Czech Technical University in Prague

Faculty of transportation sciences

Department of Transport Telematics
Study program: Intelligent Transport Systems

Methodology of the Positioning Performance Testing in Car Navigation

Master's Thesis

Author: Elshan Khalilov
Master's thesis supervisor: Ing. Petr Bureš, Ph.D.

Year: 2015

CZECH TECHNICAL UNIVERSITY IN PRAGUE
Faculty of Transportation Sciences
Dean's office
Konviktská 20, 110 00 Prague 1, Czech Republic

K620..Department of Transport Telematics

MASTER'S THESIS ASSIGNMENT

(PROJECT, WORK OF ART)

Student's name and surname (including degrees):

Bc. Elshan Khalilov

Code of study programme code and study field of the student:

N 3710 – IS – Intelligent Transport Systems

Theme title (in Czech): **Metodika testování určení polohy vozidlovou navigací**

Theme title (in English): Methodology of the Positioning Performance Testing in Car Navigation

Guides for elaboration

During the elaboration of the master's thesis follow the outline below:

- Survey of state of the art navigation device testing methodology.
- Survey of existing technology for testing of navigation devices in simulated environment.
- Proposal of different scenarios for testing of capability of navigation device.
- Comparison of the results in real and simulated environment.
- Finalizing test scenarios using prescribed form for autonomous testing.
- Algorithm for the Comparison of Positioning Performance of Car Navigation

Graphical work range: standard

Accompanying report length: stantard

Bibliography:

HOFFMANN-WELLENHOF, et. al. GNSS - Global Navigation Satellite Systems: GPS, CLONASS, Galileo and more. Wien: Springer, ©2008. xxix, 516 s. ISBN 978-3-211-73012-6.

DEMILLO, Richard A. et. al. Software Testing and Evaluation. 1st Ed. Menlo Park: Benjamin/Cummings Publ. Co., 1987. 537 s. ISBN 0-8053-2535-2.

Master's thesis supervisor: **Ing. Petr Bureš, Ph.D.**

Date of master's thesis assignment: **June 30, 2014**

(date of the first assignment of this work, that has be minimum of 10 months before the deadline of the theses submission based on the standard duration of the study)

Date of master's thesis submission: **November 30, 2015**

a) date of first anticipated submission of the thesis based on the standard study duration and the recommended study time schedule
b) in case of postponing the submission of the thesis, next submission date results from the recommended time schedule

L. S.

doc. Ing. Pavel Hrubeš, Ph.D.
head of the Department of Transport Telematics

prof. Dr. Ing. Miroslav Svítek
dean of the faculty

I confirm assumption of master's thesis assignment.

Elshan Khalilov
Student's name and signature

Prague ....June 30, 2014

Declaration

I have no relevant reason against using this schoolwork in the sense of § 60 of Act No 121/2000 concerning the authorial law.

Prague, 09.06.2015 Elshan Khalilov

Declaration

I declare that I accomplished my final thesis by myself and I named all the sources I used in accordance with the guideline about the ethical rules during preparation of University final thesis.

Prague, 09.06.2015 Elshan Khalilov

Acknowledgements

I would like to take this opportunity to express my deepest gratitude and appreciation to all the people who assisted me in preparation of this Master Thesis and who have made my overall experience at **Škoda Auto** both fun and challenging.

I am indebted to Doc. Ing. Pavel Hrubeš Ph.D., Ing. Petr Bureš, Ph.D., Ing. Petr Struhovský, Ph.D. and Ing. Jan Krejčí for the theoretical and technical insights they have shared, and in general for all the support they have provided.

I would also like to thank the entire **Škoda Auto** for providing me with all the necessary equipment and tools that made the realization of this project possible.

I would also like to express my sincere gratitude to the whole **FD ČVUT** team, which have made my study process during these two years very positive, enjoyable and interesting, I appreciate everyone for such precious knowledge and skills I have obtained during my master's studies.

Last, but not least so far, I sincerely thank my family. I would never reach any success in my life without their support. Thanks to my groupmates, especially to my groupmate and best friend Raissa Likhonina and to all of my friends for all the support and encouragement they have provided all along my studies in Czech Technical University in Prague.

Abstrakt

Název práce:	Metodika testování určení polohy vozidlovou navigací
Autor:	Bc. Elshan Khalilov
Vysoká škola:	České Vysoké Učení Technické v Praze Fakulta Dopravní
Studijní obor:	Inteligentní Dopravní Systémy
Kategorie:	Magisterská Práce
Vedoucí:	Ing. Petr Bureš, Ph.D.
Katedra (Ústav):	Ústav dopravní telematiky
Konzultant:	Ing. Jan Krejčí

Tato diplomová práce je věnována testování výkonnosti navigačního systému ve vozidle. Jsou v ní popsány základní principy GNSS, uveden přehled teorie testování, existující metodiky a technologie pro testování výkonu navigačních přístrojů. Hlavním cílem diplomové práce je metodologie testování výkonnosti určování polohy navigačních systémů, která by měla výrazně zjednodušit proces testování a ve výsledku zmenšit dobu strávenou na sestavení plánu zkoušek. Kromě toho tato metodologie umožní objasnit výsledky testů ve formě příslušných číselných hodnot jako i pomocí grafické reprezentace.

Klíčová Slova: navigace, výkonnost určování polohy, GNSS simulátor, navexp, testování, scénář testování, testovací případ

Abstract

Title:	Methodology of the Positioning Performance Testing in Car Navigation
Author:	Bc. Elshan Khalilov
University:	Czech Technical University in Prague
	Faculty of Transportation Sciences
Study program:	Intelligent Transport Systems
Category:	Master's Thesis
Supervisor:	Ing. Petr Bureš, Ph.D.
Department:	Department of Transport Telematics
Advisor:	Ing. Jan Krejčí

This Master Thesis is dedicated to the in-car navigation performance testing. In this paperwork the basic GNSS principles, testing theory overview, existing methodologies and technologies for navigation device performance testing are represented. The main goal of this paperwork is the methodology for the positioning performance testing, which should significantly simplify the testing procedures, and as a result – to reduce the time spent for the test plan creation, to clarify the test results, providing the clear resulting information in values, as well as in graphical representation.

Keywords: navigation, positioning performance, GNSS simulator, navexp, testing, test scenario, test case

Contents

Preface

Year by year, in-car positioning technologies become more and more popular, and nowadays it is very difficult to imagine our journey without positioning and navigation device, especially while travelling to some foreign city or country. Satellite positioning technologies have totally changed our lifestyle and paper maps almost disappeared from drivers' usage. Of course, with the rapid development of such a challenging technologies, at the same time we can observe a strong growth of the popularity due to the very considerable factor: navigation technologies can provide an accurate vehicle location and the road guidance function will calculate the most efficient route to the required destination point. Consequently, the navigation device will significantly reduce trip time and gasoline consumption.

The key part of successful in-car navigation development is testing. Indeed, testing process guarantees a proper functionality of any system, while testing is undertaken by persons, who are at least very familiar with its proper usage and system performance; however, the end-user satisfaction fully depends on the actual performance. Therefore, there are many testing methodologies created for different types of systems, and speaking about the in-car navigation device performance testing, I see several possibilities: laboratory testing and real-world testing. Of course, the GNSS signal simulators exist today, which give us the opportunity to simulate the GNSS signal and real-world environment in a laboratory to test a navigation device in simulated environment, but in my personal opinion, it does not provide all the necessary data to ensure that the device will have a good performance in real-world conditions. In contrast, the real-world testing can provide us a valuable performance data. It is clear, that the real-world conditions are not constant and humans are not able to control that. Nevertheless, testing both in simulated and real-world conditions can very help us to understand somehow the real-world components' affection on the positioning device performance.

This is clear, that both of the methodologies have their advantages and disadvantages. For example, real test-drive is not always affordable, especially depending on the testing area location (it may be situated in another county or even continent), and it is not possible to simulate a 100% real-world conditions in the laboratory, in spite of the possibility to simulate its key characteristics.

Nevertheless, these both methodologies exist today and both ones are very used. This thesis is dedicated to the positioning performance testing. The thesis contains a short overview of GNSS technology, testing theory, their basic principles and methodologies, as well as the survey of the existing positioning performance testing technology and methodologies; however, the key point of this Master Thesis is the comparison of the testing results in real and simulated environments.

This thesis describes the prescribed master form for the test scenarios, test scenarios filtering algorithm by the parameters as well as the algorithm for the test acceptance/non-acceptance computation, and the comparison algorithm for the results of testing of the navigation device performance. The key objective of this work is to reduce time for processing the testing, to increase the comprehension of the future test plans according to the testing requirements. The filtering application, which is called Navexp, should considerably reduce time spent on the selection the test scenarios according to the defined in advance test cases. Consequently, the comparison algorithm itself should clarify the whole picture of the test result.

1. Basic Principles and Techniques

This Chapter represents the short overview of main principles and techniques, related to the Master Thesis. The main components of the GNSS technology and the short overview of the testing theory with its main principles, as well as the usage of GNSS in automotive industry and the in-car navigation testing requirements are discussed in this Chapter.

1.1 GNSS (Global Navigation Satellite System)

Global Navigation Satellite System is an asset technique in many different fields where the positioning services are very widely used. GNSS is an integral tool of modern positioning systems and nowadays, cooperating with Traffic Information Services, it has become a very necessary instrument in our daily routine, giving us the important traffic information, and especially helping us to avoid the traffic jams or detour the maintenance roadways, choosing the most optimal driving route, considerably saving time and fuel.

GNSS consist of several wireless space-based positioning systems, which provide us the accurate location, speed and time information anytime and almost anywhere on our planet. The main GNSS technologies are: GPS (Global Positioning System), GLONASS (Globalnaya navigatsionnaya sputnikovaya sistema) and Galileo. As shown on Figure 1.1, each system contains three general components: space segment, control segment and the user segment. There are many various papers written about the GNSS, and I decided to use (Bernhard Hofmann-Wellenhof, 2008) and (J. Sanz Subirana, 2013) as a basis.

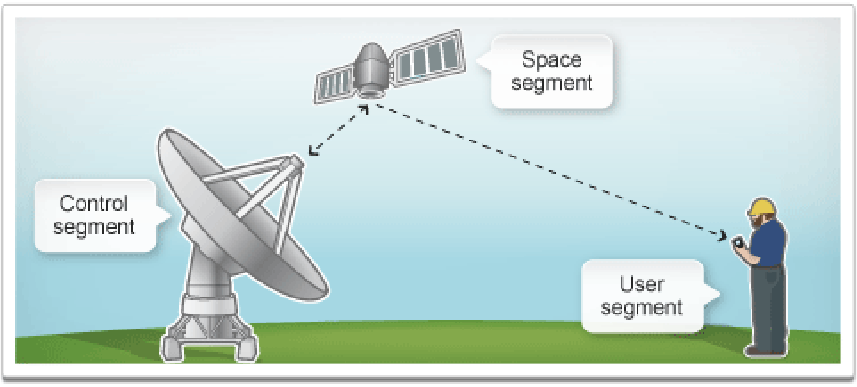


Figure 1.1: Components of GNSS
Note: The figure is taken from the NATIONAL VET CONTENT web page.

1.1.1 Space Segment

The space segment contains a set of specific number of satellites, what composes a *satellite constellation*. The amount of the satellites depends on the exact navigation system. The key role of the satellites is time and position transmission to the users. The level of minimum four satellites should be good to provide a proper performance of the navigation device. Each of the satellite equipped by atomic clock, computers, transmitter and another additional equipment to provide the control and proper system operation. The satellites are powered by solar batteries. The each satellite has specific name and the appropriate set orbital position. (Bernhard Hofmann-Wellenhof, 2008)

1.1.2 Control Segment

The control segment (also called a ground segment) is responsible for the whole system's control. This usually represents as a ground area with a building (control and operational center) and antennas. The control in this content means such tasks as system operations, maintenance, monitoring the satellites state, including their atomic clock data and the auxiliary parameters, to transmit the data messages to satellites. The control segment is also responsible for the security issues, in case if the service is not a public one, data encoding etc. (Bernhard Hofmann-Wellenhof, 2008)

1.1.3 User Segment

The user segment mainly consists of the satellite signal receivers with different purposes - navigation devices, TV receivers and other different information systems. The users are classified on users required an authorization (for example – military) and do not (civilian, for example – navigation device users). (Bernhard Hofmann-Wellenhof, 2008)

Receivers contain the antenna, specific radio frequency range, microprocessor, power source (electricity, battery, whatever), the intermediate-precision oscillator, the memory source and the user interface. (J. Sanz Subirana, 2013)

1.2 Role of GNSS Technology in Automotive Industry

GNSS technology is very widely used in automotive industry, and especially during the last few of years, we can observe a swift growth of the amount of manufactured cars, where the inside GNSS units are implemented. There are many facts prove the importance and necessity in such a technologies. (Ngah, 2006)

1.2.1 Vehicle Navigation

In-car navigation systems is the major user of GNSS technology. Many of us remember the times, when we used to use paper maps to specify our location and the proper roadway. Nowadays we just need to feed the destination data into the on-board computer, the optimal route is immediately calculated, showing all the possible routes as well, and we can be sure, that we will reach our destination. This feature especially can be very useful in such situations, while driving on unknown town or city. In-car navigation systems also contain such a features like approximate travel time calculation and fuel consumption calculation for reaching the specified destination point. Furthermore, the most of the in-car navigation systems cooperate with traffic information systems. This property helps the drivers to avoid the road sections with a heavy traffic and traffic accidents, considerably reducing a travel time and fuel consumption. The navigation devices also represent traffic signs, for example – the maximal speed limit, and when the driver exceeds a speed, the voice notification warns about that. Thus, we can conclude, that the in-car navigation systems make our journey comfortable, enjoyable and relatively safe at the same time. (Ngah, 2006)

1.2.2 Dynamic Vehicle Routing

Dynamical Vehicle Routing is widely used in delivery services. This is very useful tool, which helps coordinators to control the in-time delivery of any good according to the specified schedule. The main feature of DVR is the road guidance according to the optimal driving route, calculating arrival time and journey time. This application is also very useful for public transport industry in controlling schedule-based arrivals and departure of buses, trams, etc.

1.2.3 Tracking Rental Vehicles

This involves mainly a security issue. Obviously, when renting companies rent a car, they cannot be a 100% sure that the driver will follow the road regulations. Many companies consequently receive considerably charges, for example, for speed limit exceeding, or using toll road while driver did not pay for its usage. Therefore, such practice has been implemented: they install GPS beacons on their rental cars to obtain a monitoring possibility as full as possible, and they can be notified about any law or contract violation. The detailed information about this technology given in (Omarah Omar Alharaki, 2010).

1.3 Vehicle Satellite Navigation Systems

GNSS technology is an integral component of car navigation, however the combination of the GNSS technology with the telecommunication technology and computer, including specific GIS application, compose a fully functioning navigation device. The signals received from the satellites represent the current vehicle position on the digital map in cursor representation to the computer's display. (Gopal Dommety, 1998)

With help of such a technologies, as well as in cooperation of appropriate authorities (in our case it can be traffic information center, the ministry of transportation services, etc.), it is possible the device to calculate the shortest and fastest route. This also helps drivers to avoid the problematic road sectors (traffic jams, car accident places etc.).

Nowadays, there are plenty of manufacturers, who produce such a devices, and as the leaders of the market in GNSS Navigation Technologies, I would mention Garmin, TomTom and others.

1.3.1 Functions of Vehicle Navigation Systems

The car navigation device contains a GPS antenna and a computer – in-car computer installed to the dashboard with a GPS module, or a portative navigation device, installed on a car windshield. Receiver contains a digital map stored in memory stick (or CD-ROM) that contains the road infrastructure and represents the vehicle location on the display. Thus, to get the optimal route details it is enough to type in the destination information, and in seconds, the path optimal path is calculated. (Gopal Dommety, 1998), (Ngah, 2006)

The newest navigation devices usually have a possibility to integrate with a car multimedia system, including a microphone and the loudspeakers, so that would be possibly to tell the device the required destination details, and during a driving process, the navigation device would tell back to the driver, where to drive, in how many meters to turn, for example, left. This is very useful option indeed, when this gives the drivers an opportunity to be more concentrated on the road and less often to watch to the computer display.

1.3.1.1 Distance Calculation

The general explanation is found in (Ngah, 2006), where said, that the computer clock and the satellite clocks are synchronized. Since the center of gravity of the earth influences the rate at which the clocks change, the clocks are not synchronized with respect to each other but as a function of their velocity with respect to the center of the coordinate system, which is the center of the earth. Packets from a satellite A contain a time stamp t1 as part of the packets. Upon receiving these packets, the receiver clock reads its time t2. The frequency of transmission is f and the wavelength of transmitted radio signals is λ. As a result of the clocks synchronization, the computer determines the distance between the satellite and the car as follows:

The one-way delay of radio signals: $t = t_2 - t_1$
The speed of radio signals: $v = f\lambda$
The distance between satellite and car: $speed * delay = f\lambda\ (t_2 - t_1)$
(Ngah, 2006)

1.3.1.2 Frequency Identification

Although all GPS satellites broadcast on the same frequency, the receiver computer is still capable to distinguish satellites using their broadcast frequency. The computer has identification codes for all the satellites stored in its memory. While transmitting, each satellite includes as part of the transmitted packets its identification code. The receiver computer then determines which transmission is from which satellite by matching the identification code with the ones stored in its memory. The computer uses this frequency identification together with the distance calculation above to determine its exact distance from that particular satellite. With frequency identification, GPS receivers that do not have antennas can still make reliable use of the tiny signals they receive. (Ngah, 2006)

1.3.1.3 Coordinate Determination

The most appropriate information for this body I have found in (Ngah, 2006) as well. There is said, that for a system with n coordinates, at least n lines need to intersect in order to specify the coordinates of a point. This is the principle that the GPS system uses. The origin of the GPS system is the center of the earth. Here, $n = 3$. (Ngah, 2006)

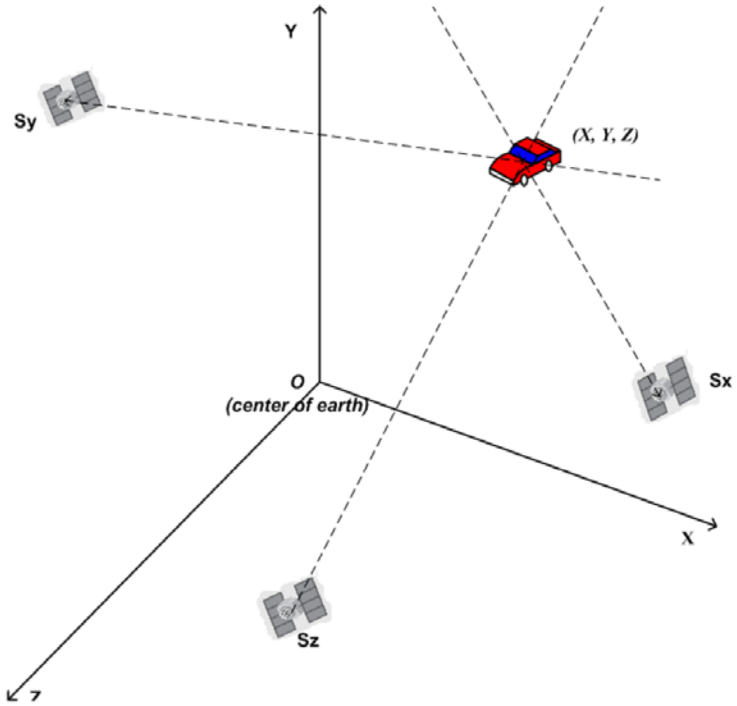


Figure 1.2: Three GPS satellites Sx, Sy, and Sz are used to geometrically determine the X, Y, and Z coordinates, respectively, of a car in the earth's 3D coordinate.
Source: (Ngah, 2006)

Each satellite specifies a line. So, at least three satellites (lines) are needed to determine the coordinates (X, Y, Z) of the car. A fourth satellite is needed to specify the time. (Ngah, 2006)

1.3.2 GNSS Accuracy

The GNSS accuracy directly depends on the atomic clock performance. The atomic clock is one of vital components of each satellite. Atomic clock is based on atomic vibration, which provides a maximal error rate in nanoseconds, and consequently, it is very precise. Atomic clock is a substantial time unit. However, due to the high cost such a technology, receivers use the less expensive quartz clock. It should be noted, that precise time measure is a very highly required parameter, and thus if we have an error approximately 0.001, the positioning error will be around 320 km, which is really quite strong deviation, however, the GNSS receivers performs the accuracy of more or less than 10 meters. Thus, the maximal clock deviation examines in nanoseconds. (Ngah, 2006)

1.4 Testing Theory Overview

In my personal opinion, testing is an earnest of success in the development and implementation of any kind of system, and the GPS navigation systems are not exclusions so far. Testing process verifies whether the system meets the requirements and follows all the prescribed functions properly or not. Testing theory has quite a large spectrum of views, and in fact, the exact rules for testing do not exist. Nevertheless, I have examined several testing theory surveys, and I tend to adopt the software testing methodologies, which are mainly discussed in (S.M.K Quadri, 2010).

Testing is a process of verifying and validating that a software application or program meets the business and technical requirements that guided its design and development and works as expected and also identifies important errors or flaws categorized as per the severity level in the application that must be fixed. (S.M.K Quadri, 2010)

I would note the point, that there are many different testing approaches, for different type of software, and mainly it depends on level of testing. Of course, testing should be done properly and efficiently, according to the established time schedule and budget. Furthermore, due to the large amount of testing limits, it is impossible to get clear that every bug and error is detected and consequently fixed.

Therefore, here (S.M.K Quadri, 2010) are following principles, what can make testing efficient and relatively easier, and as a result – to approach the testing goal as maximal as possible, even though the limitations exist.

As said in (S.M.K Quadri, 2010),

Testing is a very important quality filter and needs to be planned taking into account its goals, principles and limitations. (S.M.K Quadri, 2010)

1.4.1 Testing Goals

Very good explanation:

A goal is a projected state of affairs that a person or system plans or intends to achieve. A goal has to be accomplishable and measurable. It is good if all goals are interrelated. In testing we can describe goals as intended outputs of the software testing process. Software testing has following goals (S.M.K Quadri, 2010):

1.4.1.1 Verification and validation

It would not be right to say that testing is done only to find faults. Faults will be found by everybody using the software. Testing is a quality control measure used to verify that a product works as desired. Software testing provides a status report of the actual product in comparison to product requirements (written and implicit). Testing process has to verify and validate whether the software fulfills conditions laid down for its release/use. Testing should reveal as many errors as possible in the software under test, check whether it meets its requirements and also bring it to an acceptable level of quality. (S.M.K Quadri, 2010)

1.4.1.2 Priority coverage

Exhaustive testing is impossible. We should perform tests efficiently and effectively, within budgetary and scheduling limitations. Therefore testing needs to assign effort reasonably and prioritize thoroughly. Generally every feature should be tested at least with one valid input case. We can also test input permutations, invalid input, and non-functional requirements depending upon the operational profile of software. Highly present and frequent use scenarios should have more coverage than infrequently encountered and insignificant scenarios. (S.M.K Quadri, 2010)

1.4.1.3 Traceable

In (S.M.K Quadri, 2010) given:

> *Documenting both the successes and failures helps in easing the process of testing. (S.M.K Quadri, 2010)*

It is clear, that the whole testing process including the testing result should be comprehensibly documented, and the integral points should be mentioned in the test report. Therefore, clear and optimal documentation form for testing should be applied to ease the overview and to understand the main things like how was testing done, what was the expectation of some test case and how acceptable the testing result is. This is very useful and very important, especially if the amount of test scenarios is large.

1.4.1.4 Deterministic

> *Problem detection should not be random in testing. We should know what are we doing, what are we targeting, what will be the possible outcome. Coverage criteria should expose all defects of a decided nature and priority. Also, afterward surfacing errors should be categorized as to which section in the coverage it would have occurred, and can thus present a definite cost in detecting such defects in future testing. Having clean insight into the process allows us to better estimate costs and to better direct the overall development. (S.M.K Quadri, 2010)*

1.4.2 Testing Principles

As given in (S.M.K Quadri, 2010):

> *A principle is an accepted rule or method for application in action that has to be, or can be desirably followed. Testing Principles offer general guidelines common for all testing which assists us in performing testing effectively and efficiently. Principles for software testing are:*
>
> - ***Test a program to try to make it fail**: Testing is the process of executing a program with the intent of finding errors. Our objective should be to demonstrate that a program has errors, and then only true value of testing can be accomplished. We should expose failures (as many as possible) to make testing process more effective.*
>
> - ***Start testing early**: Start as early as possible. This helps in fixing enormous errors in early stages of development, reduces the rework of finding the errors in the initial stages. Fixing errors at early phases cost less as compared to later phases. For example, if a problem in the requirements is found after releasing the product, then it would cost 10–100 times more to correct than if it had already been found by the requirements review. Figure 1 depicts the increase in cost of fixing bugs detected/fixed in later phases.*

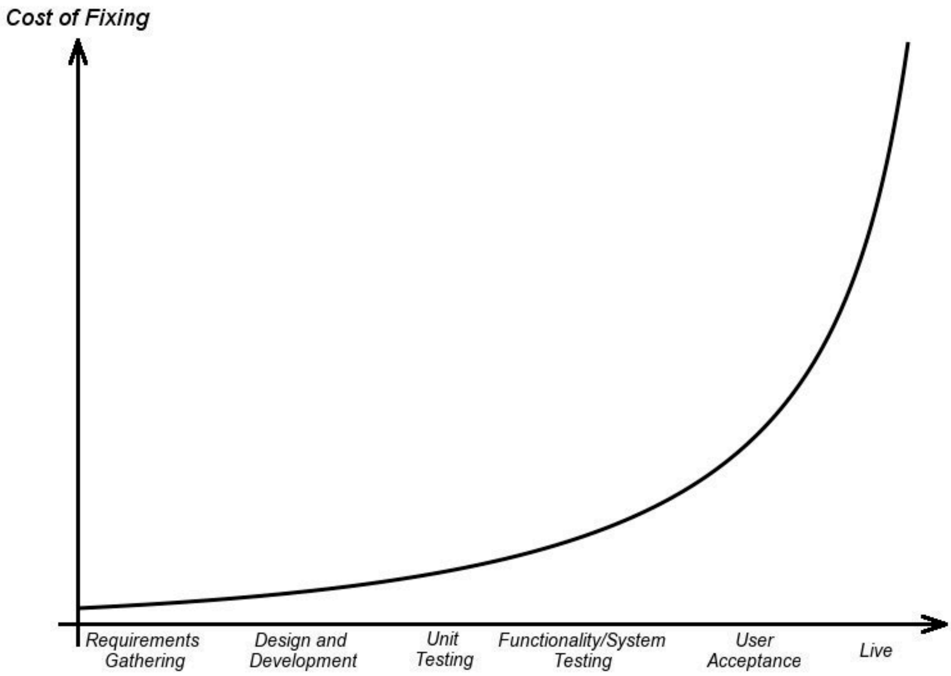


Figure 1.3: Cost of fixing bugs in different stages
Source: (S.M.K Quadri, 2010)

- ***Testing is context dependent****: Testing is done differently in different contexts. Testing should be appropriate and different for different points of time. (S.M.K Quadri, 2010)*

According to the previous quote, I would add very important point, that testing should be performed in different ways and different context. Testing time and testing conditions should vary as well. The main objective is to get sure that the software performance meets the user expectations.

