Communication link and code conversion between Vehicle and smartphone for low speed semi-autonomous maneuvering

Bahram Rahmatdoustbeilankouh
Abstract

Something that has recently gained popularity in the leading car manufacturing companies is the integration of an auto-reverse assistance system to improve the customer experience. The undeniable spread of smartphones and their significant role in human life in recent years, gave rise to the idea of designing an application to be used for reverse driving. As a result, the car company Volvo proposed an idea for this project: to implement an Android based application to facilitate reverse navigation in their trucks. From a technical point of view, the most crucial obstacle that should be addressed is implementing a secure and reliable communication link between the smartphone and the truck’s control centre. Hence, the primary goal of this thesis work is to provide a secure channel to transmit data and computing reliability of the communication link. Another objective of this project is to define a solution for auto-converting path plan function, currently developed in Matlab, to run on Android devices. In addition, this solution should enable the developer to modify the path plan function in Matlab without having to consider Android programming. The differential method has been proposed for transferring secure data transfer using Bluetooth technology. This solution not only increases the security of the communication link but also improves transmission time. Another objective has been reached by developing a middleware function using C programming language and Android Native Development Kite (NDK) between Android and Matlab. The result of these solutions provides a reliable communication link and runs the developed path plan function in Android phone.

Keywords: Android, Matlab, Bluetooth, Serial Rs-232, Android NDK, Android JNI.
Acknowledgements

I would like to express my gratitude and appreciation to:

Prof. Tingting Zhang, Mid Sweden University examiner and teacher, for supporting me through my master studies.

Carl-Johan Hol, my Volvo Truck Technology supervisor, for technical support and guidance throughout my thesis.

Dr. Rahim Rahmni, Mid Sweden University supervisor, your advice helped me finish my thesis.

Finally, a special thanks to my family. I could not have finished my study without their support.
# Table of Contents

Abstract .................................................................................................................. ii
Acknowledgements ............................................................................................... iii
Table of Contents .................................................................................................... iv
Terminology/Notation ............................................................................................. vii
Introduction ........................................................................................................... 1
  1.1. Background and problem motivation ......................................................... 1
  1.2. Overall aim ................................................................................................. 1
  1.3. Scope ........................................................................................................... 2
  1.4. Concrete and verifiable goals ..................................................................... 3
  1.5. Outline ........................................................................................................ 5
Related work ........................................................................................................... 6
  2.1. Android operation system ......................................................................... 6
      2.1.1. Linux kernel ....................................................................................... 7
      2.1.2. Native library ................................................................................... 8
      2.1.3. Application framework ................................................................. 9
      2.1.4. Application layer ............................................................................. 9
  2.2. Android Native Development Kit (NDK) ............................................. 10
  2.3. Bluetooth technology ............................................................................. 11
  2.4. Serial communication technology ....................................................... 13
      2.4.1. Synchronous serial communication ........................................... 14
      2.4.2. Asynchronous serial communication ........................................ 14
  2.5. Matrix Laboratory (Matlab) .................................................................. 15
      2.5.1. Matlab cross compiler ............................................................... 16
  2.6. AutoBox .................................................................................................... 17
Methodology and system structure ................................................................... 19
  3.1. System requirement ............................................................................... 19
      3.1.1. Communication link .................................................................... 19
      3.1.2. Code conversion .......................................................................... 22
      3.1.3. Combination of communication link and code conversion .......... 23
# Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering

Bahram Rahmatdoustbeilankouh

**Table of Contents**

2015-12-15

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2.</td>
<td>System structure when the path plan function is running on the AutoBox</td>
<td>25</td>
</tr>
<tr>
<td>3.3.</td>
<td>System structure when the path plan is running on the smartphone</td>
<td>27</td>
</tr>
<tr>
<td>3.4.</td>
<td>Verification</td>
<td>32</td>
</tr>
<tr>
<td>3.4.1.</td>
<td>The communication link</td>
<td>32</td>
</tr>
<tr>
<td>3.4.2.</td>
<td>Data compression</td>
<td>33</td>
</tr>
<tr>
<td>3.4.3.</td>
<td>Data type</td>
<td>34</td>
</tr>
<tr>
<td>3.4.4.</td>
<td>Code conversion</td>
<td>34</td>
</tr>
<tr>
<td>3.5.</td>
<td>Ethical Considerations</td>
<td>34</td>
</tr>
<tr>
<td>3.6.</td>
<td>Security suggestion</td>
<td>35</td>
</tr>
</tbody>
</table>

### Implementation of the methods

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1.</td>
<td>Implementation when the PP function runs on the AutoBox</td>
<td>39</td>
</tr>
<tr>
<td>4.1.1.</td>
<td>Implementation of the communication link for the AutoBox</td>
<td>39</td>
</tr>
<tr>
<td>4.1.2.</td>
<td>Implementation of communication link for the smartphone</td>
<td>44</td>
</tr>
<tr>
<td>4.2.</td>
<td>Implementation when the PP function runs on the Smartphone</td>
<td>46</td>
</tr>
<tr>
<td>4.2.1.</td>
<td>Run Path Plan on Smartphone</td>
<td>46</td>
</tr>
<tr>
<td>4.2.2.</td>
<td>Implementation of the Smartphone side data communication link</td>
<td>47</td>
</tr>
<tr>
<td>4.2.3.</td>
<td>Implementation of the AutoBox side data communication link</td>
<td>50</td>
</tr>
</tbody>
</table>

### Results

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1.</td>
<td>Code conversion</td>
<td>51</td>
</tr>
<tr>
<td>5.2.</td>
<td>Data communication</td>
<td>54</td>
</tr>
<tr>
<td>5.3.</td>
<td>Data compressing</td>
<td>54</td>
</tr>
<tr>
<td>5.4.</td>
<td>Communication link</td>
<td>55</td>
</tr>
<tr>
<td>5.4.1.</td>
<td>Evaluation of the serial port</td>
<td>55</td>
</tr>
<tr>
<td>5.4.2.</td>
<td>Evaluation of Bluetooth technology</td>
<td>56</td>
</tr>
</tbody>
</table>

### Conclusions

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>63</td>
</tr>
</tbody>
</table>

### Future Work

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>65</td>
</tr>
</tbody>
</table>

### Reference

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>66</td>
</tr>
</tbody>
</table>

### Appendix A: User manual

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>70</td>
</tr>
</tbody>
</table>
## Terminology/Notation

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NDK</td>
<td>Native Development Kit</td>
</tr>
<tr>
<td>UTM</td>
<td>Universal Transverse Mercator Mathematical</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>ECU</td>
<td>Electronic Control Unit</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphic User Interface</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>HMI</td>
<td>Human-Machine Interface</td>
</tr>
<tr>
<td>CF</td>
<td>Controlling Function</td>
</tr>
<tr>
<td>ISM</td>
<td>Industrial, Scientific and Medical</td>
</tr>
<tr>
<td>FHSP</td>
<td>Frequency-Hopping Spread Spectrum</td>
</tr>
<tr>
<td>GFSK</td>
<td>Gaussian Frequency Shift Keying</td>
</tr>
<tr>
<td>UART</td>
<td>Universal Asynchronous Receiver Transmitter</td>
</tr>
<tr>
<td>API</td>
<td>Application Program Interface</td>
</tr>
<tr>
<td>MCR</td>
<td>Matlab Compiler Runtime</td>
</tr>
<tr>
<td>RCP</td>
<td>Rapid Control Prototyping</td>
</tr>
<tr>
<td>ADAS</td>
<td>Advanced Driver Assistance Systems</td>
</tr>
<tr>
<td>RTI</td>
<td>Real-Time Interface</td>
</tr>
<tr>
<td>PP</td>
<td>Path Plan</td>
</tr>
<tr>
<td>APP</td>
<td>Android Path Plan</td>
</tr>
<tr>
<td>DP</td>
<td>Data Path</td>
</tr>
<tr>
<td>TX</td>
<td>Transmission</td>
</tr>
</tbody>
</table>

vii
<table>
<thead>
<tr>
<th>RX</th>
<th>Receive</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>Media Access Control</td>
</tr>
</tbody>
</table>
Introduction

1.1. Background and problem motivation

It was not that long ago that the cell phone was introduced, and, before long, people no longer needed wire and landline phones. Still, no one knew that the cell phone would turn into a small magic box. Nowadays, the smartphone plays an important role in improving our day-to-day lives. The smartphone is really the right word for the new generation of cell phones; because it has several built-in sensors such as camera, microphone and Bluetooth, making it really quite smart. When you have a smartphone it means that you have a camera, laptop, MP3-player, map, game box, all in one. The operating system operates built-in sensor and provides facility for an end user that can be easily used from a smartphone.

Nowadays, most companies attempt to connect the smartphone to their new products, and there has been competition in terms of finding a way or solution making it possible to access their products through a smartphone. For example: it would be of interest for a driver to have a car that could be parked and controlled automatically by using a smartphone, or it would be of help for truck drivers if the truck had an auto-reverse system. Driving a heavy truck with a trailer becomes more difficult and stressful when the parking space is tight. The truck driver has to have precision and use low speed maneuvers to park the long vehicle, even if the driver is experienced. When loading or unloading the vehicle or docking at loading bays, this becomes a challenge for the driver. Under these circumstance, and auto-reverse system would be useful, the truck could be parked automatically and the driver just checks that the parking process is going well.

1.2. Overall aim

A low speed semi-autonomous maneuvering system has been developed and tested successfully on a simulation box without graphical interface. The final product will run on an Electronic Control Unit (ECU) of the vehicle and requires a Graphical User Interface (GUI) that will run on a smartphone. The system developers of the final product decided to run a path plan function which is a part of low speed semi-autonomous maneuvering system, on the smartphone too, because the ECU is limited
when it comes to memory and process. The Path Plan function was developed in Matlab and a solution has to be found in order to be able to run it on a smartphone. The first objective of this project is to present a solution for running the Matlab model on the Android application without having to modify or reprogram manually. The Path Plan function is one of the significant functions of the system because it is responsible for generating a drivable path between the current and the target position. The path should be transferred to the vehicle after generation.

