Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für
Technische Informatik und
Kommunikationsnetze

Nils Braune

Telemetry Unit for a Formula Student Race Car

Semester Thesis SA-2013-67
November 2013 to February 2014

Tutor: Mahdi Asadpour
Supervisor: Prof. Dr. Bernhard Plattner

Abstract

In this thesis, we implement a wireless telemetry solution for a Formula Student race car. The Formula Student racing series is a competition organised by the Society of Automotive Engineers and is restricted to student teams representing their university. The telemetry solution is to transmit sensor data and computed values from the on-board computer as well as a video signal over a distance of 600m.

In order to realize our implementation, we first address the requirements as well as we use theoretic models to assess the possible reliability we can expect from our connection. We also analyse the previous solution in the light of the aforementioned theoretic models in order to derive possible solutions to issues encountered with the previous implementation.

Based on the findings from both the theory and the analysis of the previous system, we design our new implementation. Eventually, we evaluate our system under conditions which represent the given requirements so as to find out whether it meets the specified goals.

[1] Title image: © AMZ Racing

Contents

1 Introduction **5**
- 1.1 Motivation . . . 5
- 1.2 Task Description . . . 5
- 1.3 Related Work . . . 6
- 1.4 Overview . . . 6

2 Problem Description **7**
- 2.1 System Definition . . . 7
- 2.2 Wireless Communication . . . 7
- 2.3 Positioning . . . 8

3 Design **9**
- 3.1 Theoretic Background . . . 9
- 3.2 Previous Implementation . . . 10
- 3.3 Choice of Wireless Protocol . . . 11
  - 3.3.1 ZigBee . . . 11
  - 3.3.2 Bluetooth . . . 12
  - 3.3.3 3G/4G Cellular Networks . . . 12
  - 3.3.4 WLAN . . . 12
- 3.4 Transmitters . . . 12
- 3.5 Antennas and Positioning . . . 12
  - 3.5.1 Choice of Antennas . . . 12
  - 3.5.2 Positioning . . . 13
- 3.6 Software Client . . . 14
  - 3.6.1 Choice of Client Software . . . 14
  - 3.6.2 Extending the Intecrio Experiment Environment . . . 15

4 Evaluation & Results **19**
- 4.1 Overview . . . 19
- 4.2 Impact of the Rear Wing . . . 19
- 4.3 Range Evaluation . . . 20
  - 4.3.1 Throughput . . . 20
  - 4.3.2 Signal Strength and Noise Floor . . . 23
- 4.4 Range Evaluation with longer Pole . . . 23
  - 4.4.1 Throughput . . . 23
  - 4.4.2 Signal Strength and Noise Floor . . . 24
  - 4.4.3 Jitter . . . 25
- 4.5 Impact of Movement . . . 26
  - 4.5.1 Throughput . . . 26
  - 4.5.2 Latency . . . 27
  - 4.5.3 Jitter . . . 27
  - 4.5.4 Packet Loss . . . 28

5 Conclusion & Future Work **29**
- 5.1 Conclusion . . . 29
- 5.2 Opportunities for Future Work . . . 29

A Scripts 33
A.1 Test Scripts 33
A.1.1 Range Evaluation 33
A.1.2 Range Evaluation with 3m Pole 37
A.1.3 Impact of Movement 40

B Datasheets 53

C Example Documentation 57
C.1 docker.js 57
C.2 AMZ Wiki 58

Chapter 1

Introduction

1.1 Motivation

The Formula Student is an international race car competition for student teams representing their respective university. As race cars in this competition grow increasingly complex, so do their control systems. The cars need to compete in several categories under different conditions. Often, the race car's control units are operating using changing parameters for each of the different categories.

Using bidirectional telemetry solutions, the control system of a car can be supervised as well as it can be modified and tweaked. They provide insight into sensor data like the current SoC (state of charge of the car's HV battery) as well as system parameters like the current maximum torque permitted to the actors. Additional data collected in the car, such as video data, can also increase the understanding of the observed processes.

Wireless telemetry solutions enable these possibilities without physically accessing the car. Applications range from debugging, calibrating and testing to remotely modifying the car's behavior during races.

1.2 Task Description

The objective of this thesis is to implement a wireless telemetry solution for the AMZ (Akademischer Motorsportverein Zürich) Formula Student race car. The range of the wireless connection should cover distances up to 600m, so to cover all common race tracks of the Formula Student series. The reliability of the connection should be tested with respect to range in conditions with little to no obstacles, thus simulating a setting similar to the actual race tracks.

As part of the thesis, a suitable network technology (Bluetooth, ZigBee, WLAN, etc.) and operating frequency should be chosen to fit the given requirements and constraints. The implementation should be able to transmit sensor data as well as data from the internal state of the VCU (Vehicle Control Unit) to a laptop computer or similar device next to the race track.

Besides this, video data recorded using a GoPro camera should be transmitted over the same distance, in order to allow for further insight into the processes under consideration as well as to give a training assistance to drivers in the team currently not driving the car. Key goals of the implementation must be flexibility and ease of use, as well as minimum weight and form factor of all physical parts in the race car.

1.3 Related Work

In [3], a wireless telemetry solution for a Formula Student race car is implemented using ZigBee in the 2.4GHz spectrum. Other authors [2] have also successfully deployed ZigBee hardware for their implementation. However, the nominal maximum throughput of 250kb/s that ZigBee provides are too low for our needs. Other authors [4, 5] have made use of WLAN solutions for their telemetry implementations, with resulting ranges of up to 640m and reliable connections at 100km/h during measurements.

On the software side, various solutions to the needs of telemetry have been implemented by Formula Student teams. Solutions deploying existing software products such as National Instruments' Labview [1] as well as solutions using custom software solutions [6] have been implemented.

In this thesis, we propose a wireless telemetry system based on WLAN technology for the AMZ Racing team. Unlike in the related work mentioned above, it is required to have sufficient bandwidth for a video signal in addition to the telemetry data proceeded by the VCU. In addition to that, we propose a software solution built around the software used in the team to configure the VCU, such that the learning curve of the telemetry system is flat for members of the team. This results in a system which is specifically tailored to the needs and requirements of the AMZ Racing team.

1.4 Overview

In Chapter 2, we give an in-depth description of the problem. Chapter 3 starts with giving the necessary theoretic background, then discusses the previous approach in the light of the theoretical models and finally presents our proposed design. In Chapter 4, we describe our test setup to evaluate our system and present the results of that evaluation. In Chapter 5, we summarize our findings and give an outlook to possible future work.

Chapter 2

Problem Description

2.1 System Definition

In this section we want to introduce a definition of the system which is to be implemented. As can be seen in Figure 2.1, the system in divided into two parts: the part in the race car and the part at the base station. On the car side, the data to be transmitted comes from the vehicle control unit (VCU), which is connected to all actors and sensors in the car. It provides the data over a 100Mb/s Ethernet connection. As described in Section 2.2, the mobility of the car side requires a wireless solution to transmit the data. Therefore, the telemetry system consists of a transceiver module connected to an antenna on both the car side as well as the base station side. Those components form the hardware part of the telemetry system.

