(SOC(N) - SOC_f)^2$$
$$\mu, v, \alpha \ldots \text{factors that weight the different parameters.} \tag{2.1}$$

The goal was to minimize fuel consumption and the emission of Nitrogen oxide (NOx) and Particulate Matter (PM). The last term $\alpha(SOC(N) - SOC_f)^2$ ensures that the battery is not depleted by the end of the driving pattern. Various constraints for the engine speed, engine torque and battery State of Charge (SOC) were also defined. Based on a given driving pattern the required wheel torque and wheel speed were calculated as the inputs for the problem. Dynamic Programming (DP) was used to solve this control problem for the given driving pattern.

The DP approach was not considered feasible for real-time driving conditions. It required information supplied by future driving patterns. The

Table 2.1: Driving pattern factors affecting fuel usage and emission [2].

	Factor name	Typical parameter
1	Deceleration factor	Average deceleration
2	Factor for acceleration with strong power demand $\star$	Relative positive acceleration
3	Stop factor $\star$	Percentage of time $v < 2$ km/h
4	Speed oscillation factor $\star$	Frequency of oscillations of the speed curve per 100 s
5	Factor for acceleration with moderate power demand $\star$	Percentage of time when va is $3 - 6$ m^2/s^3
6	Extreme acceleration factor $\star$	Percentage of time when acceleration exceeds 2.5 m/s^2
7	Factor for speed $15 - 30$ km/h	Percentage of time at speed $15 - 30$ km/h
8	Factor for speed $50 - 70$ km/h $\star$	Percentage of time at speed $50 - 70$ km/h
9	Factor for speed $70 - 90$ km/h	Percentage of time at speed $70 - 90$ km/h
10	Factor for speed $90 - 110$ km/h	Percentage of time at speed $90 - 110$ km/h
11	Factor for speed > 110 km/h	Percentage of time at speed > 110 km/h
12	Factor for late gear changing from gears 2 and 3 $\star$	Percentage of time engine speed is $2500 - 3500$ rpm in gear 3
13	Factor for engine speed > 3500 rpm $\star$	Percentage of time engine speed > 3500 rpm
14	Factor for moderate engine speeds in gear 2 and 3 $\star$	Percentage of time engine speed is $1500 - 2500$ rpm in gear 2
15	Factor for low engine speed in gear 4	Percentage of time engine speed < 1500 rpm in gear 4
16	Factor for low engine speed in gear 5	Percentage of time engine speed < 1500 rpm in gear 5

Factors that have a high influence on fuel use and emissions are marked with an asterisk $\star$

algorithm was also computationally heavy. Therefore a rule-based control was developed, based on the knowledge gained from the optimal solution.

As this sub-optimal controller was only based on one specific driving pattern, it would have probably performed less efficient under different driving conditions. Therefore a multi-mode driving controller was suggested. This concept assumed that several representative driving patterns could be used to represent all driving conditions. A control algorithm was designed for each driving condition. Based on the determined driving condition, the according algorithm was chosen. Two characteristics were selected for identification: average positive power demand and the standard deviation of the positive power demand. A quadratic cost function was used to determine the driving condition. The representative driving patterns were created from simple mathematical operations in the cited work.

The controllers were tested by simulation. This showed that the multi-mode controller performed up to 5% better than the single-mode controller. This result confirms that the use of driving patterns increases controller performance. The comparison of the multi-mode controller with the theoretical DP-approach showed great potential for further performance gain. This suggests further studies investigating the prediction of driving patterns.

Rajagopalan et al. [4] have developed a control strategy for Hybrid Electric Vehicles (HEVs) using Fuzzy Logic Controllers (FLCs).

An optimization problem was formulated to calculate the optimum torque of the Internal Combustion Engine (ICE). The cost function used the following normalized input parameters: ICE efficiency and NOx, Carbon monoxide (CO), Hydrocarbon (HC) emissions. The optimum torque was derived for *all* engine speeds, resulting in the engine's torque graph over its rpm. The controller follows the idea of *load levelling*. The electric machine is used to provide the difference between required torque and the optimum torque from the ICE. Surplus energy was stored in the battery.

This optimization did not take into account the battery's SOC. To prevent the battery from depleting a FLC was introduced. This controller took the SOC and the requested torque as inputs. It contained a set of rules so that the input requirements were met, and sets the ICE to provide the necessary torque.

To further improve the performance a predictive controller was added. It was assumed the route is known in advance. Future speeds and elevation values were estimated by averaging over a set of sample points. It was assumed that these sample values were available from Traffic flow speed- and road elevation-data. The future state was compared with the current state to predict changes in speed and elevation. A FLC used these predictions to suggest the battery's charge or discharging mode to the overall controller. This improved the distribution of the SOC along the route.

Hajimiri and Salmasi [5] expand the predictive controller introduced in [4]. Future elevation and speeds were used by a FLC to set the mode of the battery. A third input, based on the battery's State of Health (SOH) was introduced. *The SOH is defined as the ability of a cell to store energy, source and sink high currents, and retain charge over extended periods, relative to its initial or nominal capabilities.* The rules of the FLC were expanded to increase the life time of the battery at the cost of increasing fuel consumption and more emissions.

Both of the works just summarized [4, 5] show an improvement in fuel efficiency by using a predictive controller. However they rely on predicting the of future speed from traffic flow information. This traffic flow data is not always available.[1] Alternatives for the speed prediction are therefore needed in such regions.

Montazeri-Gh et. al. [6,7] have employed driving condition recognition to enhance the performance of their HEV controller.

The driving data was collected by driving in real traffic conditions. Four driving conditions were defined: congested-, urban-, extra urban-, and highway condition. The pattern recognition was based on the concept of microtrips. *A microtrip is an excursion between two successive time points at which the vehicle is stopped.* From the driving data a histogram was created showing the number of microtrips per average speed. This histogram was integrated. The driving conditions were classified by assigning each condition the same number of microtrips, thus assigning each driving condition an average speed range.

To predict the future driving condition a hidden markov model was used. Such a model consists of a number of states and the transition between them. At discrete time intervals the state changes according to transition probabilities. The highest transition probabilities were stated in a *transition matrix*. This matrix was used to predict the future driving condition. It was calculated from the driving data collected beforehand.

A FLC was used. The requested torque and the SOC were used as inputs. For each driving condition the membership functions for the two inputs were optimised. In addition five road slopes ranging from -2% to $+2\%$ were added to the problem. This led to a total of 20 driving conditions, each with their own set of membership functions. The functions that corresponds to the predicted driving condition were then used.

The simulation of this controller showed that this method of predicting the driving condition, does indeed improve the performance. This means that this prediction method can be used with the methods based on traffic flow information, so that they can complement one another. In addition the influence of the speed and road grade were also analysed separately,

[1] For example was not available for the region studied in this diploma thesis.

showing that both these factors have a significant effect on the controller performance.

Liaw and Dubarry [8] have done driving pattern analyses to understand battery performance better. Driving data was collected by a fleet of purely battery-powered vehicles with an on-board data acquisition device. The collected data contained information about the motor controller, auxiliary power unit and battery management system. This includes amongst others: pack voltage, current, power and motor rpm.

The identification of the driving conditions is built upon *driving pulses. A driving pulse is defined as an active driving period between two contiguous stops in a trip.*[2] To classify a driving condition to a driving pulse the *average speed versus distance* plot, based on all collected data, was used. Five driving conditions were derived from this plot: stop-and-go, urban, suburban, rural and highway. For example a stop-and-go driving condition was characterized by low average speed and short travel distance of a driving pulse. The identification was done by fuzzy logic pattern recognition. The results from the pattern recognition were validated by comparing them with the driving condition identified by the driver of the track. This was done on a pulse by pulse, and a trip by trip basis. The membership functions were modified till the results matched the human drivers opinion.

Analogous to using driving pulses to characterize driving conditions, duty pulses were introduced to characterize the strain on the battery. The peak power of each duty pulse and the number of pulses per minute were used to classify the different strain levels. These two variables were chosen due to their importance to battery performance and life. A fuzzy logic pattern recognition, similar to the one used for driving condition recognition, was used to detect strain levels ranging from intensive to benign.

In the last step the graph representing the driving condition was overlaid with the graph representing the strain on the battery creating a vehicle usage profile. Supplementary work was planned to use this profile to extract key parameters and predict battery performance depending on road condition and driving habit. This shows the importance of driving patterns and their influence on battery performance.

Ichikawa et al. [9] investigated energy management systems along a commuting route. The aim was to provide the energy management system with the driving pattern for the complete trip beforehand. Because the efforts concentrated on a commuting route the trips were similar to each other.

The route was divided into subsections around traffic signals. This was done because the driving information around the signals is very important.

[2]The definition of a driving pulse used here is essentially the same as the definition of a microtrip used in the work by M. Montazeri-Gh [6, 7] described earlier.

Driving patterns along the route were collected. Then similar patterns were clustered together for each subsection. This reduced the amount of data that needed to be stored. From this data a state transition diagram was derived. The clustered patterns form the different states of a section and the probabilities from one state to the state in the next section were described in the transitions.

The work was expanded in [10]. External factors like the weather, the day of the week and departure time were taken into consideration. This led to different transition diagrams depending on these factors. In addition the current state of the vehicle was compared to the a priori prediction of said state. If the predicted state was wrong the state was corrected to prevent consecutive faults. After the trip was completed the database was updated with the data gained during this trip.

Ichikawa's work describes another way to create a model to predict future driving conditions based on previous measured driving patterns. The main difference to the studies done by Montazeri-Gh et. al. is the use of sub patterns instead of more generalized driving conditions.

In his dissertation Back [11] developed a model predictive control strategy with DP. The basic idea of model predictive control is as follows: Based on the current state and the model of system, a future state of the system is predicted. The controller optimizes the transition between these two states by minimising a quality criterion. The predicted value will always be distorted by disturbances. Therefore optimisation is redone at each time step. DP was used to solve the optimisation problem. The algorithm was improved for the problem to reduce its computation time.

This control strategy requires the prediction of a disturbance vector which contains the vehicle speed, acceleration and the slope of the road. The following parameters were provided from a digital map combined with data from a GPS-receiver: road slope, road curvature and speed limit. To predict the speed two factors were taken into account. The speed limit was taken directly from the map. The maximum speed v_k that can be driven in a curve is limited by the curvature[3] κ and the maximum acceptable side acceleration a_{side} affecting the driver.

$$v_k = \sqrt{\frac{a_{side}}{\kappa}} \tag{2.2}$$

The lesser of these two speed values gave the upper threshold for each point on the road. This gave a speed curve that jumped between the different thresholds. To improve the prediction of the speed, speed changes were modelled as a first-order time-delay element.

[3]See section 2.2.2 and figure 2.1 for an explanation of the curvature.

This shows how it is possible to derive a speed profile from digital map information.

An experimental design of this controller was implemented with a HEV. The test runs with an without this controller showed, that the developed controller reduced the fuel consumption of the HEV. Unfortunately is was not possible to repeat the test runs under similar conditions often enough to perform a statistical evaluation of the results. Nevertheless the practical adaptability of the predictive control concept has been proven.

Langari and Won [12, 13] developed an Intelligent Energy Management System consisting of the following components: Driving information extractor, driving situation identifier, fuzzy torque distributor and state of charge compensator.

The objective of the driving information extractor was to derive characteristic parameters from the current driving pattern. These parameters were partly based on the ones introduced in Ericsson's [2] work.

The driving situation identifier used the parameter gained from the information extractor to identify the driving condition. The roadway type and traffic congestion level were determined with the help of a neural network. A set of nine driving patterns along with their characteristic parameters were used as reference and to train the network. Another set of characteristics identified were the so-called driving trends. These were short term features of the driving pattern like *low and high speed cruise and acceleration/deceleration.* These can be easily described by average speed and acceleration. Similar to the driving trends were the driving modes. These also included *Start-up, acceleration/deceleration, cruise and stationary modes.* The difference to the trends was that the modes were identified by torque relations on the drive shaft. The last feature identified was the driving style. The terms calm, normal and aggressive driving styles were used for the classification. The relation between average acceleration and the standard deviation of said acceleration were used to identify the driving style. The driving trend, driving mode and driving style were modelled as fuzzy membership functions.

For each of the nine possible driving situations a rule base, with different membership functions, was selected for the controller. The membership functions derived from driving trend, driving mode and the battery's SOC were used as inputs. The output of the driving style identifier was a factor that was used to compensate the torque output of the controller.

The ibid. work shows how it is possible to incorporate short and long term driving condition information into a control strategy. The influence of the different components was tested by simulating the controller, with different sets of these components switched on and off.

Gong et al. [14] have investigated trip models to improve power management of a plug-in HEV. Plug-in HEVs differ from the HEVs discussed previously as they utilize more battery power and its battery can be recharged from a power socket. It was also assumed that the battery would be recharged after each trip, so its SOC should drop to a low threshold at the end of the trip. This charge-depletion mode therefore aimed to make optimal use of the cheaper and cleaner electricity and minimize the use of fossil fuel over the whole trip. Dynamic programming was used to determine this solution.

To find the optimal solution the driving pattern of the complete trip must be known in advance. Path-finding algorithms from Geographic Information System (GIS) technology searched for the driving path and divided it into segments. The GIS then provided the following data along this path: Segment length, slope, speed limit and intersection/traffic-light distribution. In addition historical and real time traffic flow were available for arterial and express roads.

Two trip models were created. The first was a simple model used for local roads where no traffic flow information was available. This model consisted of two cases. Either a constant acceleration to the maximum speed, a period at maximum speed and deceleration back to standstill, or just a constant acceleration followed by a deceleration to standstill. The second more advanced model used the speed gained from the historical traffic data to improve its performance.

The plug-in HEV was simulated with a controller optimized to make use of the complete trip model. The comparison with a standard controller showed that the new concept reduced the fuel consumption by over 50%.

The ibid. work has described how to create a model for a driving pattern based on historical traffic information. In also shows the huge potential to conserve fuel, when the controller of a plug-in HEV uses a prediction of the driving pattern of the complete trip.

Johannesson et al. [15] compared three optimal controllers with different degrees of input information to asses their potential to reduce fuel consumption. The thirst controller had access to the complete power demand over the whole route. It was used as the baseline for the other two controllers. The second controller had access to the position on the route and traffic-flow information. This simulated a vehicle with a GPS-receiver and traffic-flow information system. The last control system only knew the general environment of the track.

The studied route was carefully measured and the positions for every crossroad and other points of interest were logged. This measurement was then used as the baseline for 37 tracks taken with a GPS-receiver. All tracks were gathered under similar conditions. The time when the driving data was gathered, was deliberately chosen to avoid congested traffic.

These tracks were then used to create two models for the route. The route was modelled as a Markov-Chain Model. The difference between the two models was in accordance with the controllers, so one was position dependant and the other not. Obviously the position independent model could not take the information from the road grade into account. Dynamic Programming was used to find the optimal controller based on the route model.

The ibid. work concludes that the performance of the position dependent controller was almost identical to the ideal one. Furthermore, it looked into the effect of using only one previous drive cycle for the track information. The resulting performance was close to the ideal controller. Relying on a previously measured drive cycle would be impractical, so it was suggested the relevant information could be derived from digital maps.

Summary

In this section it was shown how driving patterns can be characterized [2] (Table 2.1).

Current driving conditions were derived, by calculating simple descriptive factors from measured patterns [3,8]. A more complex driving condition identification system, using a large number of characteristic factors was also shown [12, 13].

Prediction of a future driving condition with the help of transition diagrams based on past driving patterns, were presented [6, 7, 9, 10, 15]. Others suggest deriving the driving condition from digital maps and traffic flow information systems [4, 5, 11, 14].

The information of the driving condition was used to switch the parameters of the controller [3,6,7,12,13], for example by switching the membership function of a FLC. They were also used to select a mode for the battery [4,5] and to find an optimal solution via DP [11, 14, 15].

2.2 Mapmatching

After having described driving patterns and their application, this section will continue to look at how these patterns can be gathered from GPS tracks. It will also introduce mapmatching methods to combine track and digital map data.

2.2.1 Global Positioning System (GPS)

The GPS is a worldwide radio-navigation system formed by a constellation of 24 satellites and their ground stations [16]. The satellites are used as reference points by a receiver. By measuring the time of flight of signals from

at least three satellites the position of the receiver can be calculated by triangulation. Due to the high speed of the signals, differences in the reference time lead to high location error. Therefore a fourth signal is necessary to mitigate this error.

Accuracy of GPS

For the GPS to work accurately the speed of all signals must be equal. A GPS signal travels through the ionosphere, where it is slowed down by charged particles, and through the troposphere, where clouds cause additional speed reduction. When the signal reaches the ground it may be reflected from local obstructions causing a multi-path error. This effect is very common in towns and cities due to the high density of buildings. Additional problems are caused by incorrect satellite positions and their internal clock. The receiver also contributes with its noise.

Many of the above mentioned problems can be compensated for with Differential GPS (DGPS). A ground station acts as a stationary reference. It calculates the error of the signals and broadcasts it to the receivers for error correction. Typical error values in metres are listed in table 2.2

Another effect that can be observed in urban areas is the loss of the signal from one or more satellites. This leads to sudden jumps of the measured position. A study [17] has seen jumps of up to 400 m with GPS and up to 70 m for DGPS.

Table 2.2: Summary of GPS error sources in m [16]

Error Source	GPS	DGPS
Satellite Clocks	1.5	0
Orbit Errors	2.5	0
Ionosphere	5.0	0.4
Troposphere	0.5	0.2
Receiver Noise	0.3	0.3
Multipath	0.6	0.6

GPS-Tracks

As mentioned driving patterns are basically speed profiles. A GPS-receiver placed in a vehicle will measure the vehicle position at defined time intervals. Usually one measurement is taken every second. These sequential measured points over time are commonly referred to as a GPS-track. In addition a GPS-receiver usually also logs the speed for every point [18], resulting in speed profiles over time. Driving patterns can be directly extracted from these profiles. It was however noted in [8–10, 15], that the speed profile over

the distance is preferable to the one over time. The advantage of the profile over distance is that it can be easily compared with other profiles and map information.

2.2.2 Mapmatching algorithms

In the encyclopedia of database systems [19] mapmatching is described as follows: *Map matching denotes a procedure that assigns geographical objects to locations on a digital map. The most typical geographical objects are point positions obtained from a positioning system, often a GPS-receiver. In typical uses, the GPS positions derive from a receiver located in a vehicle or other moving object traveling in a road network, and the digital map models the embedding into geographical space of the roads by means of polylines that approximate the center lines of the roads. The GPS positions generally do not intersect with the polylines, due to inaccuracies. The aim of mapmatching is then to place the GPS positions at their "right" locations on the polylines in the map.*

The Encyclopedia categorizes mapmatching algorithms into on-line and off-line mapmatching. The on-line algorithms are typically used in vehicles, to display the current position. On the other hand, the off-line algorithms are used to interpret the trip after it was driven.

The second categorisation done by the encyclopedia distinguishes the algorithms, by how they correspond the datasets to each other. Point-to-point, polyline-to-polyline and point-to-polyline correspondences can be established.

In her work Czommer [17] has described several basic principles for mapmatching algorithms, that will be summarized here. Two general concepts were introduced: mapmatching based on the comparison of curvature profiles and based on the point-to-point comparison of the two dimensional Cartesian coordinates. The curvature profiles comparison equals a polyline-to-polyline algorithm.

Mapmatching based on curvature profiles

These methods are based on the curvatures of the road and track. The curvature can be described as the change of direction per length. The curvature κ depends of the angle $d\varphi$, between the tangents of two points separated by the distance ds. Compare with figure 2.1.

$$\kappa = \lim_{\Delta s \to 0} \frac{\Delta \varphi}{\Delta s} = \frac{d\varphi}{ds} \tag{2.3}$$

The curvature was calculated as a function over distance for the track and for the road network. A mathematical transformation, with four unknown

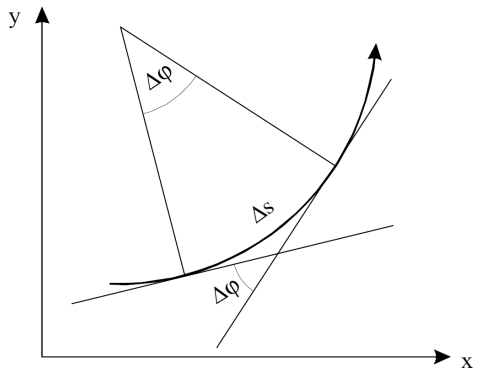

Figure 2.1: Curvature definition [17].

parameters, is defined between the two curves. The transformation function needs to be linearised. It can then be solved via the method of least squares or a kalman filter. The method of least squares was not considered real-time applicable in Czommer's work as it requires the normal equations matrix $(X^{\mathrm{T}}X)$ to be rebuilt and inverted after each iteration. The recursive kalman filter only needs to do the inversion of the normal equations matrix after all the iterations are completed.

Mapmatching based on Cartesian coordinates

The track and roads were both interpolated with line segments of constant length. Geometric transformation parameters were calculated to transform the set of track points into the road points. By analysing these parameters the best fit was chosen, thereby establishing a point to point relation between track and road. See also figure 4.3. The following transformations were evaluated: translation, similarity transform and affine transform.