Based on this scenario, another goal has been defined because it requires a communication link to connect the smartphone and the vehicle together to transmit data. This communication link should be reliable and secure because it transfers the vehicle state online from the vehicle such as speed, direction and position based on global position system (GPS) to the smartphone to be used by GUI. Also, simultaneously, this link should transfer data from the smartphone to the vehicle, like a generated path from the Path Plan function or other control signals such as start, stop or retrieve. It is clear that this communication link will play a significant role in the project and it should be acute.

To sum up, the overall aim of this project is divided into two parts. The first part consists of developing a solution to run Matlab model on the Android application and the second part consists of establishing the communication link between the smartphone and the vehicle.

1.3. Scope

As previously mentioned, this project is divided into two phases. From a technical point of view, these two phases will be implemented totally separately, but will be connected by the end of the project. The communication link sends generated data from code conversion part to vehicle while receiving special data, which the code converting function requires.

The communication link will be implemented on the Android platform with Android programming language by using built-in Bluetooth sensors to establish a reliable and secure communication link between the smartphone and the vehicle. Unfortunately, the hardware in the vehicle does not support wireless technology, so it is necessary to use a converter from wireless to a vehicle supported port, such as a serial port.
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering
Bahram Rahmatdoustbeilankouh

Introduction

2015-12-15

The Matlab model is used to program for the vehicle, with extra functions in the Matlab model provided for vehicle programming.

The code conversion part will be generated by using C programming language. Native Development Kit (NDK) will be required to run C codes in the Android kernel.

This project will focus on Android phones only and all functionalities are based on the capabilities of Android phones. Developing the same functions for others smartphones is not considered in this project. In project, smartphone refers to an Android phone. Graphic User Interface (GUI) is out of scope for this project, and GUI and user interaction will be implemented by the Human-Machine Interface (HMI) developer part. The wireless and wire technologies are used for communication, improving these technologies is also out of scope for this project. The code converted solution is specifically related to the Path Plan function and is not general solution. Finally, it should be mentioned that both part of this project are prototypes used to ensure that the concepts are feasible. The result of this thesis will be used to develop a final marketing application.

1.4. Concrete and verifiable goals

In terms of theory, reading up on Bluetooth technology, the Android operating system, Android native development kit, RS-232 serial link, Matlab and AutoBox tool is required. Because the result of this study should lead to the design of a tenacious and flexible system in both the communication and code conversion parts, the theory chapter is followed by attempting to answer the following questions:

- What is a Bluetooth technology?
- What is a RS-232 serial link?
- How many layers does the Android operating system have and what kind of services does each layer provide?
- What is Android NDK?
- How is NDK used in Android?
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering

Bahram Rahmatdoustbeilankouh

Introduction

2015-12-15

- How does an AutoBox work?
- What kind of input/output port does AutoBox have?
- What are Matlab and Matlab compiler?
- What kind of output does Matlab compiler provide for developers?

The main objective of this thesis includes designing and implementing two goals that will be described in the below.

The first goal is to implement secure and reliable communication link between the smartphone and the vehicle. This communication link should be able to send and receive data from the smartphone to the vehicle and answer to versa. The implementation of this communication link should provide answers to the following questions:

- How should the communication system work?
- How can a flexible and tenacious transmission channel be defined?
- What is the data size and type on the smartphone side?
- What kind of data is generated by the AutoBox?
- How can the security of the communication link be improved?
- Which level of Bluetooth security is suitable for this version of the application?
- How can transmission time be improved without improving the transmission technology?

The second objective of this thesis is defining a solution for the Matlab function to run on the smartphone. This solution should work automatically without human interaction. Given the time constraints of this project, achieving this target was difficult because no similar project could be found. Implementing this project requires a complete analysis of Matlab functionality as well as Android core facility. A proper implementation of this solution would answer the following questions:
What functionality has generated the C file from the Matlab compiler?

- How can the required library be provided to run the Matlab compiler generated C file?

- How can a middleware function between Matlab and Android be defined?

- How to connected middleware function to Android by using NDK?

The result of the main objectives of this thesis should be verified and the accuracy of the results should also be evaluated.

- How can the communication link be evaluated to prove quality of service?

- How can the reliability of the communication link be computed under different circumstances?

- How can the result of a converted function be verified?

- How can the accuracy of converted function be estimated?

1.5. **Outline**

The first chapter describes the main motivation of the project, its scope and final concrete goals. The second chapter provides information on basic concepts used in this article such as Android operating system, Bluetooth technology and serial link. Chapter three on methodology and system structure describes the system requirements, application structure and methodology of the system. The chapter also describes the solution to how the designed structure should be verified. Chapter four shows the how designed methodology and system structures are implemented. Chapter five provides the results. It includes an evaluation and verification of the result generated in the implementation phase. The sixth chapter provides the conclusion and where the result is discussed. Finally, the last chapter suggests future work that could to improve the application.
Related work

The theory chapter will cover required knowledge and background studies for this project, such as Android operating system, wireless technology, Android NDK, etc.

2.1. Android operation system

Android is a Linux kernel open software platform, primarily designed for touch screen mobile phones and tablets. The first beta version of Android was released in November 2007, and the first commercial version was released in September 2008, called Android 1.0. The new versions of Android have been developed under codenames and in alphabetical order since April 2009: Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich, and Jelly Bean. Each version has improved bugs of the previous version and added new features. The android operating system is developed with complete stack includes four layers, Linux kernel, native libraries, application framework and application layer [1].
2.1.1. Linux kernel

The Linux kernel that is used in Android is different from other devices used in the Linux operating system. Several Android specific built-in codes are added to drive smartphone hardware. The kernel drives hardware and makes it possible for software and applications to use the hardware. When the software needs to use the hardware, it sends a request to the kernel for accessing a specific resource and the kernel drives the hardware for a particular purpose[3].
Without kernel, the hardware will not be accessible and developers have to write code for all events of every single piece of hardware in the device. Changing the brightness of the screen, changing the volume level, answering a phone call etc. are all driven by the kernel [3].

The Android architecture is based on the Linux 2.6 kernel. The Linux kernel is used as hardware abstraction layout because the Linux kernel provides an approved driver model and several existing driver. The Linux kernel provides memory management, process management, security model, and networking in addition to a robust core operating system infrastructure. It has been improving over a time.

2.1.2. Native library

Native libraries can be found above the kernel layer. Figure 1 shows all of the green blocks in layer 2 written in C and C++ programming language. This layer provides extra flexibility for Android developers. For example, the Surf Manager manages application windows running different processes. OpenGL-ES and SGL provide 2D and 3D core graphical libraries for developers. An interesting thing about the graphics is that 3D and 2D graphics can be combined in the same application. The smartphone has a small screen, and a small screen size requires a different web browser engine. The WebKit function in the native library is responsible for rendering a web page for a small screen size [4]. SQLite is a powerful database engine that is mostly used in Android phones. SQLite provides full functionality of SQL commands in Android which can be used for a database to record a data. It is easy to use and does not require a configuration file or database administrator [5].

The main component in Android is Dalvik Virtual Machine. The core library also is located in the native library layer. Android Runtime was
designated specifically for Android as a way to allow it to run in embedded environments with limited battery, memory and CPU. Dalvik Virtual Machine runs specific files called dex, which are byte codes that are the result of converting at build time. Class files and .jar files are converted to .dex, resulting in more efficient byte code that can run well on small processor, as they use memory more efficiently. Data structures are designed to be shared across processes whenever possible and uses a highly CPU optimized bytecode interpreter. The end result is that all scenarios could have a multiple instance of Dalvik VM running on the device at the same time on an each of the processes [4], [6].

Core library is written in Java programming language and a core library contains all of the collection classes, utilities and I/O.

2.1.3. Application framework

The application framework is written in Java and is a tool that all applications use. The applications include sets that come with the phone, such as home applications, or the applications are written by Google and include applications that are written by the user. All applications use the same framework and the same APIs. The application framework layer manages access for hardware by related services by providing manager blocks. These manager blocks manage the life cycle of applications, access data and isolate application data from other applications.

For example, the content provider manager block manages access to structured sets of data. Data is encapsulated and affecting security issues. This is the standard way allowing applications to share data with code running in another process. For example, access to contact info, such as the contact list, phone numbers, addresses and names, is available for any application that would like to use or share this information[7].

2.1.4. Application layer

All applications (native or third party) are built on the application layer by using the same application framework that is provided by the layer below. Components that are supported by Android are activity, intent receiver, service and content provider.

Activity is part of the User Interface (UI) typically corresponding to one screen. For example, the email activity divided into 3 main activities.
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering
Bahram Rahmatdoustbeilankouh

One activity shows a list of your emails, but when you choose an individual message another activity will be launched, and a third will be started when you start writing an outgoing email [7].

The intent receiver is a way for your application to register your code to trigger by extra applications or is a way for your application to run other application. For example, the intent receiver of the Facebook gets your note, when your application needs to share a note on the Facebook[7].

Service is a task that does not have a UI. It has a long life and runs in the background. A good example is a music player. To play music, an activity is launched by using UI, the music starts and keeps on playing even if you navigate to another application; code keeps running working through the playlist. [7].

The contain provider is a component that allows the programmer to share data with other processes of other applications. Any application can store data that makes sense for the application. They can store files or databases, they can also make sure data is available as part of the platform, in addition to which Android provides several content providers that store common data, such as contact information and calendar information [7].

In Android, every application runs in its own process. This has several benefits, such as security and protected memory [7].