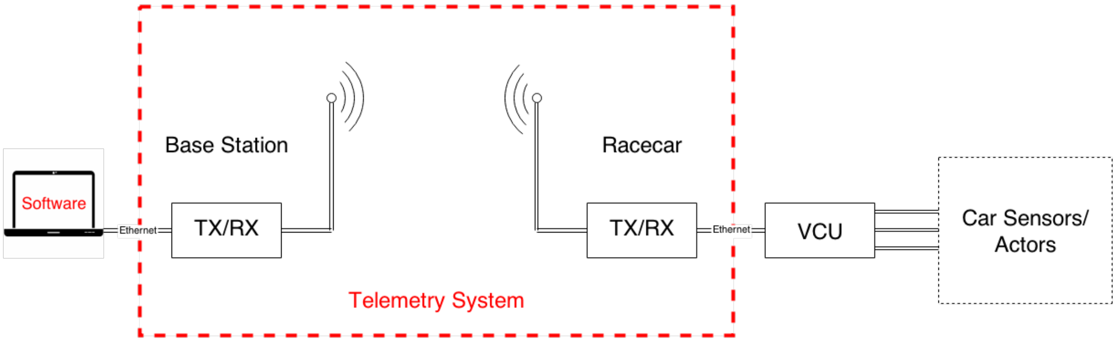


Figure 2.1: Definition of the telemetry system (red).

On the base station side, one can connect a laptop to retrieve the telemetry data. In order to to display and process the data gathered, a software part is required in addition to the hardware components.

2.2 Wireless Communication

The requirement of an unimpaired mobility for the race car leads to the choice of a wireless communication channel for our telemetry implementation. Wireless communication relies on the dissipation of electromagnetic waves in a given frequency band. This imposes several constraints on the link.

The strength of the electromagnetic field emitted from a source decreases proportionally to the inverse of the squared distance to that source. Field strength also depends on the material the field passes through. While materials of low conductivity impose higher losses to the field than air, materials of high conductivity work as reflectors to the field. Thus, every obstruction from physical objects decreases the field strength measured at the receiver's end.

Moreover, the properties of a wireless link depend on the chosen frequency band. Using a higher frequency band positively increases maximal throughput while making the link more vulnerable to the attenuation from obstruction. Also, some frequency bands are used more often than others, so that there is a risk of having interfering signals from other parties resulting in additional noise on the channel.
In order to receive a signal at one end of the channel, the received field strength of the signal coming from the transmitting end must surpass the level of the surrounding noise floor. If the signal strength falls below that level, the receiver is unable to distinguish it from the noise.

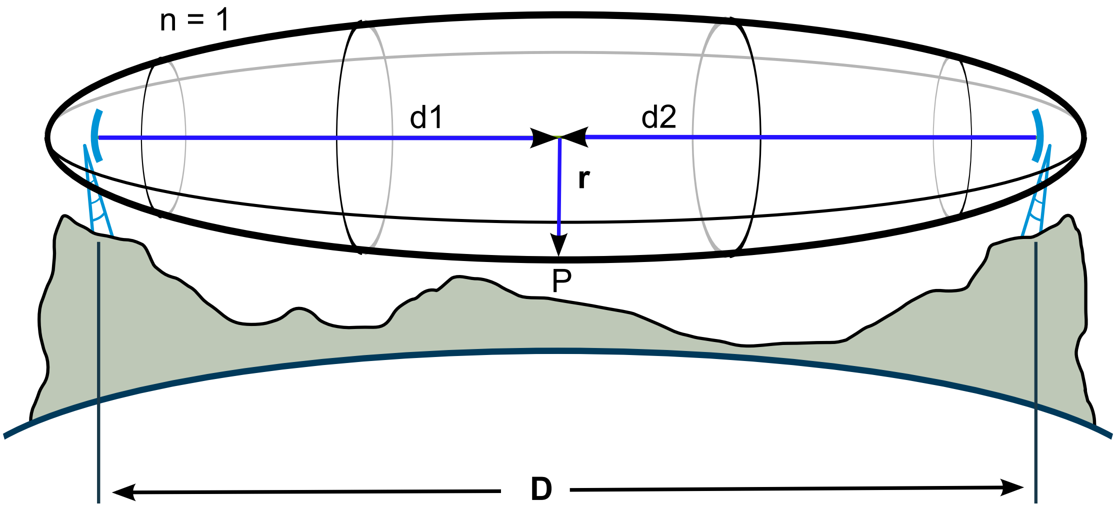


Figure 2.2: First Fresnel zone[1].

When electromagnetic waves propagate from one antenna to another, the majority of the energy is transported in the first Fresnel zone. This is the circular ellipsoid with the antennas in its focus points, as depicted in Figure 2.2. Therefore, from a geometric perspective, the quality of a wireless channel is not just defined by a direct line of sight, but instead depends on an unobstructed first Fresnel zone.
In order to meet the specified goals for the system, all of the aforementioned inherent properties of wireless channels have to be considered when designing the telemetry implementation.

2.3 Positioning

Further issues to consider are related to positioning. The data which is to be transmitted is coming from the VCU, which collects and processes all sensor data using the controller area network (CAN) protocol and controls the actors in the car. The telemetry transmitter can be connected to the VCU using an Ethernet connection. However, the VCU is placed in the monocoque so as to be proteced from physical damage. The car itself consists mostly of electrically conductive materials such as aluminium and carbon fibers. While the telemetry transmitter can be placed in proximity of the VCU in order to benefit from the same protection, it is a requirement for a wireless channel to have the least possible obstruction in the first Fresnel zone between the two antennas of a wireless link. Therefore, the positioning of the antenna in the race car influences the quality of the transmission and therefore has to be considered.

Another point to consider is weight. As many dynamic properties of the race car improve when weight is reduced, it is a concern that every component required for the wireless channel is as lightweight as possible. Furthermore, the space in the car is constrained, as smaller volumes result in less weight of material as well as less cross-section surface, which improves the aerodynamic properties of the car. Thus, small-sized components should be chosen.

[1] By Jcmcclurg [CC-BY-SA-3.0] via Wikimedia Commons

Chapter 3

Design

3.1 Theoretic Background

In this section, we want to give an overview over some of the theoretic models used to design the wireless telemetry implementation.