Further important notes, taken from (S.M.K Quadri, 2010):

- ***Define test plan****: Test Plan usually describes test scope, test objectives, test strategy, test environment, deliverables of the test, risks and mitigation, schedule, levels of testing to be applied, methods, techniques and tools to be used. Test plan should efficiently meet the needs of an organization and clients as well. The testing is conducted in view of a specific purpose (test objective) which should be stated in measurable terms, for example test effectiveness, coverage criteria. Although the prime objective of testing is to find errors, a good testing strategy also assesses other quality characteristics such as portability, maintainability and usability.*

- ***Design effective test cases****: Complete and precise requirements are crucial for effective testing. User Requirements should be well known before test case design. Testing should be performed against those user requirements. The test case scenarios shall be written and scripted before testing begins. If you do not understand the user requirements and architecture of the product you are testing, then you will not be able to design test cases which will reveal more errors in short amount of time. A test case must consist of a description of the input data to the program and a precise description to the correct output of the program for that set of input data. A necessary part of test documentation is the specification of expected* *results, even if providing such results is impractical. These must be specified in a way that is measurable so that testing results are unambiguous.*

- ***Test for valid as well as invalid conditions****: we should also test system for invalid and unexpected inputs/conditions. Many errors are discovered when a program under test is used in some new and unexpected way and invalid input conditions seem to have higher error detection yield than do test cases for valid input conditions. Choose test inputs that possibly will uncover maximum faults by triggering failures.*

- ***Test evaluation****: We should have some criterion to decide whether a test is successful or not. If limited test cases are executed, the test oracle can be tester himself/herself who inspects and decides the conditions that makes test run successful. When test cases are quite high in number, automated oracles must be implemented to determine the success or failure of tests without manual intervention. One good criterion for test case evaluation is test effectiveness (number of errors it uncovers in given amount of time).*

- ***Error absence myth****: System that does not fulfill user requirements will not be usable even if it does not have any errors. Finding and fixing defects does not help if the system built does not fulfill the users' needs and expectations.*

- ***End of testing****: testing is an ongoing process, which is potentially endless but has to be stopped somewhere. Realistically, testing is a trade-off between budget, time and quality. The effort spent on testing should be correlated with the consequences of possible program errors. The possible factors for stopping testing are:*

    - *The risk in the software is under acceptable limit.*
    - *Coverage of code/functionality/requirements reaches a specified point.*
    - *Budgetary/scheduling limitations. (S.M.K Quadri, 2010)*

1.4.3 Testing Restrictions

Speaking about testing restrictions, I would notice one very important point. In (S.M.K Quadri, 2010) said, that restriction is a principle that limits the extent of something. Testing also has some limitations that should be taken into account to set realistic expectations about its benefits. (S.M.K Quadri, 2010)

Thus, I would mention the two following restrictions, which have to be taken into account:

- The task of testing is to find out the bugs, not to find that there are no any. The absence of bugs does not guarantee that the system, software, unit, whatever, does not have any bug so far.

- Testing cannot establish that a product functions properly under all conditions but can only establish that it does not function properly under specific conditions. (S.M.K Quadri, 2010)

1.4.4 Testing Forms

There are three main testing forms exist, and the each on is defined by the level of testing. These forms are White Box, Black Box and the Grey Box testing forms.

1.4.4.1 White Box Testing

As written in (Mohd. Ehmer Khan, 2012),

> *White box testing is the detailed investigation of internal logic and structure of the code. In white box testing it is necessary for a tester to have full knowledge of source code. (Mohd. Ehmer Khan, 2012)*

I would call it "Deep Testing". White box testing is such form of testing, when the tester has access to the internal (I would say – deepest) components of the system, software, whatever. White box testing can detect the implementation bugs. If it is a software or the application, tester should know well the programming language, which the program is written on, namely – tester should be able to understand the script and define which body of the script has bad performance, resulting the wrong functionality of the program.

Within the white box testing form, tester should be skilled enough and be familiar with several important types of testing technique, for example: control flow testing, branch testing, basis path testing, data flow testing, loop testing, etc. The more detailed information given in (Mohd. Ehmer Khan, 2012).

1.4.4.2 Black Box Testing

The explanation is given in (Mohd. Ehmer Khan, 2012), where is said, that:

> *Black box testing is a technique of testing without having any knowledge of the internal working of the application. It only examines the fundamental aspects of the system and has no or little relevance with the internal logical structure of the system. (Mohd. Ehmer Khan, 2012)*

While testing according to the black box testing form, there is no need to obtain any knowledge about "how do the infernal stuff working"; the tester's task is to know – what are the functions of the device and how should be the performance of the device. While doing black box testing, tester must be familiar with a software architecture and but to the testing level, tester does not have the access to the script.

Within a black box testing, tester should be quite familiar with the following important testing types: equivalence partitioning, boundary value analysis, fuzzing, cause-effect graph, orthogonal array testing, all pair testing and state transition testing. The more detailed information represented in (Mohd. Ehmer Khan, 2012).

1.4.4.3 Grey Box Testing Technique

Very comprehensive explanation:

> *White box + Black box = Grey box, it is a technique to test the application with limited knowledge of the internal working of an application and also has the knowledge of fundamental aspects of the system. (Mohd. Ehmer Khan, 2012)*

While testing using grey box testing form, tester should obtain the internal data knowledge as well as the algorithm for the purposes of creating specific test cases. The tester should be familiar with the architectural model of the system and obtain the skills in Unified Modeling Language (UML).

As written in (Mohd. Ehmer Khan, 2012),

> *In grey box testing the codes of two modules are studied (white box testing method) for the design of test cases and actual test are performed in the interfaces exposed (black box testing method). (Mohd. Ehmer Khan, 2012)*

Within the gray box testing form, the tester should be quite familiar with the following types of testing like orthogonal testing, matrix testing, regression testing and the pattern testing. The detailed information given in (Mohd. Ehmer Khan, 2012).

1.5 GPS Test Requirements

There are many in-car GPS Navigation device manufacturers exist today, and the each one is trying to sell better product with a best performance. The success of the manufacturer depends on the user satisfaction with a device performance and quality. To be better and to be more competitive in the market, the manufacturers tend to meet the user satisfaction in the highest level.

In (Agilent Technologies Inc. , 2010) there is clearly said, what are the main factors, which determine the level of satisfaction of the device users.

- *When a GPS device is turned on, how long is it until the position of the receiver is determined?*
- *When a weak or poor signal area is encountered, can the receiver still determine its position?*
- *If the signal is interrupted and then restored, how long does it take for the receiver to recover and resume calculating its position?*
- *Accuracy of the calculated location. (Agilent Technologies Inc. , 2010)*

Of course, there are plenty of different devices with a different capabilities such as user interface, voice communication, etc., and surely with the different price. However, the most critical issue, speaking about the navigation device quality, is the receiver performance.

For the all types of the navigation applications, the most important conditions are listed in (Agilent Technologies Inc. , 2010):

- *How accurately can a position or time be determined?*
- *How repeatable is the solution?*
- *How sensitive is the receiver to interference or jamming?*
- *How rapidly can the receiver report its position? (Agilent Technologies Inc. , 2010)*

1.6 Conclusion

In this Chapter, we have provided the general overview of the GNSS technology, containing car navigation technology, its possible functions, as well as the testing theory, its main principles and techniques. Consequently, we can define the proper testing methodology for the navigation device capability. From the GPS Test Requirements we can conclude, that the key parameter of the positioning performance is – cursor movement. Thus, we can assume in advance that the most appropriate testing methodology for the in-car positioning performance is the Black Box Testing, containing scenario-based testing. From this point forward, this thesis will focus mainly on testing of the navigation device performance.

2. Survey of the Navigation Device Performance Testing Approaches

2.1 Introduction

There are many different methodologies exist today for testing of the navigation device performance. Indeed, the navigation device performance testing is quite a wide field, and there are many different equipment for different purposes and ways exist. The existing technology for testing the navigation devices will be described farther in the next Chapter. However, let us now concentrate on the existing testing methodology for the navigation device performance testing. Having practice in *Skoda Auto Company*, I have obtained the experience with several approaches of testing of the in-car navigation device performance.

All these approaches are mainly laboratory-based. However, the main difference is that the testing can be done using simulated conditions and real-world conditions.

2.2 Lab-based Testing using Simulated Conditions

While using the lab-based approach, it is possible to simulate the conditions, which are close to the reality. Indeed, as said in (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009):

> *A lab-based simulation approach involves piecing together several distinct elements of the environment that affect GNSS performance. These elements, when defined and integrated properly, provide a representation of the real world. While simulation may not be able to reproduce the full richness of real world conditions, it is ideal for repeatable characterization and optimization of receiver performance, especially when combined with other methods of testing. (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009)*

The efficient simulation needs a comprehension of the key elements, which affect the positioning performance of the navigation device. These elements can be separately designed with help of GSS6700 Multi-GNSS Simulator and using the SimGEN software, which is actually comes with the equipment together.

In this Chapter, the basic principles of the lab-based testing approach using the simulated conditions are represented. However, the more detailed information is given in (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009).

The key elements for the lab simulation are:

- *Satellite Constellation Simulation*
- *Environment and Device Antenna Characteristics*
- *Network Simulation*
- *Automation Software (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009)*

2.2.1 Satellite Constellation Simulation

As described in (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009),

The simulation of the GNSS constellation allows for the simulation of moving satellites and a moving receiver. In the listed literature there are the key elements described as well:

- ***Location**: the physical location of the receiver, expressed in terms of latitude, longitude and altitude. If the receiver is in motion, a rout needs to be specified.*

- ***Device Motion**: the path taken by a device. This is needed in order to simulate a device that is moving (as opposed to staying in one place)*

- ***Date, Time, and Length of Simulation**: directly related to the visibility of satellites for a given location. The date and time can be specified to be any point in the past, present, or the future.*

- ***Satellite Orbits**: the actual satellites available in the sky for a given date, time and location. They can be obtained from YUMA data, maintained by the US Navy.*

- ***GNSS Satellite Constellations**: the satellite constellations being simulated. Although GPS is still by far the most common, GLONASS is gaining momentum and others are coming, including Galileo and QZSS. (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009)*

2.2.2 Environmental and Device Antenna Characteristics

As given in the same source:

GNSS signals are transmitted wirelessly and are therefore subject to many environmental and device antenna factors. These elements are responsible for the majority of performance variations from device to device. Since they are also the most difficult to simulate correctly, they need special attention:

- *Multi-path/Fading Emulation: a suitable multi-path model must be created, based on the geographic elements of the given location, such as buildings, trees, rocks, etc.*

- *Atmospheric Modeling: simulation of atmospheric conditions is required to test the ability of the GNSS system to compensate for resultant navigation error.*

- *Signal Obscuration: the attenuating effect of physical elements, such as buildings and trees, on incoming GNSS signals from the sky must be accounted for.*

- *Device Antenna Pattern: as more capability gets crammed into shrinking mobile devices, the GNSS antenna performance becomes a critical factor. To simulate real-world performance in the lab, antenna pattern must be characterized and simulated. (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009)*

2.2.3 Network Simulation

As said in the application note (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009):

The A-GNSS also requires to simulate the cellular network and the assistance data that it provides to the mobile device. The key elements are:

- *Flexible Simulation: The network simulator must be capable of supporting all the cellular technologies required by the device under test, whether GSM/GPRS, UMTS and/or CDMA. The minimum requirements are typically:*

    - *Support for 3G and 2G location protocols (such as RRC/RPLP).*
    - *Ability to initiate common signaling procedures (such as voice calls and supplementary services).*
    - *Flexibility to modify key protocol information elements (such as desired horizontal accuracy and desired response time).*

- *SMLC/PDE Simulation: this is required to control the assistance data delivery and for position calculation. It is important that this simulation be flexible so that the configuration can be aligned as closely as possible with the real network. (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009)*

2.2.4 Automation Software

As described in (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009),

> *This ties everything together and controls the test settings, procedure, collection and organization of data, and representation of results. (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009)*
>
> - *Simulation Parameters: easy configuration of key parameters of the simulation components mentioned below. These may include:*
>   - *Delivery of assistance data*
>   - *Specifying initial location*
>   - *Specifying the number of measurements*
>   - *Protocol and call flow section*
>
> - *Test Scenario: determines the timing and flow of test execution. A properly designed test scenario should be able to replicate real-world scenarios much faster than is possible with field test or record-and-playback approaches.*
>
> - *Test Results: includes analysis yielding common metric and KPIs used to assess the A-GNSS device performance, as well as detailed protocol logs and positioning data:*
>   - *Sigma 1 and 2 calculations for horizontal error*
>   - *Yield calculations*
>   - *Time to first fix*
>   - *Graphical representation of data, including CDF and PDF (Spirent Communications plc., Testing Real-World A-GNSS Performance in the Lab using ULTS, 2009)*

2.2.5 Additional Notes

From my personal experience, I should highlight one very important point. The GSS6700 Multi-GNSS Simulator, with help of SimGEN software, is able to work with the GIS files, representing the start-point, end-point and the driving route. SimGEN works with NMEA file format itself, however, using additional tools, such as “Schoko-Loko-Viewer” or any another, it is easy to convert any file, containing such an information into NMEA format. I personally used to use such GIS tools like Google Maps, Mapy.Cz and Google Earth. For example, using Google Maps, it is possible to generate KML file, using Mapy.Cz – KML and GPX files. However, the end file type, which should be loaded into the system, should be NMEA.

The listed file types – KML, GPX, NMEA – the map-based graphical representation of GIS data, which contain the necessary elements, in our case they are: start-point, end-point and the driving path itself. Thus, we should only set up the appropriate simulation parameters for testing listed previously, and arrange a testing of the navigation device performance during a virtual test drive, for example, in London, while locating in Prague.

Consequently, when the appropriate vehicle movement, conditions and environment are simulated, the testing is undertaken using against the different tested navigation device units. This can be repeatable with many different devices, as well as the recorded data. This principle is discussed further.

2.3 Lab-based Testing Approach using Real-World Conditions

In the previous section, we have observed such a possibility as a simulation of GNSS constellation as well as simulation of environmental conditions for testing of the navigation device. We can simulate the GNSS signal quality for the each satellite, signal obstructions, which cause multipath effect and so on, simply saying – the proper environment which is necessary for the navigation device functioning, including the conditions which affect its functioning. Apart of the simulation, there is a precious possibility to record a real-world data as well. We call this "live environment test". Such data is unique and not repeatable.

There are many various equipment exist today for the recording such a data. Mainly, this data is recorded using a GPS antenna, vehicle and the CAN technology, and of course the record & replay device itself. From my experience in *Skoda Auto Company*, I have used the Spirent GSS6400 Record & Playback System, which is described in detail in the next Chapter. The necessary condition for the recording is – vehicle, recording unit and the planned driving route. It is possible to use a windshield camera, as well as two cameras to capture the installed in-car navigation device display and the vehicle's control panel to record the actual movement trajectory and test the already installed device as well. The purpose is to have as much clear picture as possible, which is also very used for the testing in real drive. This approach is shortly discussed further.

Of course, the test plan should be arranged with respect to the necessity of testing. It can be different urban environmental conditions, such as bypasses on highways, or streets with surrounding tall buildings. In fact, using this testing approach it is easier to create a test plane, and testers usually use different GIS tools such as Google Earth etc. to have a clear picture about the recording driving route, its surrounding and environment.

However, while recording the real-world environment, the recording device records everything what is necessary: the actual satellites constellation, satellites signal power, steering wheel rotation, vehicle speed and so on. All this is being proceeded using CAN technology (the recording device must be connected to the vehicle's CAN). Thus, when we have obtained the whole picture and the recorded data, it is very easy to replay that in the laboratory conditions for testing of one or more navigation device units.

2.4 Real Drive Testing Approach

The real test drive approach is very used in *Skoda Auto Company*. As I have mentioned previously, the three camera is installed: one camera – on to the windshield of the car to capture the driving route, the second camera – in front of the meter panel, and the third one – in front the on-board computer to capture the positioning performance.

With help of these cameras, it is possible to test the in-car navigation device performance after recording and examination the captured videos in laboratory.

During a real test drive, all the videos should be recorded synchronously, as well as synchronously should be replayed in the laboratory. And based on these video files, the tester makes a decision, whether the test is passed or failed.

The key components for such type of testing are the three installed cameras and the laptop, with previously installed specific software to operate the synchronous recording process.

The examples of captured data are depicted below:

Figure 2.1: Vehicle meter panel.
Source link is [here](#).

Figure 2.2: Vehicle on-board.
Source link is [here](#).

Figure 2.3: Vehicle driving trajectory.
Source link is [here](#).

While having the synchronous video files of this data, it is easy to detect the bugs and errors in navigation device performance visually.

The approach has its advantages and disadvantages. The significant advantage is that the navigation device is being tested in real conditions and in real usage, however using this approach it is possible to test one unit only, which is a considerable disadvantage. Nevertheless, this gap has been filled with help of the Spirent GSS6400 RPS, which gives us the opportunity to record the all necessary environmental data and test several devices replaying the scenarios in the laboratory conditions.

2.5 Conclusion

Of course, all these testing approaches have their features and purposes. As we defined previously, these approaches are mainly laboratory based, as the testing itself is being done in laboratory. It is wrong to say that one approach is better than another one, but let us highlight the main features of all of them:

Lab-based testing using simulated conditions	Lab-based testing approach using real-world conditions	Real drive testing approach
Real-world environment simulation	Real-world environment record and replay	Real-world environment
GIS source is used: - for the overview of the whole picture - to perform a testing in locations far from the laboratory	GIS source is used for the overview for the whole picture only	GIS source is used for the overview for the whole picture only
Setting of the environmental conditions required	Connection to the vehicle and the environmental conditions recording test drive required	Installation of the three cameras required
Used special computer (supplied with the technology equipment) with specific software (SimGEN, supplied with the simulation equipment, etc.)	Used any computer with installed software (CANoe, CANalyzer etc.)	Used any computer with specific software to record and replay video files.
Not portable	Portable	Portable

Table 2.1: Short features overview of the lab-based testing and a real test drive approaches.

All these approaches are very useful, and the usage of some of them directly depends on the concrete testing task, testing plan, time schedule, financial aspects and the other different aspects. However, the key point of usage is location, which is directly related to the economical and requirement issues. For example, situating in Mlada Boleslav, it is very expensive to fly, for example, to New York and make recordings of the real-world environment for testing of the navigation device performance in multipath conditions or test the device in real test drive. However, we can simulate such an environment using lab-based approach with simulated conditions and the appropriate GIS file, containing start-point, end-point and the driving path itself. On the other hand, if we need to test the positioning performance in narrow streets in cities, or narrow intersection, we can easily drive to Prague, record the data and perform all the necessary measurements in the laboratory. Here is another, not less-important issue so far: GNSS technologies. This issue includes the presence of different positioning systems (GPS, GLONASS, Galileo, etc.), the satellites signal actual quality, constellation, satellites position in the space, and so on, in spite of the 99.9% covering. Nevertheless, testing of the navigation device performance is very narrow specialization, and at the same time – very important issue today, and with the course of time and appearance of new navigation technologies, there are many new different testing approaches and technologies appear.

3. Survey of Existing Technology for Testing of the Navigation Device Performance

During my internship in Technical Development Department at *Skoda Auto Company*, I have obtained the precious experience working with two different technologies.

In this Chapter, I will provide the short description of these both technologies, as well as its specific software, taking and combining the information from the relevant sources, brochures and user guides.

3.1 Spirent GSS6700 Multi-GNSS Constellation Simulator System

As given in (Spirent Communications plc., GSS6700 GNSS Simulator Product Brochure, 2013):

> *The GSS6700 Multi-GNSS Simulator from Spirent offers simultaneous coherent GPS/SBAS, GLONASS, BeiDou and Galileo L1 signals from a single test scenario. 12 channels of each enabled constellation provide ample signals for a wide range of development and integration tasks. GSS6700 Multi-GNSS Constellation Simulator provides an easy-to-use but powerful solution for GNSS testing which can grow with our evolving needs. (Spirent Communications plc., GSS6700 GNSS Simulator Product Brochure, 2013)*

Figure 3.1: GSS6700 Multi-GNSS Simulation System
Source: (Spirent Communications plc., GSS6700 GNSS Simulator Product Brochure, 2013)

The GSS6700 Multi-GNSS Simulator from Spirent has been developed for R&D, integration and verification testing of devices that use commercial GPS, GLONASS and/or Galileo signals. (Spirent Communications plc., GSS6700 GNSS Simulator Product Brochure, 2013). The GSS6700 has following key features:

- *GPS/SBAS, GLONASS, BeiDou and Galileo supported*
- *12 independent channels of each constellation*
- *Field upgrade minimizes downtime as your needs grow*
- *Portable scenarios facilitate collaboration*
- *Class leading accuracy, fidelity and reliability*
- *Save and compare DUT data*
- *Receiver antenna pattern modelling*
- *Import motion from logged NMEA*
- *Selection of scenario generation and simulation control software available:*
    - ***SimREPLAY**: Interactive run time control with assistance data extract*
    - ***SimREPLAYplus**: Import remote trajectory, edit time, date and position*
    - ***SimGEN**: Comprehensive constellation, propagation and vehicle modelling with flexible data capture (Spirent Communications plc., GSS6700 GNSS Simulator Product Brochure, 2013)*

A full range of hardware integration signals are provided including 1PPS in / out, 10MHz in / out and hardware trigger. (Spirent Communications plc., GSS6700 GNSS Simulator Product Brochure, 2013)

Output Frequency	
GPS L1 C/A	1575.42MHz
GLONASS L1 C/A (Ch0)	1602MHz
BeiDou-2 B1	1561.098MHz
Galileo E1 OS CBOC	1575.42MHz

Signal Quality	
Spurious (Max)	-30dBc
Harmonics (Max)	-35dBc
Phase Noise (Max)	0.02 rad RMS
Frequency Stability	±5 x 10-10

Signal Accuracy	
Pseudorange	±0.002m RMS
Pseudorange rate	±0.001m/s RMS
Interchannel bias	zero

Signal Dynamics	
Max Velocity*	±600m/s
Max Acceleration*	±45m/s^2
Max Jerk*	±50m/s^3

Signal Level	
GPS/SBAS nominal	-130dBm
GLONASS nominal	-131dBm
BeiDou nominal	-130dBm
Galileo nominal	-128.5dBm
Level control range	+15 / -20 dB
Level control resolution	0.1dB
Level control accuracy	±0.5dB RSS

**Higher dynamics can be achieved at a slight reduction in accuracy.*

Physical and Electrical (Signal Generator)	
Size (mm)	449x383x89 (WxDxH)
Weight	6.5 kg
Power	110/240 V AC 50/60Hz

Table 3.1: GSS6700 Multi-GNSS Simulator Specification
Sources: (Spirent Communications plc., GSS6700 GNSS Simulator Product Brochure, 2013), (Spirent Communications plc., Spirent GSS6700 with SimGEN, 2009)

The GSS6700 Multi-GNSS Simulator is supported by a range of scenario generation and simulator control software packages including Spirent's feature-rich SimGEN. Each software package has a range of capabilities from the all-inclusive SimGEN to packages more suited to production or integration / verification applications.

A full range of hardware integration signals are provided for enhanced testing flexibility including: 1PPS input and output, external reference clock input, 10 MHz output and hardware trigger. (Spirent Communications plc., GSS6700 GNSS Simulator Product Brochure, 2013)

The figure below represents the global coverage of GNSS signal simulation opportunity.

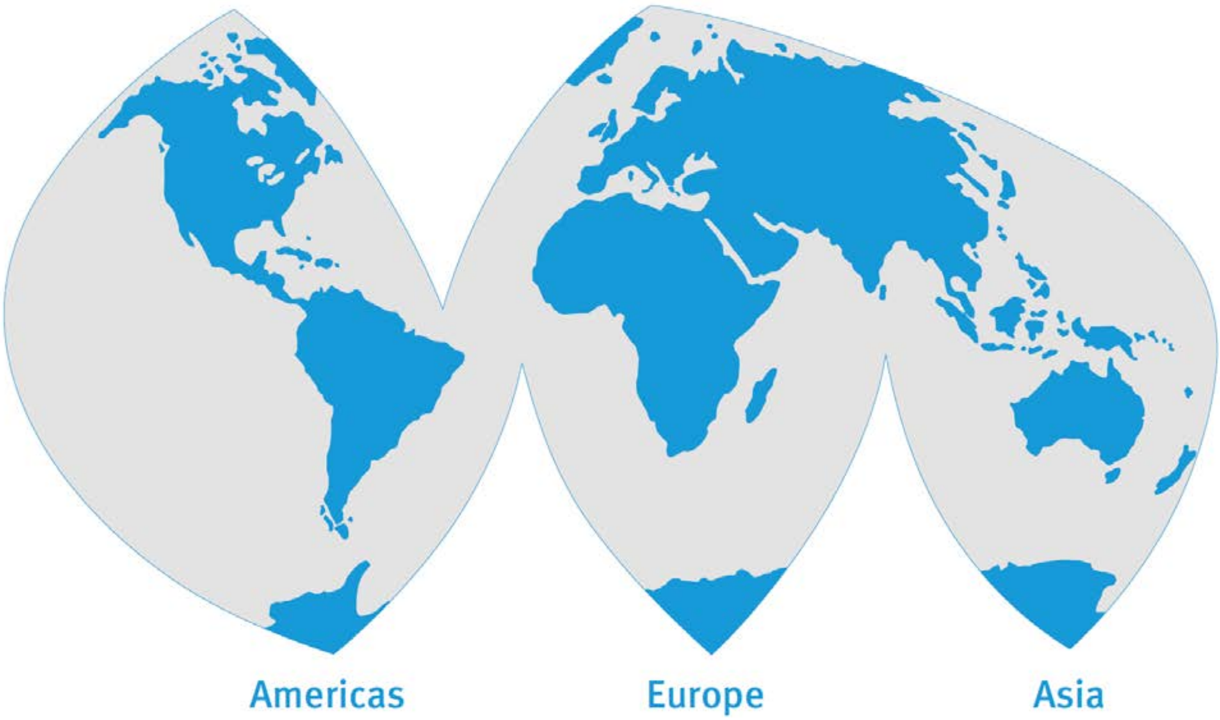


Figure 3.2: Global Coverage.
Source: (Spirent Communications plc., GSS6700 GNSS Simulator Product Brochure, 2013)

As I have mentioned previously, the specific software is available and being supplied with the GSS6700 GNSS simulator. This software is the integral component of the described equipment. The detailed information about *SimGEN*, *SimREPLAY* and *SimREPLAYplus* software is given in (Spirent Communications plc., Spirent GSS6700 with SimGEN, 2009), (Spirent Communications plc., SIMGEN SOFTWARE USER MANUAL, 2012) and (Spirent Communications plc., Spirent GSS6700 with SimREPLAY, 2009).

3.2 Spirent GSS6400 Record Playback System

There a couple of different sources exist for good explanation for the Spirent GSS6400 Record and Playback System, as well as its introduction. There are some of them:

> *A critical challenge to achieving the performance and quality of experience expectations of today's navigation and positioning systems is accurately testing user devices in complex and dynamic real-world scenarios. To address this issue Spirent Communications has introduced the GSS6400 Record and Playback System, a solution designed to improve device performance while reducing the need for and cost of field-testing. (A "Real-World" Navigation and Positioning Testing, 2011)*
>
> *With the Spirent GSS6400 Record Playback System, it's simple and quick to record real GPS, GLONASS and QZSS signals. GSS6400 also supports record playback of signals from augmentation systems such as WAAS and EGNOS. Once captured, the Record Playback System can be used in the laboratory to replay the captured environment time and time again to the device under test. (Spirent Communications plc., Spirent GSS6400 RPS, 2012)*

Also good notes:

> *The GSS6400 is designed to capture complex environments efficiently and with the fidelity to ensure playback that is truly representative of challenging real world conditions. The unit is appropriate for a wide range of test applications, from optimization of GNSS chipset performance in difficult environments through to testing mapping applications and location enabled services. (A "Real-World" Navigation and Positioning Testing, 2011)*

Very useful information about data sets given in (Data Sets Launched for GSS6400, 2012):

> *The data sets encompass a variety of recorded scenarios, ranging from drive through multi-level roads, tunnels and car parks to drive through in downtown areas with highrise buildings. The real-world effects such as atmospheric, environmental (multipath, interference, terrain obscuration) are captured, making use of data sets for debugging issues, performance tuning and optimization of GNSS receivers.*
>
> *The data sets encompass a variety of recorded scenarios, ranging from drive through multi-level roads, tunnels and car parks to drive through in downtown areas with highrise buildings. The real-world effects such as atmospheric, environmental (multipath, interference, terrain obscuration) are captured, making use of data sets for debugging issues, performance tuning and optimization of GNSS receivers.*
>
> *The data sets are accompanied by reference "truth" position data in time-stamped NMEA format, recorded using a high-grade inertial (INS) aided receiver. With the data sets, customers will able to save time and maximize their test coverage. (Data Sets Launched for GSS6400, 2012)*

Figure 3.3: GSS6400 Record & Playback System
Source: *(Spirent Communications plc., Spirent GSS6400 RPS, 2012)*

The key features of the GSS6400 Record & Playback System are represented below.

Recorder	Playback
Internal battery and vehicle DC power adapter	Single touch playback
2-bit quantization	Browser control over network
Internal HDD	OCXO for frequency stability
30 hours record time	Power level control
Single touch record	Multiple file playback
Multiple file record	Start at any point in a file
Event markers	

Table 3.2: Key Features of Recorder and Playback
Source: *(Spirent Communications plc., Spirent GSS6400 RPS, 2012)*

The GSS6400 records sampled GPS and GLONASS L1 signal data onto the internal 500GB hard drive. Sampled data is written at speeds of up to 4 MB/s onto the hard drive. The GSS6400 uses 2-bit signal sampling to achieve signal dynamic range suitable for testing high sensitivity GNSS receivers. Single bit sampling loses up to 3db of signal in the digitization process. Two-bit quantisation loses only around 0.5 dB. During playback the system upconverts the sampled data to the 1575.42MHz for GPS and 1600MHz for GLONASS L1 frequencies. An OCXO is used to provide a stable L1 frequency and accurate data playback, so preserving the code/carrier relationships of the original recorded signal. The OCXO can be locked to an external 10 MHz source for increased accuracy. (Spirent Communications plc., Spirent GSS6400 RPS, 2012)

The table below represents the specification of the GSS6400 Record Playback System.