Android supports multithreading. Multithreading makes it possible for app developers to use thread when an application needs it to carry out different tasks simultaneously; it allows each thread to be assigned to a different task. For example, full duplex communication can be implemented using two threads, one thread for receiving data and one for sending data.

2.2. **Android Native Development Kit (NDK)**

The NDK is a toolset that helps developer to use a part of their own app with native-code, such as C or C++. Then the developer can use C or C++ codes in Android code. This solution means that the developer does not have to recode and can reuse the existing code libraries written in native-code language. Native codes are run in the kernel of the Android
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering
Bahram Rahmatdoustbeilankouh

side of other native libraries as running the code in the kernel means they can run faster and improve system efficiency. Reusing the existing C or C++ libraries codes and high efficiency are advantages of the NDK method. However, the NDK method increases app complexity, and most apps do not need the Android NDK. The developer should be aware that the NDK will not be benefit for most apps and should only use NDK if it is essential for the app because it is preferable to program in C/C++. As a developer, you should look at all the requirements and chose the best solution. When in doubt about using the NDK, consider the requirements and check the Android framework APIs. It is possible the existing framework provides the functionality that you need [8].

Typical good candidates for the NDK are self-contained, CPU-intensive operations that do not allocate much memory, such as signal processing, and physics simulation [8].

2.3. Bluetooth technology

In 1994, Ericsson developed a solution for a laptop computer 'making it possible to call a mobile phone from a laptop. After 1994, several companies have improved the standard for low-power short range communication technology. Theoretically, data transmission rate is 1 mega bit (Mb) per second, but in practice it is 720 Kbit per second when there are two devices within 10m using the 2 400 MHz to 2 483.5 MHz (i.e. 2.4000 - 2.4835 GHz) short-range radio frequency band. A Bluetooth device with a 0dBm nominal antenna power can cover a ten-meter area. The optional range is 100 meters if there is a 20dBm antenna output power. This technology is capable of 'if there is a data, audio, graphical data and video file [13], [16], [17].

Bluetooth could provide a solution to the following:

- It can be used to connect a headset to make a call remotely [13].
- It can decrease computer dependency on cable to be able to connect a printer, keyboard and mouse [13].
- It can be used in a home network for remote control of devices such as the oven and air conditioning [13].
Three different ranges have been designed for Bluetooth devices. The range may vary depending on the class of radio used in an implementation:

- **Class 3 radios** – have a range of up to 1 meter with lowest power requirement. The output is 1mW [13], [18].

- **Class 2 radios** – most commonly found in mobile devices – have a range of 10 meters. Minimum output for this class is 0.25mW (-6dBm) and maximum output is 2.4mW (+4dBm). This class has optional power control [13], [18].

- **Class 1 radios** – has a range of up to 100 meters with a maximum output of 100mW (+20dBm) and minimum output of 1mW (0dBm). Class 1 provides mandatory power control ranging from 4dBm to 20dBm. This class is primarily used in industrial use cases [13], [18].

Bluetooth uses frequency-hopping spread spectrum (FHSP) radio technology with Gaussian Frequency Shift Keying (GFSK) carrier modulation. This radio technology divides data into different packets and each packet is transmitted on one of the 79 Bluetooth channels. The channel has a 1MHz bandwidth and ranges from 2402 MHz to 2489 MHz with a guard band of 2 MHz at the bottom end of the band and 3.5 MHz at the top. It usually performs 1600 hops per second, with Adaptive Frequency-Hopping (AFH) enabled [13], [16], [17].

AFH capability was designed to reduce interference between wireless technologies sharing the 2.4 GHz spectrum. AFH works within the spectrum to take advantage of the available frequency. This is done by the technology detecting other devices in the spectrum and avoiding the frequencies they are using. This adaptive hopping among 79 frequencies at 1 MHz intervals gives a high degree of interference immunity and also allows for more efficient transmission within the spectrum. For users of Bluetooth technology this hopping provides greater performance even when other technologies are being used along with Bluetooth technology [13], [16], [17].
2.4. **Serial communication technology**

An interface that can transmit only one bit at a time is called serial port. Serial port is called COM port in it computer field and is used to connect a modem or other devices. It is controlled by a chip called Universal Asynchronous Receiver Transmitter (UART) [20], [21].

Serial communication technology can transmit data to remote devices further away than parallel communication technology; in addition, a serial connection cable is simpler than a parallel connection cable. These are two important advantage of serial technology. Serial devices are made in two different sizes, 9 pin and 25 pin, both called D-type plug. A D-type plug could be either male or female.

Below table gives briefly information about pins.

<table>
<thead>
<tr>
<th>Function</th>
<th>Pin-Symbol</th>
<th>DB-9</th>
<th>DB-25</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive data (serial data input)</td>
<td>RD</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Transmit data (serial data output)</td>
<td>Td</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Request to send (informs the modem that UART is ready to exchange data)</td>
<td>RTS</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>Clear to send (modem is ready to exchange data)</td>
<td>CTS</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>Data ready state (UART establishes a link)</td>
<td>DSR</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Signal ground</td>
<td>SG</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>Data carrier detection (this line is active when modem detects a carrier)</td>
<td>DCD</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>Data terminal ready</td>
<td>DTR</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>Ring indicator (becomes active when the</td>
<td>RI</td>
<td>9</td>
<td>22</td>
</tr>
</tbody>
</table>
Synchronous and asynchronous are two communication methods for serial communication technology.

2.4.1. **Synchronous serial communication**

The sender informs the receiver when the receiver should read the next bit in the synchronous serial communication method by the sender and receiver sharing a clock between them. When the sender has no data to send at the given time, it will send a fill character. The fill character will be used to start transmission. This scenario was implemented in the most synchronous communication method. Synchronization between sender and receiver is more efficient because only data bits are transmitted, but this efficiency comes at a cost; implementation of synchronous serial communication based on sharing a clock signal requires extra writing and controlling circuits [20].

2.4.2. **Asynchronous serial communication**

Asynchronous serial communication uses extra bits to clarify the start and end of each word data in order to synchronize. Extra bits include “start bit”, “end bit” and possibly “parity bit”. When a world data is received to URT for transmission, the “start bit” is added to the beginning of the data and the “stop bit” is added to the end. The “parity bit” may be used by the receiver to perform simple error checking (Figure 3).
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering
Bahram Rahmatdoustbeilankouh

Related work

2015-12-15

Figure 3: The asynchronous transmission method [21]

The start bit forces the receiver to synchronize with the sender’s clock. After the start bit, word data is sent bit by bit; transmission time for each bit of data is the same. When all data has been transmitted, the parity bit will be sent for error checking. Odd, even or non-parity can be selected in the configuration phase. Finally, the stop bit will be sent. If the stop bit is not received on the receiver side, URT will report a framing error because word data without a stop bit is not complete. The standard serial communications hardware in the PC does not support synchronous operations. Speed of asynchronous communication is measured based on number of bits transmitted per second, which is called baud rate. Transmitted bits include start bit, data, parity bit and stop bit. 9600, 19200, 38400, 57600 and 115200 are different serial baud rates [20], [21].

2.5. Matrix Laboratory (Matlab)

Matlab was written for easy access to matrix software and was developed by the LINPACK and EISPACK projects, which together represent the state-of-the-art in software for matrix computation. Users have improved Matlab and added new features over a period of years. It has tool suitable for high-productivity research, development, and analysis. Matlab provides a different toolbox for signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and more. Toolboxes are comprehensive collections of Matlab functions (M-files) that extend the Matlab environment to solve particular classes of problems [24].

Matlab is based on five different parts:
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering  
Bahram Rahmatdoustbeilankouh

Related work  
2015-12-15

1- The MATLAB language.

2- The MATLAB working environment.

3- Handle Graphics.

4- The MATLAB mathematical function library.

5- The MATLAB Application Program Interface (API).

These five parts collaborate to provide the functionality of Matlab.

The Matlab language is a high-level matrix/array language that it provides data structures, functions, input/output methods, control flow statement and object-oriented programming features.

The Matlab working environment provides a set of tools for the user or programmer. These tools include features for managing variables and importing and exporting data in a workspace. The Matlab working environment also provides tools for developing, debugging and management.

Handle Graphic handles graphic features and provides high-level and low-level commands to enable full customization of the graphic appearance. These graphic features include two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics.

The MATLAB mathematical function libraries include all mathematical algorithms that are required to solve all types of mathematical problems.

Finally, the last part is related to API libraries making it possible for developer to interact with the C and FORTRAN programming language in their Matlab application [24].

2.5.1. Matlab cross compiler

Matlab suggests three different output types based on developer requirements.
The first solution is for a system with Matlab installed.

The next two solutions are designed for users who have restrictions when installing Matlab. Matlab compiler is a solution that lets developers run their code outside the Matlab environment. This solution significantly reduces development time and eliminates the requirement of manually translating code to a different language by building a standalone application or shared library. A standalone application and shared libraries need a specific runtime engine called Matlab Compiler Runtime (MCR). The Matlab compiler provides MCR allows developers to deploy their own applications royalty-free [27], [29].

The final solution that Matlab provides is a software component. The Matlab compiler also provides software components such as Excel add-in, Java classes and .Net as well as standalone and shared libraries for developers by wrapping a created Simulink model or Matlab function. This solution means that the developer will not have to recode. Software components need MCR as well standalone executable and shared libraries [27], [29].

2.6. AutoBox

The AutoBox is a real-time system that does not require user intervention to operate, just like an Electronic Control Unit (ECU). It is used
to operate fast function prototyping in full pass and bypass scenarios. It is appropriate for various Rapid Control Prototyping (RCP) applications such as: Chassis control, Powertrain, Body control, Electric drives control, X-by-wire applications, Advanced Driver Assistance Systems (ADAS) and Aerospace applications [31].