The translation consists of a simple shifting of the points along the x and y axes. This shift can be represented with a translation vector. The vector with the lowest deviation is selected as best fit. However the transformation using only translation cannot cope with systematic measurement errors and is therefore not practicable for this work.

The similarity transform allows the translation, rotating and scaling of the points. In addition the affine transform allows for shearing.

Czommer's comparison of results from the similarity and affine transform showed that the affine transform performs better. The reason for this is that the affine transform possesses more degrees of freedom. After the viable roads are identified it is necessary to perform plausibility checks to select the most likely road.

Based on the results of this research the mapmatching algorithm using the affine transform was chosen for this diploma thesis and will be described in more detail in section 4.1.

Chapter 3

Data Sources and Preprocessing

This chapter will present the different data sources used for this work. There are three main data sets. The road network data, the Digital Elevation Model (DEM) and the track data. The different geodetic reference systems used in this work will also be described. The second part of this chapter will explain two necessary preprocessing steps: The undulation of the geoid and interpolation of the DEM data.

3.1 Data Sources

3.1.1 Road Network

The area studied was the Austrian state of Styria. The road network data was provided by GIS-Steiermark [20] and is available free on their website. Table 3.1 holds an overview of this data. This network does not cover every road in Styria. Many roads from the capital town Graz are missing. Another noteworthy omission are the ramps to and from motorways. Figure 3.1 shows the road map and the state boundaries.

The data only holds the most basic attributes. There is no information about speed limits or tunnels. It is possible to derive the road type from the name of the road and get a general idea about the possible speed on this road.

The data was provided in the ESRI Shapefile format [21]. A geospatial vector data format commonly used by GIS software. uDig [22], an open source GIS software program, was used for easy visualization of the network.

3.1.2 Digital Elevation Model (DEM)

The road network data does not contain any information on road slopes. An additional data source was therefore needed to obtain this data. As

Table 3.1: Road network data overview [20]

Geodata	Roads
Description	Main road network
Scale	1/50000
Coordinate system	UTM N33
Format	EsriShape [21]
Structure	kilometre line
Source	State of Styria
Updated	2006
Attributes:	
STRNR	Road number
STRNAME	Road name
ROUTE	Route

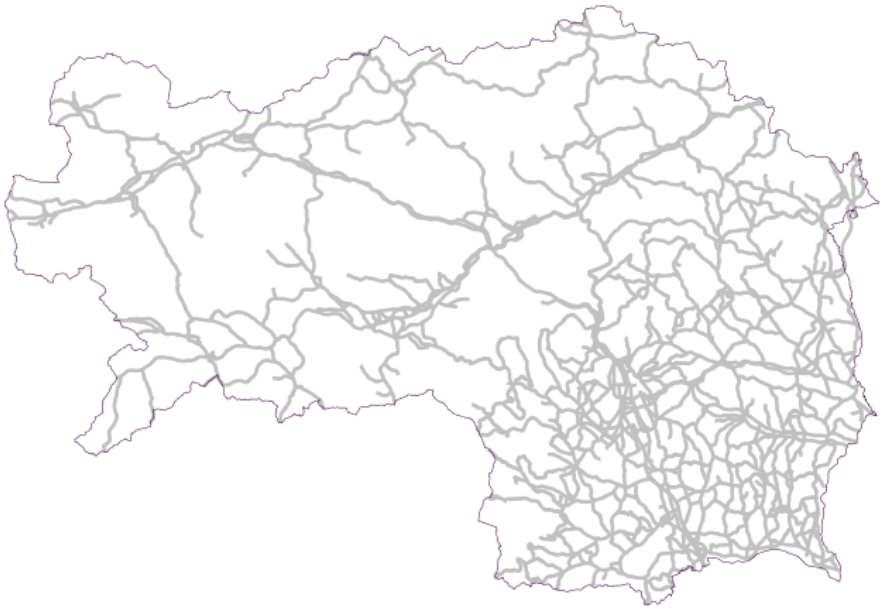

Figure 3.1: Styria road network

the basis for our Digital Elevation Model (DEM) the data provided by the Shuttle Radar Topography Mission (SRTM) [23] was used. The data from SRTM was collected by the space shuttle Endeavour in February 2000. It used the interferometric synthetic aperture radar technique to acquire the topographic data. The resolution of the data is three arc seconds[1]. Due to the measuring method, based on the reflection of electromagnetic waves, this elevation data does not necessarily represent the actual surface of the earth. Dense vegetation and buildings can cause unwanted premature reflection. Problems also occur around mountain peaks and steep slopes.

A dataset [24] derived from the SRTM was used for the highly mountainous terrain in the area studied. The parts that were void in the original data were filled. This data is available in a three times higher resolution than the original data. In the more flat eastern part this improved DEM was not available so the SRTM data was used there.

3.1.3 Track data

The track data was collected by members of the Institute of Electrical Measurement and Measurement Signal Processing during their usual trips, to and from Graz. The HOLUX M-241 DGPS unit [18] was used to collect the following data vectors of the tracks:

- time
- latitude and longitude
- height
- speed
- number of satellites visible

As mentioned in the discussion on GPS (section 2.2.1) four satellites need to be visible for correct positioning data. Measurements made with less than four satellites were therefore discarded.

3.1.4 Geodetic reference systems

The information on the reference systems presented here was taken from [25]. The data used in this work uses two different reference systems. The World Geodetic System 84 (WGS 84) and the Universal Transversal Mercator (UTM) system.

The WGS 84 uses an ellipsoid to approximate the whole earth. The latitude and longitude angles describe the relation of every point in the

[1] 1 arc second roughly equals 30 meters

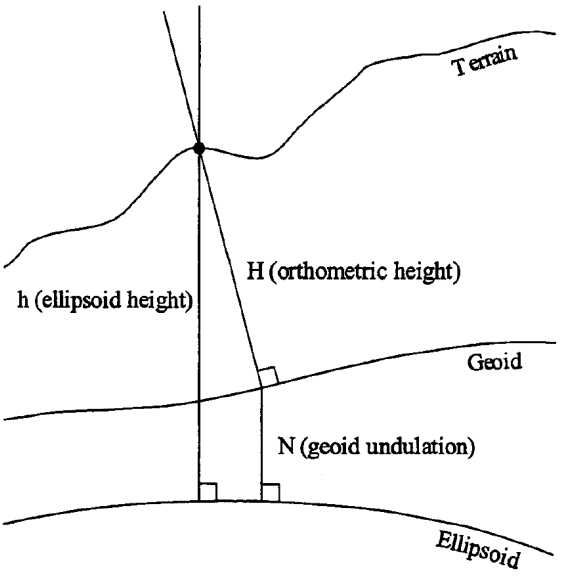

Figure 3.2: Undulation of the geoid [27]

world to the equator and a meridian near Greenwich. The WGS 84 is the standard reference system for GPS. It is also used by the DEM data.

The UTM system can be visualized as a cylinder that touches the earth at the equator. The earth's surface is brought in relation to it using a cylindrical map projection, also referred to as Mercator projection[2]. The cylinder is divided into sixty zones, each with a separate Cartesian coordinate system. The road network data is provided in UTM format. The UTM system is also used for all internal calculations to make use of the simple euclidean metric.

The transformation between the two systems is a standard GIS application. See [26] for more details.

3.2 Preprocessing

3.2.1 Undulation of the Geoid

As already mentioned GPS uses WGS 84 as its reference coordinate system. This system uses a single ellipsoid covering the whole earth as its basic altitude reference. Due to variations in the densities of the earth's crustal materials as well as terrain variations, the observed gravity of the earth varies irregularly from point to point. A gravitational equipotential surface, called

[2]More precisely each zone uses a specifically defined secant transverse Mercator projection.

the geoid, is used to define the nominal sea level. The distance between the mathematical ellipsoid and the actual geoid is called the undulation of the geoid. Since the geoid is irregular, geoid undulations must be determined point by point. See figure 3.2

A program provided by [28] was used to compute geoid height from the data supplied by the GPS.

3.2.2 Interpolation of the DEM data

The road network data consists of a long number of consecutive points forming lines. The DEM on the other hand consists of a regular grid with assorted elevation values. To assign an elevation value to each point in the road network, it is necessary to interpolate this point in the DEM.

Two methods were tested using MATLAB: The bilinear and the bicubic interpolation. The difference of the result from a few samples lies in the range of a tenth of a metre. This was deemed to be insignificant compared with the range of the GPS-signals. The bilinear interpolation was chosen and is used for the rest of this work, because it takes up less computation time.

Chapter 4

Mapmatching

The following is a more detailed discussion of the mapmatching algorithm mentioned in section 2.2.2. The method can be separated into two parts. The first part matches track parts to small road segments. The second part merges these matched parts back together.

4.1 Point to point comparison with the Affine Transformation

The algorithm is divided into three sequential steps. First a preselection of the data is done to avoid unnecessary complex computations. Then the parameters of the affine transformation are calculated. Finally these parameters are validated.

Preselection of the input data

The road network was divided into segments of constant length. These segments overlap each other. This segmentation was done to gain smaller parts for comparison. At crossroads additional segments were added that start and end at those points. This was done because there is a high chance that the track will change direction at these locations and are therefore important points of interest.

A simple bounding box comparison was done for each road and the track. If this is successful the next test compares the track bounding box with each segment of the road. If they still overlap, the bounding box of the road segment is used to cut out all connected track points within said bounding box. This new data vector is used as the input in the mapmatching algorithm using the affine transform. See figure 4.1 for a simplified example of these checks.

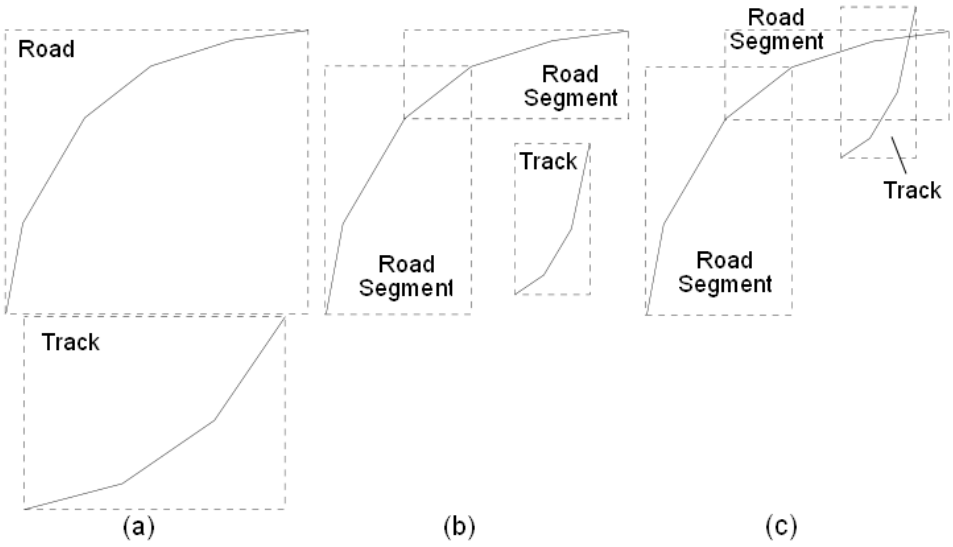

Figure 4.1: Preselection by bounding box comparison. (a) Road and track do not overlap. (b) Road and track do overlap, but the track does not overlap any road segments (c) The track overlaps a road segment, further comparisons are done with the affine transform.

Calculating the affine transformation

With the planar affine transformation it is possible to map distorted lines onto each other. This transform(4.5) applies a combination of translation(4.1), rotation(4.2), scaling(4.3) and shearing(4.4) operations. See figure 4.2.

$$t = \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{4.1}$$

$$A_r = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \tag{4.2}$$

$$A_{scal} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \tag{4.3}$$

$$A_{shear} = \begin{bmatrix} 1 & 0 \\ m & 1 \end{bmatrix} \tag{4.4}$$

$$\begin{aligned} \begin{bmatrix} \tilde{X}_i \\ \tilde{Y}_i \end{bmatrix} &= A_r * A_{scal} * A_{shear} * \begin{bmatrix} x_i \\ y_i \end{bmatrix} + t \\ &= A * \begin{bmatrix} x_i \\ y_i \end{bmatrix} + t \end{aligned} \tag{4.5}$$

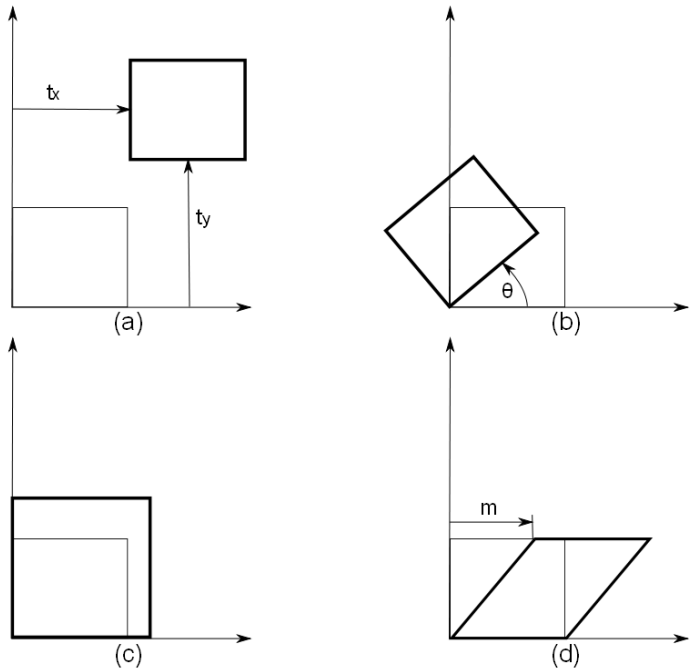

Figure 4.2: The affine transform can be separated into the following transformations: (a) translation, (b) rotation, (c) scaling, (d) shearing

To apply the affine transform on the track and road segment data, both sets must be interpolated with a constant spacial length. These new data vectors will be of unequal length. In the first step the shorter of the two data vectors of length n is compared with the first n elements of the larger vector. Next the comparison is done with the 2nd to n+1 elements of the larger vector and so on. (See figure 4.3) The actual comparison will be covered in the subsection on the evaluation of the affine transformation.

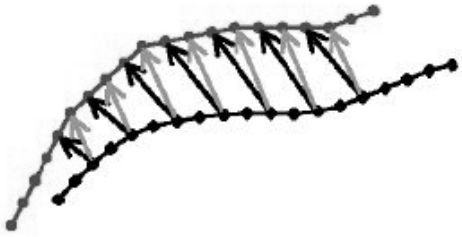

Figure 4.3: Point to point comparison with the affine transform.

In this application the track data are used as input variables x, y. The objective is to find parameters for A and t, so that the result of the transformation $\tilde{X}, \tilde{Y}$ approximates the road segment data X, Y. These data vectors must all be of the same length n. The subscript i indicates a single value from these data vectors.

There are six unknown parameters in the transform. To solve the problem unambiguously, three coordinate pairs would be needed. However the two line segments consist of more than three coordinate points. The problem of finding the parameters of the affine transform is therefore an overdetermined equation. This equation was solved by using the formulas given in [29]. These formulas are based on the least squares method and weighting factors relative to the centroid.

$$A = \begin{bmatrix} a_1 & -a_2 \\ a_4 & a_3 \end{bmatrix} \tag{4.6}$$

To calculate the parameter the centroid coordinates

$$\begin{aligned} x_c &= \frac{1}{n} * \sum_{i=1}^{n} x_i \qquad y_c = \frac{1}{n} * \sum_{i=1}^{n} y_i \\ X_c &= \frac{1}{n} * \sum_{i=1}^{n} X_i \qquad Y_c = \frac{1}{n} * \sum_{i=1}^{n} Y_i \end{aligned} \tag{4.7}$$

and the relation of all points to the centroid are needed.

$$\begin{aligned} \overline{x}_i &= x_i - x_c \qquad \overline{y}_i = y_i - y_c \\ \overline{X}_i &= X_i - X_c \qquad \overline{Y}_i = Y_i - Y_c \end{aligned} \tag{4.8}$$

With these coordinates the parameters of the affine transformation can be calculated:

$$\begin{aligned} a_1 &= \frac{\sum(\overline{x}_i\overline{X}_i) * \sum(\overline{y}_i^2) - \sum(\overline{y}_i\overline{X}_i) * \sum(\overline{x}_i\overline{y}_i)}{N} \\ a_2 &= \frac{\sum(\overline{x}_i\overline{X}_i) * \sum(\overline{x}_i\overline{y}_i) - \sum(\overline{y}_i\overline{X}_i) * \sum(\overline{x}_i^2)}{N} \\ a_3 &= \frac{\sum(\overline{y}_i\overline{Y}_i) * \sum(\overline{x}_i^2) - \sum(\overline{x}_i\overline{Y}_i) * \sum(\overline{x}_i\overline{y}_i)}{N} \\ a_4 &= \frac{\sum(\overline{x}_i\overline{Y}_i) * \sum(\overline{y}_i^2) - \sum(\overline{y}_i\overline{Y}_i) * \sum(\overline{x}_i\overline{y}_i)}{N} \\ t_x &= X_c - (a_1 x_s - a_2 y_s) \\ t_y &= Y_c - (a_4 x_s + a_3 y_s) \end{aligned} \tag{4.9}$$

The value for N is defined as:

$$N = [\overline{x}_i^2] * [\overline{y}_i^2] - [\overline{x}_i\overline{y}_i]^2 \tag{4.10}$$

The deviations are given as the difference between the transformed and the road coordinates

$$\begin{aligned} W_{x_i} &= X_i - \underbrace{(a_1 x_i - a_2 y_i + t_x)}_{\tilde{X}_i} \\ W_{y_i} &= Y_i - \underbrace{(a_4 x_i + a_3 y_i + t_y)}_{\tilde{Y}_i} \end{aligned} \tag{4.11}$$

with the standard deviations

$$\sigma_x = \sigma_y = \sqrt{\frac{\sum(W_{x_i}^2) + \sum(W_{y_i}^2)}{2n - 6}} \tag{4.12}$$

This gives all the unknown parameters needed for the affine transform (4.5).

$$\begin{bmatrix} \tilde{X}_i \\ \tilde{Y}_i \end{bmatrix} = \begin{bmatrix} a_1 & -a_2 \\ a_4 & a_3 \end{bmatrix} * \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{4.13}$$

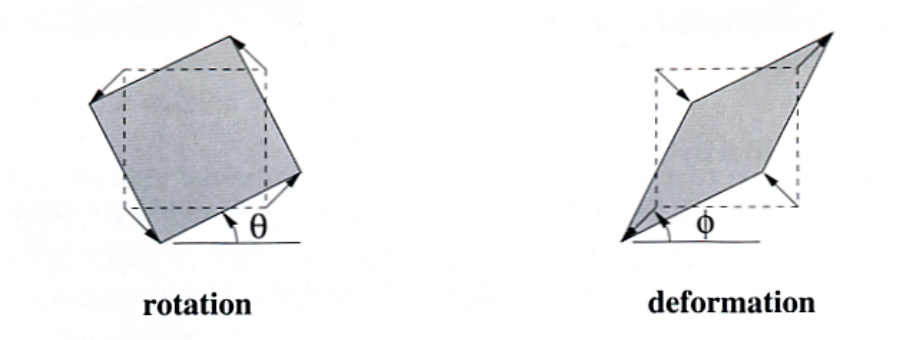

Figure 4.4: Distortions arising from the affine transformation. Rotation by $R(\theta)$ and deformation $R(-\phi)DR(\phi)$. Note the scaling directions in the deformation are orthogonal. [30]

Evaluating the affine transformation

This result (4.13) still needs to be interpreted. The translation vector $\begin{bmatrix} t_x & t_y \end{bmatrix}$ does give us a general feeling as to how far the point was moved by the transformation. To exclude roads that cross the track, it is also necessary to look at the rotation component of the matrix A. This matrix can be decomposed [30] as follows:

$$A = R(\theta)R(-\phi)DR(\phi) \tag{4.14}$$

R denotes a rotation matrix and D a diagonal matrix. These matrices can be interpreted as follows:

The coordinates are rotated by the angle ϕ, scaled by elements of D and rotated back again by $-\phi$. This characterises the deformation of the transformation. The final rotation by θ allows us to gain insights on the angle between two polylines. See figure 4.4 for a graphical interpretation.

To gain these rotation and scaling matrices (4.14) the Singular Value Decomposition (SVD) is used:

$$\begin{aligned} A = U\Sigma V^T &= (UV^T) * (V\Sigma V^T) \\ &= R(\theta) * (R(-\phi)DR(\phi)) \end{aligned} \tag{4.15}$$

From this it follows that

$$R(\theta) = UV^T \tag{4.16}$$

$$R(\phi) = V^T \tag{4.17}$$

$$D = \Sigma \tag{4.18}$$

For every comparison the parameters of the affine transform are calculated. The translation vector (4.1) and the angle θ (4.16) are used to determine if the transformation is within a *reasonable* range. The range permitted for these two tests is a classical tradeoff situation. Set the range too small and valid results will be discarded, set it too large and wrong identifications will be made. For the data used in this study the following ranges are found to be acceptable:

$$|\theta| < 10^\circ \tag{4.19}$$

$$|t_x| < 100\,m, |t_y| < 100\,m \tag{4.20}$$

The above values were determined by manually checking the result from sample tracks. The set of parameters with the minimum standard deviation (4.12) is used to gain the point to point relationship of the track to the road.