Specification	
Quantization	2 bit
Output attenuator	31dB, 1dB steps
Internal HDD	500GB
Record capacity	30 hours
Playback frequency	1575.42MHz GPS 1600MHz GLONASS
Power	12-16v DC 90-260v AC adapter supplied Internal battery supplied
Frequency reference	Internal OCXO for playback External 10 MHz in
Antenna	Active antenna supplied
Case	Hard case supplied

Table 3.3: GSS6400 Record Playback System specification
Source: (Spirent Communications plc., Spirent GSS6400 RPS, 2012)

As said in the equipment overview brochure (Spirent Communications plc., Spirent GSS6400 RPS, 2012):

The GSS6400 records and replays serial data from a wide range of data sources as well. Inertial sensors, DR sensors, reference receivers and 1pps signals can be recorded coherently with the GNSS embedded within the data file to guarantee synchronization. Additionally, the GSS6400 can log serial data into separate files for subsequent analysis or post processing. NMEA logs or Wi-Fi war-drive data are amongst the types of file that the GSS6400 can record.

Finally, I have to note, that it is easy to operate the system from the front panel, as well as from the laptop via LAN connection, using internet browser and IP address of the RPS. (Spirent Communications plc., Spirent GSS6400 RPS, 2012)

4. Proposal of Different Scenarios for Testing of the Navigation Device Performance

4.1 Introduction

The main task of scenarios is to represent the functionality of the system and system behavior, based on user operations and needs. This is mainly used in modern software development to meet user requirements. Although there are many different strategies exist for testing, there is one general and important point for all the methodologies, which helps to provide a comprehension and clear overview, which is the formal languages. Nevertheless, there is a big gap between a theory and practice often exists due to the reasons, clearly described in (Johannes Ryser, 1999).

- ***Lack in planning / time and cost pressure****: In real-world projects tests are conducted under immense time and cost pressure, as often the project at the end of the development process is behind schedule and over budget already. Detecting faults causes additional delays. As a consequence, both test preparation and execution are frequently performed only superficially. Cost and time needed for testing are hard to be estimated with reasonable accuracy. Moreover, testing is often insufficiently planned for and not enough time and resources are allocated for testing.*

- ***Lack of test documentation****: Tests are not properly prepared, no test plans are developed and tests are not documented*

- ***Drudgery****: Testing and test case development are tedious, wearisome, repetitious, error-prone and time-consuming activities which prompt fatigue and inattentive work, even if sound testing strategies and methods are applied.*

- ***Lack of tool support****: Extended tool support and more especially automatic test case generation is restricted to systems which are formally specified.*

- ***Formal languages / specific testing languages required****: Many test methods use formal specification languages or specific testing languages (thus requiring special training and education). Their application is extremely costly, they are difficult to apply and/or can only be applied to limited problems or very specific domains.*

- ***Lack of specific measures and data to evaluate testing quality****: In most projects only little testing data (error statistics, coverage measurements, and so on) is collected during testing or available from other projects. Because of missing data only little can be said about the benefits and economics of testing, different approaches can not be compared and processes can hardly be improved. The quality of tests, and thus to some extent of the product, is often not assessed. Furthermore, the missing data further aggravates the problem of accurate test planning and allocation of the necessary resources. (Johannes Ryser, 1999)*

In spite of that, testing mostly done in the last resort just to show that the system is working, not to discover the bugs in the tested system but to show that the system meets the requirements. This mainly done by the several reasons:

- *Testing is performed in the last stage*
- *Testing methodology is not integrated with a system development methodology*
- *Test cases have not been designed properly*

Of course, testing is not very easy task. Testing cannot be even simply automated: even using automated testing, it is impossible to meet the required testing coverage with respect to time and cost, using natural language. It has specific issues need to be solved, and the specific tool support is required. (Johannes Ryser, 1999)

4.2 Scenario-Based Approach

The proposal approach is the practice-oriented scenario-based approach to support systematic test case development, that utilizes early artifacts of the development process in later phases again, in order to realize synergies between closely related phases of system analysis and system test. (Johannes Ryser, 1999)

The main instructions of the scenario-based testing approach are described in (Johannes Ryser, 1999) in detail. These are:

- *Use natural language scenarios.*
- *Uncover ambiguities, contradictions, omissions, impreciseness and vagueness.*
- *Annotate the narrative scenarios and/or the statecharts.*
- *Systematically derive test cases. (Johannes Ryser, 1999)*

This is a very specific approach, and for the each system of software, the test scenario should be specific, and should correspond the appropriate development methodology. However, theoretically it seems easy.

4.2.1 Scenarios

Firstly, let us have a look at the basic definitions of the Test Scenario. Cem Karner has defined:

> ***Scenario*** is a hypothetical story, used to help a person think through a complex problem or system. (Kaner, June, 2003)

However, there are many different definitions and considerations about the nomenclature listed above exist, and there are many different sources. Depending on the system and the testing itself, we could also consider test scenario as a set of test cases.

As described in (Kaner, June, 2003), a scenario test has five key characteristics, which are:

- *Story*: writing a scenario involves writing a *story*. (Kaner, June, 2003)

This is such a kind of art, but this should contain a real story. Story in this content means something, what relates to the tested system, including (especially) such points, what can take a real place in the story.

- *Motivating*: scenario test is *motivating* if a stakeholder with influence wants the program to pass the test. A highly motivating bug report might consider the impact of possible failure on the user's business or, for example, software developer. (Kaner, June, 2003)

- *Credible*: scenario is credible if a stakeholder with influence believes it will probably happen. Sometimes we can establish credibility simply by referring to a requirements specification. In many projects, though, we will not have these specs or they will not cover the situation. (Kaner, June, 2003)

- *Complex*: as said in (Kaner, June, 2003), a complex story involves many feature. Of course, depending on the testing methodology, it is possible to create very simple scenario with one feature only as well. However, applying some methodologies it is possible to create one feature, which is more focused on developing power in simple situations. The strength of the scenario is that it can help to discover problems in the relationship among the features. (Kaner, June, 2003)

- *Easy to evaluate*: every test result should be easy to evaluate. However, the more complex the test, the more likely that the tester will accept a plausible-looking result as correct. (Kaner, June, 2003)

Cem Kaner in his work (Kaner, June, 2003) offers the following ways to create good testing scenarios:

- *Write life histories for objects in the system.*
- *List possible users, analyze their interests and objectives*.
- *Consider disfavored users: How do they want to abuse the system*?
- *List "system events": How foes the system handle then*?
- *List "special events": What accommodations does the system make for these*?
- *List benefits and create end-to-end tasks to check them.*
- *Interview users about challenges and failures of the old system.*
- *Work alongside users to see how the work and what they do.*
- *Read about what systems like this are supposed to do.*
- *Study complaints about the predecessor to this system or its competitors*.
- *Create a mock business. Treat it as real and process its data.*
- *Try converting real-life data from a competing or predecessor application*. (Kaner, June, 2003)

One more useful note:

Designing scenario tests is much like doing a requirements analysis, but is not requirements analysis. They rely on similar information but use it differently. (Kaner, June, 2003)

- *The requirements analyst tries to foster agreement about the system to be built. The tester exploits disagreements to predict problems with the system.*

- *The tester does not have to reach conclusions or make recommendations about how the product should work. The task is to expose credible concerns to the stakeholders.*

- *The tester does not have to make the product design tradeoffs. The tester exposes the consequences of those tradeoffs, especially unanticipated or more serious consequences than expected.*

- *The tester does not have to respect prior agreements.*

- *The scenario tester's work need not be exhaustive, just useful. (Kaner, June, 2003)*

4.2.2 Test Cases

There are many different definitions about the test cases. For example:

> *Test case is a set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement. (IEEE Standard 610, 1990)*
>
> *Test cases are the specific inputs that you will try and the procedures that you will follow when you test the software. (Patton, 2001)*

Cem Kaner in his work "What Is a Good Test Case?" (Cem Kaner, May, 2003) defined a test case as follows:

> *Test case is a question that you ask of the program. The point of running the test is to gain information, for example whether the program will pass or fail the test. It may or may not be specified in great procedural detail, as long as it is clear what is the idea of the test and how to apply that idea to some specific aspect (feature, for example) of the product. (Cem Kaner, May, 2003)*

Cem Kaner has also specified the following objectives of test cases:

- *Find defects*
- *Maximize bud count*
- *Block premature product releases*
- *Help managers make ship / no-ship decisions*
- *Minimize technical support costs*
- *Asses conformance to specification*
- *Conform to regulations*
- *Minimize safety-related lawsuit risk*
- *Find safe scenarios for use of the product*
- *Assess quality*
- *Correctness of product verification*
- *Assure quality*

More detailed explanation of test cases objectives given in (Cem Kaner, May, 2003).

Even within these objectives, tests can be good in many different ways. For example, we might say that one test is better than another if it is:

- *More powerful*
- *More likely to yield significant*
- *More credible*
- *Representative of events more likely to be encountered by the costumer*
- *Easier to evaluate*
- *More useful for troubleshooting*
- *More informative*
- *Appropriately complex* (Cem Kaner, May, 2003)

In spite of the presence of many different testing styles I would like to approach mainly to the testing of the positioning performance and list those testing styles, I would say – dominant styles, which relate straight to the black box testing technique. Thus, there are eleven key testing styles given in (Cem Kaner, May, 2003), which are mainly used in black box testing:

- *Function testing: testing of every single function in isolation and/or testing the functions all together or in different options.*

- *Domain testing: the essence of this type of testing is sampling. Tester reduce a massive set to a small group by partitioning the set into subsets and picking up one or more representatives from each subset.*

- *Specification-based testing: checks the system of software on every point shown in reference document. This can be design specification, list of requirements, user interface description or user manual.*

- *Risk-based testing: tend to carry high information value while testing for a problem that by some reason might exist in the software or a system.*

- *Stress testing: as defined in (IEEE Standard 610, 1990), stress testing is a testing, conducted to evaluate a system at or beyond the limits of its specified with the goal of causing the system to fail.*

- *Regression testing: design, develop and save tests with the intent of regularly reusing them, Repeat the tests after making changes to the program.*

- *User testing: done by users while using the software or a system. Simply saying – "Beta" version.*

- *State-model based testing: we model the visible behavior of the program as a state machine and drive the program through the state transitions, checking for conformance to predictions from the model.*

- *High volume automated testing: high-volume testing is a diverse grouping. The essence of it is that the structure of this type of testing is designed by a person, but the individual test cases are developed, executed, and interpreted by the computer, which flags suspected failures for human review.*

- *Exploratory testing: exploratory testing is not purely spontaneous. The tester might do extensive research, such as studying competitive products, failure histories of this and analogous products, interviewing programmers and users, reading specifications, and working with the product. (Cem Kaner, May, 2003)*

For more detail explanation of the testing styles, related to the black box testing form, see (Cem Kaner, May, 2003).

4.3 GPS Navigation Device Performance Testing Scenarios

In the previous chapter, I have described several ways for testing according scenario-based testing approach. As it was mentioned before, there is no simple formula or prescription for generating "good" test cases. However, let us concentrate now on the concrete field – Testing of the GPS Navigation Device Capability.

We live in such a period, when millions of people travel around the Earth. Taking into account the fact of globalization, the last couple tens of years, we observe a rapid growth in popularity in the highest buildings in the majority of countries. For example, if we compare how was looking Dubai in the beginning of ninetieth and how does this city look today, it is incomparably. There are a lot of tall buildings, as well as the highest skyscraper in the world, which called "Burj Khalifa"; its height is around 500 m! And this is going on almost in every developing country. Alongside with this, we can also add the necessity in the underground roads (tunnels), as well as parking. Nevertheless, this cause a serious challenge for the navigation devices.

Because GPS satellites transmit their signals with the equivalent power of a 30 watt light bulb from a distance of 20,000 km, they arrive with typical signal strength, in the best case, of –120 dBm (1×10^{-15} Watts). This is millions of times weaker than a typical home WiFi signal! These signals can easily be degraded by an additional 20-30 dB in city conditions, or blocked completely, further impacting the accuracy of GPS navigation. (Etienne Favey, 2010)

4.3.1 Challenges to Urban Navigation

Thinking of test cases creation for GPS Navigation Device testing, first of all we need to think about problems, which could be user facing with, i.e. – real world scenarios.

As we know from the Chapter 1, there are at least four GNSS visible satellites required for the navigation device to determine the vehicle location, and generally for normal functionality. Without that, the full and even proper position performance is not possible.

As mentioned in (Etienne Favey, 2010), the several significant barriers to already weak GPS signals include:

- ***Tunnels** and **parking garages**, the worst case scenario where GPS signals are completely blocked*

- ***Multi-level roads**, **overpasses** and **bridges** which can confuse GPS receiver (which road am I on?)*

- *Tall buildings, which can reflect GPS signals (**multipath propagation**) fooling a GPS receiver into thinking it is somewhere else. (Etienne Favey, 2010)*

Authors have specified in (Etienne Favey, 2010), what outcomes all this can lead to. The end result of these obstructions range from minor irritation to major problem:

- *For drivers unfamiliar with the area, navigation can be intermittent or fail altogether, especially when exiting tunnels and park garages, resulting in irritation, wasted time and fuel*
- *For commuters who may already know their way, traffic-jam avoidance services can be rendered useless*

- *For public transportation systems such as buses and trams, the loss of expected arrival times poses an inconvenience to thousands of commuters*

- *For commercial transportation services such as taxis, freight and logistics companies, the loss of positional overview and security of transported goods can have major financial ramifications*

- *Emergency vehicles such as police, fire and ambulance services are prevented from reaching the location of an incident quickly*

- *Systems used for automatic road-pricing or pay-as-you-drive insurance have insufficient data to charge for road usage. (Etienne Favey, 2010)*

To avoid these risks, testing should be done with as much as possible detailed test scenarios and test cases.

4.3.2 Proposal of Input Parameters for the Scenario Testing

As it was said before, scenario is a hypothetical story, used to help a person think through a complex problem or system, simply saying – it is a set of test cases. Test cases for GPS Device capability testing can vary from very simple (for example – positioning on *Area without obstacles*) to very challenging (for example – positioning in *underground garage,* which contains four levels).

Based on theoretical part of this chapter, I propose the following real-world parameters for test scenarios as a key input:

- **Area without obstacles**: very simple, not challenging parameter with 100% GPS signal reception.

- **Narrow streets in cities**: it can be the old narrow streets in historical parts of cities like Prague, Athens, Rome, etc. Simply speaking, roads with 1 lane, with 3-3.5 m of length, narrow streets enclosed by buildings.

- **Parallel roads**: can be challenging and can confuse the GPS receiver.

I have divided this parameter by distances from the center of the road to the center of another road. *From the most challenging situation to less challenging*. The same principle is used for the rest of parameters.

- **Less than 15 meters**, which I consider the most challenging
- **From 15 to 30 meters**, which is medium
- **More than 30 meters**, which is less challenging

- **Serpentine roads**: this parameter has been divided on:
  - **Extreme**
  - **Medium**
  - **Weak**

In this case, the parameter should be defined by tester **visually**, which one is *Extreme*, which one is *Weak*, whatever; however, in my opinion, there is no point to think of the radius, distance between centerlines and quantity of serpentine road.

- **Multipath**: firstly, let us define, what is "multipath" actually?

  "*Multipath is the propagation phenomenon that results in radio signals reaching the receiving antenna by two or more paths. Causes of multipath include atmospheric ducting, ionospheric reflection and refraction, and reflection from water bodies and terrestrial objects such as mountains and buildings."*
  (Source: http://en.wikipedia.org/wiki/Multipath_propagation)

  We take into account the height of the surrounding objects, however we do not take the distances between the road and the object. In my opinion, the distance, whatever it is, does not make a big sense even if the object is in one or more kilometers from the road and if the object is visible, the multipath effect is evident. Height is a key factor here: the tall object will cause multipath propagation with much more probability, than the lower one wherever the object is. Thus, we divide this parameter from the most challenging to the less challenging one:
  - **More than 100 meters**
  - **From 100 to 50 meters**
  - **Less than 50 meters**

- **Tunnels**: roads with no GPS signal reception. It is one of the challenging parameter. In spite of this, it is possible to divide this as well:
  - Tunnels with **forks**.
  - Tunnels with the length **more than 10 kilometers**
  - Tunnels with the length **from 5 to 10 kilometers**
  - Tunnels with the length **from 1 to 5 kilometers**
  - Tunnels with the length **less than 1 kilometer**

- **Garages & Parking**: here can be used different types of garages and parking area:
  - **Underground parking**: area without GPS signal reception
    - **More than 5 levels**
    - **From 2 to 4 levels**
    - **1 level**
  - **Above-ground parking**: area with limited GPS signal reception
    - **More than 5 levels**
    - **From 2 to 4 levels**
    - **1 level**
  - **Open air parking**: are with 100% GPS signal reception
    - **Mapped**
    - **Unmapped**

- **Highways**: there can be overpasses and gasoline station on highways, as well as junctions with different types of exits and entrances. Thus, I would divide this parameter on highways with:
  - **More than 4 lanes**
  - **Less than 4 lanes**
  - With **elevated connection**
  - With **normal connection**

- **Roundabouts**: different types of roundabouts can be considered, for example:
  - **Circle**
  - **Squared**
  - **Oval**
  - Roundabouts with **more than 4 lanes**
  - **Unusual**

- **Ferry mode**: mostly, this parameter is unmapped. In such case the key factor is map matching. This parameter is divided so, that path is:
  - **Less than 100 meters**
  - **From 100 to 500 meters**
  - **More than 500 meters**

- **Urban villages**, rural area: not very challenging parameter so far, better for map matching testing.

- **Intersections**: also can be considered as a challenge for urban navigation, especially in combination with such parameters like *Multipath* (i.e. skyscrapers), *Narrow streets in cities* etc. We can consider different intersections, for example:
  - **Wide** (for example, can contain more then 4-5 lanes or less, but containing a road divider etc.)
  - **Normal** (or medium)
  - **Narrow** (for example containint 2 lanes)

Special Situations: this parameter means all another parameters, which are not included to the all listed parameters. Combination of several parameters (for example, *"Circle Roundabout"* and *"Multipath: 50 m"*) can be included to this parameter as well.

4.3.3 Proposal of Test Cases

When creating the test cases for testing of the Navigation Device, it is necessary to take into account the main function of the device – navigation and positioning. The key instrument here is a cursor, pointing the position on map, namely – *proper cursor movement*, and *recalculating route process time*, in case of deviation from the right way, when using navigation function.

It is obviously, that *recalculating route process time* directly related to the positioning, which is represented by the cursor on a display.

Let us imagine a situation, when we are driving somewhere in the unknown terrain using the road guidance function (navigation), and somewhere, for example at the junction there is a division of the road on 2 parallel roads. Obviously, one of these roads is right, another one is wrong, while we have turned the wrong road. What should happen then? What message should we receive from the navigation device? Of course, the proper functioning application will start a recalculating route process and the cursor surely will show us, that we have left the proper roadway, and the recalculating process should take a few seconds. This is very important especially on highways on high speed.

However, I personally was observing such a situation, when the cursor was showing the wrong position, and as a result, the recalculating process was lasting quite long.

This has happened, when I was driving from Mlada Boleslav to Melnik using Road Guidance. At the place shown on Figure 4.1, I have by accident missed the turn to the road, leading to Melnik, which is the Parallel Road. Result: lost time and fuel.

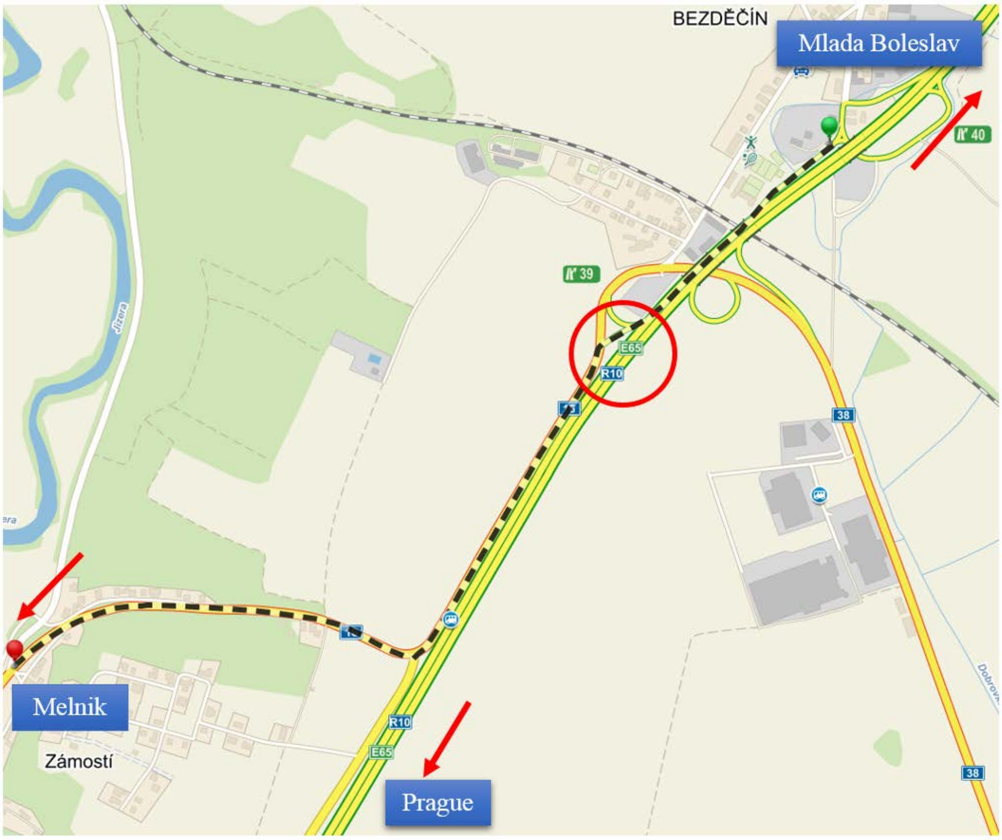


Figure 4.1: The fork contoured by a red circle. Source: www.mapy.cz

We can consider this situation as an example of the test case with the main input parameter “**Parallel Roads**” with the distance range **between 15 and 20 meters** from center to center.

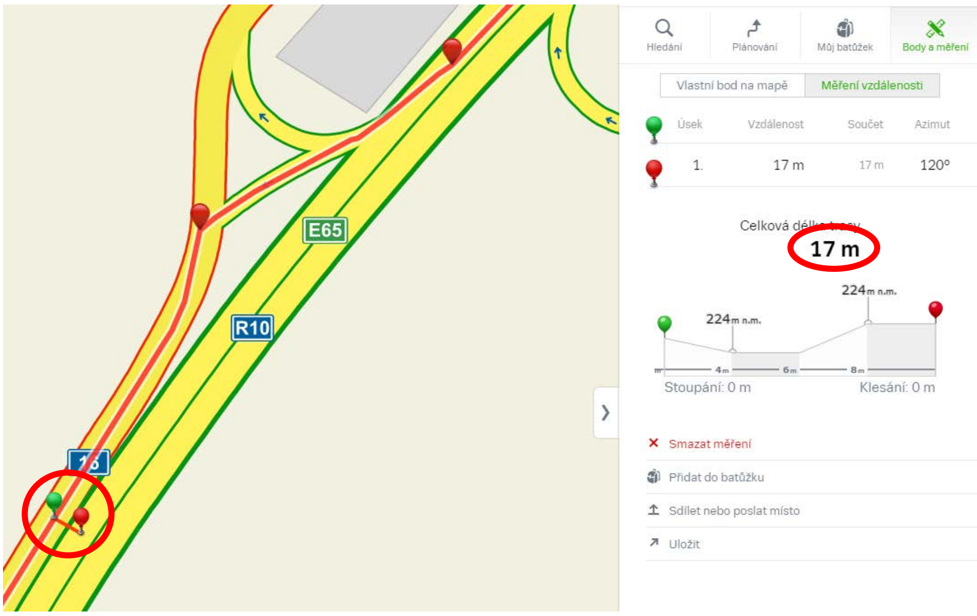


Figure 4.2: Parallel Roads with distance range between 15 and 30 meters from center to center. Source: www.mapy.cz

I personally do not see the point to test the Navigation Device in a Road Guidance (Navigation) mode, because the key factor is still the same: positioning, and as I mentioned before, the recalculating route process time directly depends on the actual cursor state.

Thus, we can derive several test cases from this situation, for example:

1. Positioning at the Elevated connection to the Highway: Here we can test *cursor movement* at the **Elevated connection** with the **Highway**.

Expectation: The cursor should move from the entrance to the highway

2. Positioning at the Parallel Roads: Here we can test a cursor movement driving on a highway and then turning to the **Parallel Road** with the distance range **between 15 and 30 meters** from center to center.

Expectation: The cursor should move according to the direction to Melnik

In fact, testing the positioning performance of the Navigation Device, the expectation and requirement in any cases is generally the same: The cursor should move according to the vehicle movement and show the actual vehicle position.

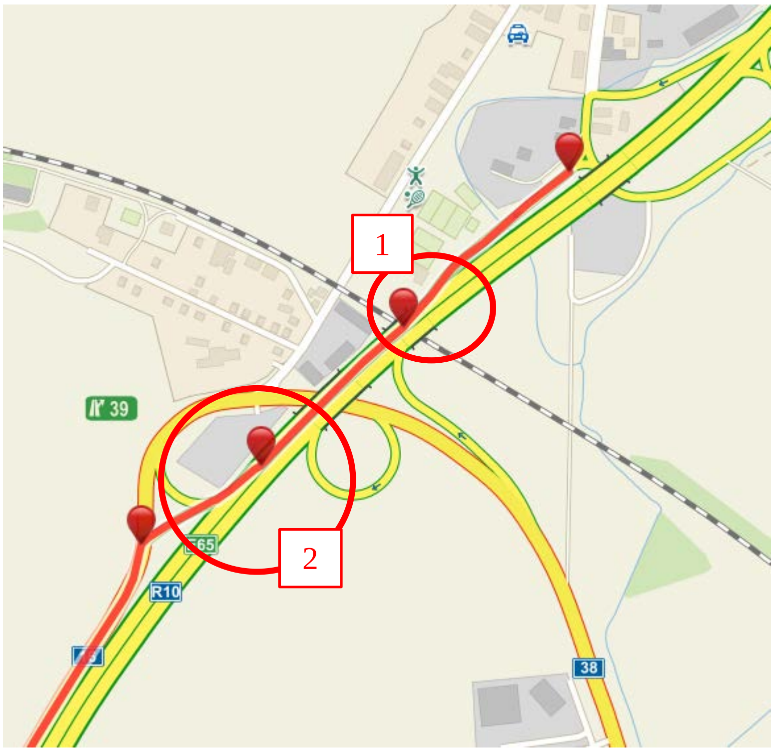


Figure 4.3: Test cases 1 and 2
Source: www.mapy.cz

All these – **Input Parameters** and **Test Cases** form a **Test Scenario**.

4.3.4 Proposal of Test Scenarios

Now we have defined the **Input Parameters** and the **Test Cases**, which could be in such type of testing. All these in combination form a single **Test Scenario**.

Thus, we have such a Test Scenario:

Test Scenario ID	Input Parameters	Test Cases	Comments	Expectation
54-EU-CZ-MLADA_BOLESLAV	**Highways:** *Elevated connection* **Parallel roads:** *15 – 30 m*	1. Connection to the Highway	Elevated connection	The cursor should move from the entrance to the highway
		2. Parallel roads	Distance range is **between 15 and 30 meters** from center to center.	The cursor should move according the real vehicle position.

Table 4.1: Example of the Test Scenario for the Navigation Device Capability testing

Describing the Test Scenario ID - ***54-EU-CZ-MLADA_BOLESLAV***:
“54” – means the number in the list of the Test Scenarios.
“EU” – Continent (in this case – Europe)
“CZ” – Country (in this case – Czech Republic)
“MLADA_BOLESLAV” – City/Town

Of course, it is vital to add another important information, such as coordinates, etc., but it will be described in Chapter 5 – *"Finalizing test scenarios using prescribed form"*.
In fact, creating the test scenarios for the Navigation device capability is a creative job. Very accessory source here is KML file with the *Start Point*, *End Point*, and the *Path* itself. It is very easy to generate and download the KML file using such popular GIS tools like www.mapy.cz or www.google.cz/maps. However, there are a lot of online services are available to design a path and generate a KML file from this.