AutoBox was developed with a mixture of high performance, comprehensive automotive input/output, and is very compact. It passes shock and vibration tests because of its robust implementation. Furthermore, AutoBox suggests versions with FPGA functionality in addition to versions with interfaces for all major automotive bus systems: CAN, LIN, K/L line and FlexRay [31].

A user can record a new application in non-volatile memory to run automatically after power-up. Program download and data analysis (hot plugging) can be approached simply by connecting a PC or notebook via the Ethernet port. The Real-Time Interface (RTI) library supplies the link between dSPACE AutoBox and the development software Matlab/Simulink/Stateflow from MathWorks. It allocates a blockset that develops the functionality and input/output abilities of AutoBox in Simulink controller models [31].
Methodology and system structure

This chapter describes which methods are suitable for solving the problems mentioned in the low level problem statement.

3.1. System requirement

3.1.1. Communication link

Before starting to design a method, important information about system conditions and requirements should be collected. As mentioned in section 1.2, project proposal chapter, to implement a communication system below the Application Data Communication Stack Model can be defined. It is always the upper layer generates data, which in turn is passed to lower layer for transition. Data changes when it passes from one layer to another. This means each layer works with a different data type.

![Diagram of Data Communication Stack Model]

**Figure 5: Data communication stack model**

Layers one and two are connected by a communication link, layer three is a middleware between communication and upper layer. Layer four uses layer three to write and read data. Layer four is HMI in the smartphone and Controlling Function (CF) in AutoBox and they are external applications, out of the project scope and not discussed.
Layer 1: the transmission layer is a physical link layer and is implemented based on available communication protocols. The smartphone supports Wi-Fi and Bluetooth and on the other side the AutoBox supports serial connection based on wire technology. It is obvious the devices will not match and to make a connection a modulator should be used to convert wire to wireless technology. Analysing system requirements proved that the Autobox serial ports are system bottlenecks. This means that regardless of which of the wireless technologies is selected the serial transmission rate 115,200 bits per second are slower than wireless technology. Next, the communication layer has to follow a serial port data transmission rate. As a facet of other communication features such as short distance, price, low energy consumption and security, the Bluetooth device has been selected by the project supervisor. The Bluetooth-Serial data converter module is used to connect the two non-familiar devices together (see Figure 6) and the converter module automatically converts the Bluetooth data packet to a serial data frame. Then the data communication stack model can be updated by adding this converter.

![Diagram of communication layers](image)

**Figure 6: Transmission layer in data communication stack model**

Layer 2: the data converting layer converts data from layer 3 to a data type that is suitable for the transmission layer. Layer three receives double data types from the upper layer, and layer one transmits byte data types. Next, the data conversion layer converts double to byte and
vice versa. Layer two can also be used to for mathematical formula, encryption and to merge data before transmission or reconstruct data after receiving on the receiver side.

Layer three: the data layer is a middleware layer used to connect layer four to layer two. The communication link, HMI and CF are under construction simultaneously, specific rules converting all of possibilities such as function name, input and output type, are required to facilitate merging the three parts when the project is finished. During development, HMI and CF developers create their own layer three with dedicated rules, while full functionality of the communication link is implemented by the communication link developer (Figure 7). During the development phase, layer four only has interaction with layer three.

![Layer three in the development phase](image)

In the merging phase relevant third layers are combined with each other. HMI and CF cannot access to layer one and two. These two layers are encapsulated and only layer three can be accessed (Figure 8).
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering

Bahram Rahmatdoustbeilankouh

Methodology and system structure 2015-12-15

![Diagram of data communication model](image)

**Layer Four**

<table>
<thead>
<tr>
<th>Smart Phone</th>
<th>Autobox</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HMI</strong></td>
<td><strong>Controlling Function</strong></td>
</tr>
</tbody>
</table>

Layer 4 is an external application that will be used in the communication layer, because it is out of scope it will not be described.

3.1.2. **Code conversion**

Code conversion is the second part of the project proposal that was described in section 1.2. A Path Plan (PP) algorithm is a function that is implemented by Matlab and can generate a drivable path between the current and the target position by using the vehicle state and the direction in desire position based on the Universal Transverse Mercator coordinate system (UTM). The vehicle state means the angle between the truck and the trailer and the UTM coordinate. A result of the path plan function there is a matrix with a 3000*6 double data type that each row of value represents small part of path.
The aim of code conversion is to find a solution to run the path plan function on the Android platform without recoding or modifying the path plan function manually. To achieve this goal, a middleware function using C programming language should be developed because both Android and Matlab compiler can provide and accept C code. As you can see in the below figure, the middleware function is generated by C programming language to provide all requirements to call Matlab generated C code. In addition Android NDK provides facilities for developers to call middleware function by using Android programming language.

![Figure 9: Code conversion scheme](image)

As an advantage of this solution is that the path plan developer can modify a Matlab model without knowing about functionality of middleware. Only new C code should be created from Matlab and the new code should be connected to the middleware.

### 3.1.3. Combination of communication link and code conversion

As described in the theory part, AutoBox is a device similar to a computer with CPU, memory and an external port. It can also process an application. The generated model in Matlab can run on AutoBox by cross compiling the model to native code and upload the code in AutoBox. CF was implemented on Matlab which means that whenever AutoBox is mentioned, CF is on it and what is needed is to record a path in right place and call the CF to use the path. CF will be controlled by three different signals called START, STOP and RETRIEVE signals.
Automation starts when CF receives the START signal. The STOP signal stops automation in all conditions and the RETRIEVE signal retrieves the recorded driven path. The controlling signals provide a solution to control automation.

A final 300-meters driven path is recorded in the memory and the path is accessible by the RETRIEVE signal. Below, a sequence diagram shows how the RETRIEVE signal works.

![Sequence diagram of the RETRIEVE signal](image)

Figure 10: Sequence diagram of the RETRIEVE signal

When the RETRIEVE signal is received by CF, the final 300-meters driven path is retrieved from memory, and the sample of this path should be transmitted to the HMI part on the smartphone side if the START signal is received in the CF function then automation will begin to drive the final 300-meters driven path.

Two different structures can be defined based on where the path plan function is run. The first structure is when the path plan function is
running on AutoBox. A result of the PP function is recorded automatically in the dedicated place and only the system requires the START signal to start automation. Before automation a sample of the generated a path should be transferred to the smartphone because the HMI part needs the data to show on the mobile screen. The second structure is the path plan function is running on the Smartphone and the result of the path plan function should be transferred to the AutoBox. The transferred a path should be recorded in dedicated place in the AutoBox before sending the START signal. It is obvious that a place for the path plan function to generate a path plays an important role in the implementation of a communication link. The following is a definition of the two different system structures based on the path plan function on the AutoBox and the smartphone.

3.2. System structure when the path plan function is running on the AutoBox

The final goal is running the path plan function on a smartphone but time is needed to reach this goal. While the path plan function is being developed to run on the smartphone, the HMI part needs access to a communication link to communicate with the AutoBox for testing. Then the path plan function will be running on the AutoBox side of CF. The presumed data communication stack model can be seen below.
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering

Methodology and system structure

Bahram Rahmatdoustbeilankouh

Methodology and system structure

2015-12-15

Figure 11: The path plan function side of the controlling function

As you can see in the above figure, the path plan function has been combined with CF in layer four, and a GENPATH signal has been assumed to generate the path. The below sequence diagram describes the sequential system structure for generating the path.
The GENPATH signal activates the path plan function to generate a path with dedicated input data which is collected by the HMI part. A result of the PP function is recorded in the AutoBox’s memory and a sample of the generated path is a matrix with 300*2 double data, which should be returned to the HMI side. If the path from HMI is acceptable, the START signal will be sent to the AutoBox to start automation because the path has been recorded in memory. Otherwise new input data for the path is collected and sent to the AutoBox to generate the new path.

3.3. System structure when the path plan is running on the smartphone

When the path plan function runs on the smartphone, it can be placed next to HMI in the data communication stack model because as a view of communication link, the path plan function is an external application requesting to use the communication link (Figure 13).
HMI has direct access to call the path plan function and can pass the final result to the bottom layer for transmission. As described in section 3.1.2, the result of the path plan function is called Data Path (DP). It is a matrix 3000*6 with double data type and these data should be transmitted to the AutoBox. DP is the biggest data receiver in layer two to transmit. Transmission time can be calculated using:

\[
\text{Transmission time} = \frac{\text{Data size}}{\text{Transmission rate}}
\]

Each double is eight byte then data size= 3000*6*8=144,000 bytes=144 kB

The communication link works on the lowest transmission rate and serial link with 11,520 bytes per second is the system transmission time.

Transmission time = 144,000/11520= 12.5 second

The HMI developer group has done user interview for GUI and the result of interview defines that 12.5 seconds is too long for a user to be waiting. The time needs to be reduced. This can be done by modifying the serial link or decreasing data size. Modifying the serial link is out of the project scope, which means that decreasing the data size should be considered.

Layer two of the data communication model provides a data type that is suitable for the transmission layer. The method used to decrease the
data size should be done before changing data to the relevant first layer. DP data type is double and every double is equal to eight bytes. A main goal of the method of reduction is to reduce data size by converting data type from double to float without losing parts of numbers because each float data type is equal to four bytes rather than eight bytes. Each element of DP is a big number, such as 6405717.61214033 and if a double number directly converts to float, a part of that number will be lost, for example 6405717.61214033 is a double number and number converted to float is 6405777.5. This means that some parts of the number have disappeared during the data conversion and in the UTM code these two numbers are different.

A solution to this problem is to use the differential method to convert double to float. This means that the differential of each element can be calculated by using the next element in the matrix. When a double number is converted to float by using the differential method, the data loss is reduced. For example:

\[ A = 6405717.61214033 \text{ and } B = 6405717.38476996 \] are two double sequential numbers. \[ C = A - B = 0.22737037 \] is still double. If \( C \) converts to float, it changes to \( D = 0.22737037 \). The difference between \( C \) and \( D \) is very small in the UTM code and can be neglected. This solution increases the accuracy of data type converting to \( 10^{-8} \).