Problems with the algorithm

As mentioned in section 3.1.1 the road data is incomplete. It is therefore impossible to determine intersections between known and unknown roads. If the track switches at such an unknown crossroad, the algorithm will match short pieces of unknown road onto the wrong road. The simplest solution for this problem would be to obtain more complete road network data.

A similar effect happens if the track does a U-turn on the road. Such an event is impossible to predict from the network data and no point of interest can be placed at that point. In such a case the algorithm may fail completely. It would be possible to determine such a turn from the track data, then split the track at this point and match both track parts separately. This case occurred very rarely in the measured data and was excluded from the analysis of the results.

If the road segment is a straight line it is possible that the affine transformation results in an angle $\theta \simeq 180^\circ$. To cover this case an additional condition was added for this angle.

$$|\theta| < 10^\circ or |\theta \pm 180^\circ| < 10^\circ \tag{4.21}$$

The check for the translation vector (4.20) prevents converse curves being detected as true.

Another problem for straight line segments is that the matched points may lie anywhere within the borders of the translation vector (4.20). The standard deviation (4.12) gives us no information on the best match, as its values will be almost zero for all possibilities. In hindsight this problem and the one above could be avoided by a more intelligent segmentation of the

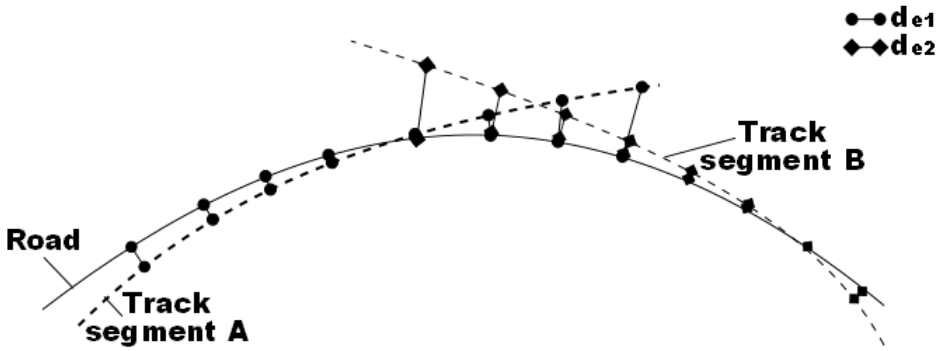

Figure 4.5: Merging of track parts. The point of transition between track segments A and B, is at the point where sum over all distance errors (d_{e1}, d_{e2}) is a minimum.

road network. Instead of just segmenting after a certain geographic length it would be possible to segment after a road segment has changed direction at least once. This intelligent segmentation was not implemented in this work.

4.2 Merging the matched data

The result from the last section consists of small track parts that match corresponding road segments. The next step is to merge these parts back together.

Merging track parts along roads

The first merging steps consist of merging track parts that lie on the same road together. As explained in the last section the track segments overlap each other, therefore the matched track parts also overlap each other. The point of transition from one part to the next was determined as follows:

The track points corresponding to the geometric nearest road points are stored to a new vector. This is done for both adjoining parts. These vectors are generally smaller than their original track parts. The distance between the road points and the nearest track points are calculated (d_{e1}, d_{e2}). (Compare with figure 4.5) An error function is defined depending on the point of transition i.

$$
\begin{aligned}
& e_i = \sum_{n=1}^{i} d_{e1}(n) + \sum_{m=i+1}^{p} d_{e2}(m) \\
& i = 1 \dots p-1 \\
& p \dots \text{number of overlapping points}
\end{aligned}
\tag{4.22}
$$

The point of transition i, is the point for which the error function (4.22) is minimal. These steps are repeated along the segments. The newly merged part is merged with the next track part in sequence.

This formula (4.22) ensures that no point will have a duplicated entry in the merged dataset. Gaps can only appear in the merged data if there are no matched tracks segments for that road part in the first place. However the merged parts were matched with different parameters. This means that the spatial relation at the transition point is not necessary smooth. The spacial distance between the points next to the transition will not correspond to the distances within a segment. See the figures 5.3 and 5.4 for an example.

Merging track parts between different roads

After merging the track parts along a road, the next step is to merge these new parts together.

The simplest case is when the sequenced parts are joined at crossroads. In this case the nearest point from the track parts to said crossroads are used as transition points. Ideally each track part would share the same point on the crossroads location. A subsequent check on the data is therefore necessary to remove such duplicate points from the data. It is also possible for small gaps to appear. This happens if one of the matched track parts overshoots the location of the crossroads. These gaps were left in the data.

If the crossroad information is not available due to incomplete map data, it is not possible to find a rule that will merge all test tracks satisfactorily. A simple user interface was implemented to take care of these cases. Two examples of such cases that commonly occurred are:

- Entering or leaving a motorway
- Two road segments running closely parallel to each other. The affine transform identified both as viable segments

Inserting track parts that were not matched

Track parts that were not matched to any road are inserted back unmodified. Such parts are identified by looking for large gaps in the index of the new matched track.

Comments

The results from the algorithm were visually verified. The resulting track and road profiles will be compared in the next chapter. It must be noted that the method described here is not state of the art. Mapmatching algorithms are a huge and complex subject. A more comprehensive examination would, however, not be feasible in this work.

Chapter 5

Code description

This chapter will outline the MATLAB-code that performed the mapmatching and evaluated the results. The aim is to give an understanding of how the different code pieces interact with each other and show the functionality of the main code blocks. Generally speaking each function or script described, will take at least one MATLAB ".mat"-file as input and store the result to a new file. Often this new file will then be the input for the next function. However several files are needed for more then one function. These files will be noted in this chapter and the table 5.1 lists them along with the functions that created them. See also the flowcharts in the figures 5.1, 5.2 and 5.12 to see where they are created and used.

Table 5.1: Important data files

function/script name	data name
Assignheights (splitA2A9roughv2)	map data
split_roads_at_intersections	split map data
find_road_intersections	crossroads
find_road_intersections_simple	crossroads simple
gpssplitting	un-matched tracks
insert_missing_org_track	matched tracks

5.1 Preprocessing the road network data

A flowchart showing the steps described in this section can be viewed in figure 5.1.

As already mentioned in section 3.1.1 the road data is provided in the ESRI Shapefile format. The `shaperead`-function, that is included in the MATLAB Mapping Toolbox™, was used to read this data into MATLAB.

The data was saved in MATLAB format for easy access[1].

The next step is to assign the road data height values by interpolating the DEM-data, as described in section 3.2.2. This was done in the `Assignheights`-script. In addition to the `height` this script calculates and adds several other values to the data.

```
1x533 struct array with fields:
    nr
    Geometry
    BoundingBox
    X
    Y
    Lat
    Lon
    height
    distance
    STRNR
    STRNAME
    ROUTE
```

The `nr` field is added to assign each coordinate an index. The latitude and longitude values were needed for the interpolation and saved in case there was further use for them. The cumulated euclidean `distance` to the first coordinate pair is also calculated for every point.

There are two motorways in the road data. These cross such a large region that the bounding box comparison shown in figure 4.1a, would be successful for most of the tracks. To reduce the computation time of the bounding box checks, these two roads were split into smaller road segments in the `splitA2A9roughv2`-script. This does not change the structure of the data. The road segments are added as different roads to the data. This step is optional, but one must make sure the same road data is used all the time, including all derived datasets.

The output file of the `Assignheights`- or `splitA2A9roughv2`-script will be referred to as *map data* for the rest of this chapter.

The next step is to split the roads into overlapping parts as outlined in the beginning of section 4.1. The function `split_roads(input_file, output_file, d_split)` splits the road data into overlapping segments of length passed in `d_split`. The `split_roads_at_intersections(road_file, intersection_file, output_file, min_dist)` function does the additional splits around crossroads. The output data of both these functions has the same structure:

[1]There are a limited number of licences at the University of Technology Graz for the Mapping Toolbox™and it is therefore not always available.

```
>> roads

roads =

1x536 struct array with fields:
    Geometry
    BoundingBox
    STRNR
    STRNAME
    ROUTE
    road_part
```

The new `road_part` field holds another struct array with the overlapping data in it:

```
>> roads(502).road_part

ans =

1x3 struct array with fields:
    nr
    X
    Y
    Lat
    Lon
    distance
    height
    BoundingBox
```

In the data structure shown one can see the different `BoundingBox` fields. One for the complete `road` and one for each `road_part`. The output of the `split_roads_at_intersections`-function will be referred to as *split map data* for the rest of this chapter. Apart from the input and output-files the `split_roads_at_intersections` function takes two additional parameters. The `min_dist` is used to ensure that the `road_part`s are over a minimum total distance. This is used to prevent very small `road_part`s that would be very easy to mapmatch and therefore are problematic for the algorithm. The other parameter is the `intersection_file` that holds the information on the crossroads.

This file is generated by the function `find_road_intersections`(`input_file`, `output_file`) and will be called *crossroads* in this chapter. The `input_file` for this function is split road data calculated in `split_roads`. The `output_file` is shown in the following struct array:

```
1x3211 struct array with fields:
    Geometry
    X
    Y
    Road_1_Nr
    Road_2_Nr
    Road_1_index
    Road_2_index
    Road_1_part_index
    Road_2_part_index
    Road_1_polyline_index
    Road_2_polyline_index
```

Each field contains a single value. `X`,`Y` are the coordinates of the crossroads. `Road_1_Nr` `Road_2_Nr` contain the strings `STRNR` from the original data for each road. `Road_1_index` `Road_2_index` are the indices to the corresponding data in the data in the `split_roads` struct array. In turn `Road_1_part_index` `Road_2_part_index` are the indices to the corresponding `road_part` array and the polyline -indices point to the actual data within the `road_part` array. As this data was generated from overlapping road parts each crossroads will have more then one entry in this array.

The `find_road_intersections_simple` function is similar to the `find_road_intersections` function. This one used the *Map data* as input value. The `Road_1_part_index` `Road_2_part_index` are missing from the output data as they refer to the *split map data*. The polyline-indices here point to the actual data within the `road_index` array. The output data of this function will be called *crossroads simple* and is needed for plotting the profiles later in section 5.3.

5.2 The mapmatching code

5.2.1 Input track data

The track data was provided by members of the Institute of Electrical Measurement and Measurement Signal Processing (EMT) in the ".mat" format of MATLAB. The data is stored in a structure called `track`. This structure contains two main fields. The `doc` field contains some general information about the track:

```
>> fieldnames(track.doc)

ans =

    'trackname'
```

```
    'driver'
    'car'
    'netweight'
    'height_model'
    'payload'
```

The actual track data is stored under the `raw` field;

```
>> fieldnames(track.raw)

ans =

    'nr'
    'timebase'
    'seconds'
    'latitude'
    'longitude'
    'height'
    'speed'
    'nsat_used'
    'nsat_vis'
    'hdop'
    'vdop'
```

The most important data for the mapmatching algorithm are the two coordinates. `nr` is an index that is used in the merging process to make sure no values are duplicated. The number of used satellites for each point are stored in `nsat_used`. These are used to determine if the point is valid or not. The `height` and `speed` values are going to be analysed after the tracks are matched. The other variables were not used in the work. They are however still available to the matched data through the index `nr`.

The mapmatching algorithm developed uses a cell array called `files`, that contains the file-names, as input. To read all input-files from a folder into the `files` variable one can use the following code snippet[2]:

```
path_inputdata = 'D:\DiplArb\GPSTRACKS\
   TrackData\';
liste = dir(fullfile(path_inputdata,'*.mat'));
files = {liste.name};
```

[2]Note that no other ".mat" files can be located in that folder for this to work.

Different `files` arrays were also saved as a ".mat"-file to make them easy to access in later steps. Here is an example of what the content of this array can look like:

```
>> files(1:3)'

ans =

    'Augasse-TUG_bus&tramway_2008Nov05-072236.
       mat'
    'Gaberl-Graz_Citroen Xsara Picasso_2009
       Jan04-154603.mat'
    'Gallsbach-Graz_Opel Astra_2008Nov11
       -150527.mat'
```

The `files` array is also going to be used later on to select which result will be analysed and displayed.

5.2.2 The main mapmatching function

The main function for the mapmatching is:

```
function complete_mapmatch_run(path_inputdata,
   files)
```

The `files` variable has just been described. `path_inputdata` is a string containing the path in which the input ".mat"-files are stored. Figure 5.2 shows a flowchart of this function.

The first part of the function sets several flags. This allows separate parts of the code to be executed by including and excluding certain sub-functions. This is mostly for debugging purposes, as usually all flags will be set to true.

```
%flags
flag_geoidundulation      = 1;
flag_gpssplitting         = 1;
flag_mapmatch     = 1;
flag_merge_trackparts      = 1;
flag_merge_trackpartsroads          = 1;
flag_merge_trackpartsroadsorg      = 1;
```

Next the names of the map files are defined along with the folder that contains them. Compare with figure 5.1 to see where this data was generated.

```
% Map data
file_map = 'Strassen_Height_A2_A9_split.mat';
file_split_map = 'Strassen_Height_Split_cross.
    mat';
file_crossroads = 'crossroads.mat';
path_map = 'D:\DiplArb\Karten\GIS STMK\
    StrassenUTM33N\';
```

Afterwards several folders are defined. These are used to save the results of every sub-function to a ".mat"-file which are used as the input of the next sub-function.

```
% path of the output geoidkorrected tracks
path_data_goidkorr = ...
'D:\DiplArb\GPSTRACKS\TrackDataGeoidkorr\';

% path of the output geoidkorrected and split
    tracks
path_data_goidkorr_split = ...
'D:\DiplArb\GPSTRACKS\TrackDataGeoidkorrSplitt
    \';
% path of the intp.exe program (Does the
    Undulation of the Geoid)
path_program = ...
'D:\DiplArb\Karten\egm96geoid\INTP_XP\';

% path of the matched data
path_data_mapmatched = ...
'D:\DiplArb\GPSTRACKS\TrackDataMapmatched\';

%path of the merged trackparts data
path_merged_trackparts = ...
'D:\DiplArb\GPSTRACKS\
    TrackDataMapmatchedMerged\';

%path of the merged data accross roads and
    trackparts
path_merged_roadparts = ...
'D:\DiplArb\GPSTRACKS\
    TrackDataMapmatchedMergedRoads\';

% path of the merged data accross roads
% and trackparts and with the origenal data
path_merged_roadparts_with_org_data = ...
```

```
'D:\DiplArb\GPSTRACKS\
   TrackDataMapmatchedMergedRoadswithorgdata
   \';
```

After that comes the main body of the function. Six sub-functions are called in sequence.

The first function called is `korrekturgeoidundulation`(`files`, `path_inputdata`, `path_outputdata`, `path_program`). This function performs the undulation of the geoid described in section 3.2.1. The path of the program that performs the actual undulation of the geoid is passed as the last parameter. The structure of the track data is not changed during this process, only the height values are changed.

The second function is `gpssplitting`(`files`, `path_inputdata`, `path_outputdata`). This function checks the number of visible satellites and discards points that were measured with fewer than 4 satellites visible. The output of this function is the last data set before the coordinate data are changed by the mapmatching algorithm. This data will be referred to as *unmatched tracks* and will often be used to plot comparisons with the *matched tracks*.

The third function performs the actual mapmatching on a single track: `map_match_track`(`file_roads`, `file_track`, `file_outputtrack`, `d_intervall`, `d_Boundingbox`). `file_roads` refers to the *split map data*. `file_track` and `file_outputtrack` are the input- and output-data. `d_intervall` is the length in metres with which the data is interpolated. The default is 5m. `d_Boundingbox` is an offset to the bounding box comparison to allow some leeway. The default is 0. An example of the resulting matched overlapping track parts can be viewed in figure 5.3.

The fourth function `merge_mapmatched_track_trackparts`(`road_file`, `input_file`, `output_file`) merges the track parts back together. The parameters are the *split map*-, input- and output-files. Compare with figure 4.5 and formula 4.22. Figure 5.4 shows the result from applying this function to the data displayed in figure 5.3.

The fifth function `merge_mapmatched_track_roads`(`crossroads_file`, `road_file`, `track_map_matched_track_merged_file`, `trackdata_org`, `output_file`, `userinput`) merges together the track parts that are matched to different roads. It uses the *crossroads*-data passed in the `crossroads_file` and the original *map data* passed in the `road_file`, to determine if the track switched to another route at crossroads. If no crossroads are found near the transition point the user may select a customized transition if the `userinput` -flag is set. This will be described in detail later.The track data prior to the mapmatching is also passed in `trackdata_org` so it can be viewed by the user and so the user may select this data if no better option is present. This process can be viewed in the transition from figure 5.5 to figure 5.6.

The last function `insert_missing_org_track(road_file, track_map_matched_track_merged_file, trackdata_org, output_file, min_nr_points)` reinserts the track-parts that were not matched to any road. The inserted data `trackdata_org` is the *un-matched track* data. The `min_nr_points` parameter specifies the minimum length of the point sequence. This is available to prevent the insertion of very small point sequences. The result of applying the function to figure 5.7 is shown in figure 5.8. The resulting data will be called *matched tracks* for future reference.

For the mapmatching function and the three merging functions, four script-files were written to plot the intermediary result of each of these functions. The allotted script files for `map_match_track`, `merge_mapmatched_track_trackparts`, `merge_mapmatched_track_roads` and `insert_missing_org_track` are `visualize_track_mm`, `visualize_track_mm_merged`, `visualize_track_mm_merged_roads` and `visualize_track_mm_final`. These script files were used to create the figures 5.3 to 5.8

User interface for custom road selection

The following two examples will show how the user interface in MATLAB is used to select the point of transition between two roads. When the function `merge_mapmatched_track_roads` cannot determine a solution for the point of transition an interface is shown in the MATLAB command window and a plot will show the problematic area.

Figure 5.9 shows the plot of the first example. One can see two parallel roads with track parts matched to them. This case was the most common in the examined data. The output of the interface is listed below:

```
Overlapping Track Nr: 29435 to 29483

Road A:       A009
Road A Index: 456

Road B:       B067
Road B Index: 9

Please select an option:
Keep the old Data          (1)
Use the new Data           (2)
Use the track Data         (3)
Advanced options           (4)
>> 1
```

The relevant track indices are shown first. Both roads are listed by name and their index in the *map data* array. The first three options simply

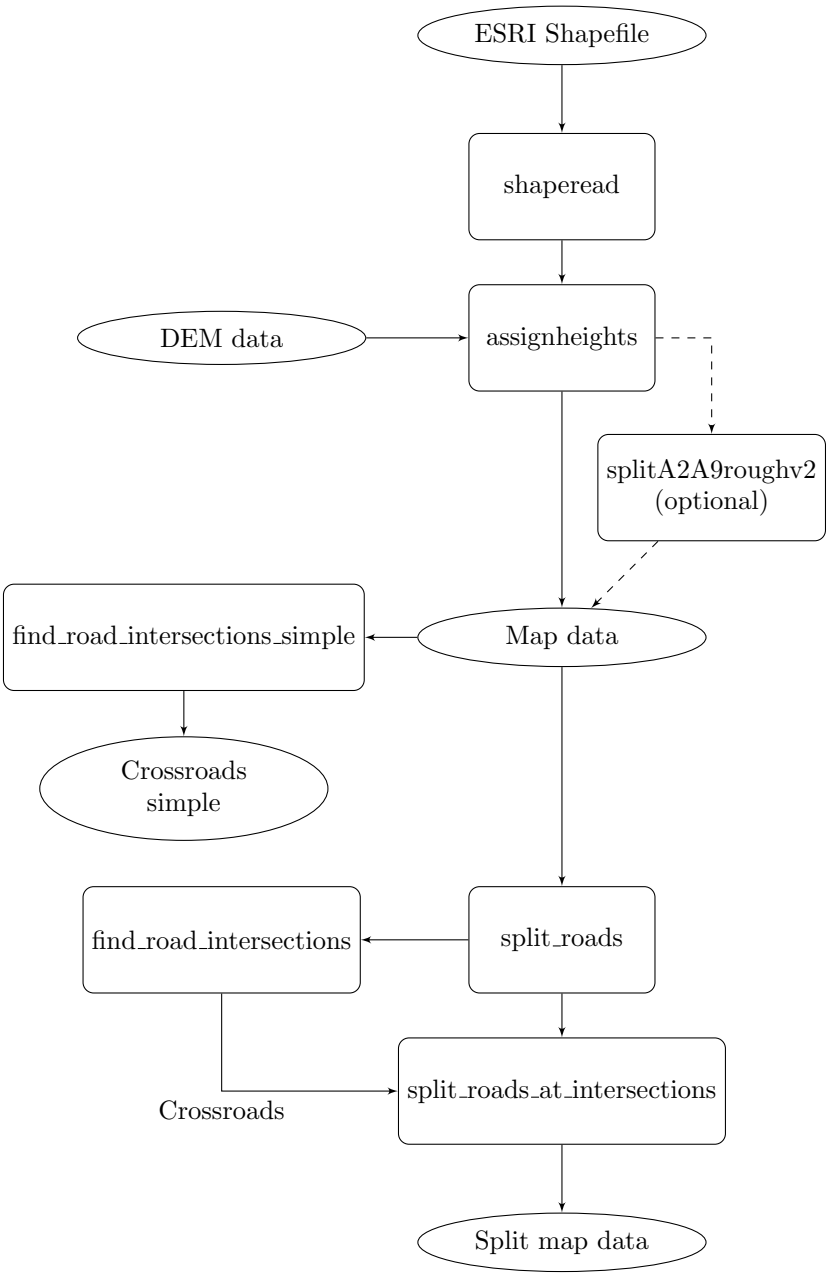

Figure 5.1: Flowchart of the road map data. This chart shows how the map data is processed before it is used by the mapmatching algorithm. The ovals depict data sources and sinks. The rectangles represent functions or scripts. The arrows show the connections between the different blocks. These connections are done by saving and loading the data to a file. Important data that is needed elsewhere is labelled at the connections.