> *Keyhole Markup Language (KML) is an XML notation for expressing geographic annotation and visualization within Internet-based, two-dimensional maps and three-dimensional Earth browsers. KML was developed for use with Google Earth, which was originally named Keyhole Earth Viewer. It was created by Keyhole Inc., which was acquired by Google in 2004. KML became an international standard of the Open Geospatial Consortium in 2008. Google Earth was the first program able to view and graphically edit KML files. Other projects such as Marble have also started to develop KML support.*
>
> *(Source: http://en.wikipedia.org/wiki/Keyhole_Markup_Language)*

Such rich resources give us a huge possibility to create many different test scenarios, containing very different parameters and very different locations all over the world. However, it is very useful indeed. Let us imagine that we need to test the cursor behavior in multipath condition. Obviously, we have no possibility to record a GPS signal in Prague in such conditions, as here are mostly not very tall buildings like, for example, in Frankfurt or Dubai.

Of course, as it was described in Chapters 2 and 3, it is possible to generate such an environment using *Spirent GSS6700* equipment. Therefore, we need to specify a possibility for testing – which test scenarios can be recorded in Reality (using *Spirent GSS6400 Record and Playback System*), which ones can be recorded from the simulated environment (using *Spirent GSS6700 Multi-GNSS Constellation Simulator System*), and of course, which ones have been already recorded.

4.3.5 Testing Environment

As we mentioned before, **Test Scenarios** can be recorded during a real test drive, using *Spirent GSS6400 Record and Playback System*, and in laboratory, in simulated environment, using *Spirent GSS6700 Multi-GNSS Constellation Simulator System*. Of course, the same test scenario can be recorded both in real and simulated environments. However, the main issue here is – location, which leads to other – economical issue.

Thus, we have several possibilities for recording the test scenarios:

- ***Spirent GSS6700 Multi-GNSS Constellation Simulator System***
  Mainly for the test scenarios, which are located abroad. Can be applied for all of the test scenarios.

- ***Spirent GSS6400 Record and Playback System***
  For the test scenarios, which are located inside the country.

4.3.6 Test Scenarios Filtering Algorithm

Being in the Czech Republic or, for example, in Austria, it is quite expensive to fly and record the test scenario, for example, to San Francisco or Moscow. However, there are lots of test scenarios can be created with different parameters, test cases and locations. Therefore, there was a strong need to create some application for filtering the new test scenarios.
Solving this problem, I have written the matrix-based algorithm for the test scenarios filtering in Matlab environment. As a key input I have used the parameters listed before (see *4.3.2 Proposal of Input Parameters for the Scenario Testing*).

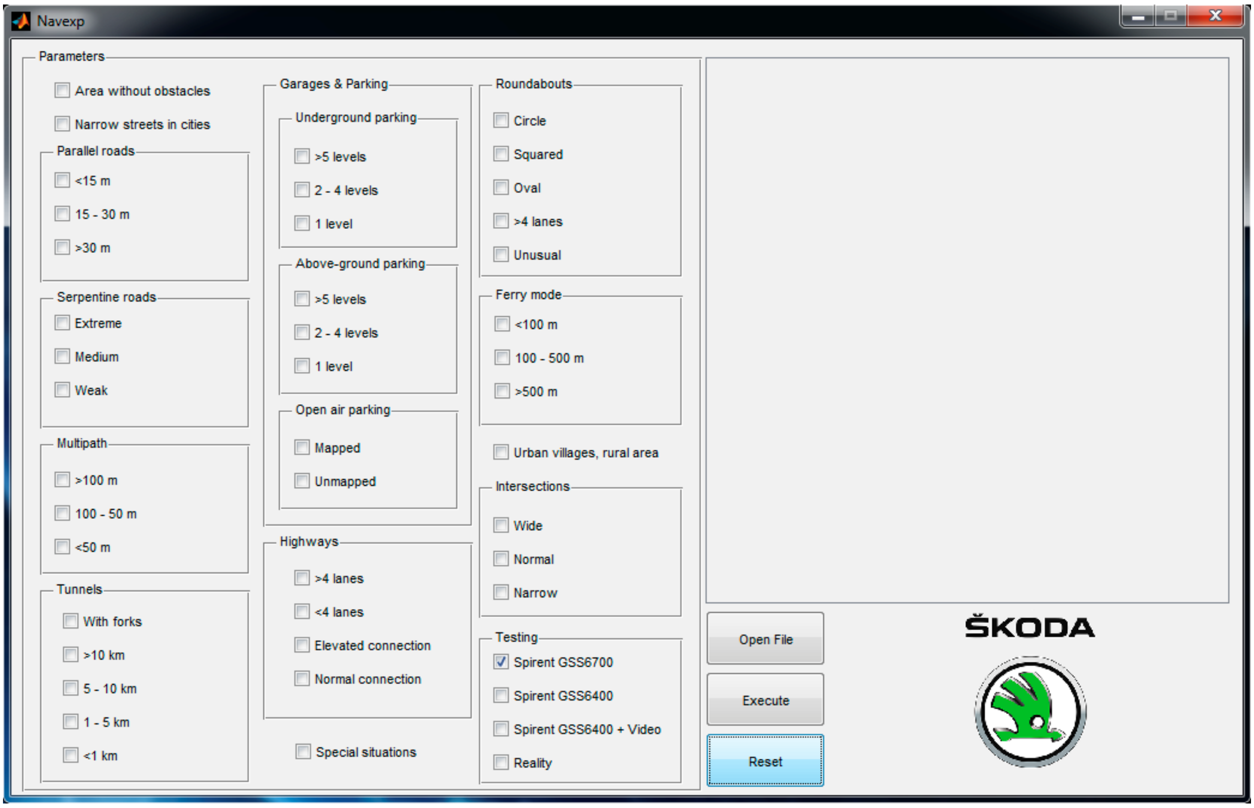


Figure 4.4: Navexp – filtering application for the test scenarios

I have called the program ***Navexp*** – short name coming from "Navigation Expertise". Navexp is the matrix-based filtering program, working with the special matrix-based database, created in advance, as well as the test scenarios, which have to be shown to the display on the right side after filtering.

4.3.6.1 Testing Parameters

To start describing the *"Testing"* panel on the interface, it must be noticed, that there is a strong filtering. If we check one checkbox in the panel, the result will be the files, related to the selected checkbox ONLY, and there is no possibility to check more than one checkbox.

Let us explain, what actually mean those parameters:

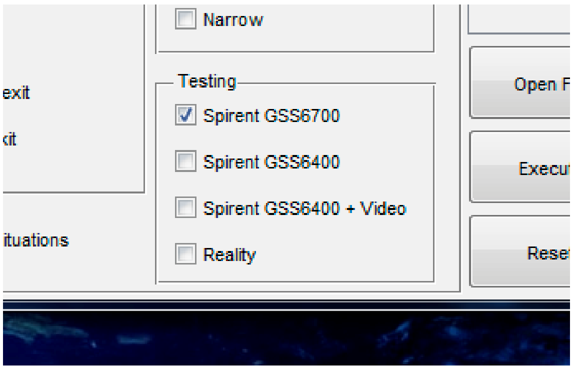


Figure 4.5: Testing Parameters

- **Spirent GSS6700**: All the test scenarios – which can be recorded both in simulated environment and in reality.
- **Spirent GSS6400**: All recorded test scenarios
- **Spirent GSS6400 + Video**: All recorded test scenarios including video recordings of the road, navigation device display and speedometer (optionally) during the real world conditions recording test drive.
- **Reality**: Test scenarios, filtered by location. For example, being in the Czech Republic, we can easily record all the test scenarios in the country. However, this can be easily changed in the database. Furthermore, it is a possibility to test the scenarios with a real drive testing approach.

Thus, if we select, for example, "*Spirent GSS6400*", as a result we will have only recorded test scenarios on display, depending on set in advance input parameters. We cannot check, for example, "*Spirent GSS6400*" and "*Reality*" both together.

4.3.6.2 Input Scenario Parameters

Speaking about the input parameters for the test scenarios, there is no necessity in strong filtering: if we select, for example "*Parallel Roads (15 – 30m)*", "*Roundabouts (Circle)*", and "*Tunnels (1 – 5km)*" al together, there will be shown the all results with all these parameters, even if one does not contain some of them.
As shown on Figure 4.6, there are several parameters selected for real test drive, namely – "*Area without obstacles*", "*Narrow streets in cities*", "*Parallel Roads (15 – 30m)*", "*Tunnels (1 – 5km)*", "*Highways (Elevated connection)*", and "*Roundabout (Circle)*" . If we take attention to the display, there is already known test scenario "**54-EU-CZ-MLADA_BOLESLAV**". As we remember, this test scenario contains test cases with *Parallel Roads* and *Highways*, but does not contain *Area without obstacles*, *Narrow streets in cities*, *Tunnels* and *Roundabouts*. To make filtering stronger, it is enough to reduce the amount of selected input parameters. However, coming back to the strong filtering of testing possibilities, we can see, that all displayed test scenarios located in Czech Republic ("*Reality*").

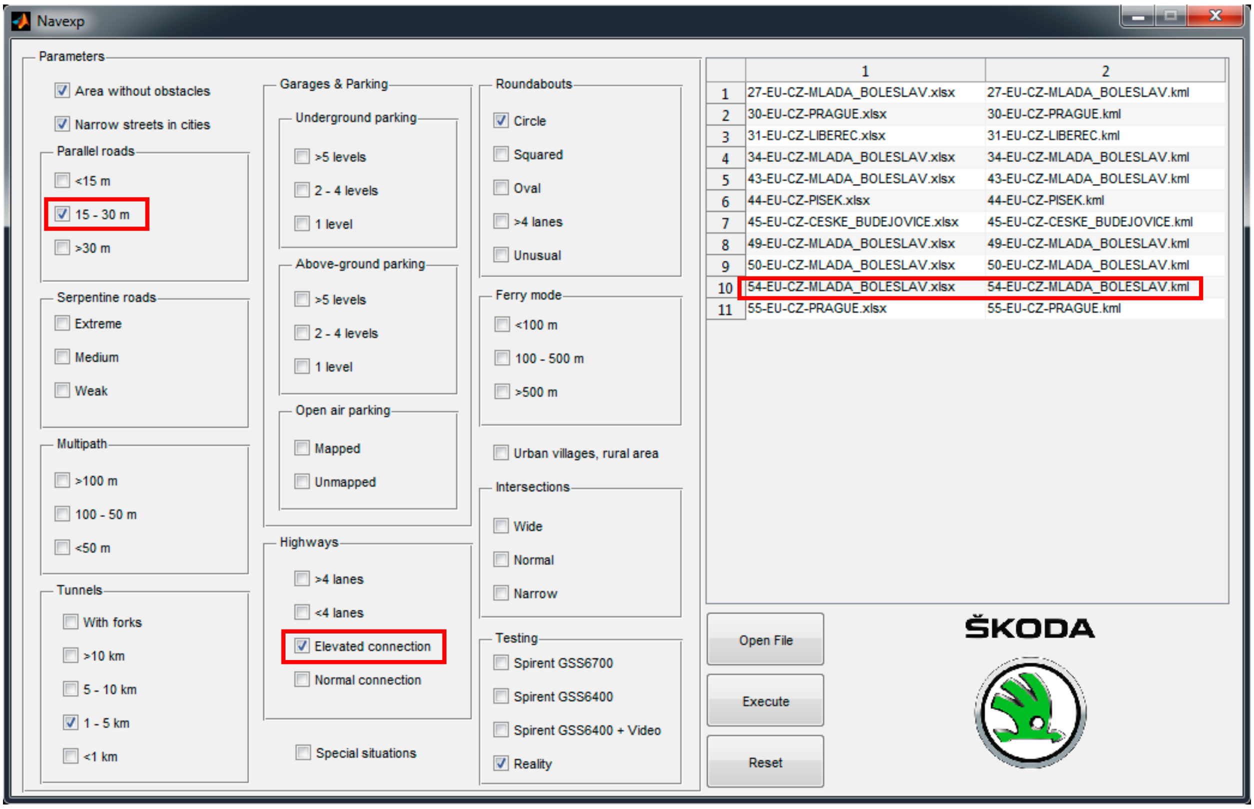


Figure 4.6: Test Scenarios filtering by input parameters and testing possibilities

Apart of the test scenario files (which stored in excel file), there are KML files as well, carrying the map information including start point, end point, and the path itself. And, if we click on it, the file will be immediately opened in Google Earth application (Figure 4.8).

The Navexp application has three buttons: *Open File*, *Execute*, and *Reset*.
Open File button responds on selection Database file, which is stored in Excel format and based on matrix, consisting of "*ones*" and "*zeros*". The database file system will be explained further.
Execute button responds for displaying the test scenarios according to chosen parameters.
Reset button corresponds for unchecking the scenario parameter checkboxes and refreshing the display.

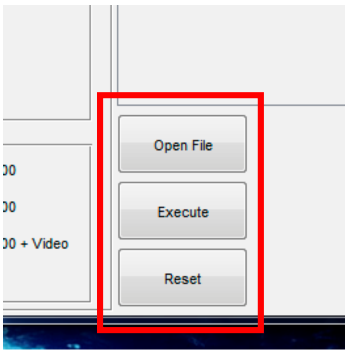


Figure 4.7: Open File, Execute, and Reset buttons

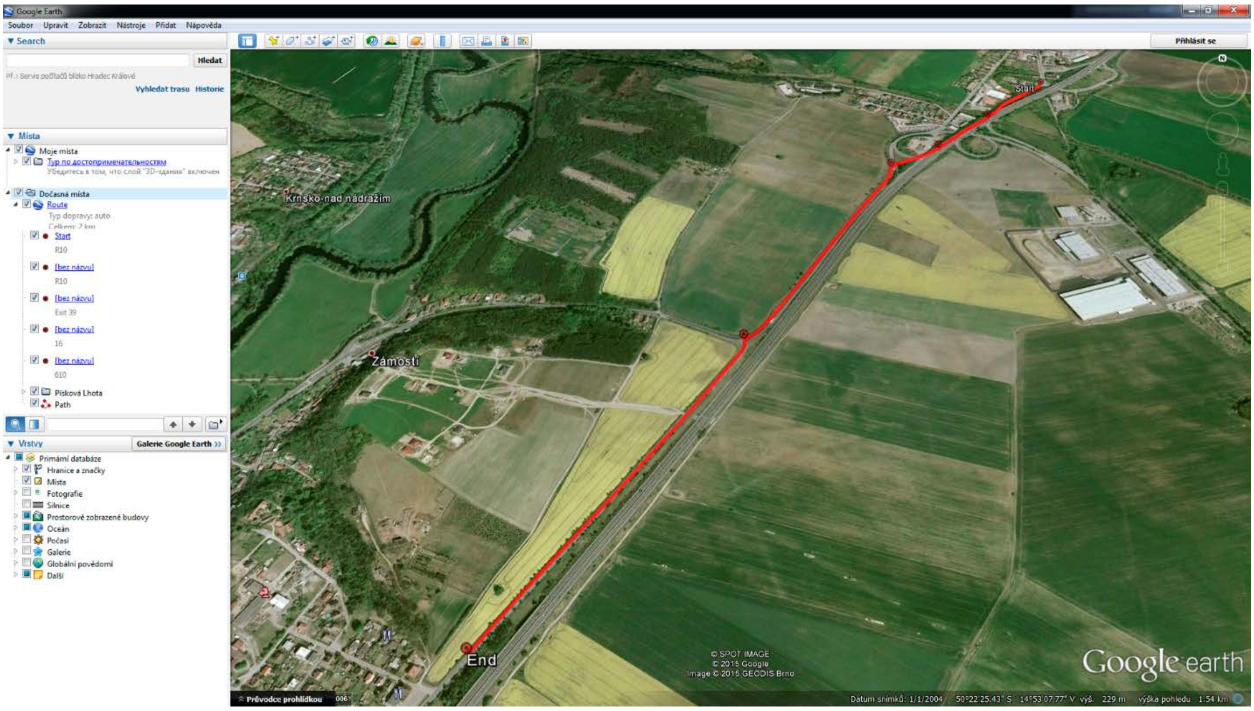


Figure 4.8: 54-EU-CZ-MLADA_BOLESLAV.kml file opened in Google Earth application

4.3.6.3 Database and the Main Principle of Test Scenario Filtering

The Navexp application works with databases, which stored in *xls* format. The database is based on matrix, which consists of "ones" and "zeros" (binary matrix). Due to the too big size of the already created matrix database, I will explain it on a more primitive example below.

Each scenario consists of different test cases, and the presence of some concrete test case in a test scenario defines by "ones"; and the absence of some concrete test cases – by "zeros".

Scenario ID (*)	Input Scenario Parameters (**)												Testing Possibilities			
	Area without…	Narrow streets…	Parallel roads			Serpentine roads			Multipath			…	GSS6700	GSS6400	GSS6400+Video	Reality
			<15 m	15–30 m	>30 m	Extreme	Medium	Weak	>100 m	100–50 m	<50 m					
#1	0	0	0	1	1	0	0	0	0	1	1	…	1	0	0	0
#2	0	1	0	0	0	0	0	0	0	1	1	…	1	0	0	0
#3	1	0	1	1	1	0	0	0	0	0	0	…	1	0	0	0
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
#55	0	0	0	1	0	0	0	0	0	0	0	…	1	0	0	1
#56	0	0	0	0	0	0	0	0	0	0	0	…	1	0	0	1

Table 4.2: Example of the matrix database for the Navigation Device Capability testing

Notes: (*) Scenario IDs are relative
(**)Due to the size, there are not all the parameters shown

The program is built in such a way: each checkbox corresponds to each column in order that name of Input Scenario Parameter must be the same both in database and in program itself. For example: Checkbox "Area without obstacles" corresponds to the column with the same name in database, which is column "B", checkbox "Narrow street in cities corresponds to the appropriate column which is column "C", etc. It must be noticed, that the column "A" in a database corresponds to the Test Scenario filenames.

	A	B	C	D	E	F	G	H	I	J	K
1	Test Scenario	Area without obstacles	Narrow streets in cities	(<15 m)	Parallel roads (15 - 30 m)	(>30 m)	(Extreme)	Serpentine roads (Medium)	(Weak)	>100 m	Multi 100 -
2	1-EU-RU-MOSCOW	0	0	0	1	1	0	0	0	0	1
3	2-EU-RU-MOSCOW	0	1	0	0	0	0	0	0	0	1
4	3-EU-RU-VOLGOGRAD	1	0	0	1	1	0	0	0	0	[illegible]
5	4-EU-AZ-BAKU	0	1	0	0	0	0	0	0	0	[illegible]
6	5-EU-RU-ARMAVIR	0	0	0	0	0	0	1	0	0	[illegible]
7	6-EU-RU-MOSCOW	0	0	0	1	0	0	0	0	0	[illegible]
8	7-EU-AZ-BAKU	0	0	1	0	0	0	0	0	0	[illegible]
9	8-EU-RU-OS	0	0	0	0	0	0	1	1	0	[illegible]
10	9-EU-RU-GIMRI	0	0	0	0	0	1	1	1	0	[illegible]
11	10-EU-RU-MOSCOW	0	0	0	1	0	0	0	0	0	[illegible]
12	11-EU-AZ-BAKU	0	0	0	0	1	0	0	0	0	[illegible]
13	12-EU-AZ-BAKU	0	0	0	1	0	0	0	0	0	[illegible]
14	13-EU-RU-MOSCOW	0	0	0	0	0	0	0	0	0	[illegible]
15	14-EU-UA-KIEV	0	0	0	0	0	0	0	0	0	[illegible]
16	15-EU-RU-ARMAVIR	0	1	0	0	0	0	0	0	0	[illegible]

Figure 4.9: Screenshot from the database: column "A" corresponds to Test Scenario filenames, the rest of the columns ("B", "C", "D", etc.) correspond to the checkboxes of the same names in Navexp application – Input scenario parameters and Testing parameters.

The main principle of filtering is simple: as we said, the database is matrix-based, as well as the application itself, and the checkboxes correspond to the column of the same names. Searching process is going through the columns, considering the values, set in advance: if the checkbox is checked, the value is "1", and if the checkbox is unchecked, the value is "0". Thus, if one checkbox is checked, there will be shown the filenames, which has test case with value "1" as a result. For example, if the checkbox ***"Parallel Roads"*** with the distance range ***15 – 30 m*** from center to center, after pressing *"Execute"* we will see on a display such Test Scenarios, which contain such a Test Case:

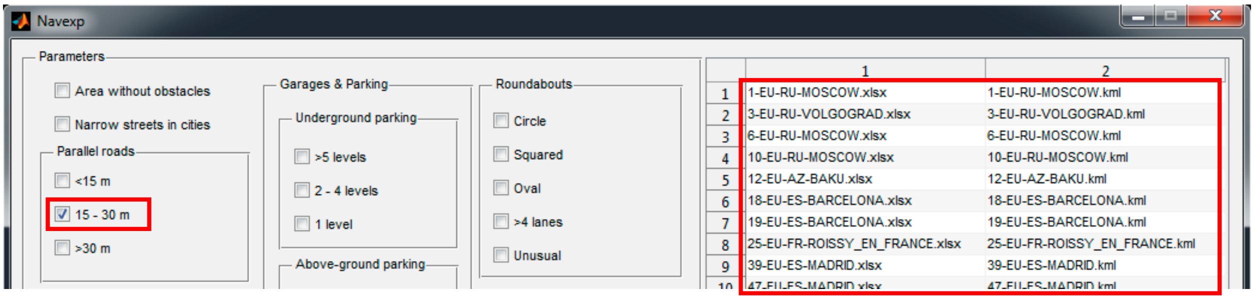


*Figure 4.10: Checkbox **15-30 m** in panel **Parallel Roads** is checked, and thus its value is "1", and as a result, the filenames, which have the same value in appropriate column are shown on the display. For comparison, see Figure 4.9.*

As the testing possibility has strong filtering, it should be set in advance, however the possibility for simulation using ***Spirent GSS6700*** equipment is set on default, as all the test scenarios can be recorded in simulated environment, using the KML files as a main source for that. Once the test scenario has been recorded (does not matter, using simulation equipment or in reality), it is necessary to change "0" to "1" in appropriate column and row in the database file. As a result, this test scenario will be appearing on display in case of filtering by recorded test scenarios, as well as the all recorded scenarios (***Spirent GSS6400*** or ***Spirent GSS6400 + Video*** filtering options).

To open those test scenario files (*xlsx* and *kml*) clicking on them on the display, it is necessary to have the stored files in some specific folder. The test scenario files must be stored at the same direction as the database file as well as *Settings* file (see Chapter 5 for more information).

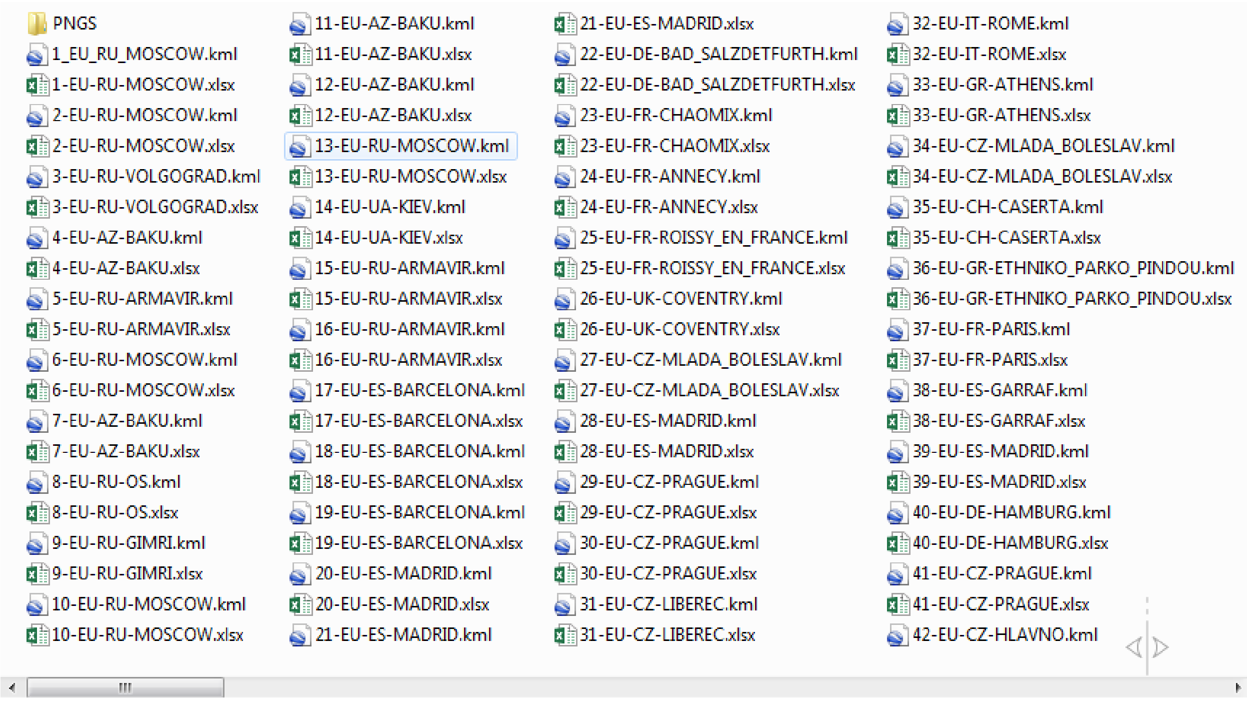


Figure 4.11: Test Scenario storage

In fact, there is a possibility to create many of the databases (also, with many other test scenarios). For example, already created database is defined by the continent location (in this case – Europe, as it is already said in filenames). Thus, in the future we can create databases corresponding to other continents (Asia, North America, Africa etc.) as well as corresponding to different countries (Czech Republic, Germany etc.) and so on. However, the key feature when creating the database is – *structure*: the database should correspond to the program algorithm.

4.3.7 Conclusion

Test scenarios for testing of capability of the navigation device can contain many different types of the test cases, much more, than was created by me, and the concrete rule for this actually does not exist. Creating of the test scenarios is quite a creative job, required basic maps understanding and imagination of any possible challenge for the positioning. Of course, there should be noticed the factors, which can influence the GPS signal reception, or even prevent it. Real-world test scenarios, chosen for the navigation device performance, can help tester to expose a real bugs during the testing process, which can real user face with during a travel, whatever it is – wrong cursor movement on the roundabout, tunnel or the highway. Navexp application can help the tester to filter test scenarios by the required input scenario parameters, its difficulties and testing options. Navexp application considerably reduces the time, spent on searching and filtering the test scenario with appropriate test cases manually, which is very important for any type of job.

5. Finalizing Test Scenarios using Prescribed Form

5.1 Introduction

In the previous chapter, we specified all possible test cases for the test scenarios. We have defined the most of the test cases as less challenging, medium and the most challenging situations, depending on each individual test case. For example, tunnels defined by lengths and presence of forks, parallel roads – by distance ranges between two centerlines and so on. But defining the test cases, containing in each test scenario, we must add a lot of another different, and not less important information, for example – expectation: (how should behave the device while driving to the roundabout or the intersection etc.), locations of the cases: (where is it?), etc. Thus, we have applied a master form for the test scenarios.

In this chapter, I will describe the master form of the Test Scenario for the navigation device performance testing in details.

As I mentioned before, we decided to store Test Scenarios in **.XLSX* format to simplify all the testing procedures. The test scenario file consists of the ***General Information***, ***Bug Report***, ***Test Report***, ***Map*** and ***Settings***, which we mostly do not operate, but have to create.

5.2 Test Scenario Overview. General Information

Every test scenario should have general information. For general overview, the most important information is – scenario content, location, length, time duration and the start point coordinates, and of course – test scenario filename and the map (to have an idea what is the country or the surrounding). The Figure 5.1 shows the general information of the randomly chosen Test Scenario.