We begin by using a link budget [7, 8], which is an accounting of all gains and losses from the transmitter to the receiver, we can model the fade margin left for the transmission of our data [7]. In the following we will use 3.1 as a starting point for our link budget:

$$P_{RX} = P_{TX} - L_{TX} + G_{TX} - L_{FSPL} - L_{DIV} + G_{RX} - L_{RX} \tag{3.1}$$

where:

P_{RX}	= received power
P_{TX}	= transmitted power
L_{TX}	= losses in the antenna cable, at connectors etc. (transmitter side)
G_{TX}	= gain transmitter antenna
L_{FSPL}	= free-space path loss
L_{DIV}	= various other losses, e.g. by (partial) obstruction of the first Fresnel zone, weather conditions etc.
G_{RX}	= gain receiver antenna
L_{RX}	= losses in the antenna cable, at connectors etc. (receiver side)

The losses in the case of free-space dissipation is the loss a link experiences when there is no obstruction by either rigid or fluid or gaseous obstacle, and can be modelled using the following equation:

$$L_{FSPL} = 20 \cdot \log\left(\frac{4 \cdot \pi \cdot d}{\lambda}\right) \tag{3.2}$$

Using 3.2, we can compute the free-space path loss for various frequencies legally usable in Europe without the use of a special license. The free-space path loss, computed for the frequencies of 868MHz, 2.4GHz and 5.5GHz and a distance of 600m can be found in Figure 3.1. As we can see, the path loss increases with increasing frequency.

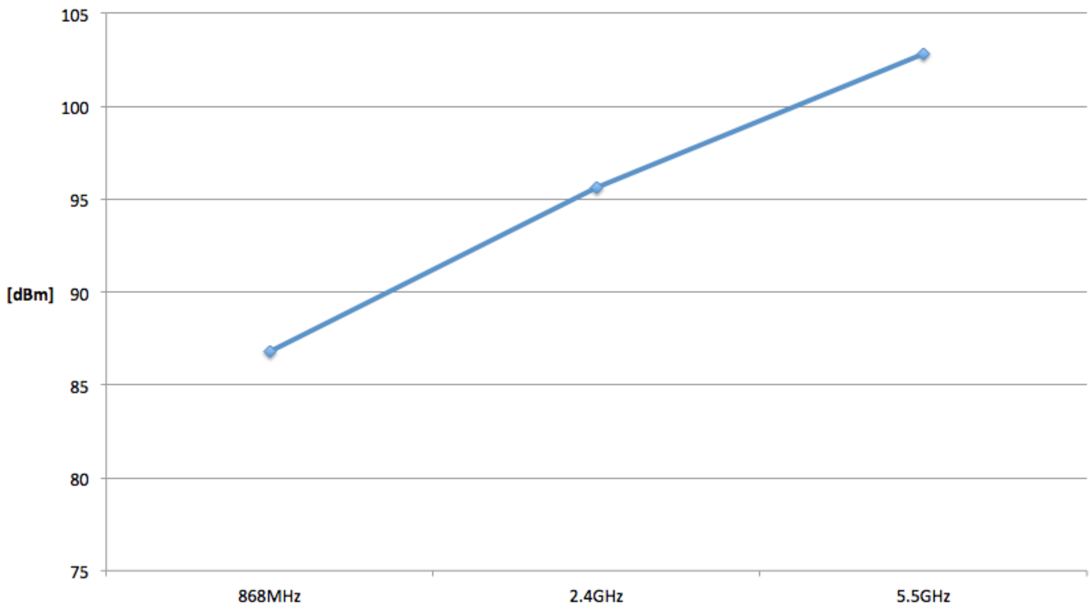


Figure 3.1: Free-space path loss at 600m.

The radius of the cross-sectional circular area of a Fresnel zone can be found using the following formula:

$$r_n = \sqrt{\frac{\lambda \cdot n \cdot d_1 \cdot d_2}{d_1 + d_2}} \tag{3.3}$$

Here, d_1 and d_2 denote the distance of the given point from the antennas, λ the wavelength and n the order of the Fresnel zone. Thus, in order to find the radius of the first Fresnel zone, one has to choose $n = 1$. Using 3.3, we can find the maximal radius of the first Fresnel zone as:

$$r_{max} = \frac{\sqrt{\lambda \cdot d}}{2}$$

where d is the distance between both antennas.

We can thus see, that while the L_{FSPL} increases with frequency, the maximum radius of the first Fresnel zone decreases with frequency. Therefore, there is a trade-off to be made.

3.2 Previous Implementation

Last year's race car had an implementation for a wireless telemetry solution. However, it was unable to cover the range of an entire race track. Thus, this implementation was analysed considering the theoretic aspects discussed in the section above and measures were derived to address possible issues.

As last year's VCU only provided CAN outputs, the telemetry solution used an Avisaro WLAN Cube CAN [12] shown in Figure 3.2, which directly forwards CAN packets over 802.11b/g in ad hoc mode. Its datasheet can be found in Appendix B. In order to reduce weight, the housing shown in Figure 3.2 was removed from the device. For the WLAN Cube, a 2dBi vertically polarised omnidirectional antenna was used. The WLAN Cube was directly communicating wirelessly with a laptop computer using the built-in laptop antenna. Using the built-in laptop antenna as well as the 2dBi antenna in the car, one loses the degree of freedom to positively affect the link budget through the use of high gain antennas. Another way of positively affecting the link budget is given up with the possible transmit power of both the laptop and the WLAN Cube.