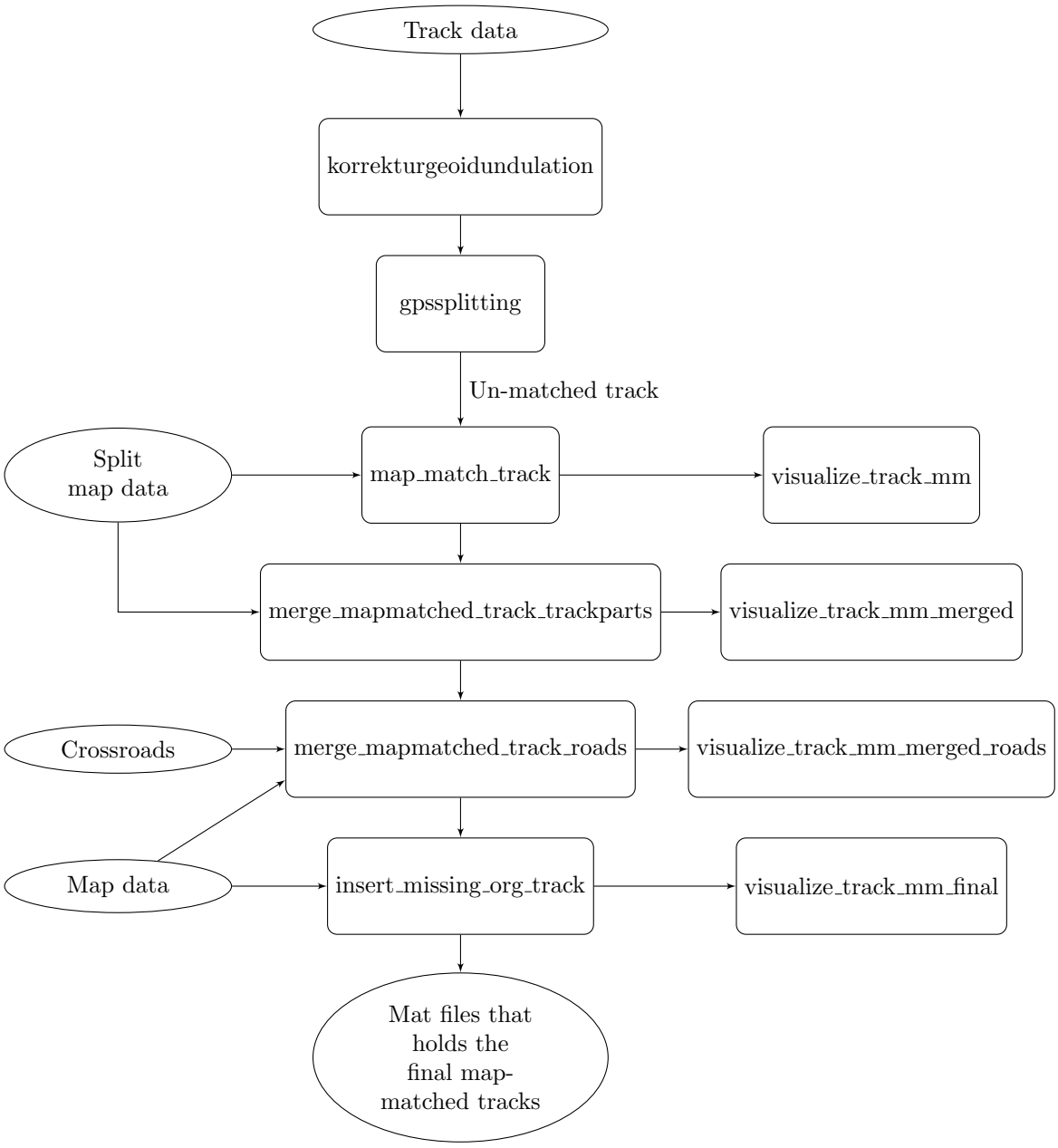

Figure 5.2: Flowchart of the track data. This chart shows the preprocessing, mapmatching and merging of the track data. The ovals depict data sources and sinks. Compare with figure 5.1 to see where these data sources were generated. The rectangles represent functions. A set of `visualize`-functions shown on the right side can be used to plot the intermediary data. The arrows show the connections between the different blocks. These connections are done by saving and loading the data to a file. Important data that is needed elsewhere is labelled at the connections.

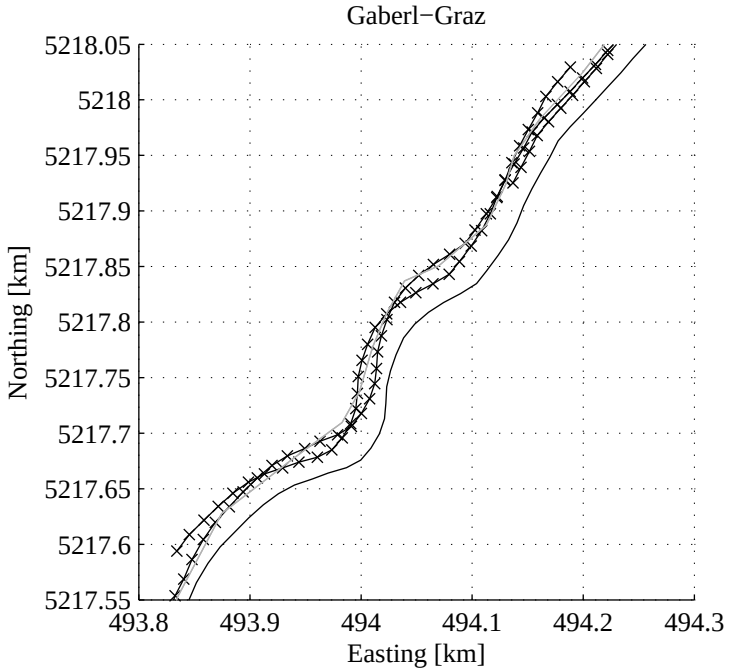

Figure 5.3: This figure shows the result from the mapmatching algorithm prior to the merging steps. Note that there are three overlapping trackparts. These are displayed with black lines with the points marked by x's. The original track is displayed as a black line. The road is displayed as a grey line. It is clearly visible that the matched trackparts are nearer to the road than the original track.

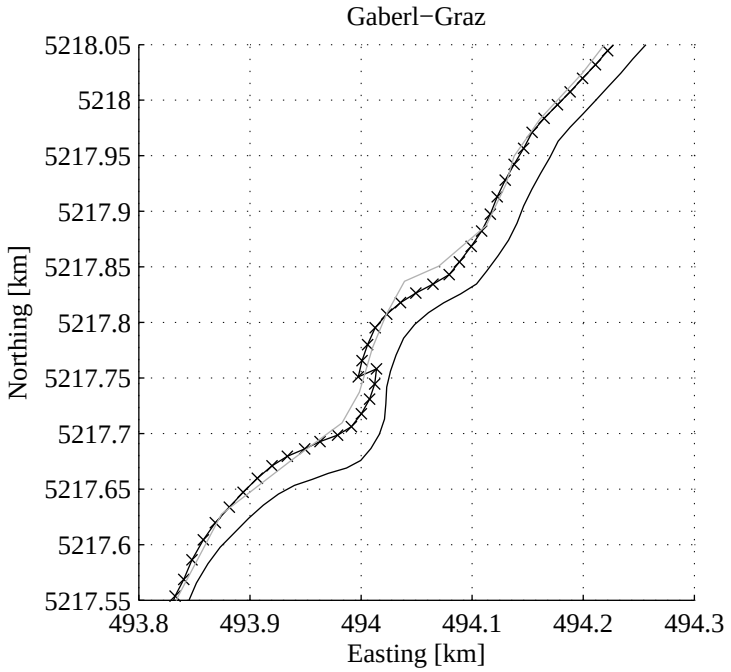

Figure 5.4: This figures shows the result from merging together the trackparts, shown in the figure above. Note that the transition point closer to the centre of the plot performs a backwards jump. See section 4.2 for the reason of this behaviour. The legend for the lines is the same as in the figure above.

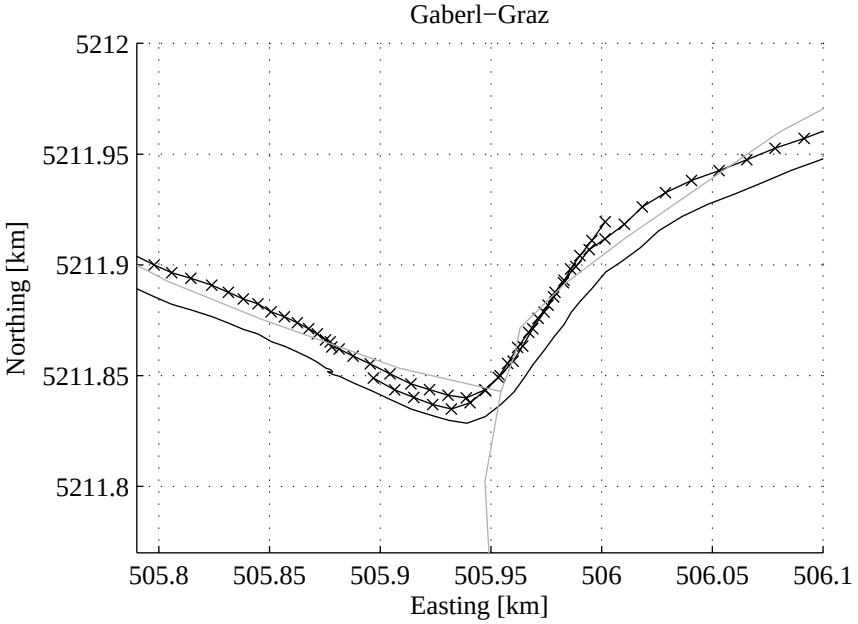

Figure 5.5: This figure shows two overlapping trackparts at a crossroads. These are displayed with black lines with the points marked by x's. The original track is displayed as a black line. The roads are displayed as grey lines.

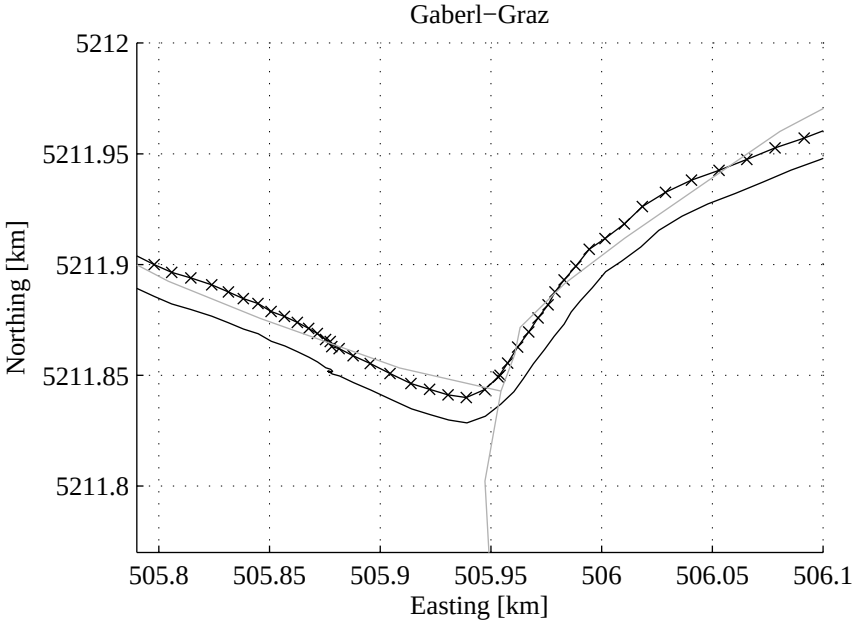

Figure 5.6: This figure shows the result of merging the two trackparts together. As expected the result is closer to the road than the original track. The legend for the lines is the same as in the figure above.

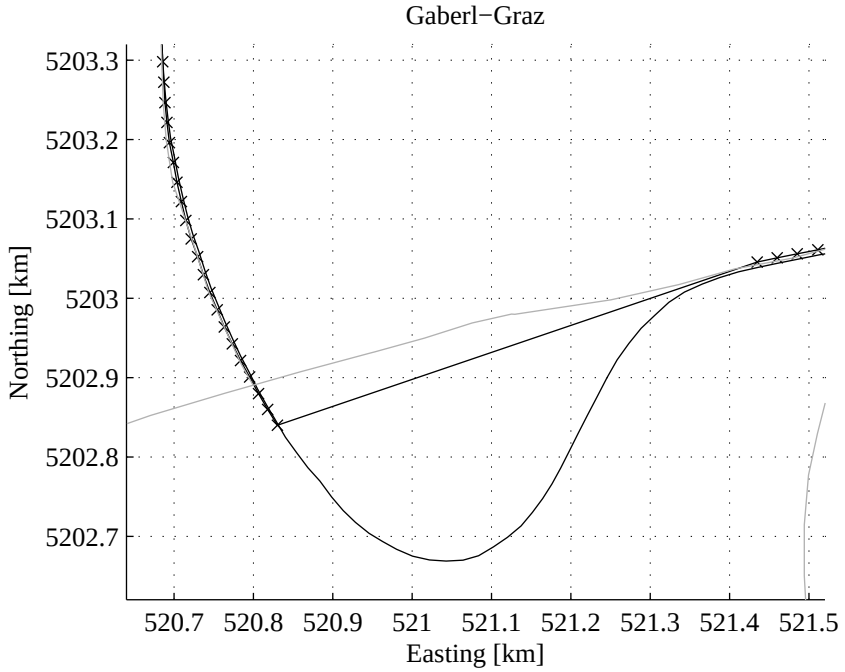

Figure 5.7: This figure shows a matched track that has lost points in the middle. This occurred because because the map data is incomplete and the middle part has no matching road. The matched track is displayed with a black line with the points marked by x's. Because the matched track is one entity, a straight line connects the separated points. The original track is displayed as a black line. The roads are displayed as grey lines.

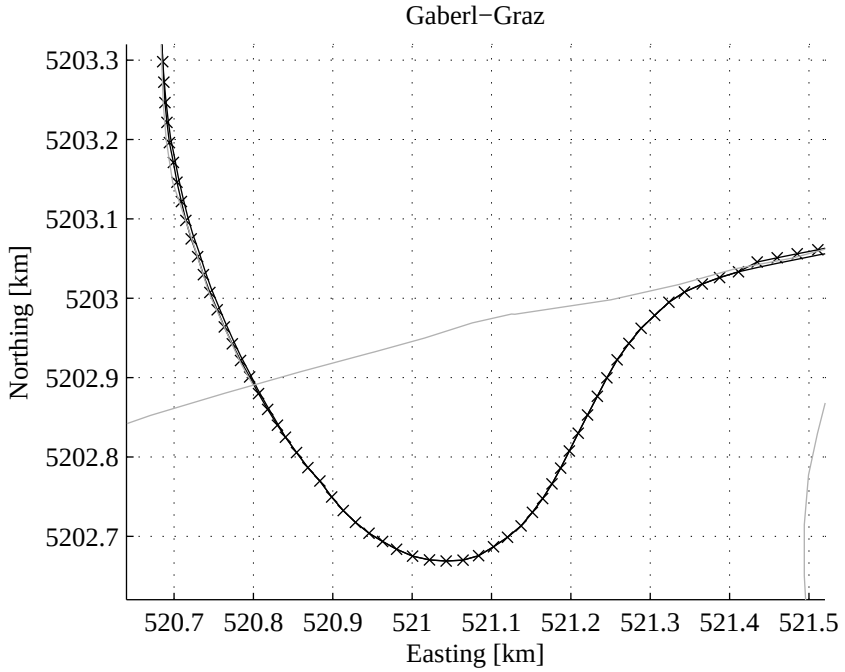

Figure 5.8: This figure shows the matched track after the missing points have been reinserted. The legend for the lines is the same as in the figure above.

choose the dataset to use. The fourth option will be discussed in the second example. The old data is matched to the motorway A009. A switch to the other road is unlikely so option (1) was chosen.

The second example is shown in figure 5.10. The track leaves the motorway A009 and switches to the road L302. Because the exit ramps of the motorways are not included in the road map data this switch cannot be matched. The manual solution chosen for this case was to switch from the track matched to the A009 to the unmatched original track data and then to the L302 using the advanced options:

```
Overlapping Track Nr: 29600 to 29639

Road A:       A009
Road A Index: 456

Road B:       L302
Road B Index: 510

Please select an option:
Keep the old Data          (1)
Use the new Data           (2)
Use the track Data         (3)
Advanced options           (4)
>> 4

Overlapping Track Nr: 29600 to 29639

Road A:       A009
Road A Index: 456

Road B:       L302
Road B Index: 510

Please select an option:
Switch from Road A to Road B           (1)
Switch from Road A to track to Road B  (2)
Back to normal Options                 (3)
>> 2
```

Option (1) is similar to option (2) but does not insert part of the original unmatched track in the middle.

```
Please enter the point to switch between Road
   A and the Track Data in %
```

```
Input may be from −100% to +100%
Any other will cancel the option
>> 10
Please enter the point to switch between the
   Track Data and Road B in %
Input may be from 0% to +200% and must be
   larger than: 10
Any other will cancel the option
>> 90
```

The actual point of switching between the different tracks is entered by a range of percentages. The overlapping segments are assigned the range from 0% to 100%. The ranges of -100% to 0% and 100% to 200% allow the original track to be inserted before and after the overlapping segment. For this example the range from 10% to 90% was chosen. This means that first 10% of the old matched track was taken. Then the range from 10% to 90% was taken from the unmatched track and the last 10% were taken from the new matched track. The result from this selection is shown in figure 5.11. Based on this plot the user can choose to confirm the selected choice or enter a different range of values.

```
Overlapping Track Nr: 29600 to 29639

Road A:       A009
Road A Index: 456

Road B:       L302
Road B Index: 510

Please select an option:
Switch from Road A to Road B           (1)
Switch from Road A to track to Road B  (2)
Back to normal Options                 (3)
Confirm selection and exit             (4)
>> 4
```

5.3 Plotting the profiles

5.3.1 Plotting a single profile

To plot the profiles the `height_profilev2`-script was used. Samples of these plots can be viewed later in chapter 6 in the figures 6.1 to 6.6. This script can

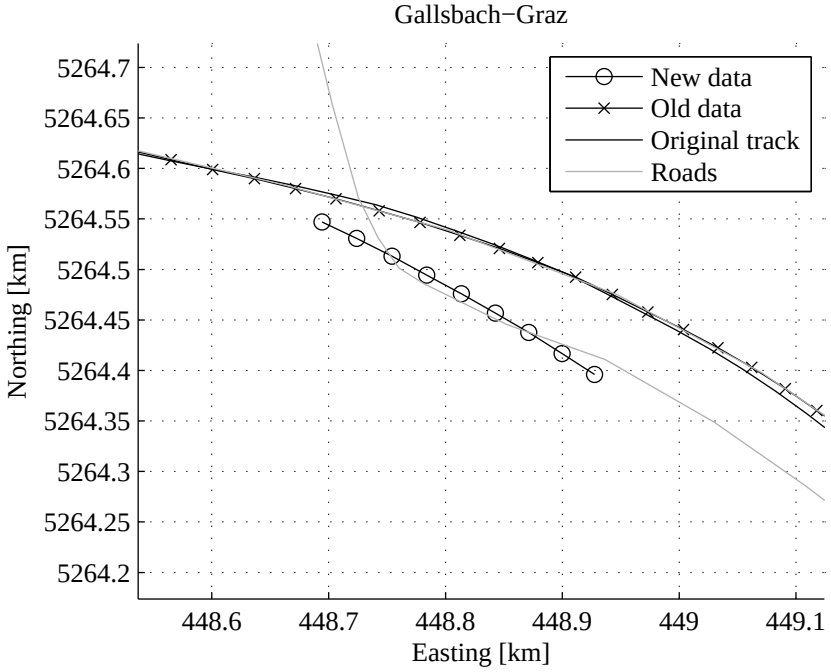

Figure 5.9: This figure shows the same track part being matched to two parallel roads. Note that there are crossroads connecting both roads. This means that the algorithm considers the possibility that the road was changed. Both options are feasible so a user input is required to resolve this situation. The x's mark the currently active track matched to the old road. The ○'s mark the possible alternative match to another road. In this case the option of keeping the old data was adopted.