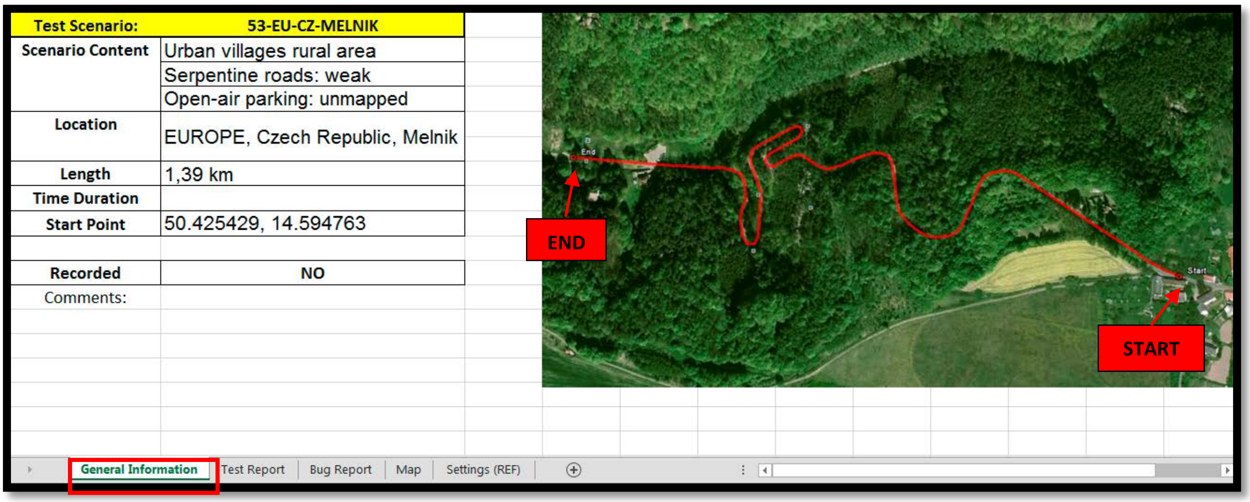


Figure 5.1: General Information of the Test Scenario

From the excel sheet shown on Figure below we see, that the scenario contains the following parameters:

- **Urban villages, rural area**

- **Serpentine roads**
    - **Medium**

- **Open air parking**
    - **Unmapped**

The concerned test scenario located in the **Czech Republic**, in immediate vicinity of **Melnik** town (what we can also extract from the filename of the test case), and its length is **1.39 km**. The start point specified by the coordinates: **50.425429**, **14.594763**.

There is also an important information to be filled after the recording:

- **Time Duration**: duration time of the testing

- **Recorded**: whether the test scenario has been recorded using GSS6400 Record and Playback system or not. In case of the concerned test scenario, it has not been recorded using Spirent equipment.

- **Comment**: here can be different comments (for example – weather conditions), as well as recording file name.

Thus, we have all necessary general information about the Test Scenario *53-EU-CZ-MELNIK*.

5.3 Test Report

Test Report is an integral part of any testing, and it must be in a Test Scenario for the navigation device performance testing as well. The main goal of the Test Report is to specify – which test cases have been passed, and which ones have been failed and why, describing the expected behavior and the actual behavior, as well as its level of acceptance or unacceptance. The Test Report overview is shown in the Figure 5.2.

ŠKODA
www.skoda-auto.cz

Test Scenario :	**53-EU-CZ-MELNIK**	Testing Duration :	**hh:mm:ss - hh:mm:ss**
Test Report No. :		Hardware Ver. :	
Location :	**Czech Republic, Melnik**	Software Ver. :	
Date :	**dd.mm.yyyy**	Map Vers. :	
Project :		Test Result :	
Testing Environment :		Tested By :	
Testing Equipment :			

Number	Content	Test Result	Rating	Location Time & Coordinates	Description	Expectation	Comments
1	Urban villages, rural area			50.4254581N, 14.5947683E	Road going through the forest	Cursor should smoothly move along the road	
2	Serpentine road			50.4268386N, 14.5877731E	Weak	Cursor should smoothly move along the serpentines	
3	Open air parking			50.4269353, 14.5846808	Unmapped	Cursor should move to match the map to the display according to the real pos.	

Amount of Test Cases: TESTED	0
Amount of Test Cases: PASSED	0
Maximal Value	0
Mean Value	#ДЕЛ/0!
TOTAL	#ДЕЛ/0!

General Information | Test Report | Bug Report | Map | Settings (REF)

Figure 5.2: Test Report overview

As we see from the figure, the Test Report contains a heading.

ŠKODA www.skoda-auto.cz	Test Scenario :	53-EU-CZ-MELNIK	Testing Duration :	hh:mm:ss - hh:mm:ss
	Test Report No. :		Hardware Ver. :	
	Location :	Czech Republic, Melnik	Software Ver. :	
	Date :	dd.mm.yyyy	Map Vers. :	
	Project :		Test Result :	
	Testing Environment :		Tested By :	
	Testing Equipment :			

Figure 5.3: Test Report - Heading

The heading consists of both general information and the empty lines, which should be filled in during the testing – specific information.

As we see from the Figure, the heading contains such lines as:

- ***Test Scenario***: here should be pointed a test scenario, which should be done in advance, when creating a test scenario. The Test Scenario naming styling has been described before in detail.

- ***Test Report №***: the same Test Scenario can be tested using different equipment – both testing equipment and tested equipment (these points are also shown as "Testing Equipment" and "Hardware"). Therefore, this line is to be filled by tester.

- ***Location***: this defines the country and the city/town. As a Test Scenario, this line is usually filled in advance.

- ***Date***: date of testing, which is to be filled in such a format: *dd.mm.yyyy*. As I said before, the same test scenario can be tested several times with different equipment, therefore, date is an important point here.

- ***Project***: this has an optional filling, which has the possibility to be improved, depending on project status. However, testing has contain several projects. In our case, the projects are related to the in-car PC equipment, including control panel and its unit.

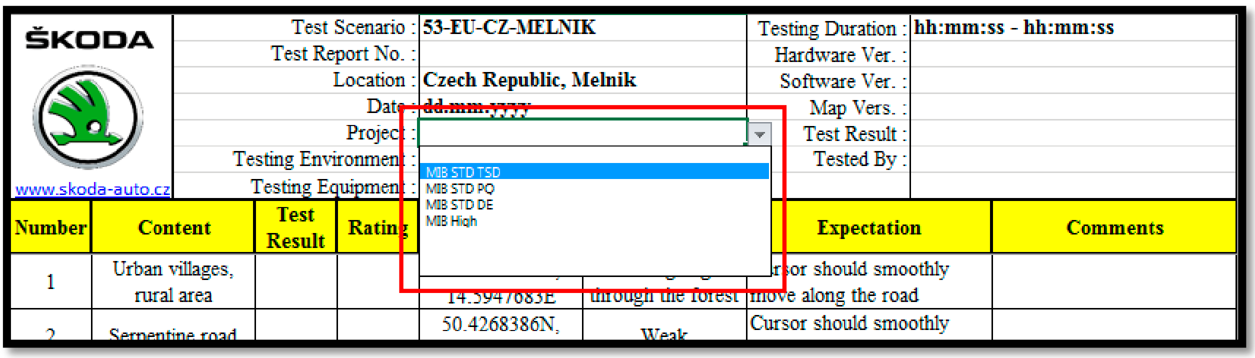


Figure 5.4: Test Report - Heading

As we see from the Figure 5.4, there is some reserved places for adding new projects under the existing ones. The detailed description of this function will be described in “Settings” subchapter.

- ***Testing Environment***: this line has only two possibilities – reality and laboratory.

- ***Testing Equipment***: this line, as well as the Project line, has possibility to be improved and currently has four options, which are:
    - *GSS6400*: Spirent GSS6400 Record & Playback System
    - *GSS6700*: Spirent GSS6700 Multi-GNSS Simulator
    - *SK371*: this is the internal project name of Skoda Octavia. This specifies mainly a vehicle version and type (it can depend on manufacturing year and the body type). That means, that the testing has been undertaken in reality, and the tested unit was that ones, which is usually used in that trim level. For example, Skoda Octavia has such a trim levels: Active, Ambition, etc.
    - *SK481*: the same meaning as SK371, however, SK481 is an internal project of Skoda Superb.

- ***Testing Duration***: this line defines a testing duration time from the start up. However, there is one important point, that testing or recording process can start in any location and in any time, even if it differs from the test scenario’s start point – for example, on a highway, where it is not allowed to stop the car. In this case, the tester should point out the arrival time on the map, namely – at the start point.

- ***Hardware Ver.***: this defines a unit version.

- ***Software Ver.***: this line is done for pointing out the software revision. This helps to define whether some test case is reducible or not.

- ***Map Ver.***: as we know, the transport infrastructure constantly develop, therefore there is a need in maps revision. This point could be also useful for navigation device performance, especially if we test it several times. It can contain unmapped area (mainly – open-air parking, etc.)

- ***Test Result***: this line has two options. This specifies whether the test scenario has passed the testing or failed.

- ***Tested By***: this line has been created to identify a tester. This line is improvable.

It should be noticed, that the same heading with exactly the same data is used in the Bud Report, which will be described in the next subchapter.

Now let us have a look at the main body of the Test Report.

Number	Content	Test Result	Rating	Location Time & Coordinates	Description	Expectation	Comments	
1	Urban villages, rural area			50.4254581N, 14.5947683E	Road going through the forest	Cursor should smoothly move along the road		
2	Serpentine road			50.4268386N, 14.5877731E	Weak	Cursor should smoothly move along the serpentines		
3	Open air parking			50.4269353, 14.5846808	Unmapped	Cursor should move to match the map to the display according to the real pos.		
							Amount of Test Cases: TESTED	0
							Amount of Test Cases: PASSED	0
							Maximal Value	0
							Mean Value	#ДЕЛ/0!
							TOTAL	#ДЕЛ/0!

General Information | Test Report | Bug Report | Map | Settings (REF)

Figure 5.5: Test Report – Main Body

As we see from the Figure 5.5, the main body of the test report consists of such important components as:

- ***Number***: this heading specifies the number of the test cases, containing in the test scenario. In case with the observed test scenario, there are only three test cases.

- ***Content***: this heading specifies the test case content, which is also specified as the key Input Scenario Parameter in *Navexp* application. In the test scenario 53-EU-CZ-MELNIK we have three different tested areas: *urban villages and rural area*, *serpentine roads* and the *open-air parking area*.

- ***Test Result***: this is a result of testing, and should be filled in for the each tested case to specify whether the test case is passed the testing of failed.

- ***Rating***: as a Test Result and the Test Report itself, Rating scale is an integral part of the Test Report, showing the result of the testing. This is a level of acceptance of the tested case.

Number	Content	Test Result	Rating	Location Time & Coordinates
1	Urban villages, rural area			50.4254581N, 14.5947683E
2	Serpentine road			50.4268386N, 14.5877731E
3	Open air parking			50.4269353, 14.5846808
			1 2 3 4 5	

Figure 5.6: Test Report – Rating

As we see from the figure above, Severity has such options: the scale from 1 to 5. Thus, we can easily define the level of acceptability of the mistake, if the device has one. If the test case accepted after testing, it can be 1 – 3. If it is failed – 4 or 5.

- ***Location Time & Coordinates***: defines the location of the some exact test case. Time means the simulation time – on which minute or hour the vehicle (cursor) has reached the test case position. Time is being defined after the recording the test scenario.

- ***Description***: this line specifies the content in details, defining the level of complexity.

- ***Expectation***: this line describes the expectation – how should the device behave while functions. Simply saying – this is a proper behavior description and based on that we do test and compare the results.

- ***Comments***: here can be any comments about the navigation device performance – how did it function, what was wrong in the behavior of the cursor etc.

The *Test Report* sheet contains an algorithm for calculating the positioning performance, which was applied to all the test scenarios in our database. The "Test Result" and "Rating" cells are the integral part of the algorithm, as well as the table, located at the right down corner of the master form.

Number	Content	Test Result	Rating	Location Time & Coordinates	Description	Expectation	Comments
1	Urban villages, rural area			50.4254581N, 14.5947683E	Road going through the forest	Cursor should smoothly move along the road	
2	Serpentine road			50.4268386N, 14.5877731E	Weak	Cursor should smoothly move along the serpentines	
3	Open air parking			50.4269353, 14.5846808	Unmapped	Cursor should move to match the map to the display according to the real pos.	

Amount of Test Cases: TESTED	0
Amount of Test Cases: PASSED	0
Maximal Value	0
Mean Value	#ДЕЛ/0!
TOTAL	#ДЕЛ/0!

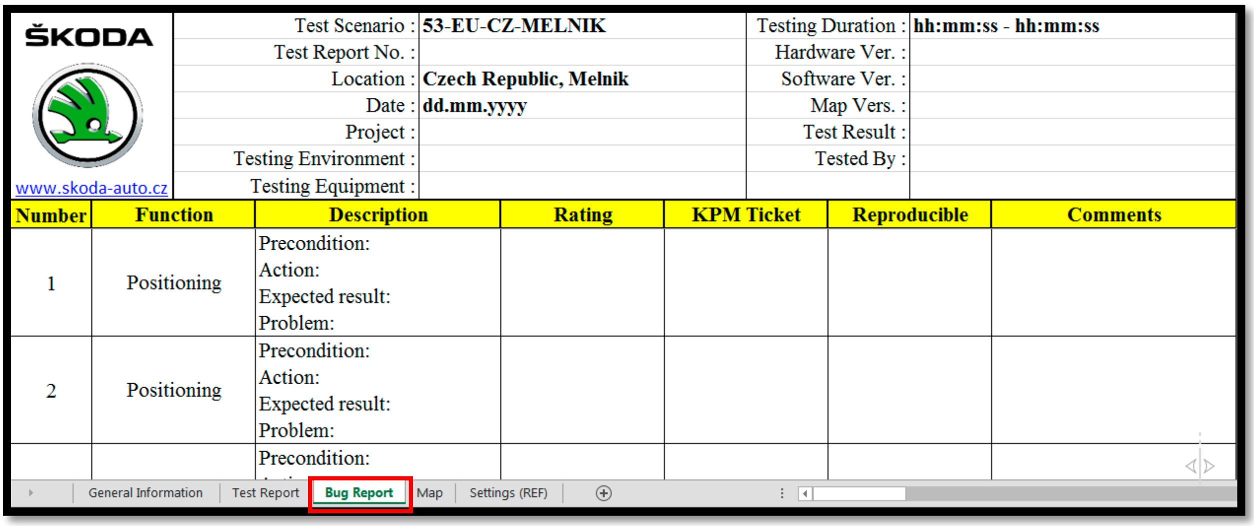
General Information | Test Report | Bug Report | Map | Settings (REF)

Figure 5.7: Positioning performance algorithm

This positioning performance calculation algorithm will be described more in details in the next chapter.

5.4 Bug Report

Bug report is a report of all the bugs which might be detected during a testing process. This is the next step after filling the Test Report sheet. As you see from the Figure 5.8, the heading of the sheet is the same as in the Test Report, and that was described in the previous subchapter, as it has exactly the same features.



Test Scenario :	53-EU-CZ-MELNIK	Testing Duration :	hh:mm:ss - hh:mm:ss
Test Report No. :		Hardware Ver. :	
Location :	Czech Republic, Melnik	Software Ver. :	
Date :	dd.mm.yyyy	Map Vers. :	
Project :		Test Result :	
Testing Environment :		Tested By :	
Testing Equipment :			

Number	Function	Description	Rating	KPM Ticket	Reproducible	Comments
1	Positioning	Precondition: Action: Expected result: Problem:				
2	Positioning	Precondition: Action: Expected result: Problem:				
		Precondition:				

Figure 5.8: Bug Report overview

Now let us consider the main content of the Bug Report:

- **Number**: number of all the bugs detected during the testing. The number of bugs directly depends on the number of test cases, which have been failed during the testing. However, if there was detected the same bug in several test cases, it is enough to have reported that bug by one number only.

- **Function**: function tested. In our case, as we decided before, we test the cursor movement function, i.e. – Positioning.

- **Description**: this content has been divided on:

    - *Precondition*: Settings of the navigation device before testing.
    - *Action*: Action(s) undertaken by tester during the testing.
    - *Expected Result*: How should the cursor behave in examined case?
    - *Problem*: What was wrong with the cursor behavior?

- **Rating**: this characteristic defines the level of acceptance of the bug, according to the internal company's KPM system, described further (Figure 5.9). Proposed options are:

    - *A1 - Safety critical*
    - *A – Unacceptable*
    - *B – Displeasing*
    - *C – Complaints*
    - *D - Information Only*
    - *DB - Decision required*

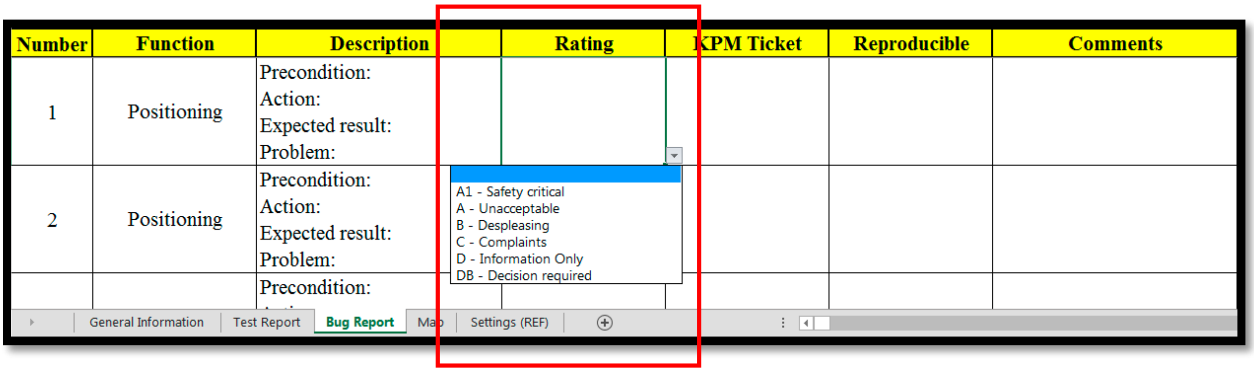


Figure 5.9: Rating options

- **KPM Ticket**: this is a number of bug in the internal bug database. KPM means *Konzern Problem Management* and is mainly used in *Volkswagen Group*.

  KPM system is an internal problem management system, simply saying – the online connection instrument between the tester and navigation device application suppliers. Thus, when the bug is found, it is being reported in a system, and supplier receives this ticket and observe the problem.

- **Reproducible**: defines, whether the bug reproducible after the navigation application revision or not, here are only two options:
  - *Yes*
  - *No*

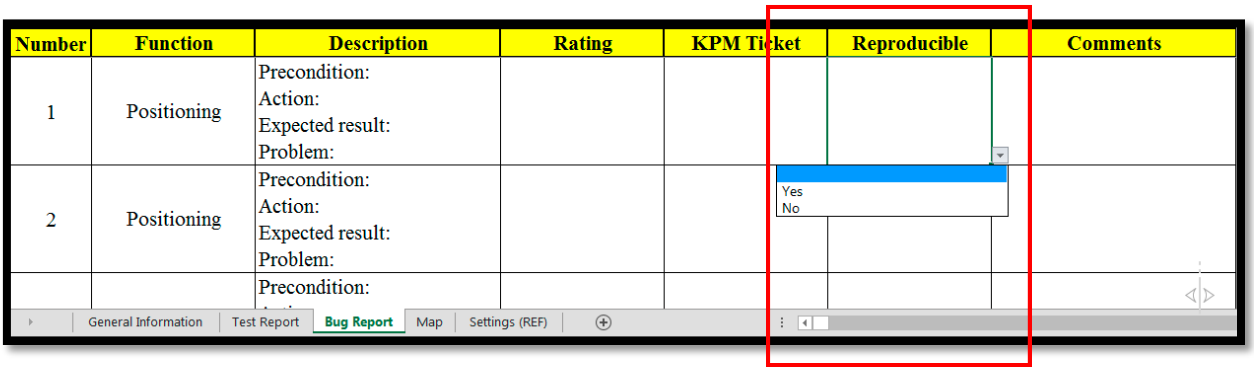


Figure 5.10: "Reproducible" option

- **Comments**: here can be different comments about the detected bug.

Mainly, the Bug Report done for the navigation application supplier, therefore, I will skip the usage of this report during the practical part of my Master Thesis. The outcome of the filled bug report should be a revision of the application, and as a result – the bugs detected during the testing have to be fixed.

5.5 Map

One of the necessary attributes of the Test Scenario is a map representation with its Start Point, End Point and the Path itself. Furthermore, there should be pointed the test cases. Depending on the test case, it can be marked as a road section or the exact or approximate location. As shown on Figure 5.11: we see, the test case #1 is marked as a piece of road – the road, going through the forest; the test case #2 – serpentine road section, and the test case #3 shows us the approximate location of the unmapped parking area.

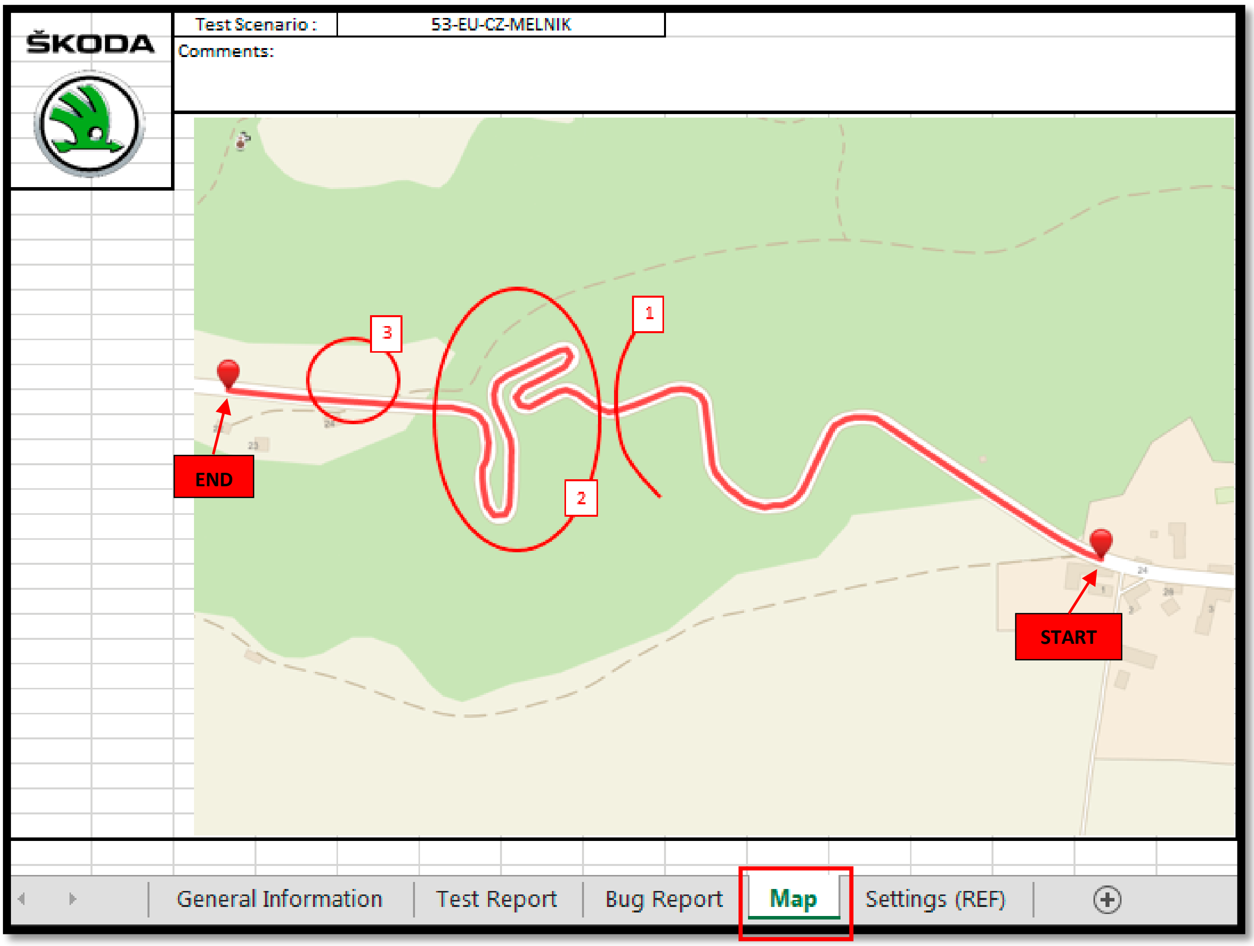


Figure 5.11: Map representation of the Test Scenario

The map is set so that it would be possible to print it out in A4 format for different goals: making marks (for example – to point the time left from the startup of the recording process or the exact location of stop engine, whatever), make different comments etc. This is very useful attribute not only to have an idea – where the test cases are, what this the area is, but to control the recording path as well – whether the driver is going right way or not. This is required due to the absence of the specific navigation application for the Road Guidance using already existing *KML* (or *GPX*, as the most of the devices support) files, what would be very useful, for example, to have it in smartphone or tablet, wherever. Nevertheless, after the filling all necessary information, the all data should be transferred from the paper map to the electronic copy, located to the sheet “Map” in the Test Scenario file.

5.6 Settings

This is not a documentation part of the test suit. This is done mainly to <u>simplify</u> a selection of the appropriate option for some lines in three sheets – ***General Information***, ***Test Report*** and ***Bug Report***, which have to be fill in within a testing process. Especially this is very useful if the option is <u>updatable</u>, for example - "*Tested By*" line. It is clear, that persons change the positions after some time – some are getting higher, some change the work places. Therefore, there was a need to make such possibility so that it would be very smart and user friendly. I have created a special ***Settings*** excel file, which gives a possibility to change the options delete or update them. The *Settings* key file is shown on Figure 5.12 below.

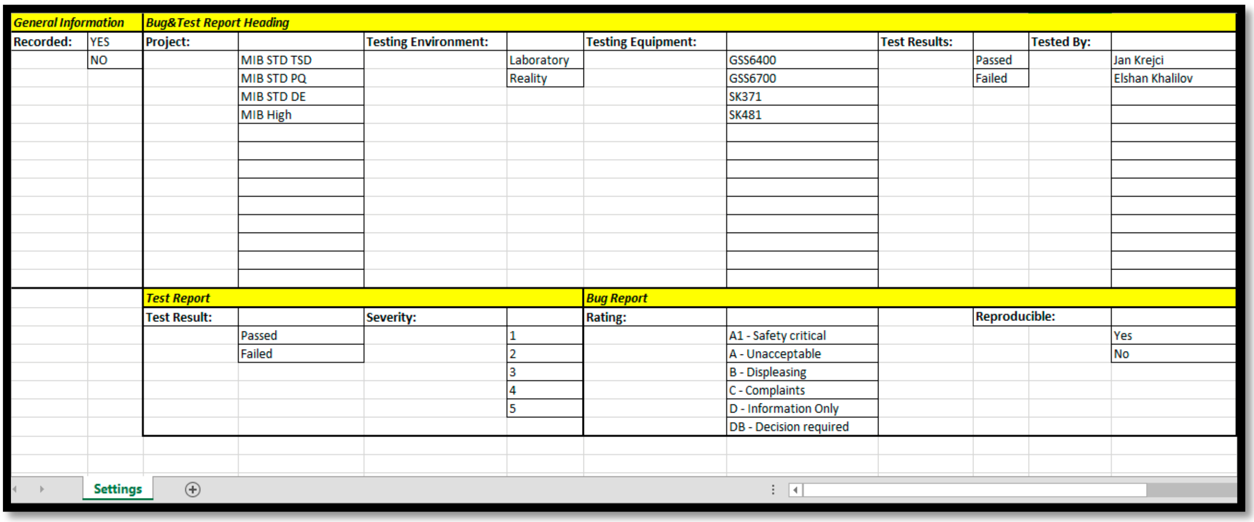


Figure 5.12: Settings file

From the Figure 5.12 we see the sections: ***General Information***, ***But & Test Report Heading***, ***Test Report*** and ***Bug Report***. This is mainly shows us, for which section it is done. We can also see that several lines have reserved places for updating:

- ***Project***: this is mainly project managers' task to update the project status – is the project closed or not, whatever, and of course it is necessary to have reserved place for new projects.

- ***Testing Equipment***: due to the development processes and time, any equipment is getting more modern, and the amount of the testing equipment is really, much more, than shown in the already existing options.

- ***Tested By***: as I said before, during some time the state of staff membership, which is responsible for testing, changes – for example, due to having more projects, there is a need to hire more employers for testing. Therefore, it is necessary to have reserved lines in this option.

The each listed above line can be easily improved by adding or deleting some option. So, how does it work? In my opinion, the most optimal way is <u>*referencing*</u>.

The each Test Scenario file has “*Settings (REF)*” sheet with exactly the same view as Settings file has.