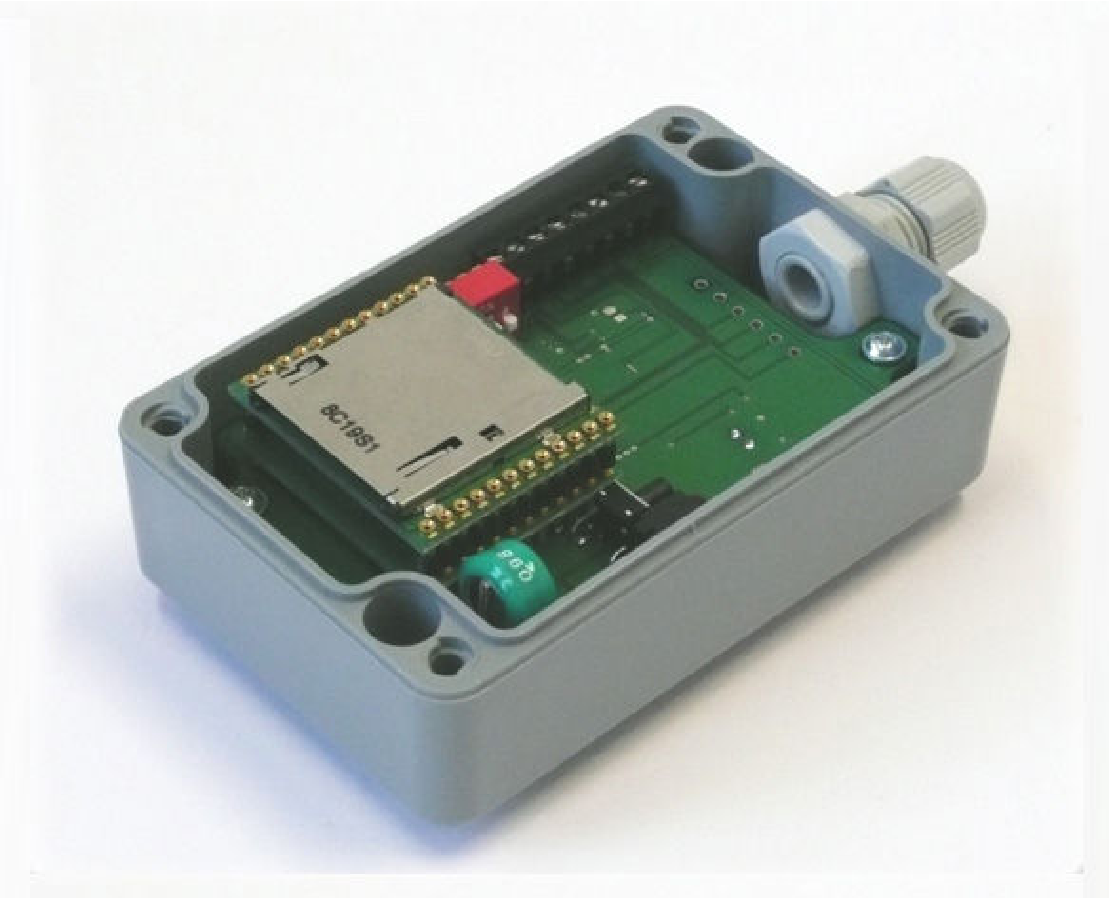


Figure 3.2: Avisaro WLAN Cube CAN[1].

Furthermore, the antenna in the car was mounted behind the main roll hoop, which consists of steel and thus works as a reflector to electromagnetic waves, obstructing the field. Altering the positioning of the antenna can improve this situation.

3.3 Choice of Wireless Protocol

In this section, we explain the reasoning behind the choice of wireless protocol used in our design. For this, we go through a number of wireless protocols and compare their properties to the given requirements.

3.3.1 ZigBee

ZigBEE is a wireless communication protocol using the IEEE 802.15.4 standard as base for the PHY and MAC layers. It is designed for low-cost and low-power devices and the usage of ISM frequency bands. It is able to operate in multiple frequencies, including the 868MHz band for Europe, the 915MHz band for the US and the 2.4GHz band worldwide. However, available bandwidths and thus available throughput rates are not the same for all frequency bands. While on 868MHz there is an available bandwidth of 300MHz, allowing for up to 20kb/s, on 2.4GHz there is room for up to 250kb/s [11].

While the simplicity of the protocol is an advantage, both maximal throughput rates are not sufficient for our requirements, which is thus ruling this option out.

[1] © Avisaro AG

3.3.2 Bluetooth

Like ZigBEE, Bluetooth is a wireless communication protocol operating in the 2.4GHz ISM frequency band. It provides advantages like simple setup of connections and rebustness to interferences through adaptive frequency hopping spread spectrum (AFHSS).

It can provide nominal throughputs of up to 24Mb/s. However, as it is conceived as a wireless standard for peripheral devices, the availability of higher power devices is reduced compared to WLAN, and mostly limited to USB Standard A or B type connectors, which the VCU doesn't provide.

3.3.3 3G/4G Cellular Networks

Another possible consideration is the usage of 3G or 4G wireless network devices, which are designed for usage in mobile contexts and providing high data rates. However, devices using these standards operate in commercially licensed frequency bands, forcing end users to use the networks of service providers which can afford a license for these frequency bands.

This brings up the issue of unreliability, as there is no guarantee that network coverage on the event locations is given. Furthermore, as at the event dates may people will be using the infrastructure provided by the service provider, the availability of the network further decreases. And lastly, the service providers do not offer their services for free, increasing the costs of this option during operation.

3.3.4 WLAN

In order to meet the bandwidth requirements given, WLAN was chosen as wireless technology stack. It provides a higher throughput than other wireless protocols in the same frequency band of 2.4GHz. Furthermore, as WLAN is a commonly used technology stack, there is a plethora of available options for possible solutions.

3.4 Transmitters

For the transmission of our telemetry data, we choose the Ubiquity PicoStation M2 HP [13]. Its datasheet can be found in Appendix B. It provides a data transmission at an output power of up to 28dBi while having a weight of less than 80g. As described in Chapter 2, it is connected to the VCU using an Ethernet connection and is powered via PoE at an input voltage of 12V.

In addition to standard WiFi communication using 802.11g, it provides a proprietary addition from Ubiquity called airMAX. airMAX uses TDMA instead of the standard CSMA/CA to coordinate the usage of the shared medium in order to increase throughput. In Chapter 4, we will show the results of our evaluation for both airMAX switched on and off.

3.5 Antennas and Positioning

3.5.1 Choice of Antennas

As noted in Section 3.1, the use of high-gain antennas can increase link quality. However, a higher antenna gain is intrinsically tied to a reduction in beam width. Thus, it is necessary to choose antennas according to the requirements of the given use case. In our case, we have one antenna on the car, which must be as lightweight as possible and which needs to be able to provide a 360° beam width in horizontal direction, as the car can assume any possible yaw angle. In vertical direction, however, we can choose a lower beam width in order to increase gain.

These requirements lead to the choice of a vertically polarised omnidirectional antenna. This is a widely-used type of antenna, as it fits many common use cases, which increases the range of possible models to choose from. However, this also includes that many other devices on the same frequency band use the same polarisation, which might increase the noise floor.

The chosen antenna is a 4dBi rubber duckie antenna for the 2.4GHz frequency band. It has a vertical beam with of 50° and a horizontal beam width of 360°.

The base station antenna is to cover an angle of 180° in order to cover the entire race track at Formula Student events. As the number of spots where the base station can be put is restricted by the event rules, a smaller horizontal beam width is not always able to cover the entire track. Also, as the antenna can be larger in size than the antenna in the car and as the vertical beam width can be reduced with this antenna as well, we can choose a high gain antenna. However, because the antenna in the car has to be vertically polarised, we must choose a vertically polarised antenna for the base station as well.

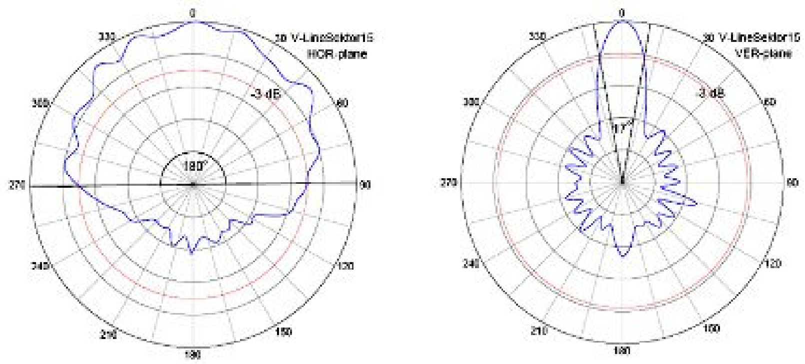


Figure 3.3: Radiation pattern of sector antenna[2].

We deploy a 15dBi 180° vertically polarised sector antenna for our base station. Its radiation pattern can be seen in Figure 3.3. Using a sector antenna provides us with two advantages: on one hand, having a sector antenna rather than an omnidirectional antenna increases the possible gain, while on the other hand, the directivity of the antenna reduces the influence of interferences coming from behind the antenna.