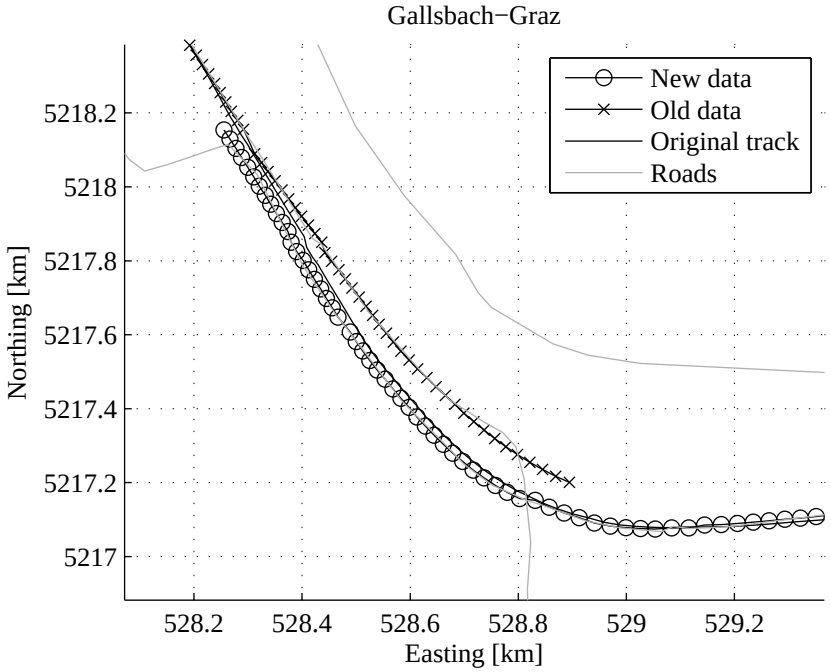

Figure 5.10: The figure shows an example of a track leaving a motorway and switching to a road that runs parallel to this motorway. Because the exit ramp is missing in the map data the algorithm cannot find the correct transition point. User input is therefore necessary to resolve this situation. The solution chosen in this case was to insert a piece of the original track between the old and new part to ensure a smooth transition. The original track is displayed as a black line. The roads are displayed as grey lines. The old trackpart is displayed as a black line with the points marked by x's. The new trackpart is displayed as a black line with the points marked by ○'s.

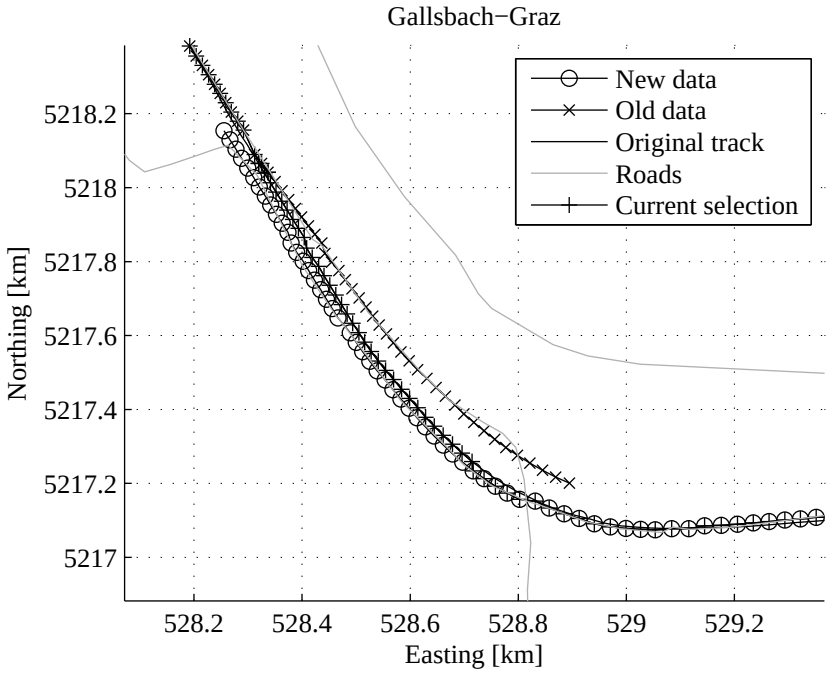

Figure 5.11: Basically this figure is the same as the one above. In addition to this the selection described above is shown as a black line with the points marked by +'s. This plot is used by the user to verify the selected input.

plot the height and speed profile for every road that the track is matched to, as well as display a contour plot of the area around the road. The different plots can be enabled or disabled by a set of flags at the start of the script. There is also an option to save the plot to a file:

```
plot_profile = 1;
plot_height_profile = 1;
plot_speed_profile = 1;

plot_heightmap = 1;
plot_delta = 1000;

save_data = 0;
pic_outputfolder_all = 'D:\DiplArb\Doku\Pics\
    Results\singleprofile\';
figure_size = [12  9];
```

`plot_delta` enables the placing of distance points highlighted by a + in the contour plot. The script requires the mapmatched track, the track data before the mapmatching was applied, the *map data* and the *crossroads simple* data. The necessary path and file-names are also set at the start of the script.

```
file_map = 'Strassen_Height_inter_A2_A9_split.
    mat';
file_xroads = 'crossroads_simplemitsplitA2_A9.
    mat';
path_map = 'D:\DiplArb\Karten\GIS STMK\
    StrassenUTM33N\';

path_org_data = 'D:\DiplArb\GPSTRACKS\
    TrackDataGeoidkorrSplitt\';
path_mapmatched = 'D:\DiplArb\GPSTRACKS\
    TrackDataMapmatchedMergedRoadswithorgdata
    \';
```

5.3.2 Plotting overlaid profiles

While the plotting of a single profile is relatively simple, plotting several profiles in the same figure is a more complicated matter. It is necessary to choose the road sections for which the profiles are to be plotted. Then the actual tracks on theses sections need to be selected before plotting them. The overview of this process can be viewed in the flowchart in figure 5.12.

The selection of the road section is done in the `visualize_overlaid-tracks`-script. This script requires the *map data*, the *crossroads simple* data and the *matched tracks*. The location of these data files is set up at the beginning of this script:

```
file_map = 'Strassen_Height_inter_A2_A9_split.
    mat';
path_map = 'D:\DiplArb\Karten\GIS STMK\
    StrassenUTM33N\';

file_xroads = 'crossroads_simplemitsplitA2_A9.
    mat';

path_mapmatched = 'D:\DiplArb\GPSTRACKS\
    TrackDataMapmatchedMergedRoadswithorgdata
    \';

path_xroad_list = 'D:\DiplArb\GPSTRACKS\
    crossroads_lists\';

liste = dir(fullfile(path_mapmatched,'*.mat'))
    ;
files = {liste.name};
```

The *matched tracks* are selected with the help of the `files`-variable and by specifying the different paths for the datasets. If only certain tracks are needed for examination[3], the `files`-array must only contain the names of these tracks.

On execution the script will display a plot showing the density of tracks on the road network. See figure 5.13 for an example. This plot is then used to select the road section. The road section is defined by crossroads at its beginning and end. These are selected by right-clicking on them. The selection must be done in sequence. To make the selection easier it is possible to zoom in with a left-click and to zoom out again by a double left-click. The selection process is ended by pressing the ENTER key. Then an user interface will be displayed in the MATLAB Command window. This allows verification of the input and adjustment to it:

```
Selected Crossroads:
1.) Crossroad Index: 160, Road A: B068, Road B
    : A002
2.) Crossroad Index: 1133, Road A: A002, Road
    B: A002Z
```

[3]For example only tracks going in one direction.

```
3.) Crossroad Index: 279, Road A: B073, Road B
   : B067AD

Please select an option:
Insert new Crossroad       (1)
Find Crossroad             (2)
Delete Crossroad           (3)
Display Crosssroad details (4)
Check data                 (5)
Exit                       (6)
```

The first two crossroads share the same road so that part of the selection is correct. The crossroads (2) and (3) do not share the same road so there is no direct connection between them. That part of the selection is therefore incomplete. This can be confirmed by selecting the `Check data` (5) option.

```
Link: 12 is valid.
Link: 23 is invalid.
Check failed!
```

Choosing the first option (1) will allow the user to select one more crossroads from the map and enter it at a specific location in the list.

```
>> 1

Selected Crossroad:
Crossroad Index: 59, Road A: B065, Road B:
   A002
Please select the index after which the
   Crossroad will be inserted: 0 - 3
(0 to insert at the start, any other key to
   cancel
>> 0

Selected Crossroads:
1.) Crossroad Index: 59, Road A: B065, Road B:
    A002
2.) Crossroad Index: 160, Road A: B068, Road B
   : A002
3.) Crossroad Index: 1133, Road A: A002, Road
   B: A002Z
4.) Crossroad Index: 279, Road A: B073, Road B
   : B067AD
```

```
Please select an option:
Insert new Crossroad        (1)
Find Crossroad              (2)
Delete Crossroad            (3)
Display Crosssroad details  (4)
Check data                  (5)
Exit                        (6)
```

This however requires the knowledge of which crossroads were missed in the first place and is therefore not always applicable. It is possible to search for crossroads between two crossroads using the `Find Crossroad` (2) option. In the example shown the missing crossroads is the one where the A002 and the B073 meet.

```
>>2

Please select the 1.st Crosstroad: 1 - 3
(the next Crossroad in the list will be used
   as the 2nd Crossroad.)
>>3

The folowing Crossroads where found:
1.) Crossroad Index: 276, Road A: B073, Road B
   : A002, X = 535976.5436, Y = 5207036.1305
2.) Crossroad Index: 1172, Road A: B067AD,
   Road B: A002Z, X = 536293.0762, Y =
   5208972.9831

Select the Crossroad you want to insert: 1 - 2
>>1


Selected Crossroads:
1.) Crossroad Index: 59, Road A: B065, Road B:
    A002
2.) Crossroad Index: 160, Road A: B068, Road B
   : A002
3.) Crossroad Index: 1133, Road A: A002, Road
   B: A002Z
4.) Crossroad Index: 276, Road A: B073, Road B
   : A002
5.) Crossroad Index: 279, Road A: B073, Road B
   : B067AD
```

```
Please select an option:
Insert new Crossroad        (1)
Find Crossroad              (2)
Delete Crossroad            (3)
Display Crossroad details   (4)
Check data                  (5)
Exit                        (6)
```

The data sequence is now valid. There are two more options. `Delete Crossroad` (3) allows the deletion of one crossroads and `Display Crossroad details` (4) shows all the data available of a particular crossroads. To save the sequence the `Check data` (5) option must be selected and return a positive result.

```
>>5

Link: 12 is valid.
Link: 23 is valid.
Link: 34 is valid.
Link: 45 is valid.
Check successful

Save                        (1)
Save and Exit               (2)
Continue                    (3)
>>2
Path: D:\DiplArb\GPSTRACKS\crossroads_lists\
Enter file name: >>A002toB073
File saved
```

The `path` is specified at the start of the script file. The file just generated is passed on to `crossroads_list_process`(`crossroads`, `xlist`, `outputfile`)-function as the `xlist` parameter. The `crossroads` parameter is the *crossroads simple* data. This function inserts all the crossroads through which the chosen route passes. These are used to synchronise the tracks and to check if tracks join or leave the route. This is done in the `calculate_overlaid_height_profile`-script. It makes use of the data just generated as well as the *map data* and the *crossroads simple* data to do so.

The actual plotting of this data is done in the `plot_overlaid_height_profilev3`-script. In addition this script needs *map data* and the *crossroads simple* data. Various examples of the generated height and speed profiles are shown in chapter 6.

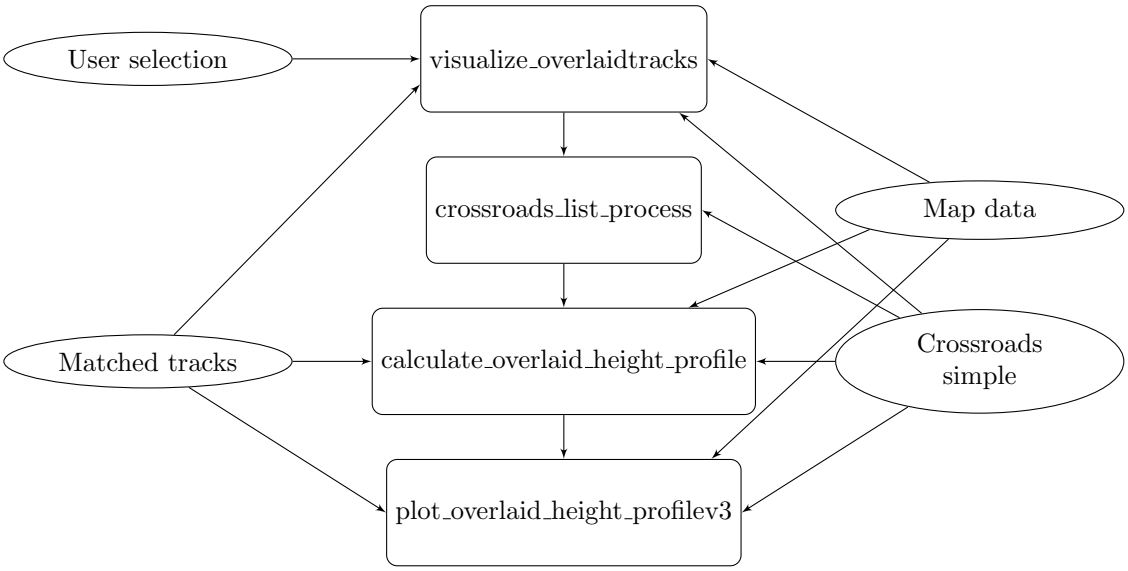

Figure 5.12: Flowchart of the profile data. This chart shows necessary steps that the profiles need to go through, to create the overlaid profiles plots. The ovals depict data sources. The rectangles represent functions or scripts. The arrows show the connections between the different blocks. These connections are done by saving and loading the data to a file.

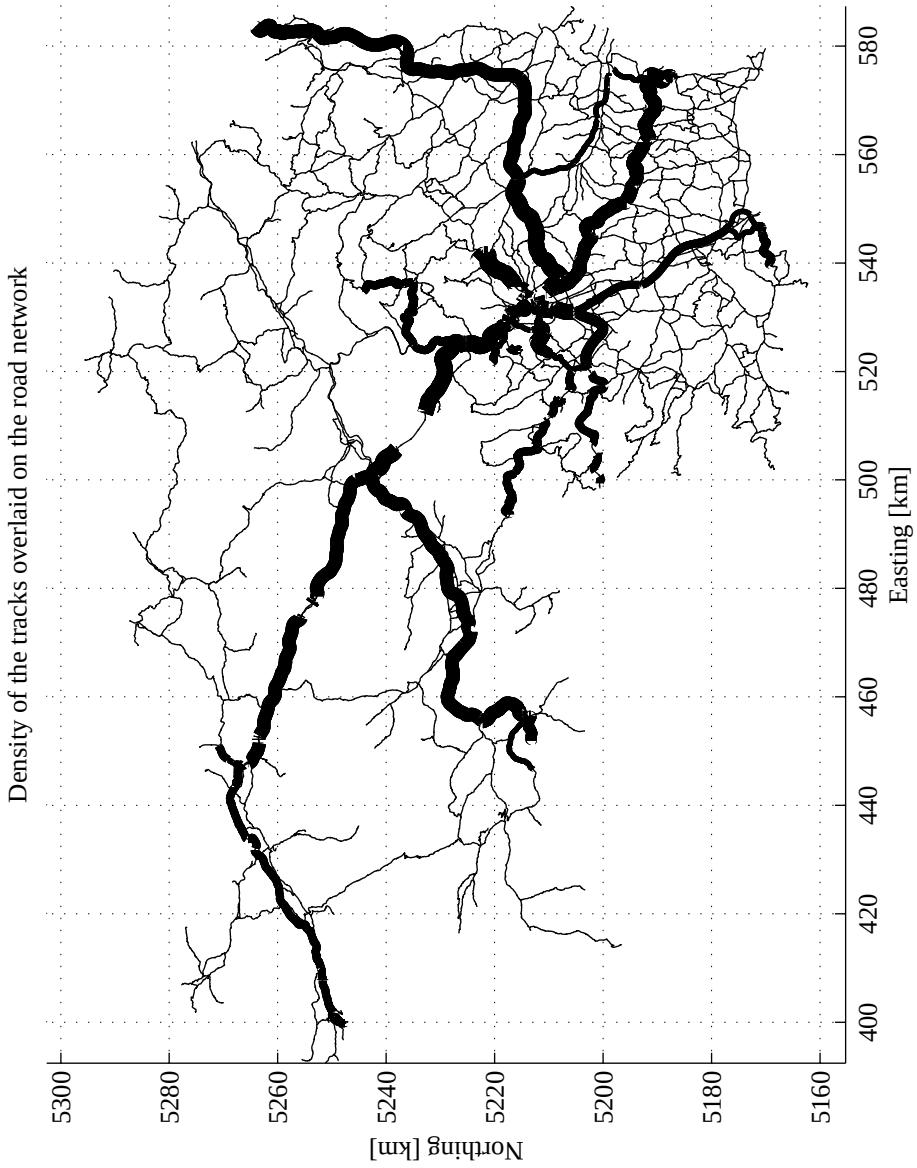

Figure 5.13: Density of the tracks. The tracks are plotted on the road network. The roads are displayed as thin lines. The tracks are shown as thick lines. The thickness of the line indicates the number of tracks on that route. This plot is used to select a route and to plot the overlaid profiles for that route.

Chapter 6

Results

In the last chapter the tracks were brought into relation with the road travelled on. In this chapter the height profile from individual tracks will be compared to the profile provided by the map. It will continue to analyse multiple tracks on the same road. This will be done by comparing their height profiles with each other and the road profile. It will also look into speed profiles, comparing multiple profiles on the same route and trying to establish relationships between them and features of the map.

6.1 Comparison of track and road height profile

The following figures show samples of track and height profiles. The samples were chosen to represent different behaviours that were observed across the resulting profiles. To compare the two profiles the difference between them is calculated and plotted. As the curves do not share the same points along the distance it is necessary to interpolate both curves equidistantly. This deviation along the distance is an indicator of how well the GPS- and map height data match each other. For easy comparison across the datasets the mean and standard deviation of this new curve are also calculated and displayed. These mean values and the standard deviation values range from 0 to $\pm$ 20 m for almost all tracks. To gain better insight into this range of values a contour plot of the surrounding area is also appended.

The profiles displayed in figure 6.1 match each other fairly well. The contour plot 6.2 shows that the road passes through a relatively flat area.

In figure 6.3 the track profile is below the road profile for almost the whole distance. The corresponding contour plot 6.4 shows that the road runs along steep slopes for most of the course. This means that small errors in the located position result in relatively high errors in the associated height value. One possible error source is due to the interpolation of the grid elevation. Roads will cut into hill sides and use bridges to span valleys. The bilinear interpolation smooths such areas.

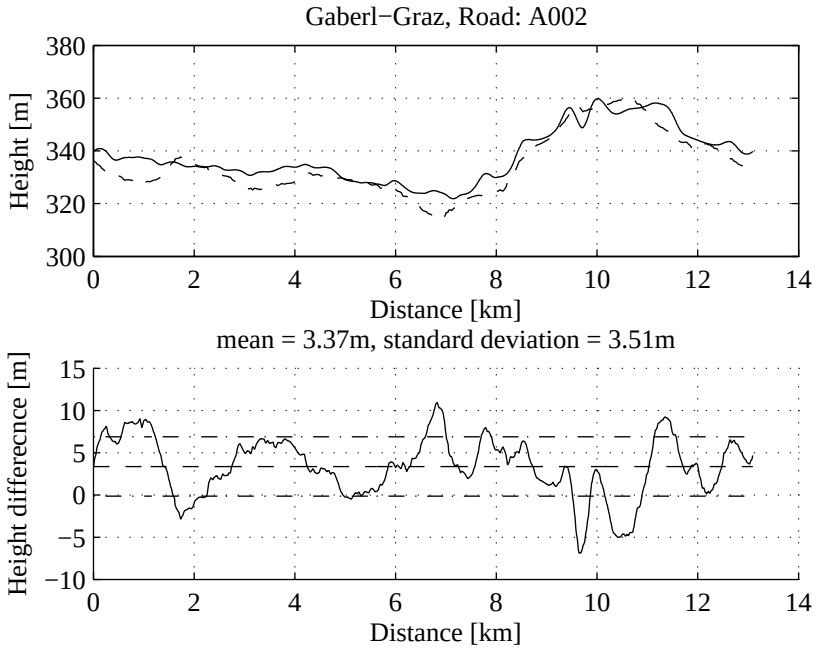

Figure 6.1: Sample height profile. The top graph shows the profile of the road (solid line) and of the GPS track (dashed line). It shows that the road and track profile follow each other well. This is confirmed by the graph beneath that shows the difference between those two profiles (solid line) and the mean and standard deviation values.