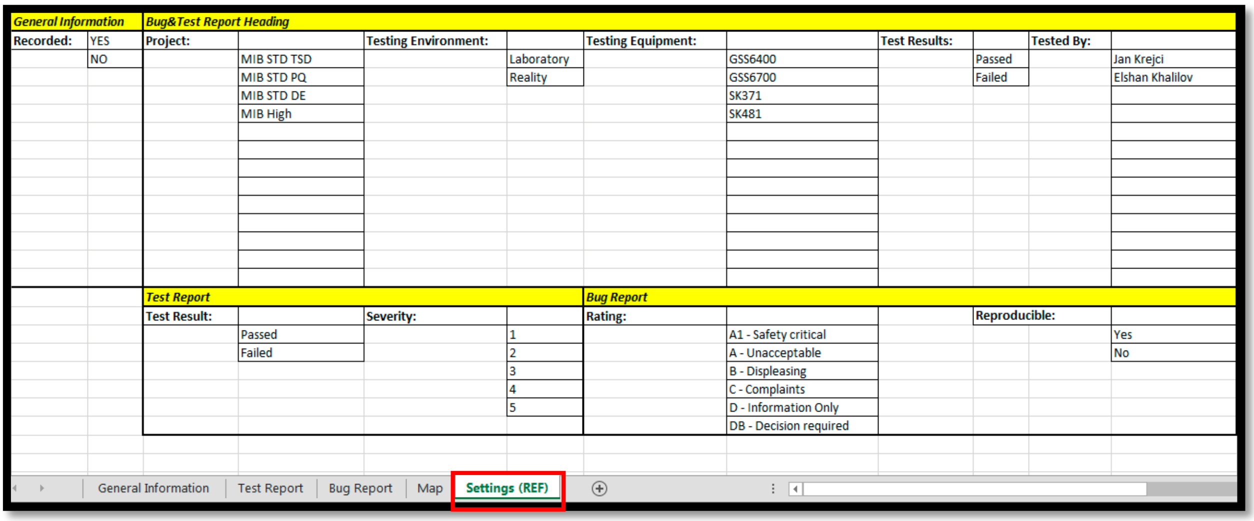


Figure 5.13: “Settings (REF)” sheet.

Each option in the “*Settings (REF)*“ excel sheet is transferred from the original *Settings* file using “*reference*” function in *MS Excel*. Thus, when we change any option in the original file, it is automatically changing in *Test Scenario* files. Therefore, when we open a *Test Scenario* file (anyone), we can see a window at the starting of the application – request to refresh the file. When clicking “*Refresh*”, we update all of the lines with the possibility of selection (Project, Testing Equipment, etc.), as well as “*Settings (REF)*” sheet itself, due to the last update in the original *Settings* file. The excel formula is very simple: we just type a “=” sign in an option cell in the “*Settings (REF)*” sheet, and then click on the appropriate cell in the original *Settings* file, and it is done. As a result, clicking on any option in “*Settings (REF)*“ sheet in the *Test Scenario* file, we will see, for example: “*=Settings.xlsx!D5*”, as shown on Figure 5.14.

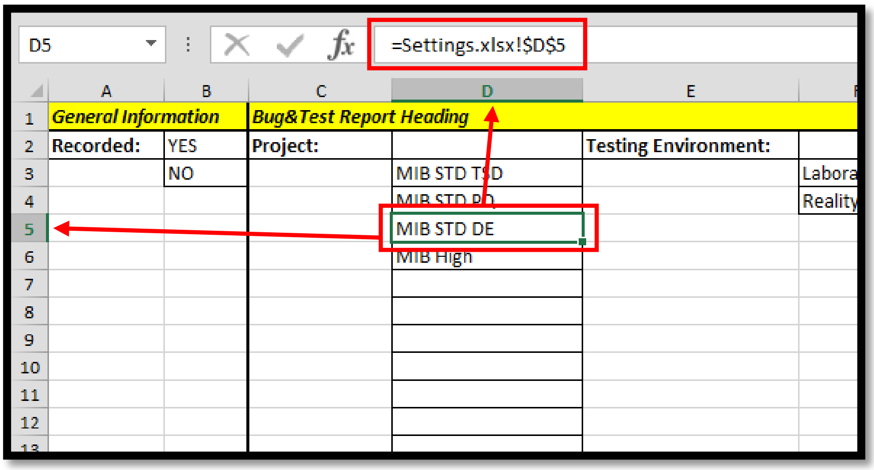


Figure 5.14: The information in the cell “D5” of the Test Scenario’s “Settings (REF)” sheet is transferred from exactly the same location – from cell of the same name (“D5” cell) of the Settings file.

It must be noticed that the *Settings* file should be located in the same root with the all Test Scenario files so that it would be easy to find this out in case of lack of connection between the files. This can happen after moving the *Settings* file to another location, as well as moving the all folder. If this has happened, it is very simple to restore that: at the beginning, when we open a Test Scenario file, there will appear a window, where the Excel application asking about connection, giving us two possibility: to *continue* working with the file, or to *change the connections*. We have to choose the second option ("*change the connections*"), and then we need to brows the original "*Settings*" file, and if there will be "*Ok*" under the "*Connection State*" line, it means that the connections is right, and everything will function. This should be done for the each *Test Scenario* file only once, until the location of the *Settings* file is changed.

5.7 Conclusion

We have finalized the prescribed form of the test scenarios for testing the navigation device performance. All necessary points have been taken into account during creating the standard test scenario form and this master form has been applied for all the already created test scenarios, and the can be easily used in the future, when creating the new scenarios. This form of the test scenario provides all the necessary information about tested scenario:

- ***General Overview*** of the Test Scenario, including all necessary information,

- Detailed description of the each test case in the ***Test Report***, which contains the ***location*** of the test case, ***expectation***, etc., as well as very important for the tester function - ***the algorithm***, which calculates the level of performance of the navigation device.

- ***Bug Report***, which is done according to the Skoda Company's bug register system.

- ***Map*** with pointed numbers of the test cases – this is done for more clear overview, what is the test scenario about.

- ***Settings***, which were described in previous subchapter, is very useful tool to simplify the filling in some lines with specific information, as well as to simplify the updating some data, which, I am sure, is very important.

This is very user-friendly form for test scenarios. Apart of this, using such a form has a very important and useful feature – time saving, which is a vital for all types of business. This is especially important for the test scenarios, which contain many different test cases. As a part of the master form, the ***Test Report*** sheet contains the algorithm for calculating and comparing the performance of the navigation device. This algorithm is described in the next Chapter.

6. Algorithm for the Comparison of Positioning Performance of Car Navigation

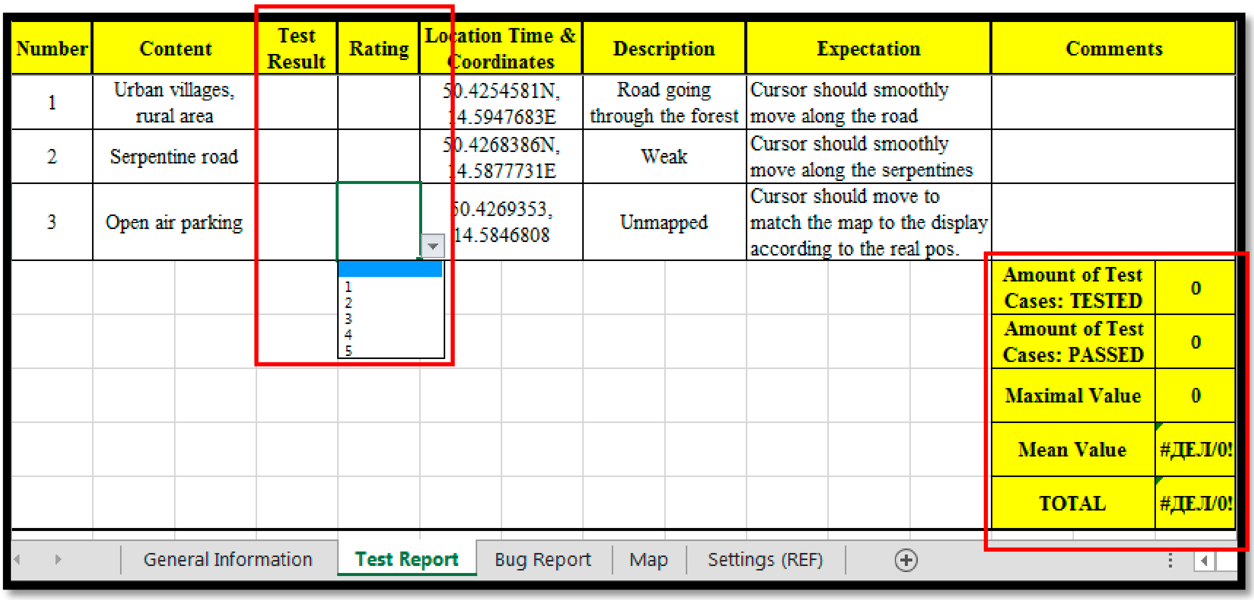
6.1 Introduction

In the current Chapter, I will summarize the mentioned previously necessary points, and based on that I will describe the algorithm for the comparison of positioning performance of car navigation. We have specified the prescribed form for the test scenarios, and as it was mentioned before, the *Test Report* sheet, which is an integral part of the test scenario master form, contains an algorithm for calculating the acceptance of the each test scenario.

I have to notice, that this acceptance calculation algorithm is the main body of the whole algorithm. So, how does it work? Let us turn back to the prescribed form of the test scenario, namely – to the *Test Report* sheet.

6.2 Test Scenario Resulting Algorithm

The Figure below represents a *Test Report* sheet, as a part of the Test Scenario master form. However, let us concentrate on the cells in the red frames. These parameters contains an algorithm for calculation the acceptance of the test scenario.



Number	Content	Test Result	Rating	Location Time & Coordinates	Description	Expectation	Comments
1	Urban villages, rural area			50.4254581N, 14.5947683E	Road going through the forest	Cursor should smoothly move along the road	
2	Serpentine road			50.4268386N, 14.5877731E	Weak	Cursor should smoothly move along the serpentines	
3	Open air parking			50.4269353, 14.5846808	Unmapped	Cursor should move to match the map to the display according to the real pos.	

Amount of Test Cases: TESTED	0
Amount of Test Cases: PASSED	0
Maximal Value	0
Mean Value	#ДЕЛ/0!
TOTAL	#ДЕЛ/0!

Figure 6.1: Key parameters for the algorithm for calculation the acceptance of the test scenario.

As we already know, “Test Result” column has only two options, which are “Passed” of “Failed”. In fact, “Rating” column also tells us, whether the test case passed the testing or failed. However, it contains some kind of numbering system, which defines the level of the positioning performance.

There are five options, and all these are numbers from “one” to “five”. It seems like something similar with an examination marking system in secondary school, but it is a bit stricter, for example, if the pupil gets “4”, it is acceptably, however, it is not excellent so far. In our testing examination system, mark “4” is not acceptable. The table below describes this more in details.

Rating		Resulting Decision
1	Excellent	Passed
2	Good	
3	Sufficient	
4	Insufficient	Failed
5	Very bad	

Table 6.1: Rating of the testing result

Thus, if we examine the test case from “one” to “three”, so the test case has passed the test. If there is “four” or “five” – the test case has failed. As I mentioned before, the key testing point is a positioning, which represents in a cursor movement on a display of the navigation device. The tester in this case takes a teacher’s role, giving marks to the test cases; however, the tester giving marks on his own, based on the actual cursor movement during the testing and his own evaluation, namely – knowledge of the proper functionality of the navigation device. As it was said in my theoretical part, there is no exact rules for testing, and the black box testing is not an exclusion so far.

Now let us have a look at the block, located at the right bottom of the Test Report, which shown on Figure 6.2.

Amount of Test Cases: TESTED	**0**
Amount of Test Cases: PASSED	**0**
Maximal Value	**0**
Mean Value	**#ДЕЛ/0!**
TOTAL	**#ДЕЛ/0!**

Figure 6.2: Test Scenario’s resulting block

This is a resulting block of the whole Test Scenario, but to be more accurate – of the whole test cases, which have been tested. Creating this algorithm, I have taken into account, that there can be such a situations, when it is necessary to test some exact test cases, based on some specific input parameter, for example – if it is necessary to test only the Roundabouts. This is an issue of time, what can be very important to take into account.

Thus, we have in the resulting box several points:

- ***Amount of test cases TESTED***: shows how many test cases have been tested and proceeded. In fact, this block counts the each cell in the column “Rating” with the value, which is more than zero, not depending whether some test case has passed the test or not.

- ***Amount of test cases PASSED***: shows how many test cases has passed the test. This block counts the each cell in the column "Rating" with the value, which is more than zero, BUT less than four. Thus, it gives us a sum of all the test cases, which have passed the test.

- ***Maximal Value***: shows the maximal evaluation of the test cases, as result of the test. This block helps us to recognize, what is the worst evaluation in the whole test scenario.

- ***Mean Value***: shows the mean value of the whole values in the test scenario in the "Rating" column. It probably seems that the mean value does not give any vital result, however, in my opinion, having the mean value it is easy to define the average status of the positioning performance, and the most iterative value among the test cases. The Mean Value, as we know, has simple formula, and in our case, it looks so:

$$\text{Mean Value} = \frac{\text{The sum of values for all the test cases}}{\text{Total amount of the test cases}}$$

- ***TOTAL***: shows us the status of the tested test scenario. Mainly, this shows us – whether all the test cases have passed the test or not. In fact, the block "Total" is based on such a simple formula:

$$\text{Total} = \frac{\text{Amount of test cases PASSED}}{\text{Amount of test cases TESTED}} * 100$$

  The total result counts in percentage, and obviously, if the result is less than 100%, it means that some test case has failed the test. The failed Test Case must be rechecked and reported using a Bug Report.

If we have a look at the resulting block now, we will see some error there in blocks "Mean Value" and "Total":

Amount of Test Cases: TESTED	0
Amount of Test Cases: PASSED	0
Maximal Value	0
Mean Value	#ДЕЛ/0!
TOTAL	#ДЕЛ/0!

Figure 6.3: Errors in "Mean Value" and "Total"

This caused by well-known mathematical reason: it is impossible to divide on zero. Surely, considering the current test scenario, testing has not been undertaken, and there are no results. Therefore – amount of test cases are both TESTED and PASSED is equal to zero, as well as the maximal and mean values.

Thus, we have completed an algorithm for calculation of the positioning performance of the navigation device for the whole test scenario.

6.3 Algorithm for the Comparison of Positioning Performance in Car Navigation

In fact, testing of the positioning performance of the navigation device is quite complicated and major process, requiring a personal analysis and of course – great attention. However, when we speak about the comparison, we face with a very important point: graphical representation of the testing results. These graphical representations can be used with different purposes – meetings, presentations, for personal representation about the performed work, etc. In this part of my Master Thesis I will describe the next important application for achieving the goal and to approach to the topic. As I mentioned before, the main resulting value for the each test scenario is total percentage, representing the summarized passage level of the whole test scenario. However, as we know, the each test scenario can contain many different test cases, for example, the driving route can contain roundabouts, intersection, presence of the tall buildings etc. But here appears another question: how to create the resulting diagram for the comparison of positioning performance, for example, if we test something using a GNSS Simulation system and Record & Playback system, or if we test two or more different units with different applications? In my opinion, that would be right to split the test scenarios by the test case type. Therefore, I have created a resulting excel spreadsheet.

As shown on the figure 6.4, the spreadsheet excel file contains an overall spreadsheet – a list of all the test cases we have included in the test scenarios with exactly the same results calculation algorithm, as the each test scenario has. It means that we are able to know the total percentage value of the set of some specified kind of test cases. The quantity of specified test cases directly depends on the amount of test scenarios we have in the test scenario storage, and surely this can be updated with a new test scenarios and consequently – with a number of test cases.

Overall Test Cases Spreadsheet		Amount of Test Cases: TESTED	Amount of Test Cases: PASSED	Maximal Value	Mean Value	TOTAL		Navexp	EU TS Storage	EU Matrix
Area without obstacles		0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
Narrow streets in cities		0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
Parallel roads	<15 m	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
	15 - 30 m	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
	>30 m	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
Serpentine roads	Extreme	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
	Medium	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
	Weak	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
Multipath	>100 m	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
	100 - 50 m	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
	<50 m	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
Tunnels	with forks	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
	>10 km	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
	5 - 10 km	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
	1 - 5 km	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
	<1 km	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				
Underground	>5 levels	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!				

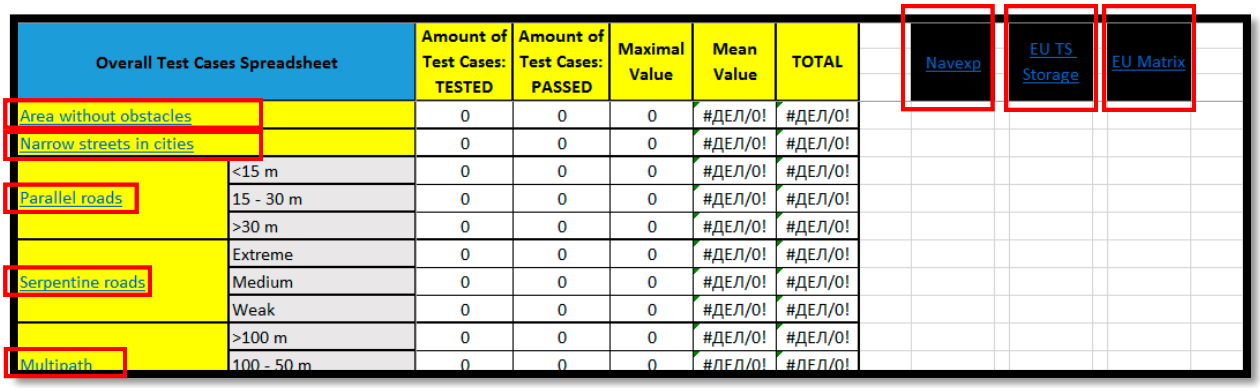
Overview | TESTED | Plane area without obstacles | Narrow streets in cities | Parallel Roads | Serpentines | Multipath | Tunnels | Garages&Parkin

Figure 6.4: Overall test cases spreadsheet

It must be noted, that errors in cells “Mean Value” and “TOTAL” are caused exactly by the same reason, as described in previous subchapter.

To simplify the Spreadsheet file operation, I have implemented the buttons in the excel file using *hyperlink* function.

In the overview sheet, all the listed test cases are represented as buttons. For example, if we need to see the results of some test case in details, we do not need to scroll right to find out a corresponding sheet, we just click on some test case name, and will immediately move to the appropriate sheet.



Overall Test Cases Spreadsheet		Amount of Test Cases: TESTED	Amount of Test Cases: PASSED	Maximal Value	Mean Value	TOTAL
Area without obstacles		0	0	0	#ДЕЛ/0!	#ДЕЛ/0!
Narrow streets in cities		0	0	0	#ДЕЛ/0!	#ДЕЛ/0!
Parallel roads	<15 m	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!
	15 - 30 m	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!
	>30 m	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!
Serpentine roads	Extreme	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!
	Medium	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!
	Weak	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!
Multipath	>100 m	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!
	100 - 50 m	0	0	0	#ДЕЛ/0!	#ДЕЛ/0!

Figure 6.5: The buttons are highlighted by the red frame

Furthermore, three more buttons have been implemented:

- ***Navexp***: it is possible to run the Navexp application from the overview of the Spreadsheet.

- ***EU TS Storage***: this button opens the root folder with all the test scenario files, containing both excel and “kml” files, as well as Settings file and the matrix database itself.

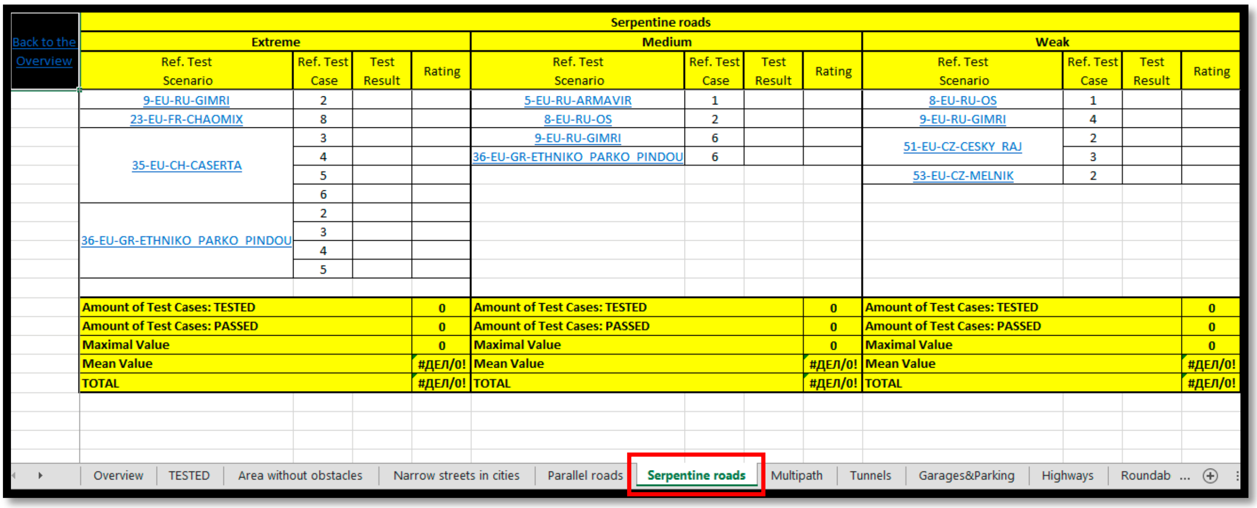
- ***EU Matrix***: this button opens the EU matrix database, which is an integral instrument for Navexp application’s functionality.

Thus, apart of the overall spreadsheet, the excel file contains the number of auxiliary sheets, which contain the result information in details. The number of the auxiliary sheets is equal to the number of test cases. The all values, which are represented in the overall spreadsheet, are linked from these auxiliary sheets, however, the values that is shown in those test case sheets by-turn linked from the test scenario files. Consequently, when the test is done and the test report is filled, the resulting values in the Spreadsheet with its appropriate test case and its sheet are automatically updated.

However, let us have a look at one of the sheets we have in the Spreadsheet excel file, and as an example we will consider such a test case type as “Serpentine roads”.

As the serpentine roads are specified as extreme, medium and weak, as well as the most of other parameters are specified by complexity, exactly such a division has been done in the overall spreadsheet to define the positioning device behavior on different types of serpentine road.

As we see in the Figure 6.6, the calculation algorithm for the each type of serpentine road is applied.



Back to the Overview	Serpentine roads											
	Extreme				Medium				Weak			
	Ref. Test Scenario	Ref. Test Case	Test Result	Rating	Ref. Test Scenario	Ref. Test Case	Test Result	Rating	Ref. Test Scenario	Ref. Test Case	Test Result	Rating
	9-EU-RU-GIMRI	2			5-EU-RU-ARMAVIR	1			8-EU-RU-OS	1		
	23-EU-FR-CHAOMIX	8			8-EU-RU-OS	2			9-EU-RU-GIMRI	4		
	35-EU-CH-CASERTA	3			9-EU-RU-GIMRI	6			51-EU-CZ-CESKY_RAJ	2		
		4			36-EU-GR-ETHNIKO_PARKO_PINDOU	6				3		
		5							53-EU-CZ-MELNIK	2		
		6										
	36-EU-GR-ETHNIKO_PARKO_PINDOU	2										
		3										
		4										
		5										
	Amount of Test Cases: TESTED			0	Amount of Test Cases: TESTED			0	Amount of Test Cases: TESTED			0
	Amount of Test Cases: PASSED			0	Amount of Test Cases: PASSED			0	Amount of Test Cases: PASSED			0
	Maximal Value			0	Maximal Value			0	Maximal Value			0
	Mean Value			#ДЕЛ/0!	Mean Value			#ДЕЛ/0!	Mean Value			#ДЕЛ/0!
	TOTAL			#ДЕЛ/0!	TOTAL			#ДЕЛ/0!	TOTAL			#ДЕЛ/0!

Overview | TESTED | Area without obstacles | Narrow streets in cities | Parallel roads | Serpentine roads | Multipath | Tunnels | Garages&Parking | Highways | Roundab ...

Figure 6.6: Serpentine roads sheet

The “Serpentine roads” sheet contains:

- ***Back to the Overview** button*: helps us to switch back to the first sheet, which is the “Overview” of the spreadsheet.

- ***Ref. Test Scenario***: referenced Test Scenario name; shows us, which test scenario contains the observed test case(s). Furthermore, the Test Scenario names contains *hyperlinks* to the corresponding Test Scenario files. The same principle, as Test Case names in the Overview sheet: *clicking on the Test Scenario filename, the proper file automatically opens*. In this case, clicking on the Test Scenario name, the appropriate Test Scenario file opens.

- ***Ref. Test Case***: referenced Test Case number; shows us, what the number of the observed test case in the Test Scenario is.

- ***Test Result*** and ***Rating***: this data is linked from the Test Scenario files’ Test Report sheet and is being automatically updated when the test cases result values there changing.

The same form and principle are applied for all the sheets containing the detail information about some exact test case, and the final value in the in the bottom block is represented in overall spreadsheet.

Speaking about updating a number of test cases, we have to take into account the Navexp filtering application as well, which has been built according to our current testing requirements, and this must be noted, that the input testing parameters on the main panel signify the test cases themselves.

Thus, using such a spreadsheet form of the algorithm for comparison of positioning performance, we can easily create a graphical representation of the results for every single kind of test case with its parameters, using different methodologies of testing the navigation device performance and different equipment.

Note: the spreadsheet form has specific feature: it is built on a basis of links and hyperlinks to the Test Scenario files and their storage, Navexp application and the matrix database. Therefore, root changing for the all or at least one of used resources or components means that it will be necessary to update the link routing to provide proper functionality of the algorithm.

7. Positioning Performance Testing and Testing Results Analysis

7.1 Introduction

This Chapter is dedicated to the comparison of testing results in real and simulated environment. In this Chapter, the real measures of testing of the navigation device performance will be undertaken. We will concentrate on some concrete test cases, which are located in the Czech Republic. The step-by-step operations will be described in this Chapter, as well as their results will be published and discussed.

The main goal of this Chapter is to prove the usability and useless of the prescribed form of test scenarios, the computation algorithm, and the methodology generally, providing and comparing the test results for two different navigation units, tested using the real-world environment – the actual (real test drive) and the recorded one.

7.2 Procedures Undertaken

During the measures, the following steps have been undertaken.

7.2.1 Test Scenarios Preparation

First, we have to define the concrete test cases for testing. We decided to test *Underground parking (2-4 levels)*, *Parallel roads (15 – 30 m)* and *Roundabouts (Circle)*, which are located in the Czech Republic. With help of Navexp application, as shown below, we have several test scenarios filtered.

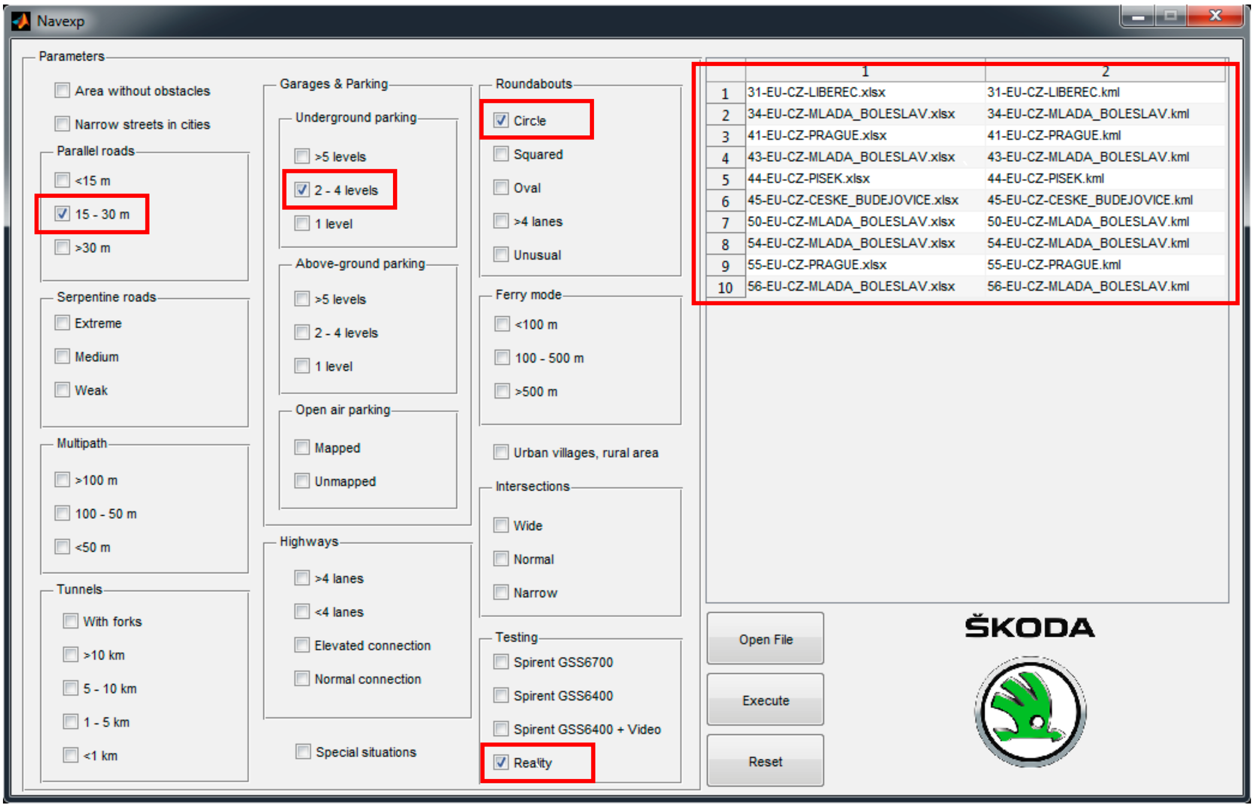


Figure 7.1: Filtering results

As it is shown in the resulting list on the display, we have the all test scenarios located in the Czech Republic. Due to our location and time limit, we have decided to record those scenarios, which are the closest ones to the laboratory, namely – in and/or nearby Mlada Boleslav.