3.5.2 Positioning

As mentioned in Section 2.3, most parts of the race car consist of electrically conductive material and therefore work like reflectors to the propagating electromagnetic waves. Hence, in order to have a field which is not obstructed by components of the race car itself, the antenna in the car must be at a location it can dissipate its field as freely as possible.

The Formula SAE rules [14] require each car to have roll hoops to protect the driver in case the car rolls over in the event of an accident. The largest roll hoop, the so-called main roll hoop (MRH), is to protect the driver's head. Its top part is one of the highest points of the car. Therefore, positioning the antenna at the top of the MRH provides two advantages: it minimizes the impact of obstruction by parts of the car and it increases the part of the first Fresnel zone above the ground.

Figure 3.4 depicts the altered positioning of the antenna compared to last year. While in the car from the last racing season, the antenna's field was obstructed by both the rear wing and the MRH, positioning the antenna on top of the MRH circumvents a large part of the obstruction, leaving only parts of the field obstructed by the rear wing.

[2] © Cyberbajt

Figure 3.4: Changed positioning of the antenna on the MRH.

3.6 Software Client

When the telemetry signal arrives from the car to the user's laptop, a client software needs to handle the data and display it to the user. Considering several options, in Subsection 3.6.1 we describe the evaluation which led to the actual choice for the client software, while in Subsection 3.6.2, the extension mechanisms of the chosen software are described. Using those extension mechanisms, we can change the user interface to fit the given requirements.

3.6.1 Choice of Client Software

In this section, we describe the decision process leading to the choice of the client software. We consider three options: running the telemetry solution using the XCP protocol and Vector CANape as client software, writing an own client software as well as using the Intecrio Experiment Environment, which is a client software shipped together with the VCU.

XCP & Vector CANape

XCP [16] is a network protocol for measurement and calibration of control devices and the successor of CCP [17] . Unlike it's predecessor, which was restricted to the CAN protocol, it is capable of operating on more than one transport medium.

By having an exact representation of the occupation in the memory of the supervised device, in our case the VCU, contained in a so-called A2L file, XCP can directly read and write the vales of interest to and from the device. Vector CANape [18] is a software which provides this functionality. It also allows for the display of video inputs in its interface besides the representation of the controller data.

However, the ETAS VCU only provides an XCP bypass, meaning that while control devices connected to the VCU can use the connection to the VCU as a relay to a monitoring PC, while the VCU itself does not support XCP. Thus, this solution is not an option.

Custom Client

Another option considered is to write an own client software to handle and display the telemetry data. For this to work, the ability to get the data from the VCU in a format which can be parsed is a requirement. As it is not possible to receive XCP packets originated from the VCU, as described in the section above, another option has to be considered.

The VCU comes with a client software named Experiment Environment, which will be described in more detail in the following subsection. It uses a proprietary protocol to transmit the data between the VCU and the Experiment Environment. Therefore, being able to process this protocol would allow to replace the Experiment Environment with a custom made client software.

However, a Wireshark[3] analysis of the traffic between the VCU and the laptop running the Experiment Environment shows that the data is encrypted using TLS. Therefore, the data cannot be directly accessed, which inhibits this approach as well.

Experiment Environment

As mentioned in the subsection above, the VCU is shipped with a client software, the Intecrio Experiment Environment. As the Experiment Environment is used in the process of programming the VCU, the users of the telemetry solution will already be familiar with the software when using it for telemetry.

It provides possibilities to flash the VCU as well as measuring and calibrating the signals processed by it. Using widgets called instruments, the received signals can be displayed and inspected.

The Experiment Environment's UI is extensible through the means of the .NET framework using Windows Forms [19], which allows for the development of custom instruments to extend the user interface.

The extensibility as well as the considerations given in the previous subsections lead to the choice of the Intecrio Experiment Environment as the software side of the telemetry solution.

3.6.2 Extending the Intecrio Experiment Environment

Widget Development

To meet the team's needs, custom instruments are created in addition to the natively provided ones. The .NET language used to implement the additional instruments is C#. Each instrument consists of two parts, a manifest file ending in .Plugin.xml as well as a DLL containing the actual code. Every instrument must have a reference to EE.Widgets.interfaces.dll and must be in the ETAS.EE.Widgets namespace. In Figure 3.5, a UML representation of the most important classes and interfaces can be seen.

In the following table, we give an overview over the different roles of the interfaces and their implementing classes shown in Figure 3.5:

[3] wireshark.org

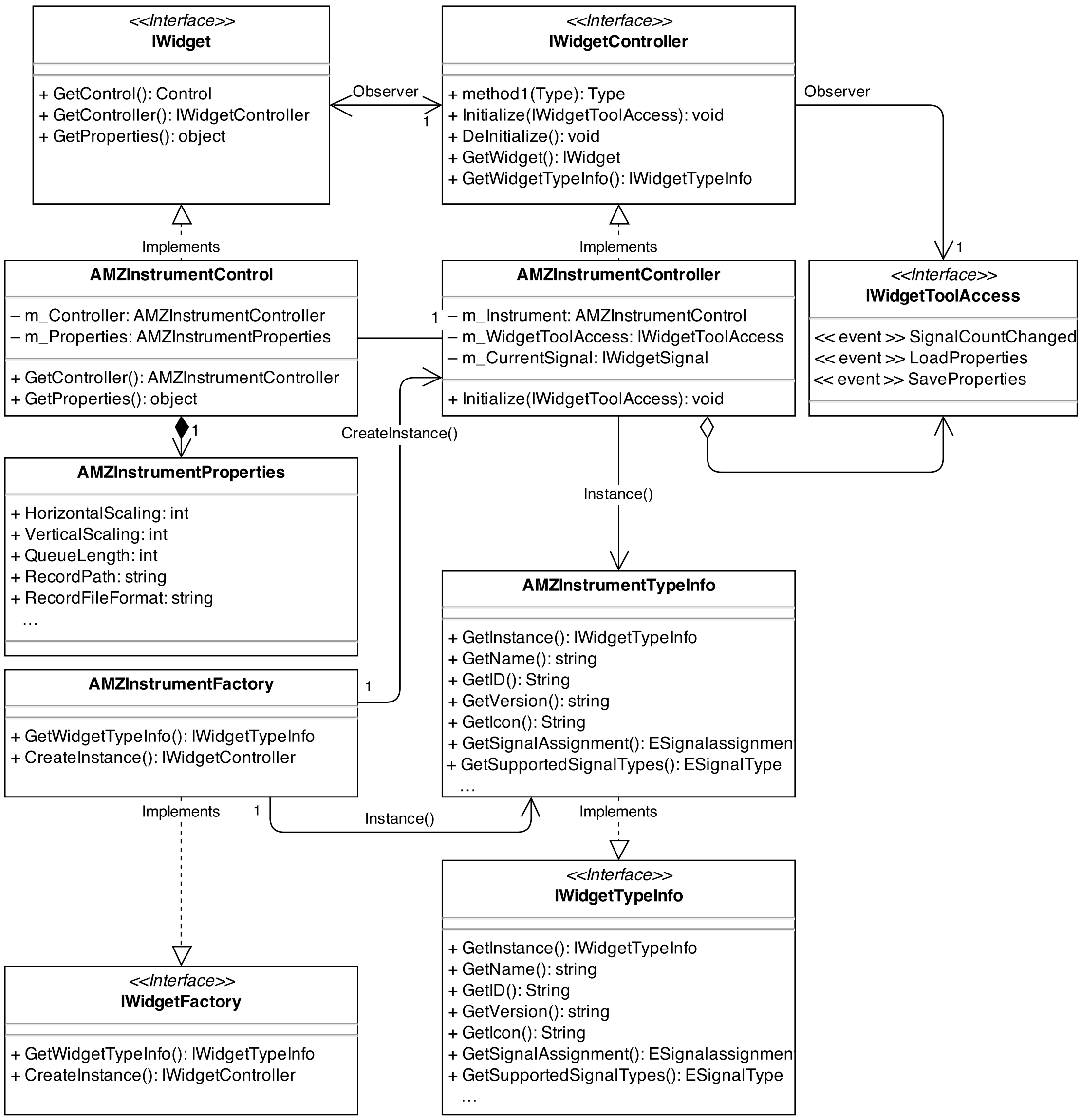


Figure 3.5: UML pattern of an instrument.

IWidgetTypeInfo — identifies the instrument type, contains the instrument-specific information displayed in the list of available instruments in the application, defines, which and how many incoming signals can be assigned, and sets default values for the size and the layout

IWidgetFactory — factory class loaded at application start, creating new instrument instances and providing the type info

IWidget — the view, represents and renders the visual part of the instrument

IWidgetController — the controller, contains the control logic of the instrument

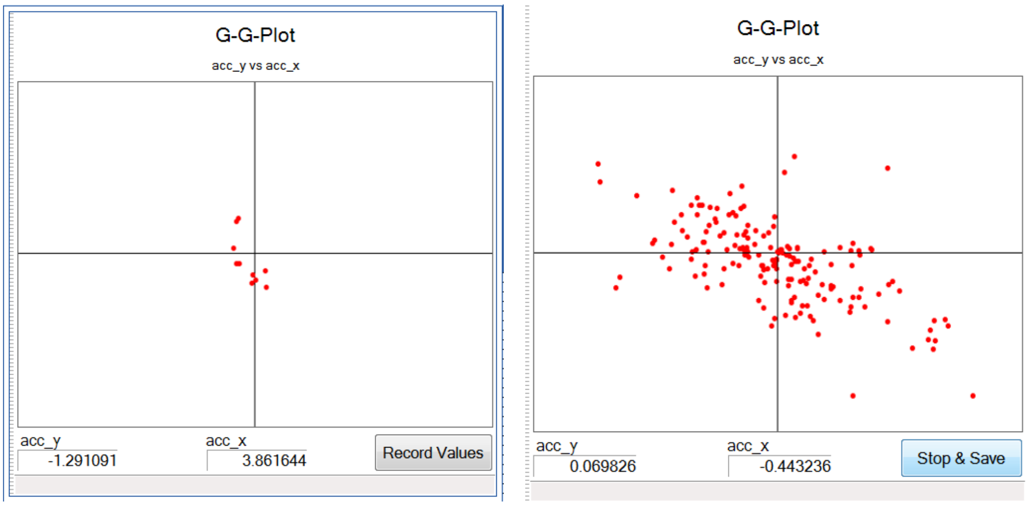
IWidgetToolAccess — an object implementing this interface is passed to the controller. gives access to the telemetry data, provides load/save functionality

As the know-how needs to be kept within the team for the next generation of members, a detailed and in-depth documentation is an explicit requirement. To meet this goal, two measures are taken: setup procedures and configurations are documented in a wiki [20], while code documentation is done using docker.js [21]. As an example, a screenshot of a documentation file generated with docker.js as well as a wiki page from the AMZ wiki can be found in appendix C.

To demonstrate some of the capabilities of the extension mechanisms of the Experiment Environment, two of the implemented widgets are described in more detail: the GG-plot widget and the video widget.

GG-Plot

A GG-plot is a way to display two different acceleration values in one graph. A common use case in the context of car development is the display of the lateral versus the longitudinal acceleration values. As an example, our implementation of a GG-plot can be seen in Figure 3.6.



(a) normal operation (b) recording

Figure 3.6: GG-plot implementation as an instrument in Experiment Environment.

The implementation in Figure 3.6 takes two signals from the Experiment Environment and displays them on the plot, which is implemented using [24]. It includes a number of features. In general, the plot not only displays the current value but keeps the history state of the last several values. The number of past values can be customized as described below.

Using the "Record" button on the lower right of the widget, as seen in Figure 3.6a, one can prevent the deletion of past values entirely. A possible use case for this is when one is not interested in the current situation but more generally a longer term profile of the measured situation. For example, one could get a profile of a driver's behaviour and give him feedback to optimize his driving performance. After the "Record" has been clicked, it swaps to a "Stop & Save" button, as shown in Figure 3.6b, allowing the user to turn back to the previous behaviour as well as saving the recorded profile to an image file.

The GG-plot's look and behaviour can be changed using the .NET property grid [22, 23]. This allows to change the number of past values displayed on the graph, the scaling in horizontal as well as in vertical direction, the storage location and file extension for the recorded file when clicking the "Stop & Save" button and the background color.

Video Widget

To be able to display the video signal in the same tool as the telemetry data, a video widget was implemented. It can receive a live stream and display it next to the other instruments the user chooses to use. It is implemented using a WebBrowser control [25]. The WebBrowser control offers to render web pages using the Trident browser engine also used in the Internet Explorer. This allows us to use it to include ActiveX scripts into our widget.

To display the video, we use the ActiveX plugin of VLC media player, which is a free and open source video player and streaming software. The ActiveX plugin can play and stream video and audio in all codecs the standalone player can decode. Thus, by embedding the ActiveX VLC player pointed to the streaming source in a local HTML file which is displayed by the WebBrowser control in the widget, we can get the video stream directly into the Experiment Environment.

Chapter 4

Evaluation & Results

4.1 Overview

In order to evaluate our design, several tests were conducted. First, given the chosen positioning within the car, the impact of the rear wing on the signal was measured. The results of this measurement can be found in Section 4.2.

In Section 4.3, we discuss the setting and results of a range test measuring throughput, signal and noise levels as well as latency using a pole of 180cm length. In order to reduce obstructions in the first Fresnel zone at longer distances, the same tests were conducted using a 300cm long pole. Results for this evaluation can be found in Section 4.4.

Furthermore, as we have an antenna on a moving vehicle, tests were conducted to measure the effect of movement on the wireless transmission. They can be found in Section 4.5.