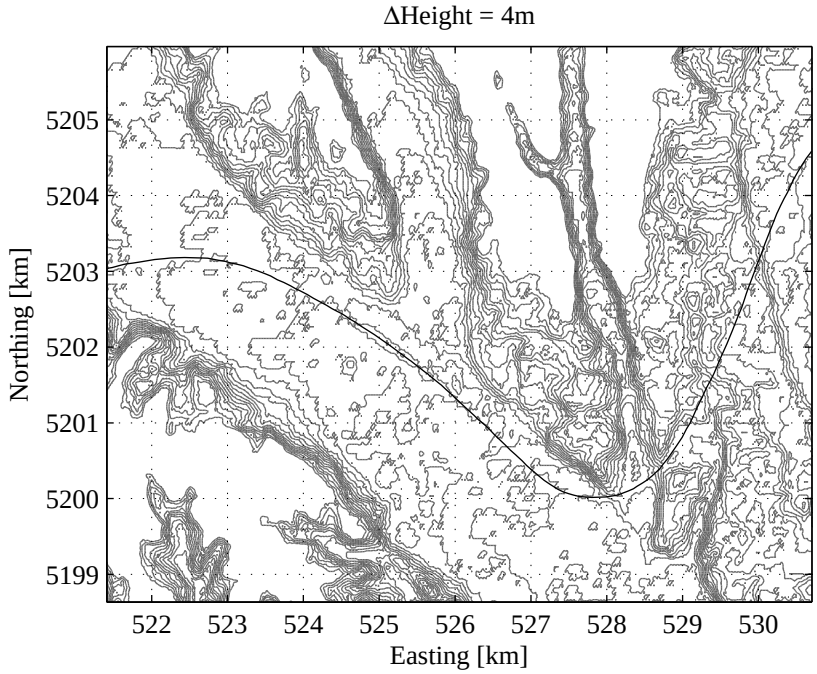

Figure 6.2: Contour plot displaying the height level for figure 6.1. The black line represents the road course. This plot shows that the profile in the figure above passes through a relatively flat area.

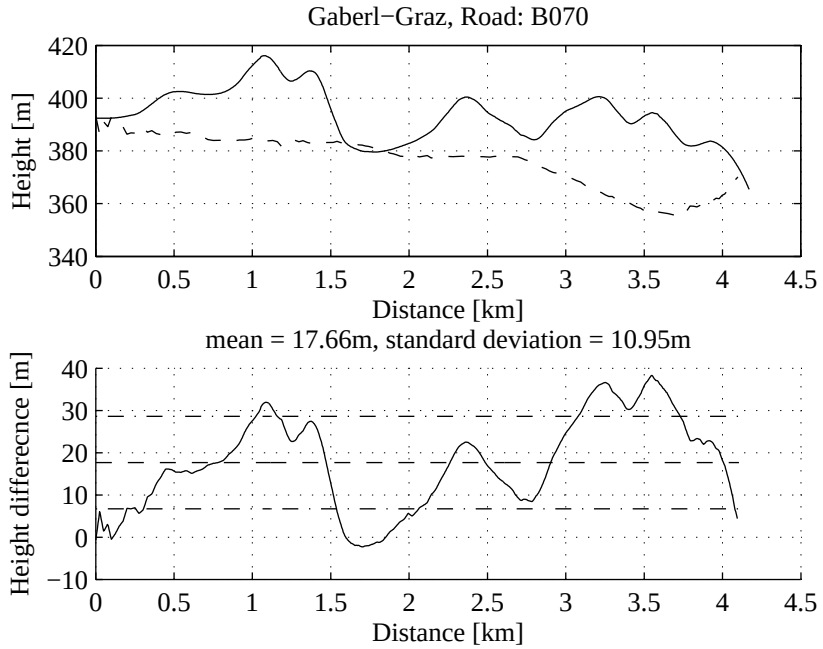

Figure 6.3: Sample height profile. The top graph shows the profile of the road (solid line) and of the GPS track (dashed line). Theses two profiles are not similar at all. The graph beneath shows the difference between those two profiles (solid line) and the mean and standard deviation values. This second graph shows that there are differences between to two curves almost up to 40 m.

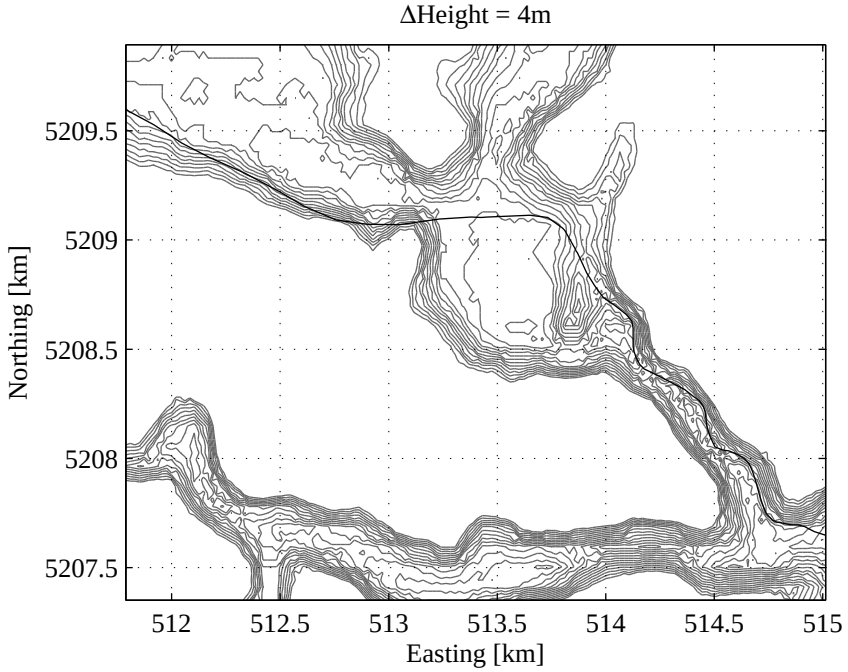

Figure 6.4: Contour plot displaying the height level for figure 6.3. The black line represents the road course. This plot shows that the road mostly runs parallel to the height-lines. This is possibly the reason for the great differences shown in the figure above.

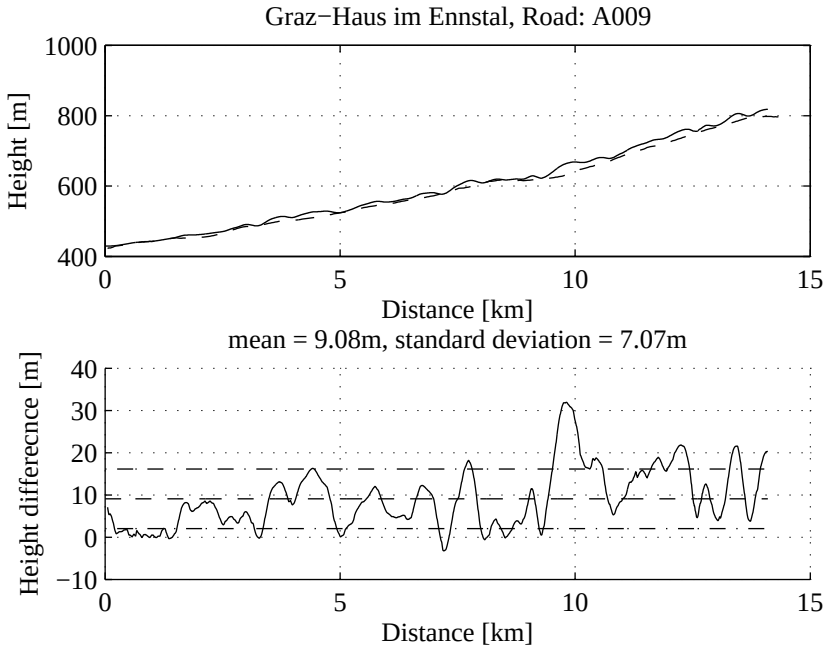

Figure 6.5: Sample height profile. The top graph shows the profile of the road (solid line) and of the GPS track (dashed line). The two curves follow each other fairly well. The graph beneath shows the difference between those two profiles (solid line) and the mean and standard deviation values. This makes its clear that there are several difference peaks. Note the particular large one at kilometre 10.

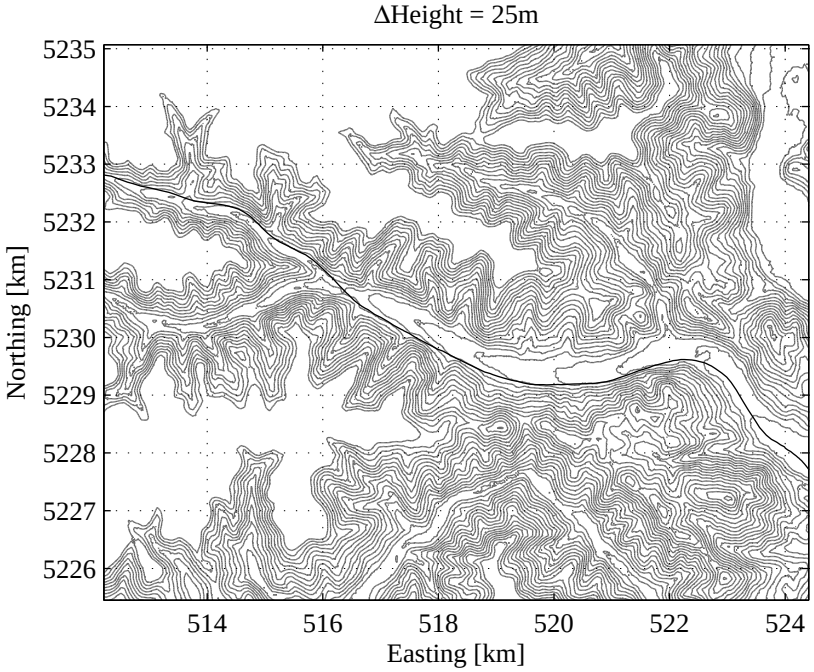

Figure 6.6: Contour plot displaying the height level for figure 6.5. The black line represents the road course. The track goes from east to west. Note that the valley the road goes up and becomes narrower the higher the road climbs.

Figure 6.5 is an example of a decent match of a road and track profile. There are however some peaks visible in the graph showing the difference between track and road. The largest peak is most notable at kilometre 10. The area surrounding the road is steep as shown in figure 6.6. The track goes from east to west. As the road gradually climbs upward, the valley gets narrower. This means that the track is subject to the same problems as the one shown in figure 6.3 the higher it goes.

6.2 Analysing multiple tracks on a road

The discussion of the single tracks has brought up some varying results. Because only single tracks were used, the analysis was done on only a single sample. This means the measurement uncertainty is completely unknown.

To remedy this issue, multiple tracks on the same route will be analysed next. To do this the profiles need to be synchronized to one another and to the road, so that the same road segment is used for all shown profiles. However the tracks do not cover exactly the same distance. For example curves may be taken at different radii or the lanes may have been switched. This means different tracks will accumulate differences over the distance covered. It was therefore necessary to reset the distance of all tracks after some time. The crossroads were chosen for these reset points as they were easily viewable on the map.

Analysing height profiles

The road segment that is going to be looked at in detail is part of the B072. This road forms a significant part of a commuting route between Graz and Kumberg. Figure 6.7 shows the contour plot of this road section. The length of the segment is over 11 km and the altitude difference around 150 m. The road-section starts in the city of Graz on the southern edge of the plot. The suburb of Mariatrost[1] is passed approximately 3.5 km later. The road then crosses a hill in a relatively open area.

In figure 6.8 all 14 track profiles along this road segment are shown. The tracks cover both directions of the road segment. For easy comparison figure 6.8 and the following figures 6.9, 6.10, 6.11 use the same starting- and end-point, regardless in which direction they were driven. This means that the tracks from Graz to Kumberg start on the left of the plot at zero and end at the right end of the plot. The tracks from Kumberg to Graz start at the right end of the plot and end at zero distance on the left.

Figure 6.8a shows all track profiles compared to the profile of the road. The tracks are clustered together in a band. It is also noticeable that one track has far too low altitude values at the starting point. The start value

[1] Easting $\sim$ 537.5 km, Northing $\sim$ 5217.5 km

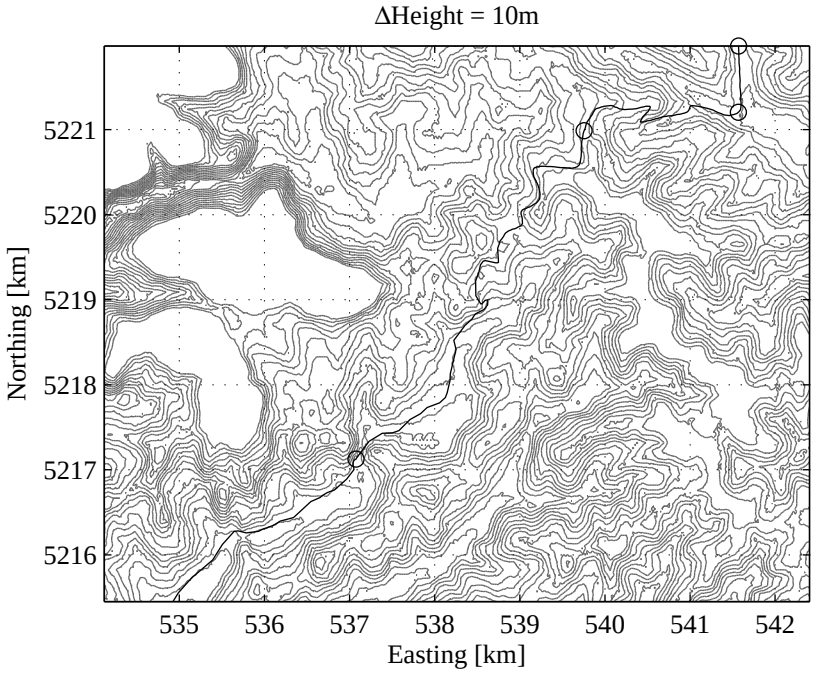

Figure 6.7: Contour plot for multiple track analysis. The black line represents the road course. The crossroads, that the road passes, are marked with a circle ○. The city of Graz is located at the southern end of the road.

has been cut off in the figure as it was over a 100 m below the other tracks. This is possibly due to not giving the GPS-receiver sufficient time to settle in. Figure 6.8b shows the average height profile of the tracks again compared to the road profile. The average height profile was gained by calculating the mean of the height values at each distance point. The average height profile is mostly below the road profile. This can be seen clearly when looking at figure 6.8c which shows the difference between the road and the average height profile. One can also see that after a distance of 4 km the road- and average height profile match each other better. Kilometre 4 is approximately the point where the road leaves the suburban area. As discussed in 3.1.2 the DEM is less reliable in such areas and probably the reason for this discrepancy.

In the preceding discussion it is assumed that the direction of the track is irrelevant. To verify this assumption the tracks were separated based on their direction. Figure 6.8 was then redrawn using only the tracks from Graz to Kumberg 6.9 and vice versa 6.10.

There are 6 tracks from Graz to Kumberg which are shown in figure 6.9a. The track that starts at too low an altitude is present in this dataset. In addition another track jumps soon after the starting point. This can also be seen in the plot with all the tracks 6.8a but is more visible here where there are fewer tracks. The most likely reason for this is a change in the visible satellite constellation. These two problems combined with the relatively low number of total tracks lead to a high mismatch between the average track profile and the road profile until the first kilometre. This is clearly visible in figure 6.9c. Otherwise the result matches the result with all the tracks fairly well.

Figure 6.10a shows the result with the 8 tracks from Kumberg to Graz. Those tracks do not include the two tracks with problematic data after the point kilometre 0. The computed curves 6.10b,c at this point are therefore better than the curves 6.8b,c that include all tracks. Again the result matches the result with all the tracks fairly well.

Based on these three plots it looks like the height profile is indeed independent of the direction it was driven. However the following discussion on the measurement uncertainty contradicts this.

In the last three plots the relation between the digital map data and the GPS-track data was discussed. Figure 6.11 is used to look at the Measurement uncertainty of the GPS-tracks. One can think of the GPS-tracks as a height measurement for each point on the road. Subsequently the mean value[2], standard deviation[3] and measurement uncertainty can be calculated

[2]The resulting mean height profiles have been already been shown in the figures 6.8, 6.9 and 6.10

[3]As there is only a limited number of measurments there is only an estimate of the standard deviation. To simplify the text will refer to the estimate as standard deviation.

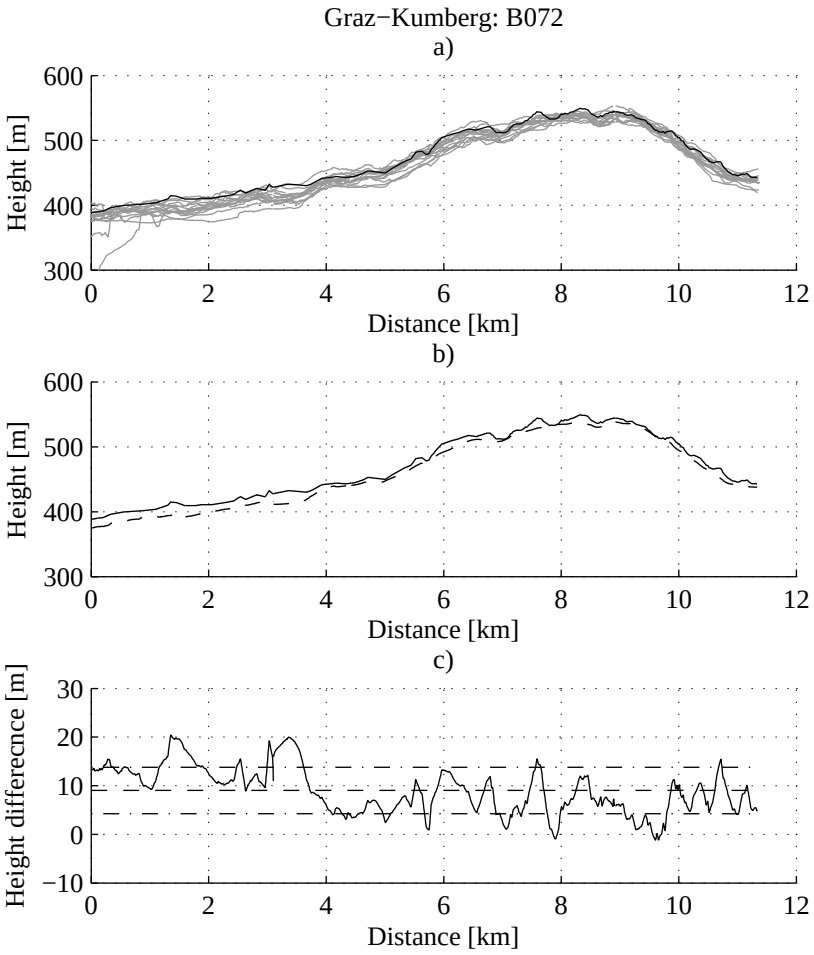

Figure 6.8: Overlaid bidirectional height profiles. a) The profile of the road (black). All tracks along this road (grey). It is clearly visible that all track follow the road profile closely. The track profiles are mostly lower than the road profile. b) The profile of the road (black). The mean of the tracks across the height values (dashed). c) The difference between the road and the mean track profile.

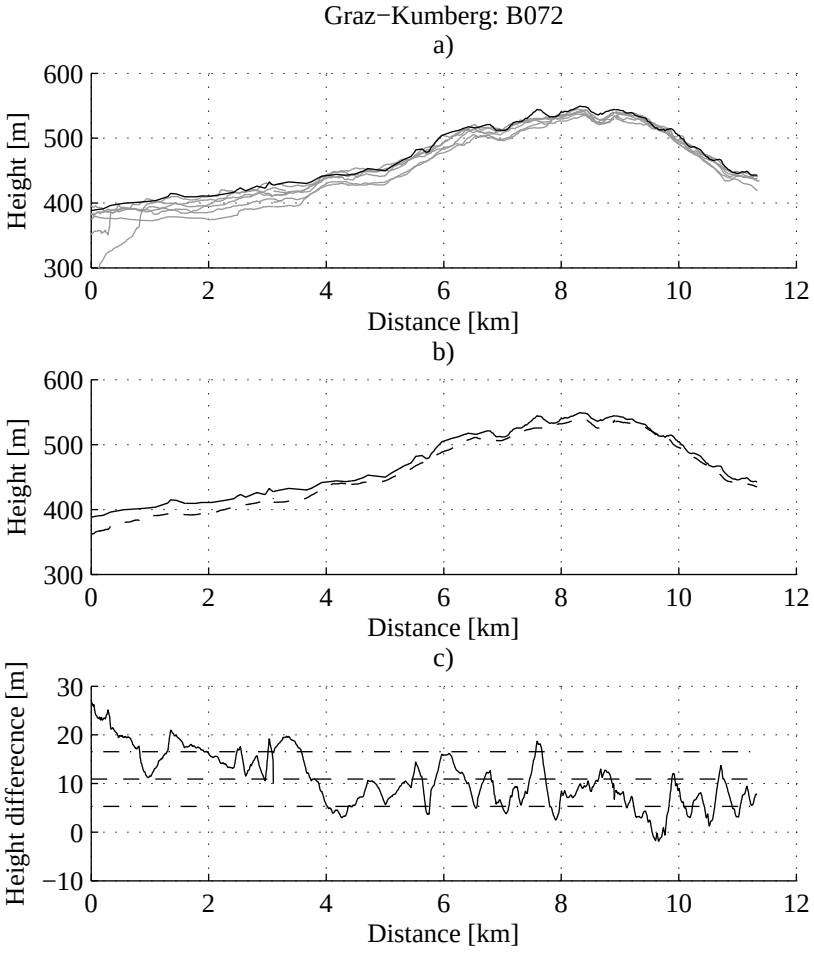

Figure 6.9: This figure shows the same information as figure 6.8. The only difference is that it only shows the profiles from Graz to Kumberg. This is a subset of the profiles shown in figure 6.8. The aim of this plot was to verify that the height profile is independent of the direction it was driven. Based on this plot and figure 6.10 it looks like the height profile is indeed independent from the driving direction. a) The profile of the road (black). All tracks along this road (grey). b) The profile of the road (black). The mean of the tracks across the height values (dashed). c) The difference between the road and the mean track profile.