Thus, we have the following test cases, corresponding to the listed test scenarios.

Parallel roads:

- 34-EU-CZ-MLADA_BOLESLAV

Circle roundabouts:

- 54-EU-CZ-MLADA_BOLESLAV

2-4-levels underground parking:

- 56-EU-CZ-MLADA_BOLESLAV

Even though some of the test scenarios contain other different test cases, we will make recordings for all the test scenarios including all the test cases, and we will test them as well.

Now let us summarize the all test cases, which we finally have to test:

34-EU-CZ-MLADA_BOLESLAV:

- Circle Roundabouts: three test cases
- Parallel roads (<15 m): two test cases

54-EU-CZ-MLADA_BOLESLAV:

- Elevated connection to the highway: one test case
- Parallel roads (15 – 30 m): one test case

56-EU-CZ-MLADA_BOLESLAV:

- Underground parking (2 – 4 levels): one test case

The map of the each test scenario has been printed out in A4 paper format in color to simplify the recording procedure of the real-world environment during the test drive.

7.2.2 Testing Details

Before the start of recording the data, we have defined a test plan. Thus, we have three test scenarios, and we will apply them separately for both tested navigation units.

We have filled the important information about the testing into the Test & Bug Report headings in advance.

ŠKODA	Test Scenario :	**ID-CONTINENT-COUNTRY-CITY**	Testing Duration :	**hh:mm:ss - hh:mm:ss**
	Test Report No. :		Hardware Ver. :	
	Location :	**Country, City/Town**	Software Ver. :	
	Date :	**dd.mm.yyyy**	Map Vers. :	
	Project :		Test Result :	
	Testing Environment :		Tested By :	
www.skoda-auto.cz	Testing Equipment :			

Figure 7.2: Test & Bug Report headings

This have been done for the all chosen test scenario, namely we have specified the **Date**, **Project**, **Testing Environment**, **Testing Equipment**, **Hardware Version** and **Test Engineer**. Software Version and the Map Version can be added depending on the requirements, but mainly optionally. In our case, we can ignore that. The day of testing is **22.05.2015**, and the Testing Equipment we have chosen is **SK371** (internal project specification), as we have decided to record the data and test the navigation device performance for SK371 Project, which is related to **Skoda Octavia**. For more detail information, please, see ***Appendix B***.

7.2.3 Test-Drive

It is quite enough to have the information, which carry test scenario files, such as map, location details and start point coordinates to easily identify this and reach the start point to start recording the real-world conditions and the video for the testing the navigation device performance.

As I have mentioned previously, we have decided to test the navigation units, which are mainly suitable for the latest model of Skoda Octavia car. Appropriately, the test drive has been performed using Skoda Octavia.

Figure 7.3: Skoda Octavia RS

Source: taken from the official Skoda Auto web page.

The GSS6400 device has been connected to the CAN system of the vehicle, and the three cameras have been installed to capture the navigation data from the meter panel and the on-board navigation application.

Due to the confidentiality agreement with Skoda Auto Company, it was impossible to provide the pictures from the vehicle or from the laboratory to explain it more in detail. However, the typical picture after the recording test-drive is represented on the screenshot on Figure 7.4.

Figure 7.4: MIB Logger screenshot

MIB Logger is a specific internal company software with the main purpose for recording, playing back and analysis of the synchronized video taken from the vehicle for the positioning performance testing. GSS6400 RPS by-turn was controlled from the front panel to record the real-world conditions.

During the test-drive, I have made notes on the printed maps, such as start time, time left from the recording start up, end-point of recording, test duration etc. Immediately, after the recording, all the necessary information has been added into the Test Scenario files map sheets. The information is about the recording process mainly to recognize the location in time and to simplify the testing process.

I have to note, that the time from cameras and time from the GSS6400 by some reasons can be different, but this does make sense only during the testing process – obviously, if we test the unit using recorded environment using RPS, we have to specify the time information, which has been set in the device. Correspondingly, if we test the device, which has been installed and used during test-drive (real drive testing), we use the time information from videos.

One very important note: the recorded environment can be used for testing of several navigation units. Therefore, the time information, as well as the file name, which has been extracted from the GSS6400 RPS with the all necessary data after recording must be added into the all test scenarios, which has been recorded during the test-drive.

Thus, the Matrix Database file, the General Information, Map and headings of Test and Bug Report sheets of the each Test Scenario file have been updated. Now it is clear that the Test Scenario is recorded; and can be filtered with testing possibility parameter "Spirent GSS6400 + Video". Test & Bug Report headings, as well as the Maps are shown in ***Appendix B***, where is the recorded data represented, **not the tested one**.

Once we have all necessary preparations done and obtained the all required data, we can start our testing processes.

7.2.5 Navigation Device Performance Test Results Analysis

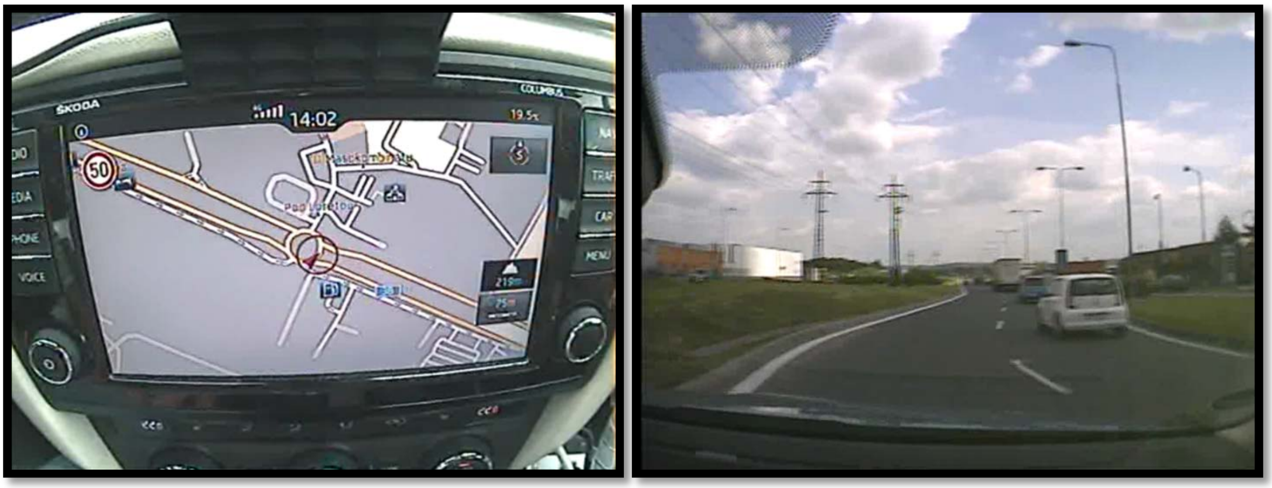
All the testing have been done visually using both replaying the recorded by GSS6400 RPS real-world environment and video files, which were capturing all necessary points described previously. I would note, that the performed testing have given us quite interesting results. But, first let us begin from the real drive testing, namely – testing, which has been done using the video files.

7.2.5.1 Real Drive Testing Results Analysis

The tested unit in this case is **MIB HIGH**. The results, obtained during the real test drive testing are represented. Let us examine the each test scenario separately.

34-EU-CZ-MLADA_BOLESLAV

During the testing, some insignificant bugs have been detected. Mostly it was rough cursor movement and delays. The worst performance of the navigation device has been detected in the Test Case #2, at the roundabout. As shown on the figures below, the cursor was showing as the



vehicle is moving around the roundabout, however, at that time it was on the roadway.

Figures 7.5A and 7.5B: The cursor and the vehicle position at the same time

In spite of this, I have considered this cursor behavior as "**3**" by our accepted rating range, as the cursor has jumped to the actual vehicle position in half a seconds. Thus, the Test Scenario has passed with the **Maximal Value 3** and the **Mean Value 1.67**.

Amount of Test Cases: TESTED	6
Amount of Test Cases: PASSED	6
Maximal Value	3
Mean Value	1,67
TOTAL	100,0%

Figure 7.6: Test Scenario has passed

54-EU-CZ-MLADA_BOLESLAV

The same as previously, the worst cursor behavior has been detected in one test cases, namely – in Test Case #2. It was showing the wrong vehicle position in maybe less than half a second, thus having short delay.

Figures 7.7A and 7.7B: The cursor and the vehicle position at the same time

From the Figures below it is clear that the cursor represents the incorrect vehicle movement trajectory. However, in this case I take into account the vehicle speed, which was around 80 km/h, and the fact, that in half a second the cursor jumped to the proper position.

Thus, the Test Scenario has passed with a **Maximal Value 3** and the **Mean Value 2.5**.

Amount of Test Cases: TESTED	6
Amount of Test Cases: PASSED	6
Maximal Value	3
Mean Value	1,67
TOTAL	100,0%

Figure 7.8: Test Scenario has passed

56-EU-CZ-MLADA_BOLESLAV

This Test Scenario, containing one Test Case only has shown the worst result among the all ones all together. Taking into account that fact, that the vehicle has entered into the underground parking in 1 minute and 40 seconds after starting up and spent there 3 minutes, driving around the parking, in my opinion, the performance had to correspond at least to the rating “3”. However, let us consider this bug a bit more in detail.

First, I have to note, that when the car has entered into the parking, the positioning performance was quite acceptable. But, roughly in a minute, the cursor had performed the strange movements, jumping from the road to road, which are actually on the ground. Cursor did not represent the actual vehicle movement trajectory and position. Wrong positioning representation was observing when the vehicle was approaching to the exit, however, when the vehicle has left the garages, simply saying – when the navigation unit has started to receive the satellite signals, the cursor movement has normalized. Nevertheless, the Test Scenario is failed with a **Maximal Value 4**, as the vehicle movement trajectory was not represented by the cursor.

Here are two screenshots, explaining the positioning performance then entering and leaving the parking.

Figures 7.9A and 7.9B: The vehicle position while entering the parking

Figures 7.10A and 7.10B: The vehicle position while leaving the parking

As we see from the Figures 7.10, the cursor was showing the wrong position when the vehicle was leaving the underground parking.

Thus, the **MIB HIGH** navigation unit has passed two test scenarios, and has failed the one. The Test Result for the each Test Scenario is represented in ***Appendix C***.

Amount of Test Cases: TESTED	1
Amount of Test Cases: PASSED	0
Maximal Value	4
Mean Value	4,00
TOTAL	0,0%

Figure 7.11: Test Scenario has failed.

7.2.5.2 RPS Testing Results Analysis

During the test drive, which has been shown previously, at the same time we have recorded that real-world environment. We have used that recording to test another navigation unit, which has shown almost similar results, as previous one.

During this type of testing, the **MIB STD DE** unit has been tested. Unfortunately, there is not possibility to make a video from the tested on-board computer in laboratory due to the confidential agreement, but several screenshots from the display have been presented.

<u>34-EU-CZ-MLADA_BOLESLAV</u>

The examined Test Scenario has fully passed the test with the **Maximal Value 1**, which shows the best result. Indeed, the cursor movement was properly represented the vehicle actual movement trajectory, as well as the expectations of the every single test case.

Amount of Test Cases: TESTED	7
Amount of Test Cases: PASSED	7
Maximal Value	1
Mean Value	1,00
TOTAL	100,0%

Figure 7.12: Test Scenario has fully passed the test

<u>54-EU-CZ-MLADA_BOLESLAV</u>

This test scenario has also passed the test and the positioning performance has shown quite good result with the **Maximal Value 2**. Namely – the Test Case #2. I have applied the mark **2** to this case because of the a little bit rough cursor movement. However, the cursor was representing the actual vehicle location.

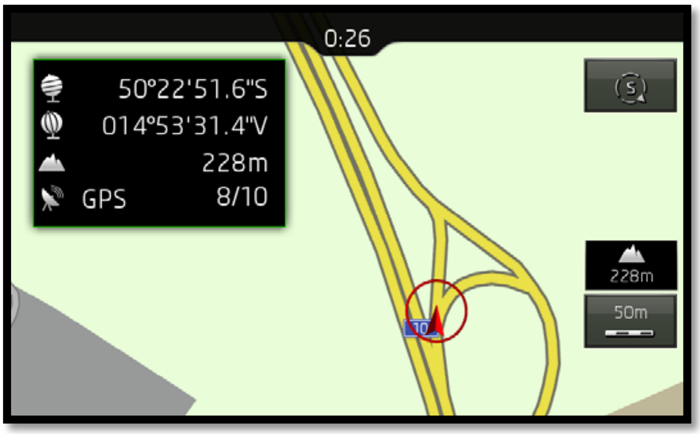


Figure 7.13: Slightly rough cursor movement in Test Case #2.

Amount of Test Cases: TESTED	7
Amount of Test Cases: PASSED	7
Maximal Value	1
Mean Value	1,00
TOTAL	100,0%

Figure 7.14: Test Scenario has passed.

56-EU-CZ-MLADA_BOLESLAV

Using the RPS data, this test scenario totally failed the test. The scenario has the worst rating mark at all, which is **5**. Cursor did not represented the actual vehicle position during movement in the underground parking. Furthermore, the cursor was showing the false vehicle position, and while leaving the garages, cursor was showing the wrong position. MIB HIGH in this case was showing slightly better result. The couple of screenshots are represented below.

Figures 7.15A and 7.15B: Cursor is outside the parking location. The real parking location is highlighted by the green hatched frame.

Thus, here is test a result.

Amount of Test Cases: TESTED	**1**
Amount of Test Cases: PASSED	**0**
Maximal Value	**5**
Mean Value	**5,00**
TOTAL	**0,0%**

Figure 7.16: Test Scenario has totally failed.

As well as the **MIB HIGH**, **MIB STD DE** has passed two test scenarios and failed one. The test report sheets fort the each test scenario is represented in ***Appendix C***.

7.3 Comparison of Testing Results

After the testing is done, we can compare the results, which we have obtained during testing of two different units in two environments, using the each test scenarios for the each unit.

Using created in advance the Overall Spreadsheet it is possible to have an overall test case information. As we know, once the resulting value is changed, it is automatically changed in the Spreadsheet as well. As we decided to use the test scenario files from database for information only now, we have stored the tested scenario files in another folder with a specific name. The same we can easily do in a spreadsheet as well: while we obtained the values, we can easily copy them, create a new excel sheet and paste the values there, but pasting not as formulas, but as values, and of course filter unused test cases, and we get such a table as shown on the figures below.

Real Test Drive + MIB HIGH Test Results (27.05.2015)

Overall Test Cases Spreadsheet			Amount of Test Cases: TESTED	Amount of Test Cases: PASSED	Maximal Value	Mean Value	TOTAL
Parallel roads		<15 m	2	2	2	1,5	100%
		15 - 30 m	1	1	3	3	100%
Garages & Parking	Underground parking	2 - 4 levels	1	0	4	4	0%
Highways		Elevated connection	1	1	2	2	100%
Roundabouts		Circle	4	4	3	1,75	100%

Figure 7.17: Test Results of the Real Test Drive

GSS6400 RPS + MIB STD DE Test Results (27.05.2015)

Overall Test Cases Spreadsheet			Amount of Test Cases: TESTED	Amount of Test Cases: PASSED	Maximal Value	Mean Value	TOTAL
Parallel roads		<15 m	2	2	1	1	100%
		15 - 30 m	1	1	2	2	100%
Garages & Parking	Underground parking	2 - 4 levels	1	0	5	5	0%
Highways		Elevated connection	1	1	1	1	100%
Roundabouts		Circle	4	4	1	1	100%

Figure 7.18: Test Results of the testing performed in Recorded Environment (with help of GSS6400 Record and Playback System)

Having these tables, we can easily compare the results of two different units, tested in two different approaches. It is clear shown, that the result is mainly the same: both units have passed test in two test scenarios, and failed in the same one scenario.

However, if we take attention on the Maximal and Mean values, we will see some difference: it is clearly shown, that in the most cases, MIB STD DE unit, which has been tested with the recorded environment, give much better values, than MIB HIGH, which has been tested in reality. In spite of that, there is a considerable exclusion: MIB HIGH has the worst result when was tested in underground parking in real test drive.

The comparison diagrams, comparing the total percentage, maximal and mean values, are represented in the ***Appendix D***.

Conclusion and Future Recommendations

In this Master Thesis, the basic principles and techniques of GNSS with its main components and testing theory, as well as the existing methodology and technology for the navigation device performance testing have been described. The necessity of all this theoretical knowledge consists in a background of the whole system, how does it works and how does it mainly done.

During the research, the important points have been noted, and the main goals of the Master Thesis have been achieved.

First, we have considered the test scenario as a set of specific test cases. Indeed, this is very optimal definition for our type of testing. Thus, we can consider test scenario as a tested route or path, and the different road sections such as roundabouts, intersection, etc. as set of different test cases.

The Navexp filtering algorithm has been built to simplify the tester's work to select the proper test scenario according the selected testing parameters. These parameters correspond to the test case names. Indeed, this application considerably reduces the time and provides the clear overview of the test scenarios, which does the tester have in the storage. The application is matrix based, and works with the matrix database, which is very simply to update, when adding new test scenario. All the procedures have been described in this thesis as well.

The prescribed master form for the test scenario has been applied. Apart of the specific data, such as project name, testing equipment etc. (headings of Test & Bug Report), the applied master form for test scenario also carries the general, and of course – integral information for such type of testing, which is represented in Test Report sheet – test cases specifications, as well as their location coordinates, the expectation, and result rating. Rating is a key parameter of the all test scenarios, rating defines whether the test case passed the test or failed. This rating mode directly connected with the computation algorithm of test passage, which measures in percentage. This computation algorithm is also can summarize the whole picture state.

To obtain the whole state picture, the general spreadsheet has been created. This is quite complex file, consists of links and hyperlinks. The key point which are shown there are: test cases, and the relative test scenarios. The goal consists in a test result value for the every single test case. Using this spreadsheet, it is easy to represent graphically the state of the navigation device performance. It is also possible to compare the different testing results, for example, between several units and using different testing approaches, as was shown in our real example in Chapter 7.

Indeed, in Chapter 7 we have proved that this methodology works pretty well, significantly simplifies the tester's job, reducing the time spent, and really helps to avoid different accidental mistakes, which can do every person. The testing technique, described in this thesis makes testing relatively simple, and at the same time – clear. In fact, creating test scenarios is quite a time consuming job. However, the prescribed master form for test scenarios positively affects on that, making the process simply and even interesting.

Thus, we can conclude, that the suggested methodology for testing the navigation device performance is very useful, time reducing and comprehensive.

For the future, I would recommend to use more test scenarios to get more complex picture, describing the navigation devices performance state. I also recommend to keep the database updated properly, periodically update the test scenarios (as well as the matrix database, as an integral part of Navexp application), according to the actual traffic state (new roads, highways, road sections etc.). It would also be nice to develop the Navexp software to widen the filtering options to get the faster and more user-friendly access to the already used and stored test scenarios, and of course to be in touch with new testing technologies. Indeed, new testing technologies appear year by year, as the navigation device performance testing is quite important and topical problem of today.

References

A "Real-World" Navigation and Positioning Testing. (2011, February). *Coordinates - Magazine on Positioning, Navigation and Beyond*, p. 41.

Agilent Technologies Inc. . (2010). GPS Test Requirements. In *GPS Receiver Testing Application Note* (p. 4). USA.

Bernhard Hofmann-Wellenhof, H. L. (2008). Satellite-based positioning. In *GNSS – Global Navigation Satellite Systems* (pp. 6 - 7). Wien: Springer-Verlag.

Cem Kaner, J. P. (May, 2003). *What Is a Good Test Case?* (pp. 2 - 7). Florida: STAR East.

Data Sets Launched for GSS6400. (2012). *The Business*, p. 27.

Etienne Favey, A. S. (2010, August). Intelligent solutions for modern urban navigation. *u-blox' Dead Reckoning for Automotive Applications*, p. 4.

Gopal Dommety, R. J. (1998). Current General Applications of GPS. In *Potential Networking Applications of Global Positioning Systems (GPS)* (pp. 13 - 14). Ohio: The Ohio State University.

IEEE Standard 610. (1990). *IEEE Standards Collection: Software Engineering 1994 Edition*. Institute of Electrical & Electronics.

J. Sanz Subirana, J. J.-P. (2013). GNSS Architecture. In *GNSS Data Processing, Vol. I: Fundamentals and Algorithms* (p. 18). Leiden: ESA Communications.

Johannes Ryser, M. G. (1999). In *A Practical Approach to Validating and Testing Software Systems Using Scenarios* (pp. 1 - 4). Zurich: SR Institute.

Kaner, C. (June, 2003). An Introduction to Scenario Testing. (pp. 4 - 9). Florida Tech.

Mohd. Ehmer Khan, F. K. (2012). A Comparative Study of White Box, Black Box and Grey Box Testing Techniques. *(IJACSA) International Journal of Advanced Computer Science and Applications Vol. 3*, 12 - 14.

Ngah, E. N. (2006). Car GPS Navigation Systems. In *GPS Technology: Optimizing Car Navigation* (pp. 14 - 16). Amsterdam: Vrije Universiteit Amsterdam.

Omarah Omar Alharaki, F. S. (2010, September 6). The Integration of GPS Navigator Device with Vehicles Tracking System for Rental Cars Firms. *(IJCSIS) International Journal of Computer Science and Information Security, Vol. 8*.

Patton, R. (2001). *Software Testing.* SAMS.

S.M.K Quadri, S. U. (2010, September). Software Testing – Goals, Principles, and Limitations. *International Journal of Computer Applications (0975 – 8887), Vol. 6*, 7- 10.

Spirent Communications plc. (2009). Spirent GSS6700 with SimGEN. *Fully Flexible Multi-GNSS Simulation System*.

Spirent Communications plc. (2009). Spirent GSS6700 with SimREPLAY. *Scenario Replay Multi-GNSS Simulation System*.

Spirent Communications plc. (2009, September). Testing Real-World A-GNSS Performance in the Lab using ULTS. *Application Note*.

Spirent Communications plc. (2012, December 13). SIMGEN SOFTWARE USER MANUAL. *SOFTWARE FOR THE SPIRENT RANGE OF SATELLITE NAVIGATION SIMULATOR PRODUCTS*, p. 2.5.

Spirent Communications plc. (2012). Spirent GSS6400 RPS. *Record & Playback Systems*.

Spirent Communications plc. (2013). GSS6700 GNSS Simulator Product Brochure. *Flexible Multi-GNSS Constellation Simulator Systems*.

APPENDICES

Appendix A

List of Abbreviations

CD-ROM – Compact Disk Read-Only Memory

DR – Decoder

DUT – Device Under Test

DVR – Dynamic Vehicle Routing

EGNOS – European Geostationary Navigation Overlay Service

GIS – Geographical Information Systems

GLONASS – Globalnaya Navigatsionnaya Sputnikovaya Sistema (Rus.)

GNSS – Global Navigation Satellite System

GPS – Global Positioning System

INS – Inertial Navigation System

KML – Keyhole Markup Language

KPM – Konzern-Problem-Management

Navexp – Navigation Expertize

NMEA - National Marine Electronics Association

OCXO – Oven-Controlled Crystal Oscillator

PLTS – Position Location Test System

QZSS – Quasi Zenith Satellite System

R&D – Research and Development

RPS – Record & Playback System

RF – Radio Frequency

SATA – Serial Advanced Technology Attachment

SBAS – Satellite Based Augmentation System

ULTS – Universal Location Test System

UML – Unified Modeling Language

WAAS – Wide Area Augmentation System

Wi-Fi – Wireless Fidelity

Appendix B

This annex carries recording information.

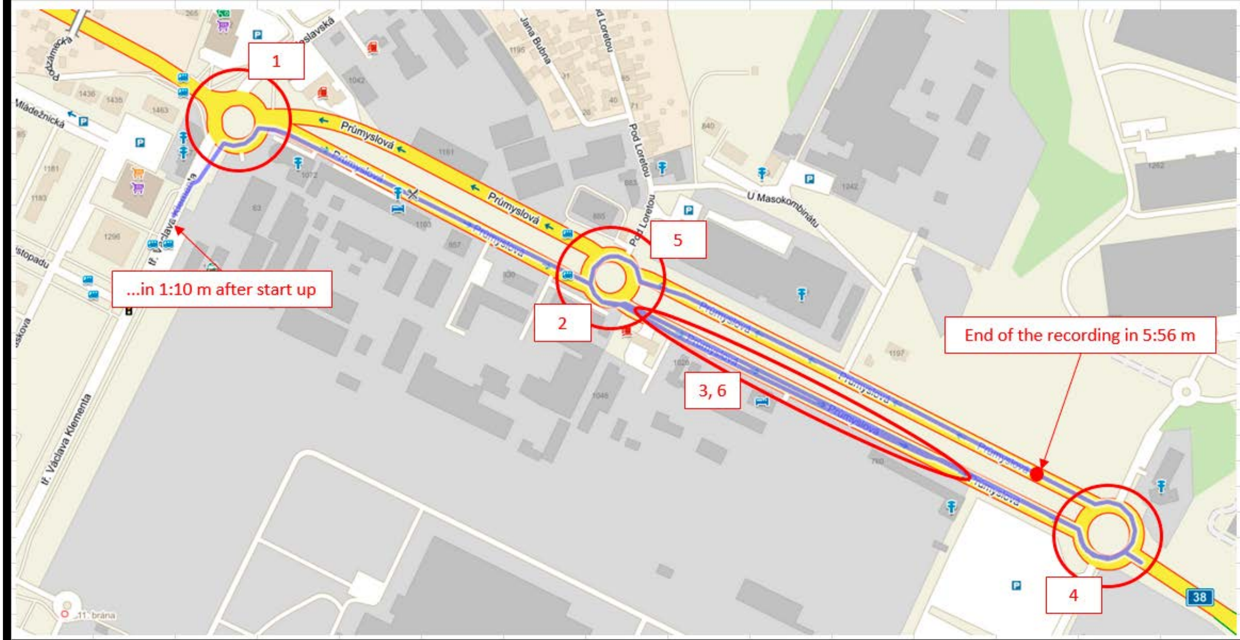
34-EU-CZ-MLADA_BOLESLAV

Test & Bug Report Headings

ŠKODA	Test Scenario :	**34-EU-CZ-MLADA_BOLESLAV**	Testing Duration :	**13:36:15 - 13:42:41 (6 min. 36 sec.)**
	Test Report No. :		Hardware Ver. :	**X41**
	Location :	**Czech Republic, Mlada Boleslav**	Software Ver. :	
	Date :	**22.05.2015**	Map Vers. :	
	Project :	**MIB High**	Test Result :	
	Testing Environment :	**Reality**	Tested By :	**Elshan Khalilov**
www.skoda-auto.cz	Testing Equipment :	**SK371**		

Map

ŠKODA	Test Scenario :	34-EU-CZ-MLADA_BOLESLAV
	Comments:	Duration: 6 min. 36 sec.