The below pseudo code describes how the differential method and the change to layer one data type can be implemented.

### Table 2: The differential method

```plaintext
Function DifferentialMethod(double DP[3000*6]){
    Byte First_Row[1*48]; //a double is equal to eight Bytes
    Byte Rest_Row[2999*24]; // a float is equal four Bytes
    Float Result [2999*6];
    For (I=3; I<=3001; I++){
        For (column=1; column <=6; column++) {
```
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering
Bahram Rahmatdoustbeilankouh

Methodology and system structure
2015-12-15

The data converting layer on the AutoBox side should create the correct data. The first row should convert byte to double and the remaining data should convert byte to float. The reverse differential method will be calculated based on the first row and the remaining rows. The below pseudo code shows a solution to how the data layer on the AutoBox side can be implemented.

Table 3: The reverse differential method

<table>
<thead>
<tr>
<th>Function Reverse DifferentialMethod(Byte ReceivedData)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Double Result [3000*6];</td>
</tr>
<tr>
<td>Int Row [1*6] = 1;</td>
</tr>
<tr>
<td>Float Rest_Row[2999*6];</td>
</tr>
<tr>
<td>If (ReceivedData is first row)</td>
</tr>
<tr>
<td>Result (Row)= ConvertByteToDouble(ReceivedData);</td>
</tr>
<tr>
<td>If (ReceivedData is rest of rows)</td>
</tr>
<tr>
<td>Rest_Row= ConvertByteToFloat(ReceivedData);</td>
</tr>
</tbody>
</table>
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering

Bahram Rahmatdoustbeilankouh

Methodology and system structure

2015-12-15

For (I=Row+1; I<=3000; I++)

    Result (I) = Result (I-1) + Rest_Row (I-1);


To facilitate reconstruction of data on the receiver side, extra bytes are added to the data when data Converting layer converts data on the sender side and data overhead is inevitable.

The below sequence diagram shows messages passing between the smartphone and the AutoBox. If we compare this sequence diagram with the previous structure’s sequence diagram, it is clear that the number of messages passing between the two devices has reduced and the compression formula in the data converting layer has decreased the data size. The suggested differential method has also increased the security of the system because finding the relation between the numbers is difficult.
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering
Bahram Rahmatdoustbeilankouh

Methodology and system structure
2015-12-15

Figure 14: Sequence diagram for system structure when the path plan function is running on the smartphone.

3.4. Verification
There are four critical parts in this project, and all of which should be verified. The first part is the reliability of the communication link, the second is the data compression, the third is the data conversion and the final one is the code conversion. Each part should be analyzed and a solution should be described for verification.

3.4.1. The communication link
The communication link is responsible for the transmission of data, but improving of Bluetooth technology and the serial link are out of the project scope, but however the reliability of system should be clarified. The communication link is implemented using Bluetooth and serial technology by using a modulator to convert Bluetooth packets to a serial data frame. The serial technology is less sensitive to noise than the Bluetooth technology in this system, because the serial link works with a
fixed length in a fixed environment. When the data has been received from the Bluetooth part of the modulator, it creates a data frame and transfers the data to the AutoBox by serial link. In this scenario the reliability of system is in control of the Bluetooth technology because it has vulnerable in different situation. We can assume a fixed value for the reliability of the serial link such as $\alpha$ when calculating the system reliability. The Bluetooth technology requires evaluation on transmission time using a fix data size. The Bluetooth transmission time can be changed by increasing or decreasing distance between two Bluetooth devices and introducing an obstacle between them.

When the implementation of the project is complete, the reliability of the communication link should be evaluated. In this system reliability means the maximum distance between two devises that the communication link works well. To calculate the exact distance, two parameters are of importance: 1. the relation between transmission time and there being an obstacle and 2. the relation between successful connection and there being an obstacle at different distance such as 0.25, 3, 6, 11, 13.5, 16 and 18.5 meters.

The first parameter in this system should be calculated on: 1. wide area and 2. When there are two kind obstacles between devices at difference distances 0.25, 3, 6, 11, 13.5, 16 and 18.5 meters.

The second parameter is important because two Bluetooth devices should be connected together others before beginning communication and trying to connect may fail if the two devices are far from each other or if there is an obstacle between them. A successful connection should be evaluated in: 1. wide area 2. when there are two types obstacles between devices at difference distances 0.25, 3, 6, 11, 13.5, 16 and 18.5 meters. The successful connection can be evaluated 20 times in each situation.

When the values for both parameters are calculated, the system reliability can be selected by combining two values.

### 3.4.2. Data compression

The differential method improved transmission time by decreasing data size as described in section 3.3. The decreased data size will decrease and improve transmission time. This allows us to compare transmission
time before and after the effect of the differential method on the data. This solution explains improvement of transmission time.

3.4.3. Data type
As mentioned, when data passes from one layer to another layer, data type is changes to the required the data type format. If this data conversion has an effect on pure data then the effect should be calculated. This is important because we need to know to what extent data conversion destroys the original data. To calculate data distortion, 1000 numbers of the biggest data, i.e. the PP function result, are selected at random before conversion in layer three on the sender side. These 1000 numbers are subtracted by the same numbers that are received in layer three on receiver side after conversion.

The result of the subtraction shows data conversion distortion and also that the effect of distortion on the system should be evaluated by analyzing data.

3.4.4. Code conversion
When implementing the PP function in Android, called Android Path Plan (APP), the result of the APP function should be verified. A way to do this is by comparing the result of the APP function with the PP function result, because the PP function result is the original result and the APP function should generate the same. We know that the PP and APP functions results are a path and we can plot these paths on the map. If the plotted paths overlap, it means that the two results are the same and the APP function results are acceptable.

3.5. Ethical Considerations
The ethical considerations of this project include the privacy of the intellectual property and proprietary knowledge that belongs to Volvo. Certain knowledge was required to carry out the case study and this knowledge may not be shared publicly outside of the Volvo network. In addition, the full implementation details of the case study including the code base and affiliated data set is proprietary information that may not be shared as part of the thesis report. To ensure that proper ethical conduct, several steps have been taken. Privacy agreements were signed at Volvo, and a Volvo computer was allocated to be used for the case study. All Volvo sensitive data was kept on the Volvo computer,
which was equipped with data encryption software. The information related to the case study was reviewed and authorized before the completion of this thesis. In addition, Volvo employees who participated in the questionnaires were informed that the data they submitted would be included as part of the thesis report.

3.6. **Security suggestion**

As we know that in the low speed semi-autonomous maneuvering system a driver controls gas, brake as well as the clutch pedal and gear. The semi-autonomous maneuvering system only controls the steering wheel. This means that the driver has to sit in the truck to push the pedals and change the gear. Maximum distance between the two Bluetooth devices is 2 meters when automation starts; the driver can stop automation by grabbing the steering wheel if there is a problem with the low speed semi-autonomous maneuvering system or the truck goes in the wrong direction. Because the driver is inside of the truck and there is a small distance between the two devices, the system does not require high level security; Media Access Control (MAC) address filtering and Simple Pair Protocol [32] are quite enough for the first implementation of the system.

Several types of attacks can be attempted for a Bluetooth device having no security configuration such as Denial-Of-Service attacks on pairing and authentication, eavesdropping, Man in the middle attacks, offline/online PIN checking and message modification[33][34]. For example, a denial-of-service attack can be caused by a third party application by repeatedly sending an authentication request on a fake link key or the attacker relays pairing information using their own unit to connect other devices later on, when two Bluetooth devices are doing the hand-shake [33], [35]. Each of these attacks can be considered a serious threat to automation system.

To avoid any security vulnerability and to increase the safety of the automation in the final product when the diver controls the system from outside of the truck, security features must be added. Because any unverified person accessing data over the communication channel will constitute a risk when automation is running. Three important security features should be added on the system: 1. a suitable pair solution, 2. authentication and finally data encryption. This part of the report
describes security suggestions for the final product. This security suggestion should be implemented on the final product and implementation of suggestion for this level is optional.

A pair protocol is the first and fundamental security level required for the Bluetooth communication link. The protocol defines how a secret key can be sent to other devices safely [32]. A simple pair protocol is a well-known pairing protocol that works based on a human-verifiable protocol. It means that image or audio matching will be verified by manually and in this solution the secret key will not be transmitted over the air. After paring, the generated secret key is stored in memory to be used for at authentication and data encryption steps[32], [33], [35].

Authentication should be added to the Bluetooth connection feature while the two devices are doing the handshake before establishment of communication link. The idea of authentication is that the other device belongs to the pair should be verified. The other device will be verified by making a challenge-response scheme from the verifier device. The challenge is a random number made by the verifier sent over the air to the other device. The other device has to calculate a response based on a special algorithm with input random number, Bluetooth address and shared secret key. The response is sent back to the verifier device to compare with its own calculated number. If the calculated number matches with the received number then other device will be authenticated successfully. This security solution blocks unauthorized devices to access the automation and increases the safety of the autonomous maneuvering system.

Another security feature that can be added on the Bluetooth communication link is data encryption. Data encryption decreases chance of attackers reading and changing the data during automation. The idea is to make the data unreadable to unauthorized persons while being transmitted over the air between the Bluetooth devices. Data will be encrypted before being sent with an encryption algorithm and decrypted by the receiver with the same algorithm. Decryption is can only be done by the authorized paired device because only the trusted device has the necessary information to perform decryption. Data encryption minimizes the risk of hackers changing data and maximizing the safety of the autonomous maneuvering system. The below figure presents a
sequence diagram of authentication and data encryption for the Bluetooth security feature [35], [36], [37].