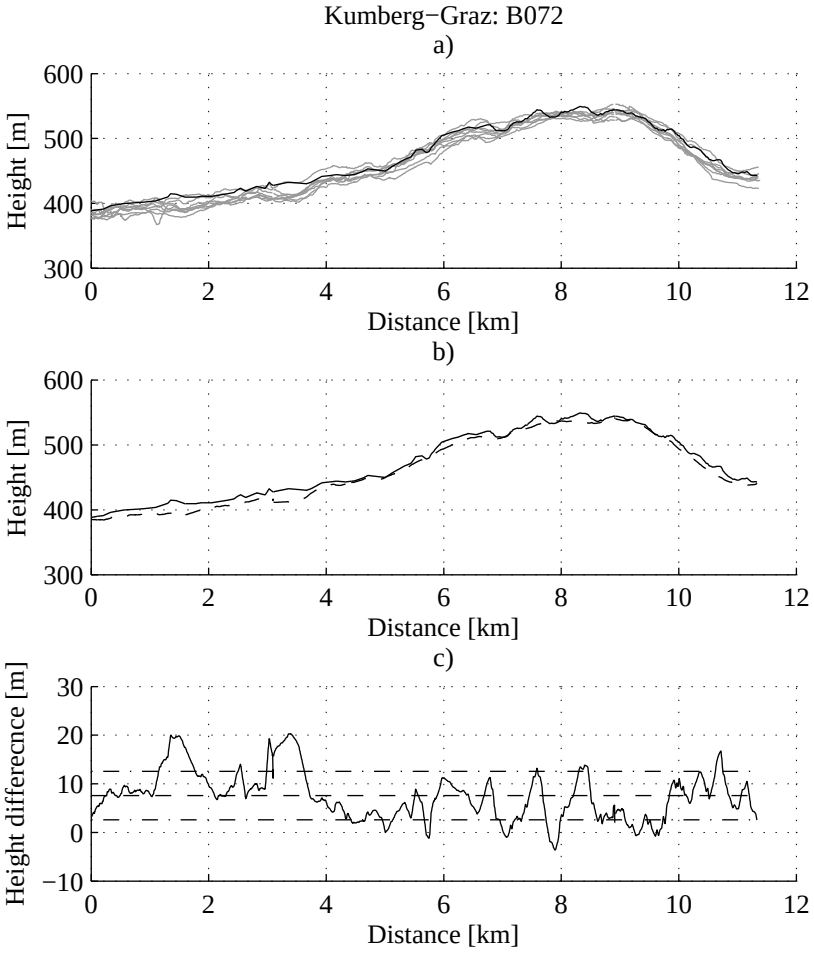

Figure 6.10: This is the counterpart to figure 6.9 showing only the profiles from Kumberg to Graz. a) The profile of the road (black). All tracks along this road (grey). b) The profile of the road (black). The mean of the tracks across the height values (dashed). c) The difference between the road and the mean track profile.

for each of these points. The difference between the mean height profile and the individual profiles of the tracks are displayed in grey. This shows how much the individual tracks diverge from their mean profile and from each other. The resulting curves are again calculated and displayed for all tracks in figure 6.11a and for tracks only going in the same direction in the figures 6.11b,c. The two black curves show two times the standard deviation (2σ) of the grey curves. This means that 95% of all height profiles lie within the the band spanned by the two black curves. In general most of the tracks diverge from the mean profile within a band of ±20 m band in all three plots. The random measurement uncertainty u_r can be described by the following formula:

$$u_r = \frac{t}{\sqrt{n}}\sigma \tag{6.1}$$

With n representing the number of measurements and t describing a confidence factor that is derived from the *Student's t-distribution*, t depends on the number of measurements n and a confidence level $1 - \alpha$. For the aforementioned band of ±20 m and the corresponding standard deviation $\sigma = 10$ the random measurement uncertainty is shown in table 6.1. This reasoning can also be applied to the speed profiles shown later on. The different standard deviation must be applied to recalculate the measurement uncertainty for them. See the German standard DIN 1309 [31] for more information.

Table 6.1: The random measurement uncertainty of the GPS-height with $\sigma = 10$

n	t	$\frac{t}{\sqrt{n}}$	u_r	t	$\frac{t}{\sqrt{n}}$	u_r
	$1-\alpha = 95\%$			$1-\alpha = 99\%$		
6	2.57	1.05	10.5	4.03	1.65	16.5
8	2.37	0.84	8.4	3.50	1.24	12.3
14	2.16	0.57	5.7	3.01	0.80	8.0

A particularly interesting point in these plots is around kilometre 8. At this point the band for the bidirectional plot narrows down to ±10 m and in the directional plots the band narrows down even more. Looking back to the height profiles shown in the figures 6.8, 6.9 and 6.10 one can see that this point is at the top of the hill. The difference between the mean height profile and the road profile is also very low at this point. No satisfactory explanation was found for this phenomena.

Another interesting observation is that the band spanned by the curves from Graz to Kumberg (figure 6.11b) narrows down to ±10 m for the distance from kilometre 8 to the end of the track. This cannot be observed for the tracks going in the other direction (figure 6.11c). This observation

suggests that the measurement uncertainty of a track depends on the direction in which the road is driven. Looking at the height profile one can see that this track segment consists of driving across the top and then down the slope of the hill. A similar downhill segment for the track in the opposite direction is between kilometre 8 and 4. However comparing these two downhill segments with each other shows no similarities. One can therefore *not* correlate this behaviour to the slope of the road. A possible explanation for this behaviour is that the GPS-receiver was positioned near the front screen of the vehicle when the track was collected. The vehicle therefore diminishes the GPS-signal by a different amount depending on the orientation of the vehicle. Signals travelling through the front window are received strongest. This theory suggests that a vehicle with a GPS-receiver positioned on its roof would not encounter this effect.

Analysing speed profiles

After analysing height profiles this text will continue on to the speed profiles on the same route. The speed of the vehicle is influenced by several surrounding conditions. Some of these influences depend on the driving direction, while others do not. For example crossroads that are approached from a different direction may be subject to a different right of way. Another example is a road that has a different number of lanes depending on its direction. On the other hand the same speed limit is usually applied to both directions of the same road segment. Also speed limitations due to the curvature of the road apply to both directions.

One of the important factors influencing driving speed is is the speed limit of the road. The default speed limits in Austria are: 50 km/h ($\sim$14 m/s) for built-up areas, 100 km/h ($\sim$28 m/s) for rural areas and 130 km/h ($\sim$36 m/s) for motorways. For approximately the first 5 km, the route studied passes through Graz and Mariatrost and the 50 km/h limit applies. There are also two smaller built-up-areas along the route. Nadischhöhe lies around the 7th kilometre of the route and Fasslberg approximately between the 8th and 9th kilometre. For the rest of the route the default speed limit of 100 km/h applies. Vehicle speed is also restricted by a number of sharp bends.

Figure 6.12 shows all the speed profiles on the route (a) and the mean over these profiles (b). The crossroads have been marked with a circle $\bigcirc$ on the plot (b). For the first kilometre of the route one can observe frequent accelerations and decelerations. Some tracks show the decelerating of the vehicle down to standstill. This section is right in the city of Graz and indicates heavy, stop-and-go traffic conditions. In the rest of the built-up-areas the tracks are within a relatively small band around the speed limit. This can be viewed in more detail in figure 6.15 which shows the difference between the mean speed profile and the individual speed profiles

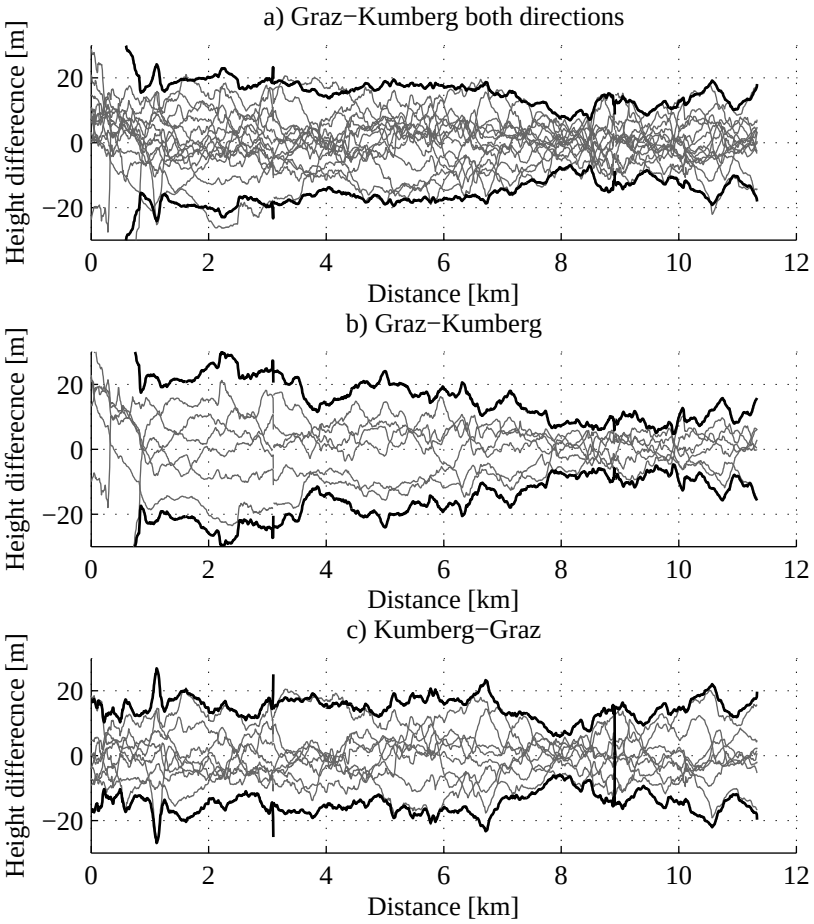

Figure 6.11: Grey: Differences between single tracks and their mean curve. Black: Two times the standard deviation (2σ) of the difference curves. This means 95% of all the track differences lie between those black curves. This figure visualizes the measurement uncertainty of the height profiles. The difference between the three graphs is their driving direction. a) Difference for all tracks. b) Difference for the tracks from Graz to Kumberg. c) Difference for the tracks from Kumberg to Graz. An interesting point lies at kilometre 8. Here the 2σ-band is lowered to ± 10 m for all three graphs. It is also interesting that from kilometre 8 to the end of the track the band in graph (b) is considerable narrower then the one in graph (c). This implies that the measurement uncertainty is dependent on the driving direction.

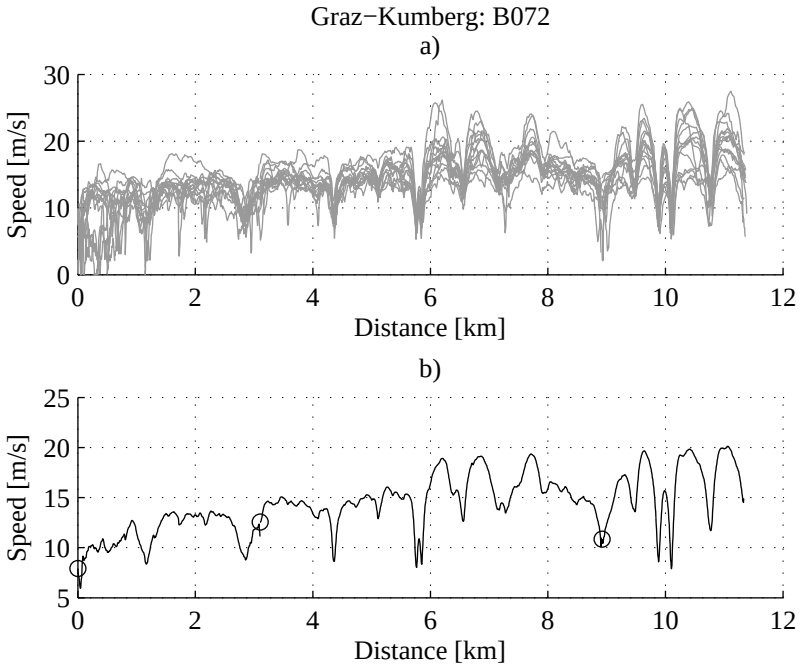

Figure 6.12: Overlaid bidirectional speed profiles. a) The speed profiles of all tracks. b) The mean of the speed profiles with crossroads marked by a circle ○. The first kilometre lies in the city of Graz and shows highly varying speeds, that indicate stop-and-go traffic. There are two particular noticeable features around kilometre 6 and 10. These two pairs of minimums correspond to a pair of hairpin bends on the map.

of the tracks. In comparison the speeds vary a great deal on the rural sections. When looking at the peak speeds in theses areas one can see speed differences of 10 m/s and more between the different tracks. This is likely due to different traffic congestion levels and other factors, that are not directly dependent to the vehicle's position on the road. In comparison the speed minimums are very short. Ignoring the stop-and-go period in Graz, most of these minimums are repeated at the same position on the road for all tracks. Looking at the tracks separated by the direction they were driven in, in the figures 6.13 and 6.14, there is little difference to be noted. The easiest way to compare them is by looking at the mean speed profiles derived from the directional tracks. The largest differences can be observed around the points of kilometres 6.1 and 7.8. The difference is about 3 m/s or 10 km/h. By conferring with the driver it was established that the reason for the speed difference around kilometre 6.1 was due to him using the engine brake going downhill, which limited the speed. To a lesser degree this explanation also applies for the second point at kilometre 7.8.

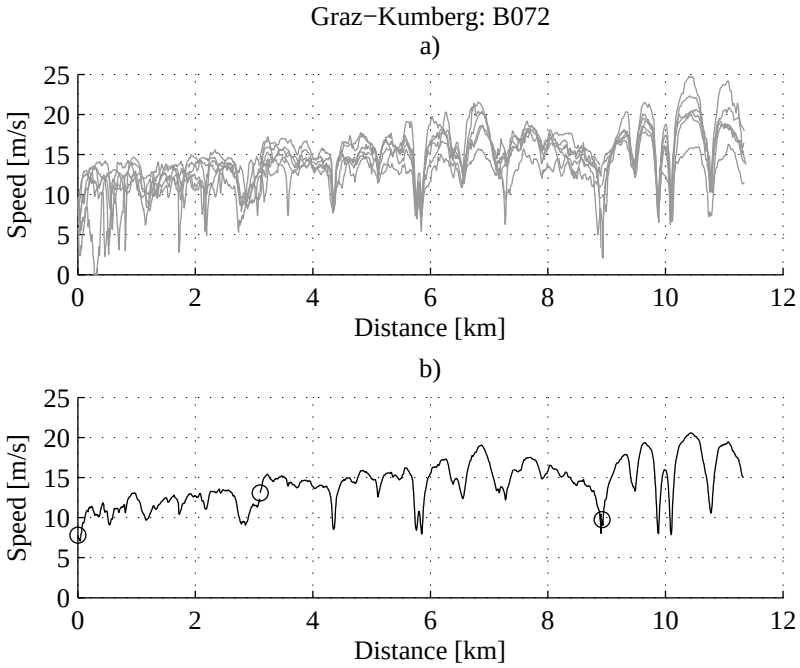

Figure 6.13: Overlaid directional speed profiles from Graz to Kumberg. a) The speed profiles of the tracks. b) The mean of the speed profiles with crossroads marked by a circle ○.

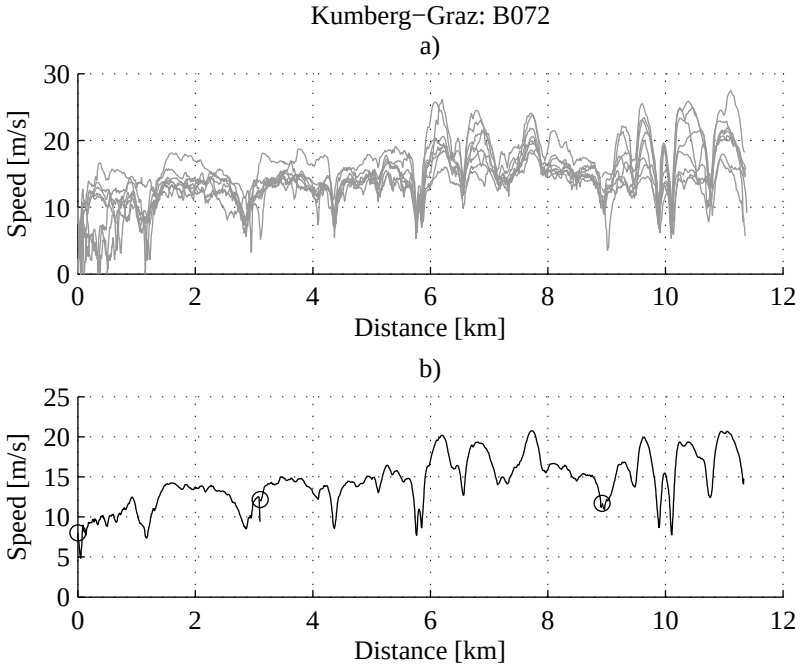

Figure 6.14: Overlaid directional speed profiles from Kumberg to Graz. a) The speed profiles of the tracks. b) The mean of the speed profiles with crossroads marked by a circle ○.
The two figures 6.13 and 6.14 are used to show the difference between the speed profiles depending on their direction. The two profiles are similar to each other.

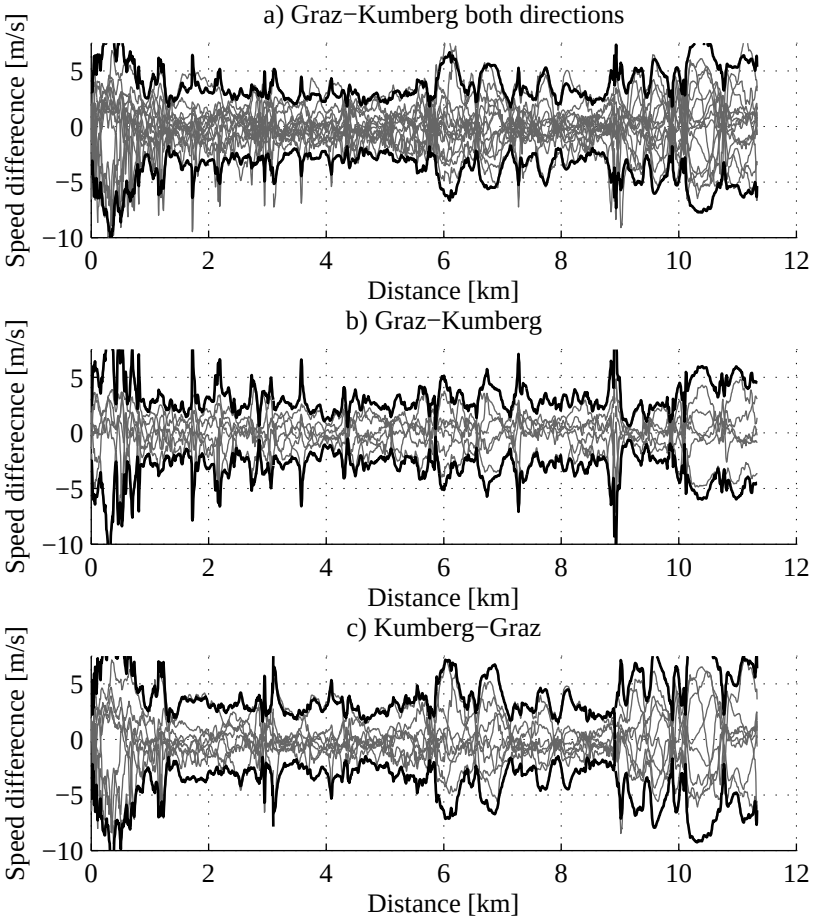

Figure 6.15: Difference between single tracks and their mean speed curve. a) Difference for all tracks. b) Difference for the tracks from Graz to Kumberg. c) Difference for the tracks from Kumberg to Graz. The stop-and-go traffic condition in Graz is noticeable on the left side of all three graphs. Graph (c) shows a considerably higher variation of the speed in the right halve than does graph (b).