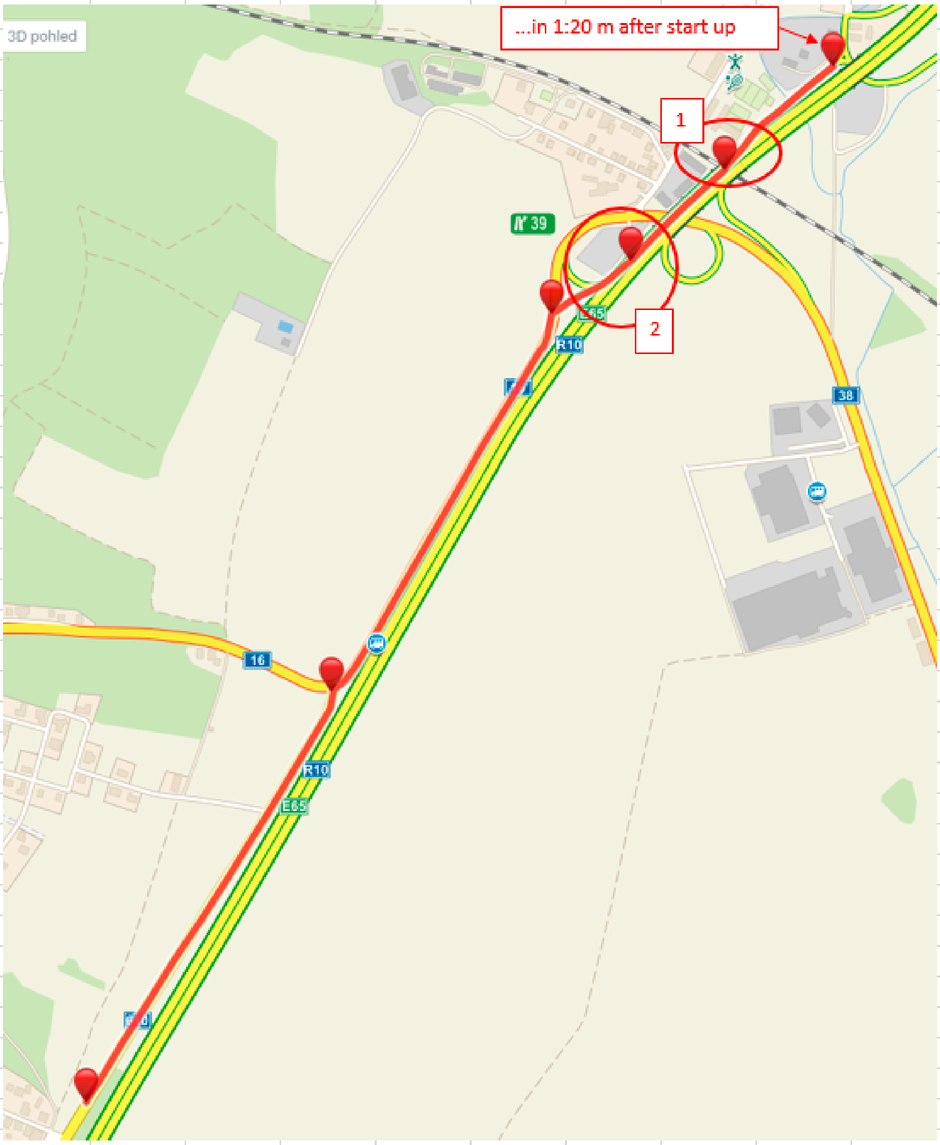
54-EU-CZ-MLADA_BOLESLAV

Test & Bug Report Headings

ŠKODA	Test Scenario :	**54-EU-CZ-MLADA_BOLESLAV**	Testing Duration :	**14:34:16 - 14:38:04**
	Test Report No. :		Hardware Ver. :	**X41**
	Location :	**Czech Republic, Mlada Boleslav**	Software Ver. :	
	Date :	**22.05.2015**	Map Vers. :	
	Project :	**MIB High**	Test Result :	
	Testing Environment :	**Reality**	Tested By :	**Elshan Khalilov**
www.skoda-auto.cz	Testing Equipment :	**SK371**		

Map

ŠKODA	Test Scenario :	54-EU-CZ-MLADA_BOLESLAV
	Comments:	Duration: 3 min. 48 sec.



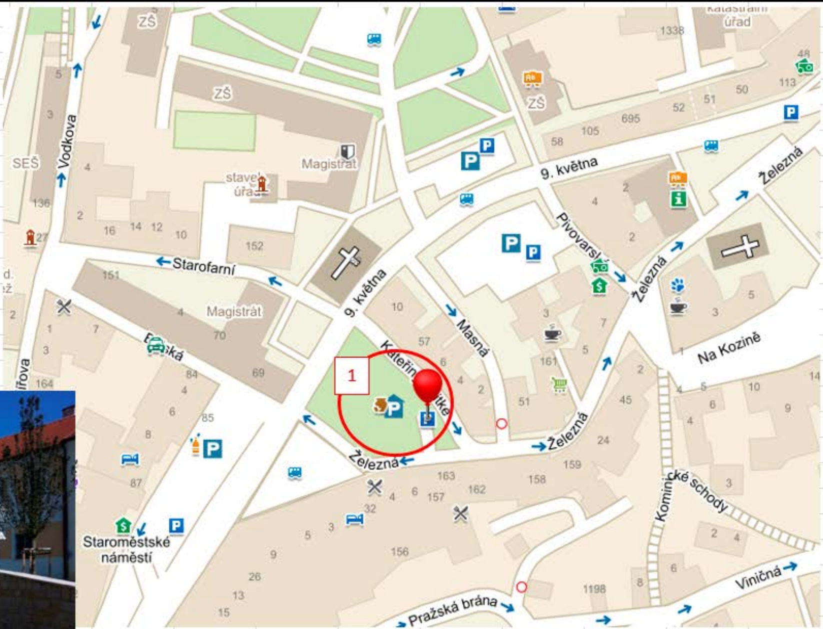
56-EU-CZ-MLADA_BOLESLAV

Test & Bug Report Headings

ŠKODA				
	Test Scenario :	56-EU-CZ-MLADA_BOLESLAV	Testing Duration :	13:49:49 - 13:55:51 (6 min. 2 sec.)
	Test Report No. :		Hardware Ver. :	X41
	Location :	Czech Republic, Mlada Boleslav	Software Ver. :	
	Date :	22.05.2015	Map Vers. :	
	Project :	MIB High	Test Result :	
	Testing Environment :	Reality	Tested By :	Elshan Khalilov
www.skoda-auto.cz	Testing Equipment :	SK371		

Map

ŠKODA			
	Test Scenario :	56-EU-CZ-MLADA_BOLESLAV	
	Comments:	Duration: 6 min. 2 sec.	Entrance to the parking in 1:40 m after start up
			Exit from the parking in 4:45 m



Appendix C

Test Results: Real Test Drive & MIB HIGH unit

34-EU-CZ-MLADA_BOLESLAV

Test Scenario :	34-EU-CZ-MLADA_BOLESLAV	Testing Duration :	8 min. 16 sec.
Test Report No. :	HIGH01	Hardware Ver. :	X41
Location :	Czech Republic, Mlada Boleslav	Software Ver. :	
Date :	27.05.2015	Map Vers. :	
Project :	MIB High	Test Result :	Passed
Testing Environment :	Reality	Tested By :	Elshan Khalilov
Testing Equipment :	SK371		

ŠKODA www.skoda-auto.cz

Number	Content	Test Result	Rating	Location Time & Coordinates	Description	Expectation	Comments
1	Roundabout	Passed	2	50.432352, 14.924114	Circle	Cursor should smoothly move around the roundabout	Cursor was moving a little bit roughly
2	Roundabout	Passed	3	50.430633, 14.930584	Circle	Cursor should smoothly move around the roundabout	Cursor was moving roughly and jumped to the actual vehicle position with a very short delay
3	Parallel Roads	Passed	1	50.4303278N, 14.9308169E	<15 m	Cursor should move along the proper road way	Cursor has smoothly moved to the proper driving way
4	Roundabout	Passed	1	50.427979, 14.938890	Circle	Cursor should smoothly move around the roundabout	Cursor was moving smoothly
5	Roundabout	Passed	1	50.430633, 14.930584	Circle	Cursor should smoothly move around the roundabout	Cursor was moving smoothly
6	Parallel Roads	Passed	2	50.4303278N, 14.9308169E	<15 m	Cursor should move along the proper road way	Cursor has jumped to the actual vehicle position a very with a short delay

Amount of Test Cases: TESTED	6
Amount of Test Cases: PASSED	6
Maximal Value	3
Mean Value	1,67
TOTAL	100,0%

54-EU-CZ-MLADA_BOLESLAV

ŠKODA	Test Scenario :	**54-EU-CZ-MLADA_BOLESLAV**	Testing Duration :	**3 min. 39 sec.**
	Test Report No. :	**HIGH02**	Hardware Ver. :	**X41**
	Location :	**Czech Republic, Mlada Boleslav**	Software Ver. :	
	Date :	**27.05.2015**	Map Vers. :	
	Project :	**MIB High**	Test Result :	**Passed**
	Testing Environment :	**Reality**	Tested By :	**Elshan Khalilov**
www.skoda-auto.cz	Testing Equipment :	**SK371**		

Number	Content	Test Result	Rating	Location Time & Coordinates	Description	Expectation	Comments	
1	Highway	Passed	2	50.383673, 14.896406	Elevated connection	Cursor should move from the entrance to the highway	Cursor was moving to the highway a bit roughly	
2	Parallel roads	Passed	3	50.381429, 14.892989	15-30 m	Cursor should move according to the direction to Melnik	Cursor was moving roughly and has jumped to the actual vehicle position with a short delay	
							Amount of Test Cases: TESTED	**2**
							Amount of Test Cases: PASSED	**2**
							Maximal Value	**3**
							Mean Value	**2,50**
							TOTAL	**100,0%**

56-EU-CZ-MLADA_BOLESLAV

ŠKODA	Test Scenario :	**56-EU-CZ-MLADA_BOLESLAV**	Testing Duration :	**5 min. 46 sec.**
	Test Report No. :	**HIGH03**	Hardware Ver. :	**X41**
	Location :	**Czech Republic, Mlada Boleslav**	Software Ver. :	
	Date :	**27.05.2015**	Map Vers. :	
	Project :	**MIB High**	Test Result :	**Failed**
	Testing Environment :	**Reality**	Tested By :	**Elshan Khalilov**
www.skoda-auto.cz	Testing Equipment :	**SK371**		

Number	Content	Test Result	Rating	Location Time & Coordinates	Description	Expectation	Comments	
1	Underground parking	Failed	4	50.411372N, 14.9040942E	2-4 levels	Cursor shouls move according to actual vehicle movement trajectory	Cursor was relatively smoothly moving when was entering the underground parking. Strange cursor movement has been detected inside. Cursor did not represented the actual vehicle movement trajectory, and was matching to the different roads on the ground surface.	
							Amount of Test Cases: TESTED	**1**
							Amount of Test Cases: PASSED	**0**
							Maximal Value	**4**
							Mean Value	**4,00**
							TOTAL	**0,0%**

Test Results: RPS & MIB STD DE unit

34-EU-CZ-MLADA_BOLESLAV

ŠKODA

www.skoda-auto.cz

Test Scenario :	**34-EU-CZ-MLADA_BOLESLAV**	Testing Duration :	**13:36:15 - 13:42:41 (6 min. 36 sec.)**
Test Report No. :	**STD01**	Hardware Ver. :	**X76**
Location :	**Czech Republic, Mlada Boleslav**	Software Ver. :	
Date :	**27.05.2015**	Map Vers. :	
Project :	**MIB STD DE**	Test Result :	**Passed**
Testing Environment :	**Laboratory**	Tested By :	**Elshan Khalilov**
Testing Equipment :	**SK371**		

Number	Content	Test Result	Rating	Location Time & Coordinates	Description	Expectation	Comments
1	Roundabout	Passed	1	50.432352, 14.924114	Circle	Cursor should smoothly move around the roundabout	Cursor was smoothly moving around the roundabout
2	Roundabout	Passed	1	50.430633, 14.930584	Circle	Cursor should smoothly move around the roundabout	Cursor was smoothly moving around the roundabout
3	Parallel Roads	Passed	1	50.4303278N, 14.9308169E	<15 m	Cursor should move along the proper road way	Cursor turned and was moving on the proper way
4	Roundabout	Passed	1	50.427979, 14.938890	Circle	Cursor should smoothly move around the roundabout	Cursor was smoothly moving around the roundabout
5	Roundabout	Passed	1	50.430633, 14.930584	Circle	Cursor should smoothly move around the roundabout	Cursor was smoothly moving around the roundabout
6	Parallel Roads	Passed	1	50.4303278N, 14.9308169E	<15 m	Cursor should move along the proper road way	Cursor turned and was moving on the proper way
7	Roundabout	Passed	1	50.427979, 14.938890	Circle	Cursor should smoothly move around the roundabout	Cursor was smoothly moving around the roundabout

Amount of Test Cases: TESTED	**7**
Amount of Test Cases: PASSED	**7**
Maximal Value	**1**
Mean Value	**1,00**
TOTAL	**100,0%**

54-EU-CZ-MLADA_BOLESLAV

ŠKODA

www.skoda-auto.cz

Test Scenario :	**54-EU-CZ-MLADA_BOLESLAV**	Testing Duration :	**14:34:16 - 14:38:04 (3 min. 48 sec.)**
Test Report No. :	**STD02**	Hardware Ver. :	**X76**
Location :	**Czech Republic, Mlada Boleslav**	Software Ver. :	
Date :	**27.05.2015**	Map Vers. :	
Project :	**MIB STD DE**	Test Result :	**Passed**
Testing Environment :	**Laboratory**	Tested By :	**Elshan Khalilov**
Testing Equipment :	**SK371**		

Number	Content	Test Result	Rating	Location Time & Coordinates	Description	Expectation	Comments
1	Highway	Passed	1	50.383673, 14.896406	Elevated connection	Cursor should move from the entrance to the highway	Cursor smoothly moved to the highway
2	Parallel roads	Passed	2	50.381429, 14.892989	15-30 m	Cursor should move according to the direction to	Cursor moved to the parallel road a little bit roughly

Amount of Test Cases: TESTED	**2**
Amount of Test Cases: PASSED	**2**
Maximal Value	**2**
Mean Value	**1,50**
TOTAL	**100,0%**

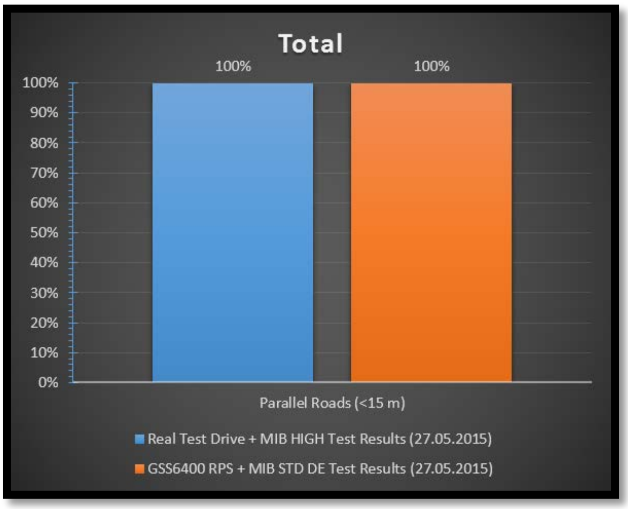
56-EU-CZ-MLADA_BOLESLAV

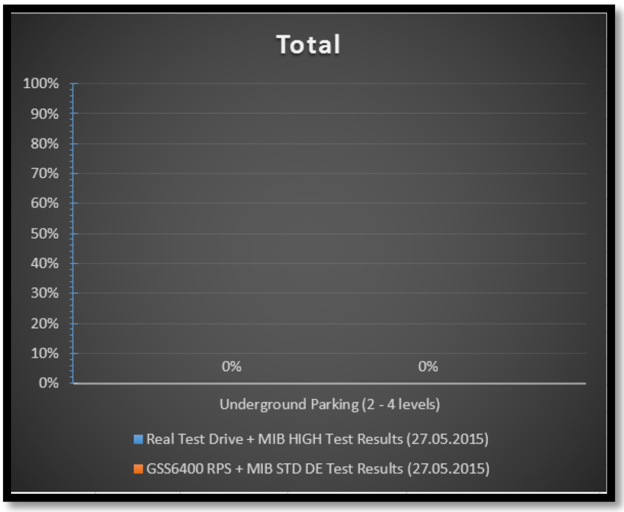
ŠKODA www.skoda-auto.cz	Test Scenario :	**56-EU-CZ-MLADA_BOLESLAV**	Testing Duration :	**13:49:49 - 13:55:51 (6 min. 2 sec.)**
	Test Report No. :	**STD03**	Hardware Ver. :	**X76**
	Location :	**Czech Republic, Mlada Boleslav**	Software Ver. :	
	Date :	**27.05.2015**	Map Vers. :	
	Project :	**MIB STD DE**	Test Result :	**Failed**
	Testing Environment :	**Laboratory**	Tested By :	**Elshan Khalilov**
	Testing Equipment :	**SK371**		

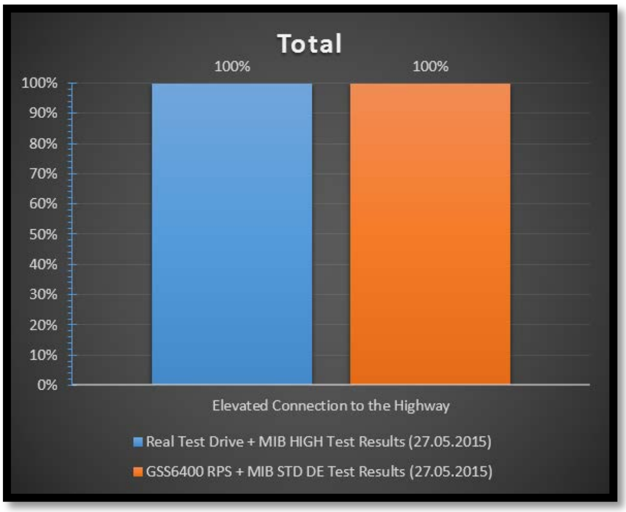
Number	Content	Test Result	Rating	Location Time & Coordinates	Description	Expectation	Comments	
1	Underground parking	Failed	5	50.411372N, 14.9040942E	2-4 levels	Cursor should move according to actual vehicle movement trajectory	Cursor did not represent the actual vehicle position. Strange cursor movement has been detected. Cursor was showing the false vehicle location inside the underground parking and when the car has left parking.	
							Amount of Test Cases: TESTED	**1**
							Amount of Test Cases: PASSED	**0**
							Maximal Value	**5**
							Mean Value	**5,00**
							TOTAL	**0,0%**

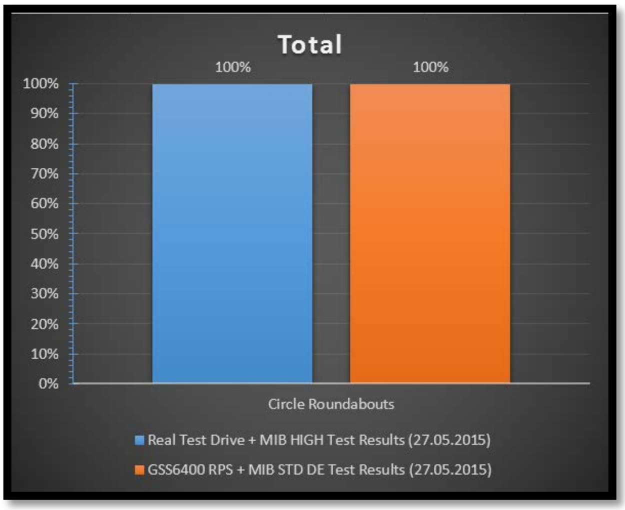
Appendix D

Total Test Case Passage Comparison

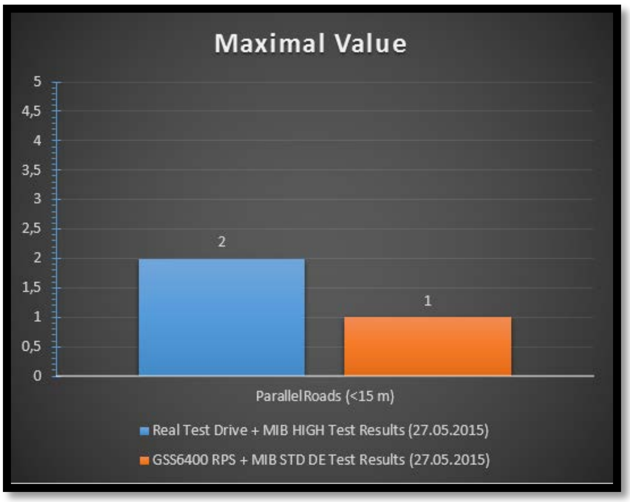


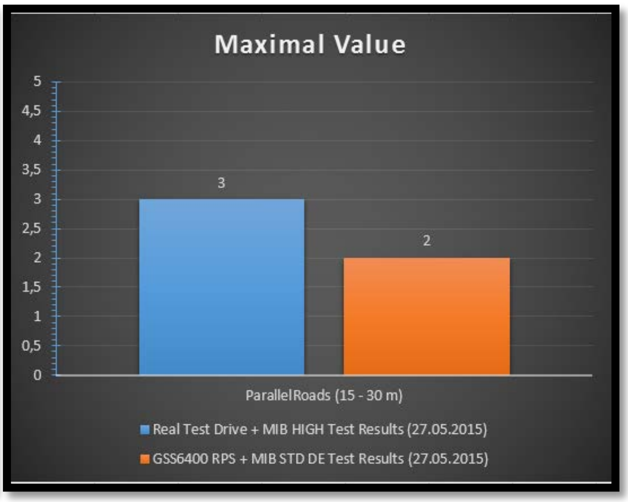


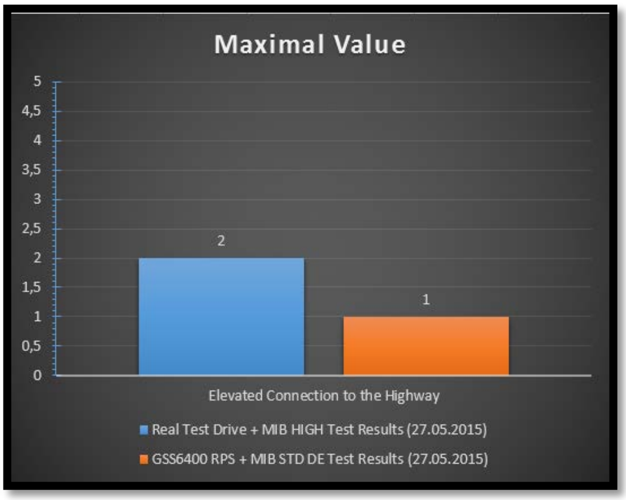


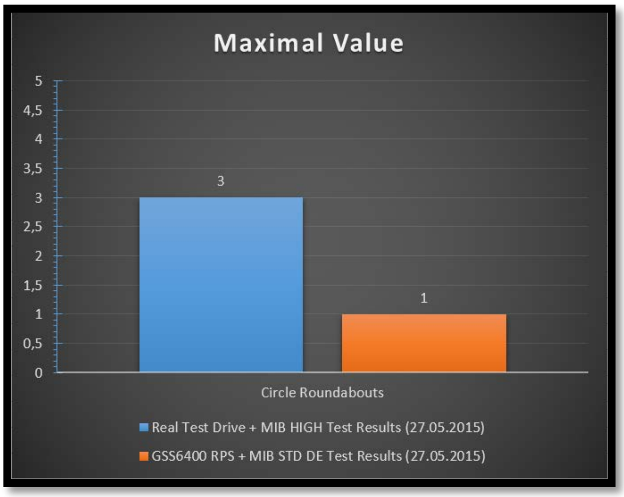


Maximal Values Comparison

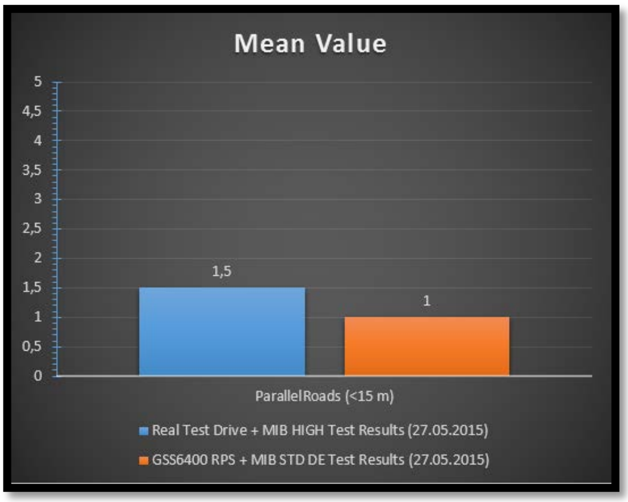


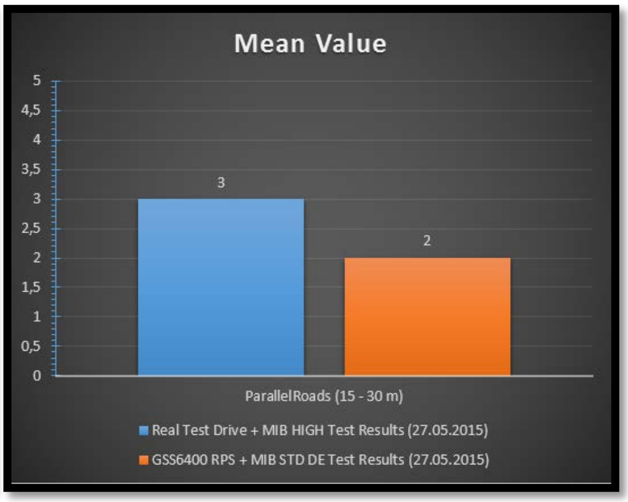


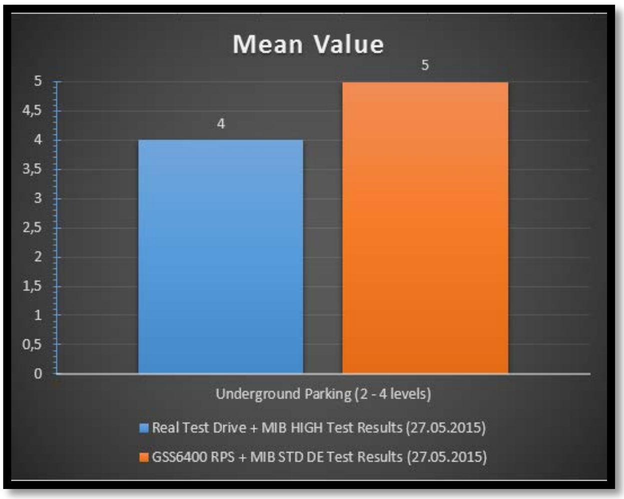


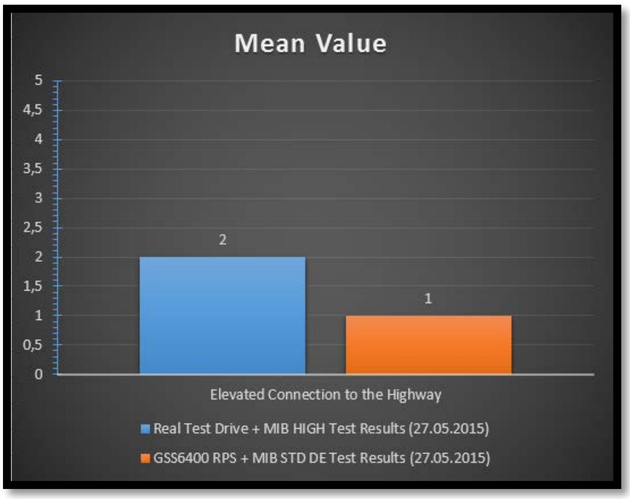


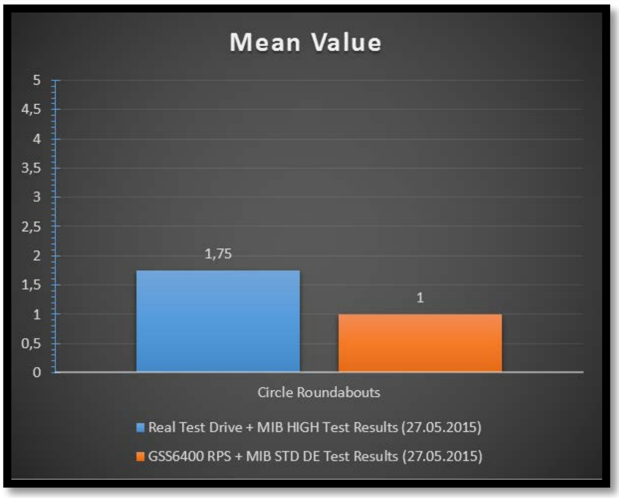
Mean Values Comparison






Mean Value
5
4
Underground Parking (2 - 4 levels)
Real Test Drive + MIB HIGH Test Results (27.05.2015)
GSS6400 RPS + MIB STD DE Test Results (27.05.2015)


Mean Value
2
1
Elevated Connection to the Highway
Real Test Drive + MIB HIGH Test Results (27.05.2015)
GSS6400 RPS + MIB STD DE Test Results (27.05.2015)


Mean Value
1,75
1
Circle Roundabouts
Real Test Drive + MIB HIGH Test Results (27.05.2015)
GSS6400 RPS + MIB STD DE Test Results (27.05.2015)