4.2 Impact of the Rear Wing

As described in Section 3.5.2, the antenna in the car is mounted on top of the MRH. This increases distance to the ground and reduces the number of obstructions by parts of the car. However, one part of the car which can still influence the communication is the rear wing.

As of the season 2015, there will be a new rule [15] (specific to Formula Student Germany), which prohibits the mounting of parts to the car which are higher than the MRH. The reasoning behind the introduction of this new rule is that often, the rear wing reduces the visibility of the TSAL, which is a red indicator light mounted to the MRH and used to indicate that the tractive system is energised. Therefore, a rule request to the organisers was made to query whether it would still be possible to install an antenna on top of the MRH, with affirmative answer. Thus, from next season onwards, the obstruction of the field by the rear wing will be reduced.

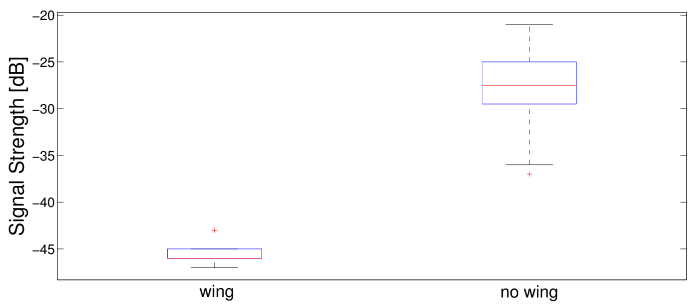


Figure 4.1: Signal level with and without obstruction by the rear wing.

Until then, the rear wing might still get into the way of the telemetry link, which was why measurements were taken so as to assess the influence of the rear wing. For this, the signal strength was first measured without obstruction by the rear wing, at a distance of 5m. Then, the same measurement was taken with the rear wing between the two antennas. As can be seen in Figure 4.1, the influence of the rear wing is an attenuation of about 18dB, which corresponds to a reduction of range with a factor of 64 in the direction the signal is completely obstructed by the rear wing.

4.3 Range Evaluation

The evaluation took place on a free field, as shown in Figure 4.2, with minimal obstruction, as these conditions resemble the testing facilities the race car will be tested at as well as most race tracks it will compete at.

Figure 4.2: Satellite picture of the test location[1].

The testing range was mostly flat, with no trees obstructing the line of sight. The base station antenna was mounted on a pole which allowed it to be positioned 1.8m above the ground. We measured link capacity, noise and signal strength in intervals of 50m up to a distance of 600m. Additionally, every 100m we made the same measurements with the base station antenna turned to 45° and 90° with respect to the omnidirectional antenna, so as to measure the dependency of the link quality to the angle.

The tests were conducted by using the test script which can be found in Subsection A.1.1 of Appendix A.

4.3.1 Throughput

Throughput was measured using iperf3[2], which is a free and open source connection measurement tool for measurements on the transport layer. With this tool, the maximal throughput of a connection was measured using the TCP mode. At each distance, the throughput was

[1] © Google

[2] http://code.google.com/p/iperf/

measured in airMAX mode as well as in normal WLAN mode without proprietary additions. The results can be found in Figure 4.3. As can be seen, throughput is close to constant up to a distance of 400m and then begins to drop off.

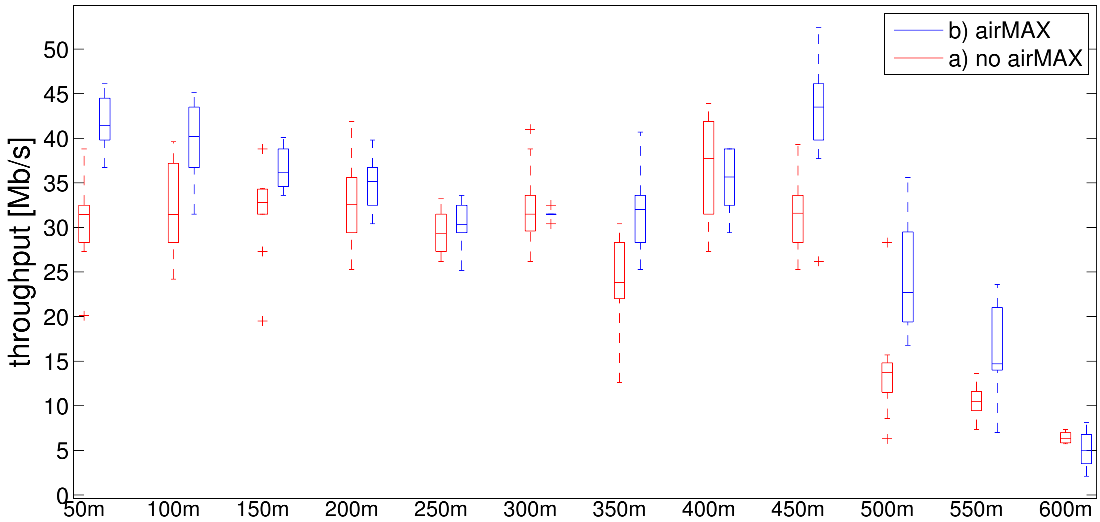


Figure 4.3: Throughput with and without airMAX enabled.

Also, in average the use of airMAX increases the maximum available throughput. To illustrate this more clearly, in Figure 4.4 the difference of the median throughput using airMAX and the median throughput without using airMAX is shown. The median overall difference is 3.8Mb/s.

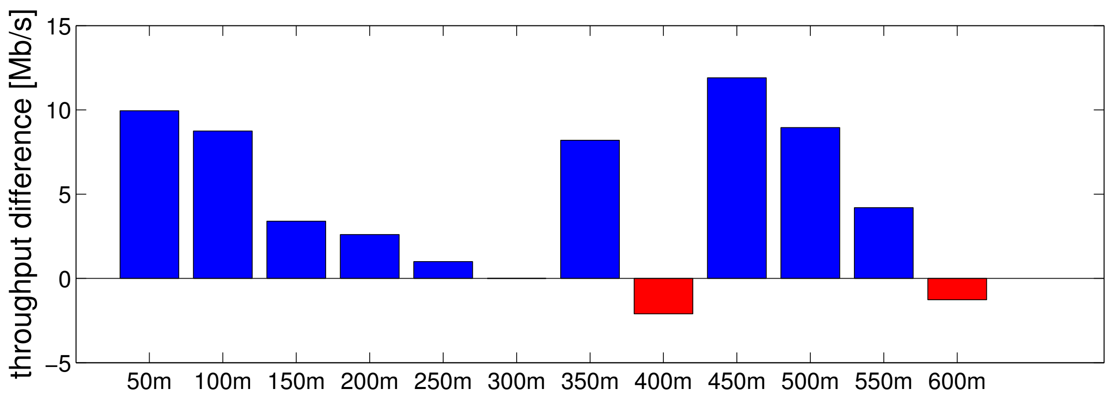


Figure 4.4: Difference in median throughput when using airMAX compared to not using airMAX.