![Sequence diagram showing the security suggestion](image)

**Figure 15:** Sequence diagram showing the security suggestion [34], [35], [36]

To sum up, two Bluetooth devices should be paired before the handshake and in this step the information required that is related to other device is record on the memory. Sometimes it is not necessary to make a connection directly after pairing step. For this reason the authenticity of the other device should be verified when the handshaking process starts. The authentication solution verifies the validity of the other device. Data encryption is another security feature that can be added after connection. This solution keeps data fare from the hacker while
being transmitted over the air. This security suggestion will cover of Bluetooth security deficiency and maximize precision of the automation.
Implementation of the methods

This chapter the methods described is implemented. Two different structures were defined in the system methodology which means that two different implementations should be implemented. The first implementation is based on the PP function running on the AutoBox and the second the implementation describes implementation when the PP function runs in the smartphone.

4.1. Implementation when the PP function runs on the AutoBox

When the communication link is implemented full duplex data passes between the smartphone and the AutoBox. Full duplex means that while a device sends data, it is also able to receive data. The sender and receiver side have been implemented on the AutoBox and smartphone separately.

Section 4.1.1 describes how the sender and receiver have been implemented on the AutoBox and section 4.1.2 will describe the smartphone implementation.

4.1.1. Implementation of the communication link for the AutoBox

The AutoBox developer company provides three Simulink blocks for the implementation of serial link; setup, transmission (TX) and receive (RX). The setup block provides commands for the configuration of a serial communication link such as Baud Rate, Data, Parity bit, Stop bit and Start bit. The below table shows the configuration parameters that are assigned to the serial link.

<table>
<thead>
<tr>
<th>Baud Rate</th>
<th>Data</th>
<th>Parity Bit</th>
<th>Stop Bit</th>
<th>Start Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>115,200</td>
<td>8</td>
<td>None</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The TX block works independently from the RX block, for each of them a Data Communication Stack Model is implemented.
The transmission block (TX) in the AutoBox

A vehicle state is 48 bytes and it is sent ten times per second to the smartphone by the TX block. An IFstatement function controls and keeps transmission bytes on 48 bytes until the RX block receives a GENPATH or RETRIEVE signal. The figure below shows the IFstatement function and a red block is transmission block.

As described in the methodology chapter, when the RX block receives a GENPATH or RETRIEVE signal, a sample of the generated path, which is a matrix 300*2 double data type, should be sent to the smartphone. The IFstatement function will then increase the TX transmission bytes to 1008. The 48 bytes of the vehicle state are combined with 960 bytes of the sample of the path until all path samples sent. The TX block runs 10 times per second as 1,152 byte per sample time (115200/10*10=1152) is the maximum transmission rate that can be assumed for serial link. The IFstatement function increases the transmission rate to 1008 instead of 152 bytes per second as it does not decrease the transmission packet numbers, which are five in both cases. The below formulas show why both results are the same.

Sample of the path size = 300*2*8 = 4,800 bytes

Transmission byte for the path with 1008 bytes per sample time is equal to 1008 -48=960 bytes per sample time and transmission byte for the path with 1152 bytes per sample time is equal to 1152 -48= 1104 bytes per sample time. The packet number can be calculated using the below formula.

Packet number = Sample of path size/Transmission byte
Packet number = \[\frac{4800}{960}\] = 5 packets

Packet number = \[\frac{4800}{1104}\] = 5 packets

The IFstatement function increases transmission bytes to 1008 for five sample times for the transmission of the sample of the path and when complete it decrease transmission bytes to 48 automatically. The RETRIEVE signal has the same process as the GENPATH signal.

To implement layer two of the data communication stack, which is Data Converting layer, two different converter functions have been generated. The first double to byte converter function is used to convert six double vehicle state values to 48 bytes output data related to layer one. The second function was generated to convert a matrix with 300*2 double data type that is used to record the sample of the path for byte. The figure below shows both converter functions.
The receive (RX) block in the AutoBox

All received data is passed to a function called the data interpreter function. This function is responsible for converting byte data type to double, analyzing received data and activating a correct output link. Based on different input data from the RX block to the data interpreter function, one or some of the output lines are activated.

The RX block receives 96 bytes in each sample time. When 96 bytes is converted to double, it is an array with 12 double elements. Each controlling signal is mapped to a specific number and when data interpreter function receives one of the numbers, the relevant output controlling signal is activated. The below table shows the relevant mapping.
Table 5: Mapping number to signal

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>6</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>RETREIVE Signal</td>
<td>CLEAR Signal</td>
<td>STOP Signal</td>
<td>START Signal</td>
</tr>
</tbody>
</table>

For example if array values are 5, it means that the START signal should be activated by assigning number 1. The below figure shows the RX block and the data interpreter function.

![Diagram](image)

Figure 18: layer one and layer two from data communication stack model for receiver side.

Also if the first element of this array is 1 and last element is -1 then this array relates for the PP function input data to generate a path. Then relevant output signals and also the GENPATH signal to generate the path are activated.
4.1.2. Implementation of communication link for the smartphone

A full duplex the Bluetooth communication link is implemented with two major classes, connectionToBluetooth and ConnectedThread. These two classes are related for layer one of data communication stack which is transmission layer. Then the Smartphone communication link can be divided to 1- Receiver side and 2- Transmission side.

**Receiver side of the smartphone communication link**

The connectionToBluetooth is responsible for creating and disconnecting a connection with target Bluetooth device. The target device is clarified as a default device by using a MAC address. Simple Pair Protocol has been done manually before start the application. Without connecting to target device communication link does not work. Some of common the errors are: target device is turned off then it should be connected to power supply or the Smartphone’s Bluetooth device is been off then it should be turned on. If the connection is established, the ConnectedThread class will be called otherwise function returns related error message to solve the error and connection is failed.

When communication link has been established without any error then the ConnectedThread class is called. The ConnectedThread class is extended from thread and this thread always listens to channel for receiving a data. These data include vehicle state and sample of the path.

Layer two of data communication stack which is data converting layer, is implemented in the ConnectedThread class also. When the ConnectedThread class receives bytes data, eight bytes convert to double directly for further process.

When the AutoBox wants to send the sample of path, for a start packet sends 999 and for an end packet sends -999. It means that the path sample sits between 999 and -999 numbers. Then it would be easy for the Smartphone to identify vehicle state from the path sample and based on how transmission packet has been generated in the AutoBox, the same solution is used to reconstruct a received data in the Smartphone. When all values of the path were received and were converted to double, these values will be passed to layer three of data communication stack by recording the values in a DataModel class. The DataModel class
is implemented by HMI part relevant for the Data Layer of Data Communication Stack model. When the sample path is ready to use, a flag value is changed to true for notifying HMI.

**Transmission side of the smartphone communication link**

A maximum data size should be transmitted from the Smartphone to the AutoBox is a 10 double numbers that are related for input data of the PP function. For clarifying different received data in the AutoBox, a start number with value 1 and an end number with value -1 should be added to those numbers. Then maximum number is increased to 12 double numbers. When the HMI collects data about a path, data are sent to a pathPlanInputData function of the DataFrame class. The pathPlanInputData function converts the path input data to 96 bytes array which is included the start and the end numbers. Those 96 bytes are returned as a result of pathPlanInputData function.

After converting data is timing to send data by using a sendData function which is provided by the connectionToBluetooth class. This function accepts 96 bytes data and transmits data over the Bluetooth to the AutoBox.

When data packet related for the PP function is received in the AutoBox side, the GENPATH signal is activated automatically by data interpreter function of the AutoBox. It means that Smartphone never sends GENPATH signal and the data interpreter function of the AutoBox automatically actives GENPATH signal.

As a weakness of the RX block in the AutoBox is a number of received data should be defined manually and it is fixed data. Then all data should be packed in the same size in the Smartphone for transmission to the AutoBox. It means that each transmission packet should be followed as maximum 12 double numbers or 96 bytes data size.

As described before, a data mapping table was defined to map a controlling signal to specific numbers (table 6).
Table 6: Data mapping table

<table>
<thead>
<tr>
<th>RETREIVE Signal</th>
<th>CLEAR Signal</th>
<th>STOP Signal</th>
<th>START Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>0</td>
<td>6</td>
<td>5</td>
</tr>
</tbody>
</table>

All of these numbers are a double and each of them should be reduplicated 11 times to be full transmission packet before sending to the AutoBox.

The RX block of the AutoBox keeps last received data and affects last received data in every sample time while RX block is idle. Then non effected number which is named CLEAR Signal is decided to send to the AutoBox after each last packet data for removing those effects. As a view of the data interpreter function in the AutoBox side, the CLEAR Signal means do nothing and it helps system to work correctly.

4.2. Implementation when the PP function runs on the Smartphone

Implementation of system structure has been designed in part 3.3 of methodology chapter should be generated in three different step. The First step is running the PP function on the Smartphone, the second step is implementing Bluetooth communication link based on designed structure and finally communication link on the AutoBox should be generated to send and receive data.

4.2.1. Run Path Plan on Smartphone

A PathPlanGenerator function of NDKLib class is a Native abstract method. Abstract method means that body of the PathPlanGenerator function will be defined later and native command tells to compiler that body of this function is C and related file is recorded in JNI folder.

The PathPlanGenerator function plays middleware role between Android and C programming language. When other Android classes need to call the PathPlanGenerator function, they can call as well as an Android class. Body of the PathPlanGenerator function is C and provides all requirements to call C files that they are generated from Matlab/Simulink generated the PP Simulink model. This solution releases dependency of Matlab generated C code form Android and the PP function developer.
in the Matlab can modify the PP function in Matlab without being worry about running new version of the PP function in Android because just needs to generate new C code from modified Matlan/Simulink model and pasts file in JNI folder and compiles all code again.