Table 6.2: Double minimum positions

Distance [km]	Easting [km]	Northing [km]
5.8	538.6	5219.0
10	540.5	5221.2

There are two particularly noticeable features that consist of two consecutive minimums in quick succession just before kilometre 6 and around kilometre 10. These two features are roughly at the map coordinates shown in table 6.2. Looking up these positions on the map displayed in figure 6.7, one sees that these coordinates point to two pairs of hairpin bends. It can therefore be concluded that these speed reductions are primarily due to the extreme curvature at these bends. To gain more insight a detailed view of the difference between mean and track profiles is shown in figure 6.16. The points of minimum speed of the averaged curve have been marked with dashed lines. These points differ slightly depending on the direction driven. For the first pair of bends (b1, c1), the section between the minimums is focused in a narrow ±2 m/s band. In comparison to this the second section (b2, c2) shows a speed band of the default ±5 m/s. The speed band at the actual minimums is however in the ±2 m/s area. The road segment between this second pair of bends is well over double the length of the first segment, giving the driver considerably more time to accelerate to higher speeds between the two bends as can bee seen in the speed profile 6.12. As noted the speed in these curves has a very low variance across the tracks. This allows the conclusion that the speed in these curves is mainly dependent on their curvature. This matches the suggestion by Back [11] discussed in section 2.1, that the maximum possible speed depends on the curvature of the road and the acceptable side acceleration affecting the driver. It must be noted that these tracks were gathered by the same driver and vehicle. The acceptable side acceleration may differ for different drivers. However the repeatability of the speed profile at these bends suggests, that these points can be considered specific characteristics of the road. The speed minimums near the kilometres 6.5 and 10.8 can also be associated with sharp bends in the road.

The other points of interest are the two crossroads along the route. As already mentioned the crossroads have been marked by a circle in the figures displaying the speed profiles 6.12, 6.13, 6.14 and the contour plot of the road on the map 6.7. Figure 6.17 shows the magnified view of the relevant sections. As the crossroads were used to synchronise the different tracks to each other, there are small gaps in the tracks at these points. In hindsight this choice of reset points was not a good idea. Contrary to the profiles around the curves, these profile vary a great deal from each other. Looking at the first crossroads (b1),(c1) one can see a speed indent. Surprisingly this

indent is only on the left side of the crossroads, one would expect that the tracks from Kumberg to Graz (c1) would reduce speed before they reach the crossroads rather then after they passed it[4]. A subset of the tracks, shown in plot (c1), does however show a similar profiles to each other. As there was no explanation found for this behaviour from the available map data, the driver was consulted for an explanation. It turned out that the road had the right of way for this section, so the crossroads had little impact on the speed. However a short distance from the crossroads, in the direction of Graz, there is a short speed limit of 30 km/h. This additional information explains the profile behaviour sufficiently. On the other hand the plots for the second crossroads (b2), (c2) conform to the behaviour from the available map data. In the plot (c2) one can see some tracks crossing this point unimpeded. The others slow down by different degrees and then cross the point accelerating. The tracks in the other direction show similar behaviour. Several tracks just pass the crossroads. Two tracks slow down before the crossroads and again after it. This is probably due to interference by other vehicles. The work by Ichikawa et al. [9], presented in section 2.1, clustered together similar driving patterns around crossroads and assigned probabilities to their occurrence. This idea can be easily applied to the second crossroads. Judging by sight, one can conclude it is most likely that the vehicle can pass the crossing unimpeded.

This concludes the discussion and analysis of the results. A summary of these results along with a perspective on the direction for further research will be given in the next chapter.

[4]Remember that the driven direction is from right to left for these tracks.

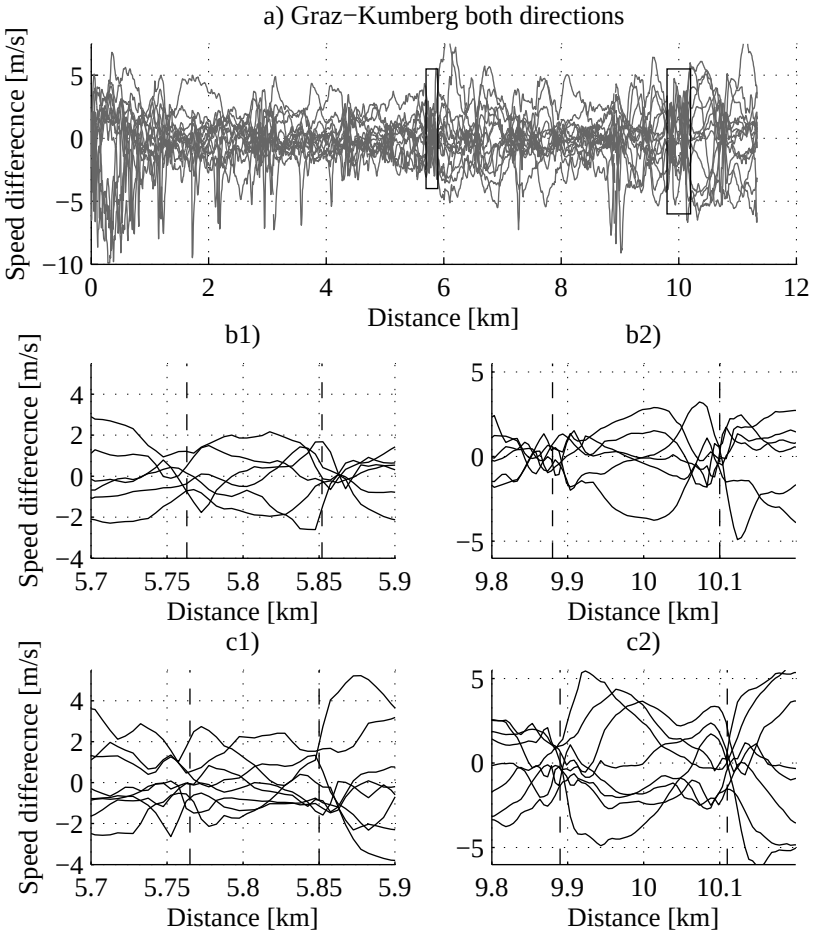

Figure 6.16: Detailed view of the speed difference between the mean speed curve and single tracks in sharp bends. The bends are marked by vertical dashed lines in the detailed view. a) Overview displaying the difference for all tracks and marking the zoomed area. b) Detailed view of the two curve pairs for the tracks from Graz to Kumberg . c) Detailed view of the two curve pairs for the tracks from Kumberg to Graz. It is notable that the speed variation at these bends are lower during the rest of the track. The lower variation between the first pair of bends, shown in (b1) and (c1) is attributed to the smaller distance between them giving the driver less time to accelerate.

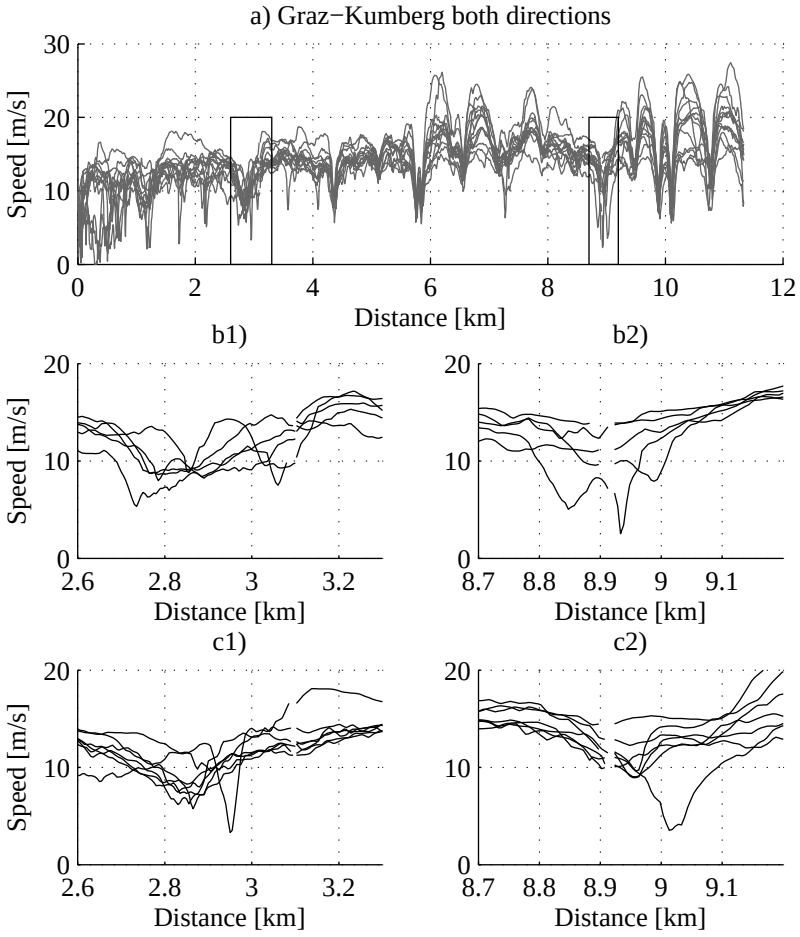

Figure 6.17: Detailed view of the speed near crossroads. The crossroads can be seen in the detailed view as there are gaps in the data at their positions. The 1st crossroads are located at ∼3.1 km, the 2nd ∼8.9 km. These locations are also marked by the gaps in the profiles. The behaviour of the speed curves at the 2nd crossroads (b2), (c2) is as expected. On the other hand the behaviour at the 1st crossroads was not expected. The vehicle passed the crossroads unimpeded, but slowed down before (b1) or after (c1) that crossroads. This was because the vehicle had right of way at the crossroads and a speed limit at the point where the slower speed occurred.
a) Overview displaying the speed for all tracks and marking the zoomed area.
b) Detailed view of the two crossroads for the tracks from Graz to Kumberg.
c) Detailed view of the two crossroads for the tracks from Kumberg to Graz.

Chapter 7

Conclusion

This last chapter will point out possible alternatives for the input data sources. The more interesting results from the last chapter will be summarized. It will also suggest which material is worth more extensive studies based on these results.

The quality of the input and measurement data, that was presented in section 3.1, will increase in the near future. For example the availability and accuracy of digital maps will continually increase.

The DEM data collected by the Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) was released to the public in June 2009 [32]. ASTER is a sensor mounted on a satellite. The sensor is an advanced multispectral imager covering a wide spectral region from the visible to the thermal infrared.

Satellite navigation systems will also increase in number and accuracy. With the Galileo project the European Union is creating an additional navigation system. There are also plans to improve the GPS and increase its accuracy [33].

As described in chapter 4 a mapmatching algorithm was developed. However due to the fact that the author of this diploma thesis had no previous experience about this subject, the time and effort needed to develop that algorithm was underestimated. Therefore there was not enough time to create and verify a model for driving conditions based on the digital map data. However chapter 6 has given a lot of indications what sort of information can be extracted from the digital map data. This will provide a starting point for future research.

The comparison of single GPS height profiles with the road profile provided mixed results. One of the three profiles[1] showed differences of almost

[1] See figure 6.3

40 m between GPS and road height data. The profiles shown were chosen to represent all the results. This means that the problem was not a single occurrence. Because only the information from a single track was used no real conclusions could be drawn from them. However the theory was proposed that such behaviour could be explained by the relation of the road to the height map. Roads running parallel to contour lines were more likely to suffer from such problems than roads in plain country or roads running perpendicular to the contour lines. More work needs to be done to verify this theory. Providing it can be proven to be true, it would enable the ability to assign each height profile a degree of confidence on its accuracy.

According to this multiple tracks along the same route were analysed and the mean height profile was compared with the road profile. The average difference between these two profiles was ∼10 m. The standard deviation of the averaged GPS heights was within 10 m for most points. Unfortunately only one route was available for this analysis. It would be interesting to do a similar test with multiple tracks on one of the more problematic single tracks.

One needs to remember that the aim is to provide a HEV controller with the necessary information to act more efficiently. The relevant information from the height profile is the change in potential energy E_p:

$$\Delta E_p = mg\Delta h \tag{7.1}$$

This formula depends on the gravitational constant g, the mass of the vehicle m and the height difference Δh. This means that the controller does not need the absolute elevation level. Instead the change of elevation is required. In the figure 6.8b comparing the mean profiles of the tracks and road profile derived from the SRTM data, it was established that the trend of the two profiles match each other well. The applicability of using these profiles will depend a lot on how far ahead the controller looks. For larger sections the impact of small discrepancies will be less pronounced as shown in figure 6.8c.

Based on these results the following method is outlined to make use of the height profile. The road height profile is available to the vehicle from a digital map. The vehicle is equipped with a GPS-receiver but no previous height profiles are available. The profile from the GPS-receiver is based on the current and past points. It acts as a feedback to verify if the information from the map is usable. The usability is determined by the requirements of the controller.

To improve the performance, past GPS-tracks can then be included to expand the basic idea.

In the second part of chapter 6 the speed profiles were analysed. Unsurprisingly the speed limits were found to have a major impact on the vehicle speed. In the sections where the speed limit of 50 km/h was applied, the observed speeds were around the limit. In the rural areas with a speed limit of 100 km/h the speeds varied a great deal. Unfortunately the speed limits were not available from the digital map data, but acquired by interviewing the driver about them. Obviously this is not feasible for any practicable implementation. The speed limits need to be available in digital form in conjunction with the road map, to make any attempt to predict the speed profile.

A strong correlation was found between the speed and sharp bends of the road. In such bends the maximum speed is mainly determined by the curvature. The curvature can be calculated from the digital road map data. If the maximum speed derived from this data is significantly less than the speed limit of the road, a deceleration to curvature speed can be predicted.

Due to the fact that the observed route only included two crossroads, their analysis was inadequate. The impact of crossroads on the speed, is highly dependant on the right of way. One case showed that while the vehicle was travelling on a major road, the crossroads had little to no influence on its speed. A prediction of the speed profile at this point, based solely on digital maps data, looks very complex if not impossible. Using past tracks and clustering similar profiles together, as suggested by Ichikawa et al. [9], looks more promising.

The discussion above has provided suggestions on how the prediction of height and speed profiles could be achieved. It is however important to not forget the context in which these profiles are needed in the first place. What the controller ultimately needs is the future energy or power demand. The height and speed profile need to be converted into a corresponding energy profile. The interesting question is: which of the features of the height and speed profiles have the most significant fuel and emissions saving potential? This could be done by simulations on a model. Profiles with varying accuracy and specific features can be used as inputs. A controller optimized for those features can then be used and the resulting fuel and emission levels of the model can be observed. Pinpointing such features would help to focus the direction of further research.

Bibliography

[1] A. Sciarretta and L. Guzzella. Control of hybrid electric vehicles. *Control Systems Magazine, IEEE*, 27(2):60–70, 2007.

[2] Eva Ericsson. Independent driving pattern factors and their influence on fuel-use and exhaust emission factors. *Transportation Research Part D: Transport and Environment*, 6(5):325 – 345, 2001.

[3] Chan-Chiao Lin, Soonil Jeon, Huei Peng, and Jang Moo Lee. Driving pattern recognition for control of hybrid electric trucks. *Vehicle System Dynamics*, 42(1):41, 2004.

[4] A. Rajagopalan, G. Washington, G. Rizzoni, and Y. Guezennec. Development of fuzzy logic and neural network control and advanced emissions modeling for parallel hybrid vehicles, December 2003.

[5] M. H. Hajimiri and F. R. Salmasi. A fuzzy energy management strategy for series hybrid electric vehicle with predictive control and durability extension of the battery. In *ICEHV. IEEE Conference on Electric and Hybrid Vehicles*, pages 1–5, 2006.

[6] M. Montazeri-Gh, A. Ahmadi, and M. Asadi. Driving condition recognition for genetic-fuzzy HEV control. In *3rd International Workshop on Genetic and Evolving Systems GEFS*, pages 65–70, 2008.

[7] M. Montazeri-Gh and M. Asadi. Application of vehicle telematic system in fuzzy-based HEV control. In *Information and Communication Technologies: From Theory to Applications. ICTTA. 3rd International Conference on*, pages 1–6, 2008.

[8] Bor Yann Liaw and Matthieu Dubarry. From driving cycle analysis to understanding battery performance in real-life electric hybrid vehicle operation. *Journal of Power Sources*, 174(1):76–88, 11/22 2007.

[9] S. Ichikawa, Y. Yokoi, S. Doki, S. Okuma, T. Naitou, T. Shiimado, and N. Miki. Novel energy management system for hybrid electric vehicles utilizing car navigation over a commuting route. *Intelligent Vehicles Symposium, IEEE*, pages 161–166, 2004.

[10] Y. Yokoi, S. Ichikawa, S. Doki, S. Okuma, T. Naitou, T. Shiimado, and N. Miki. Driving pattern prediction for an energy management system of hybrid electric vehicles in a specific driving course. In *30th Annual Conference of IEEE Industrial Electronics Society. IECON.*, volume 2, pages 1727–1732 Vol. 2, 2004.

[11] Michael Back. Prädiktive Antriebsregelung zum energieoptimalen Betrieb von Hybridfahrzeugen, 2006.

[12] R. Langari and Jong-Seob Won. Intelligent energy management agent for a parallel hybrid vehicle-part i: system architecture and design of the driving situation identification process. *IEEE Transactions on Vehicular Technology*, 54(3):925–934, 2005.

[13] Jong-Seob Won and R. Langari. Intelligent energy management agent for a parallel hybrid vehicle-part ii: torque distribution, charge sustenance strategies, and performance results. *IEEE Transactions on Vehicular Technology*, 54(3):935–953, May 2005.

[14] Gong Qiuming, Li Yaoyu, and Peng Zhong-Ren. Trip-based optimal power management of plug-in hybrid electric vehicles. *IEEE Transactions on Vehicular Technology*, 57(6):3393–3401, 2008.

[15] L. Johannesson, M. Asbogard, and B. Egardt. Assessing the potential of predictive control for hybrid vehicle powertrains using stochastic dynamic programming. *Intelligent Transportation Systems, IEEE Transactions on*, 8(1):71–83, 2007.

[16] Ltd Trimble Navigation. Gps tutorial, September 2009. http://www.trimble.com/gps/.

[17] Renate Czommer. Leistungsfähigkeit fahrzeugautonomer ortungsverfahren auf der basis von map-matching-techniken, 2000.

[18] Holux technology inc. website, November 2009. http://www.holux.com/JCore/en/home/index.jsp.

[19] Christian S. Jensen and Nerius Tradišauskas. Map matching. In LING LIU and M. TAMER ZSU, editors, *Encyclopedia of Database Systems*, pages 1692–1696. Springer US, 2009. 10.1007/978-0-387-39940-9_215.

[20] GIS-Steiermark. Styria road network data, March 2009. http://www.gis.steiermark.at/.

[21] ESRI. Esri shapefile technical description, 1998.

[22] GIS-Steiermark. User-friendly desktop internet gis, March 2009. http://udig.refractions.net/.

[23] Tom G. Farr, Paul A. Rosen, Edward Caro, Robert Crippen, Riley Duren, Scott Hensley, Michael Kobrick, Mimi Paller, Ernesto Rodriguez, Ladislav Roth, David Seal, Scott Shaffer, Joanne Shimada, Jeffrey Umland, Marian Werner, Michael Oskin, Douglas Burbank, and Douglas Alsdorf. The shuttle radar topography mission. *Reviews of Geophysics*, 2007.

[24] Jonathan de Ferranti. User-friendly desktop internet gis, January 2009. http://www.viewfinderpanoramas.org/dem3.html.

[25] Norbert de Lange. *Geoinformatik in Theorie und Praxis.*

[26] Steven Dutch. Converting utm to latitude and longitude (or vice versa), February 2009. http://www.uwgb.edu/dutchs/usefuldata/utmformulas.htm.

[27] J. Seager, P. Collier, and J. Kirby. Modelling geoid undulations with an artificial neural network. volume 5, pages 3332 –3335 vol.5, 1999.

[28] NGA/NASA. Nga/nasa egm96, n=m=360 earth gravitational model, March 2009. http://earth-info.nga.mil/GandG/wgs84/gravitymod/egm96/egm96.html.

[29] Rainer Joeckel Franz Josef Gruber. *Formelsammlung für das Vermessungswesen.* Teubner, 2007.

[30] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[31] DIN 1319, Grundlagen der Messtechnik.

[32] ASTER GDEM Validation Team: METI/ERSDAC, NASA/LPDAAC, USGS/EROS. Aster global dem validation summary report, 2009. Available from: http://asterweb.jpl.nasa.gov/gdem.asp.

[33] F. Vejrazka. Galileo and the other satellite navigation systems. pages 1 –4, apr. 2007.

Acronyms

EMT Institute of Electrical Measurement and Measurement Signal Processing

HEV Hybrid Electric Vehicle

ICE Internal Combustion Engine

SOC State of Charge

SOH State of Health

FLC Fuzzy Logic Controller

DP Dynamic Programming

PM Particulate Matter

NOx Nitrogen oxide

CO Carbon monoxide

HC Hydrocarbon

NEDC New European Driving Cycle

FTP-75 Federal Test Procedure

rpm revolutions per minute

GPS Global Positioning System

WGS 84 World Geodetic System 84

UTM Universal Transversal Mercator

DGPS Differential GPS

GIS Geographic Information System

DEM Digital Elevation Model

SRTM Shuttle Radar Topography Mission

ASTER Advanced Spaceborne Thermal Emission and Reflection Radiometer

SVD Singular Value Decomposition

List of Figures

List of Tables