4.2.2. Implementation of the Smartphone side data communication link

The Communication link has been implemented for the PP function on the AutoBox can be reused by modifying the DataFrame class and adding new class which is called a PathPlanGeneratedData when the PP function is running in Smartphone.

The Differential method which described in section 3.3 is implemented in the PathPlanGeneratedData class by mentioning that a maximum Bluetooth packet size is 115 bytes because the Rx block in the receiver side, runs 100 times per second. The maximum data size can be received in each sample time is 11520/100 = 115.2 bytes. By the help of the Differential Method function, 27 double numbers can be sited inside each packet. It means that the Differential method function is decreased data size from 216 bytes to 112 bytes. The PathPlanGeneratedData class wraps 27 numbers of the path inside each Bluetooth packet.

As described before result of the PP function are a matrix 3000 rows and 6 columns with double data type. For transmitting of each column with 3000 rows requires 111 full packets and left 3 numbers. First 111 full packets are transmitted then 3 reminding numbers from each column can be merged in one packet in the end.

Thereupon six columns are available and 111 packets for each column require and one packet requires for all left data then transmission packet numbers are 111*6+1= 667. The path should be sent to the AutoBox in 667 different packets.

The path matrix can be assumed like a puzzle with 667 pieces. Each piece should be put in the correct place in the receiver side then special address has been assigned for each piece by helping of row and column of matrix. Two extra bytes 113 and 114 are used to record these values. Then combination of byte 113 and 114 can be signed just for unique place. The byte number 113 has been used to address each column by mapping a unique number for each column and the byte number 114
has been used to clarify row. Below table describes relevant number for each column.

**Table 7: unique number for each column**

<table>
<thead>
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
<th>Column 4</th>
<th>Column 5</th>
<th>Column 6</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>70</td>
</tr>
</tbody>
</table>

The 3000 rows of the path matrix are divided in 111 different groups that each group is included in 27 rows. Then a number which is recorded in the byte number 114 shows related group number. By helping group number, beginning and ending of row can be calculated to record data in correct place. Receiver side by using a simple formula can find beginning and ending of rows.

Value of byte number 114 = X

Beginning of the row = (X-1)*27+1

Ending of the row = X*27

For example: in the below figure, byte number 113 is 20 and byte number 114 is 1. Those values mean data relate for column 2 and the beginning row is 1 and the ending row is 27.
Implementation of the methods
Bahram Rahmatdoustbeilankouh

Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering

Table 1: Example of how a packet data is created in Data Converting Layer

<table>
<thead>
<tr>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
<th>C5</th>
<th>C6</th>
</tr>
</thead>
<tbody>
<tr>
<td>365501.775863</td>
<td>6405768.993294</td>
<td>0.100000</td>
<td>-1.927511</td>
<td>-0.001000</td>
<td>1.000000</td>
</tr>
<tr>
<td>365501.775863</td>
<td>6405768.895910</td>
<td>0.200000</td>
<td>-1.927610</td>
<td>-0.002000</td>
<td>1.000000</td>
</tr>
<tr>
<td>365501.775863</td>
<td>6405768.805846</td>
<td>0.300000</td>
<td>-1.927910</td>
<td>-0.003000</td>
<td>1.000000</td>
</tr>
<tr>
<td>365501.775863</td>
<td>6405768.712211</td>
<td>0.400000</td>
<td>-1.928260</td>
<td>-0.004000</td>
<td>1.000000</td>
</tr>
</tbody>
</table>

Step 1

Step 2

Step 3

Figure 19: Example of how a packet data is created in Data Converting Layer

For more information, three steps can be assumed. In the first step 27 rows of column that should be sent are selected. Then second step is time to calculate Differential Method for those data. Result of Differential Method should be converted to byte data type and byte number 113 and 114 are calculated based on column number and row number. Finally byte array with 114 elements are passed to Transmission Layer for transmitting.

The addressing solution helps receiver for ordering the data to generate the same the path matrix.

Byte number 113 also is used to send the controlling signal.
For example when the receiver read number five in byte number 113, it understands that the Start signal should activate.

4.2.3. Implementation of the AutoBox side data communication link

A new data communication link is defined for running the PP function in the Smartphone. The new structure is mostly the same as previous structure which the PP function was running in the AutoBox. Then the same configuration is used to new serial link by the same values that are available in table 4.

The TX side works the same as before because the vehicle state should be transmitted to the Smartphone. When the RX receives the RETREIVE signal sample of recorded path should be sent to the Smartphone. The GENPATH signal is not used because of running the PP function in the Smartphone. The IFstatement function and data converting layer which was implemented for previous structure is a reusable for this structure.

As described in section 3.3, reverse the DifferentialMethod function should be implemented in Data Converting Layer of Data Communication Stack Model for receiver side. Then the data interpreter function is the best place to implement reverse differential method. Because when all data are received by the RX block and passed to the data interpreter function, reverse differential method is applied on data to extract original data. After extracting original data is time to reconstruct a path matrix with 3000 rows and 6 columns. Reconstructing is done by using of two extra bytes that they are added by sender. As we know, the path matrix is received in 667 different Bluetooth packets data and extra bytes 113 and 114 are used to reconstruct received data.

The path matrix is recorded in dedicated place after reconstructing the entire matrix in the data interpreter function.
Results

The project had two goals. The first goal was implementing a communication link between the smartphone and the Autobox. The second goal was code conversion. All desired goals were implemented. Now, the result of implementation should be verified. This chapter describes verification and evaluations of the results based on implementation section 4.

As described in section 3.4, four different parts should be verified.

1. Code conversion
2. Data type
3. Data compression
4. The communication link

Verification of the results can be divided into two parts, code conversion and data communication.

5.1. Code conversion

A solution suggested to verify the APP function is by comparing the APP function result with the PP function result when two results are plotted on a map. As we know, the result is a matrix with 6*3000 elements. Columns one and two are related for UTM coordinate X and Y values. These values show the path that the truck should take. The X-axis represents the X value and the Y-axis represents the Y value of the path on the plot. The path is generated from information such as current vehicle UTM position, destination and angle between head and trailer. The path is requested when the truck is positioned as in the below figure. The figure shows the path.
Figure 20: The PP function running in the AutoBox

All these values affect the shape of the path. These values are used to generate a new path by the APP function the X and Y axes are represented by the X and Y values. The following figure shows the result of the path that is generated from the APP function. The new path has the same shape as the previous.

Figure 21: The PP function running on the smartphone
To make sure that the two results are the same, two generated paths are plotted on the screen. It is clear that the two results are the same because the two paths completely overlap. The upper path is the APP function result which completely covers the path plan function result. The result of the two plotted paths on the screen are presented in the below figure.

Figure 22: The two different results of the PP function are the same when the function runs on different devices.

For clarification the two results have been further zoomed in and a part of the above figure is shown in Figure 23. The figure clearly shows the two paths overlap and the APP function result is acceptable.
5.2. Data communication

To verify the data passing, 1000 random numbers of path values are selected before transmission. On the receiver side the same 1000 number are selected and these numbers are subtracted. The result of subtraction is 10-8. The two UTM positions with the 10-8 variation refer to the same position. We expected the 10-8 variation because of the differential method’s effect on the data conversion.

5.3. Data compressing

The result of the PP function is:

\[\text{[3000 rows} \times 6 \text{columns} \times 8 = 144,000 \text{ bytes or } 144 \text{ Kbytes.]}\]

The effect of the differential method on the data size is

111 packets \times 6 + 1 = 667 packets \times 114 bytes data = 76,038 \text{ bytes or 76.038 Kbytes.}

This means that the differential method decreases the data size by 47.195%.
5.4. **Communication link**

The communication link plays an important role between the two devices when it comes to transmitting data. Because of this, the reliability of the communication link in different situations should be evaluated.

5.4.1. **Evaluation of the serial port**

A serial port based on wire technology does not suffer from the same problems as wireless technology. In this project the serial port is used with a fixed length in a fixed environment. Based on the different distances that Bluetooth can support, the COM port always transmits 76,038 bytes data in 6.67s while the COM port works with a 115200 Baud rate.

![Figure 24: Serial link transmission time](image)

In theory the serial port is able to transmit 76.038 Kbyte data in 6.60s, in reality the hardware device used in this project, can transmit the same amount of data in 6.67s. Based on the result, the performance of the serial port is 98.95%.
The result of the PP function is data with a size of 144 Kbytes. The data can be transmitted in 12.5s with a Baud rate of 115200. The data size is decreased to 76.038 Kbytes by using the differential method, which decreases the transmission time to 6.67s, which means that the differential method improves the transmission time by 46.64%.

![Figure 25: The differential method improves the transmission time by 46.64% transmission time.](image)

5.4.2. Evaluation of Bluetooth technology

When it comes to Bluetooth technology, environmental conditions and the distance between two the Bluetooth devices affects the transmission time. As a result, reliability of the Bluetooth communication link should be calculated under different conditions and distances. Evaluation for Bluetooth is done based on two different directions, face to face (FTF) direction and back to face direction (BTF). The Figure 26 shows the two directions.
As the human body affects the Bluetooth wavelength and decreases signal wavelength then both directions should be evaluated.

The reason behind the evaluation of the communication link is to be able to calculate the effect of obstacles on Bluetooth devices. To evaluate the effect of the obstacle, the target Bluetooth device is placed in box made of wood and the target Bluetooth device is covered in foil. The evaluation is based on three conditions wide area, obstacle and covered by foil for the distances 0.25, 3, 6, 11, 13.5, 16 and 18.5 meters. All evaluations are based on a fixed data size: 76.038 Kbyte. Figure 27 shows transmission time under the different conditions.
Under all conditions, transmission times increases when distance between the two devices increases. As you can see in the above figure, obstacles affect the transmission time more than the foil cover and the transmission time suddenly starts to increase at a distance of 6m as the system breakpoint in the obstacle environment is 6m. After 6 meters, transmission time greatly increases, but if we mention on the Y-axes we can see that this incremental is very small. For covered in foil and wide area the increase is smooth and close, the different between the two diagrams is that foil is a thin obstacle. There is one more Bluetooth characteristic that affects transmission time which will be discussed later.
Comparing BTF and FTF (Figure 28) over a wide area shows the effect of the human body on the Bluetooth wavelength. 11 meters is a critical distance for BTF because beyond 11 meters, transmission time increases, it becomes more than 6.67ms and the Bluetooth device becomes slower than the serial link. If the diagram of transmission time with obstacle for FTF is added to the comparison diagram for FTF and BTF, the diagram shows a difference between the transmission times for the different directions. As you can see in Figure 29, the blue line almost closely follows the green line for more than 12 meters, its increase is smooth and the part about the red line is incomprehensible.
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering
Bahram Rahmatdoustbeilankouh

Results
2015-12-15

The above information can be summed up in a table. The below table shows the relation between distance and transmission time (ms) under different conditions.

Table 9: The results of transmission time in different conditions

<table>
<thead>
<tr>
<th>Distance Area</th>
<th>0.25 m</th>
<th>3m</th>
<th>6m</th>
<th>11m</th>
<th>13.5m</th>
<th>16m</th>
<th>18.5m</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wide Area (F-T-F)</td>
<td>4124.8</td>
<td>4127.9</td>
<td>4130.1</td>
<td>4137.7</td>
<td>4141.2</td>
<td>4150.3</td>
<td>4159.7</td>
</tr>
<tr>
<td>Foil obstacle</td>
<td>4129.4</td>
<td>4135.1</td>
<td>4138.2</td>
<td>4141.7</td>
<td>4145.8</td>
<td>4153.5</td>
<td>4165.7</td>
</tr>
<tr>
<td>Obstacle</td>
<td>4133.5</td>
<td>4138.9</td>
<td>4141.9</td>
<td>4206.0</td>
<td>4212.7</td>
<td>4225.4</td>
<td>4310.0</td>
</tr>
<tr>
<td>Wide Area (B-T-F)</td>
<td>4133.0</td>
<td>4145.0</td>
<td>4172.0</td>
<td>7919.9</td>
<td>32944.8</td>
<td>54083.5</td>
<td>82298.67</td>
</tr>
</tbody>
</table>
The wood obstacle attenuates the Bluetooth wavelength more than the foil obstacle, but there is little difference between them in the 18.5m, and both are acceptable. The BTF direction has the worst transmission time and after 6 meters the transmission time is not acceptable.

Another Bluetooth characteristic that has an effect on the reliability of the communication link is the two devices handshaking to establish connection. Sometimes a noise or a weak wavelength two devices from handshaking, which results in no connection or connection fail. This means that the, relation between distance and handshaking is important for communication reliability and this should be evaluated.

The evaluation is implemented on 0.25, 3, 6, 11, 13.5, 16, 18.5, 20, 22.5 and 25 meters distance at a distance of wide area, obstacle and covered with foil environment for FTF and BTF. 20 handshaking attempts are recorded for each distance under different environmental conditions.

![Figure 30: Successful handshaking under different conditions](image)

The diagram in Figure 30 shows successful handshaking at different distances and under different conditions. After a distance of 6m successful handshaking starts to decrease for BTF and ate distance of 11m BTF is out of service. When it comes to wide area, Bluetooth can carry out handshaking up to 16.5m after which performance decreases.
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering

Bahram Rahmatdoustbeilankouh

Results

2015-12-15

The below table shows successful handshaking at different distances and in different areas.

Table 10: Successful hand shaking results

<table>
<thead>
<tr>
<th>Distance Area</th>
<th>0.25 m</th>
<th>3m</th>
<th>6m</th>
<th>11m</th>
<th>13.5m</th>
<th>16m</th>
<th>18.5m</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wide Area (F-T-F)</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>95%</td>
<td>85%</td>
</tr>
<tr>
<td>Foil obstacle</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>95%</td>
<td>65%</td>
<td>60%</td>
</tr>
<tr>
<td>Obstacle</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>45%</td>
<td>25%</td>
<td>5%</td>
</tr>
<tr>
<td>Wide Area (B-T-F)</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>40%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
</tbody>
</table>

As we can see in the table, foil obstacle and wood obstacle the success rate starts to after 11 meters, decrease but for BTF it is difficult to connect it after 6 meters.
Conclusions

The thesis project had two main goals. The first goal was implementing a reliable and secure communication link between a smartphone and the AutoBox, the second goal was implementing a solution to automate conversion of a Matlab model to run on an Android smartphone. As described in the result chapter, both goals have been achieved and the results have been evaluated. Based on achieved results we can estimate the reliability of the communication link and the accuracy of the code conversion.

As described in the results chapter, the Bluetooth technology has two characteristics affecting Bluetooth transmission time have affected on the Bluetooth transmission time. An evaluation can be found in the previous chapter. Tables 9 and 10 in the results chapter showed the result of transmission time in different situations and the result of hand shaking.

Finally, the reliability of the communication link should be calculated by combining table 9 and 10, which is what is shown in Table 11.

<table>
<thead>
<tr>
<th>Table 11: The reliability of the communication link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wide Area(F-T-F)</td>
</tr>
<tr>
<td>Distance</td>
</tr>
</tbody>
</table>

This table shows the maximum distance of different areas. It is clear that the obstacle decreases the Bluetooth wavelength and decreases the coverage area when we compare the distances in wide area between FTF and BTF. As we can see, the worst reliability is related to wide area with BTF. However, it is acceptable because in a real life situation, inside a truck, the maximum distance between two devices is around 2 meters and the communication link works without problem.
When it comes to layer two of the data communication stack model, data conversion layer is responsible for converting data type and calculating differential method. When data is transferred from the upper layer, it should be converted to byte, and when data is transferred from the lower layer, it should be converted from byte to the require data type. This is because the transmission layer used byte data type. This data conversion affects the data. The accuracy of the data conversion was $10^{-8}$ and has been neglected because it did not affect the final result.

The differential method could increase complexity and improve security. The main advantage of the differential method was decreasing transmission time by data size compression. This solution could compress the data size by 47.195%. As described in section 5.3, data compression decreased transmission time from 12.5s to 6.67s and it optimized the communication link by 46.64%. A transmission time of 6.67s is too long for the users. A user is busy for following construction related for HMI part, the communication link is sending the path data in background. Under upper circumstance, the user does not understand passing 6.67ms transmission time.

The second goal of this thesis project was to convert the Path Plan function developed by Matlab, to run on Android. A middleware function was a connector between Android and Matlab. This function provides a facility to developer to develop the Path Plan function on the Matlab without having knowledge about Android programming. If the Path Plan function is modified in Matlab, new C code will be generated by Matlab. The middleware function accepts this new C code. The modification is hiding from Android part of the middleware function. As mentioned in section 5.1 the APP function could generate the same result as the Path Plan function. In fact, the APP function is autogenerated from the Path Plan function and can run on the Android smartphone. As expected, when the application was tested in a real situation on the truck, the functionality of both functions was the same.
Future Work

This project is a first version of an implementation of a reverse assistance system on a smartphone. The main aim of this project was to find a solution to be able to run the PP function on a smartphone and estimate the reliability of the communication link based on Bluetooth technology. All results are significant for the implementation of the next version and in the final product these results should be considered. In the implementation of the final product, the following should be considered:

- The communication technology, if the driver would like to leave the truck and use the application outside the truck, further away than 6 meters.

- If high speed data rate is required the communication link should be replaced with other wireless technology, such as Volvo Telematic Gateway.

- The serial port in the communication link is a system bottleneck and it should be replaced with high speed wire technology such as an Ethernet cable.

- The body of the PathPlanGenerator function still needs modification for the transfer of data from C to Android.

- The security of the communication link needs to be improved and in section 3.6 there is a suggestion that could be useful if Bluetooth technology is selected for the final product. Based on the selected communication technology, a high level of security should be considered and implemented.
Reference

Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering

Bahram Rahmatdoustbeilankouh


Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering

Bahram Rahmatdoustbeilankouh

Reference
2015-12-15


Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering

Bahram Rahmatdoustbeilankouh

Reference 2015-12-15


[38] S. Misra, “A very simple user access control technique through smart device authentication using Bluetooth communication”, IEEE International Conference on Electronics, Communication and Instrumentation (ICECI) 2014, Pages 1 - 4
Appendix A: User manual

Run C code in Android

This part describes how the modified PP function in Matlab can be connected to Android. All scenarios are described step by step.

1. Android NDK application should be downloaded from the below link. Select the correct file according to your computer hardware.


2. Downloaded files should be unpacked and folder path added in the Windows system path.

3. Generate a new C file from the modified PP function and past C files in the JNI folder.

4. In the Windows command prompt, go the project folder and run the ndk-build command. If you get the same results as in the below figure, it means your code is compiled correctly and is ready for use.
If you encounter an undefined reference error like the one in Figure 32, you can see that the error relates to `rt_atan2 (double, double)` function.
Communication link and code conversion between vehicle and smartphone for low speed semi-autonomous maneuvering
Bahram Rahmatdoustbeilankouh

Appendix A: User manual

2015-12-15

Next, find the source .h file where the body of function is defined and add it as header file in com_bluetoothserialtest_NDKLib.cpp file. In this example, you should add the #include<rt_atan2.c> line in the com_bluetoothserialtest_NDKLib.cpp file and recompile the code by using the ndk-build command.

An important rule for modification of the PP function in Matlab is the following:

1. If you have changed the global variable name in the Simulink model, you should change the variable name to the same name as in the Simulink model in the com_bluetoothserialtest_NDKLib.cpp file shown in the below figure.
Figure 33: Modify variable name

2. Make the PP function a subsystem and create a C file from the subsystem.