As mentioned above, every 100m measurements at angles of 45° and 90° of the main radiation direction were taken. The results for the measurements without use of airMAX are shown in Figure 4.5, the results of the measurements with airMAX turned on can be found in Figure 4.6. From the aforementioned figures one can take that throughput drops at 90° when compared to the throughput measured at 0° or 45°. At 600m, the signal at 90° was to weak ensure a reliable connection.

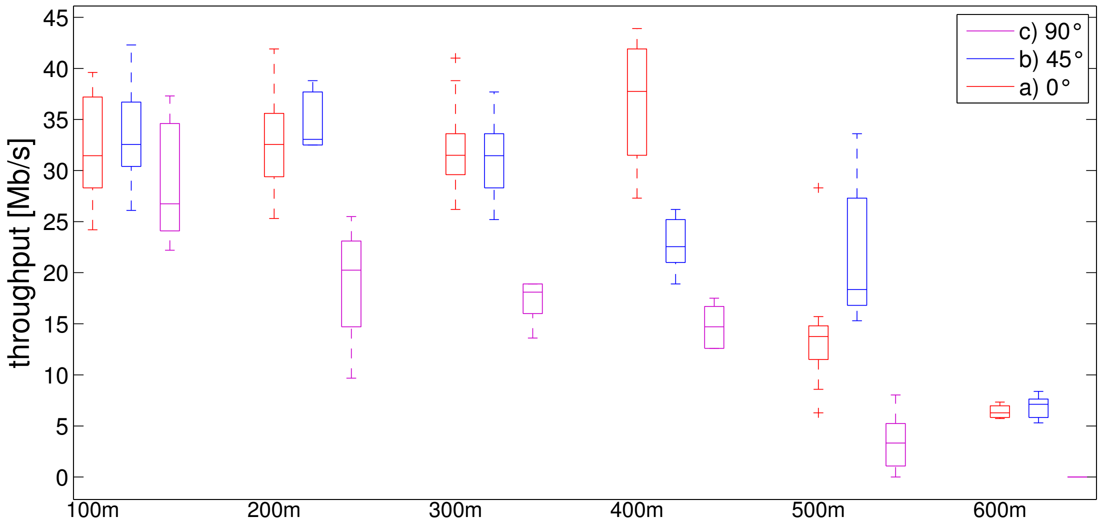


Figure 4.5: Throughput at different angles without usage of airMAX.

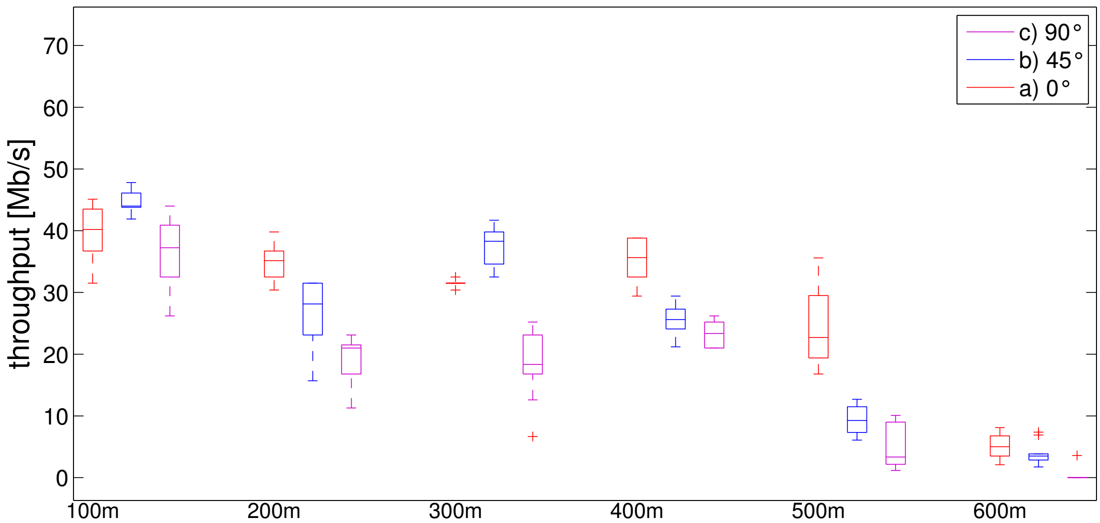


Figure 4.6: Throughput at different angles with usage of airMAX.

4.3.2 Signal Strength and Noise Floor

So as to have a measure of the connection quality on the link layer, the signal and noise levels were measured at each distance. As mentioned in Section 3.1, one should ensure a sufficient phase margin in order to have a reliable connection. In Figure 4.7 can be seen that while the level of the noise floor remains constant, the received signal strength decreases with increasing distance.

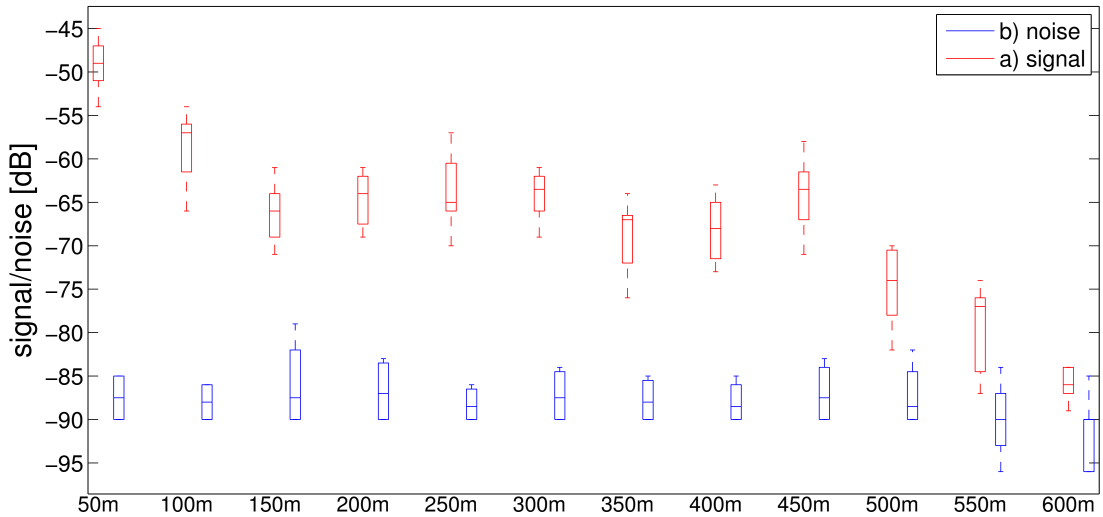


Figure 4.7: Signal & noise levels.

4.4 Range Evaluation with longer Pole

In Chapter 2.2, we described hoe the (maximal) radius of the first Fresnel zone increases with the distance between the two endpoints. Thus, as we cannot alter the height of the antenna mounted onto the car, only the pole of the stationary antenna can be changed to reduce obstruction in the first Fresnel zone by the ground.

In this test series, a pole of 3m height was used for measurements starting from a range of 300m up to 700m in steps of 100m. Additionally to the measurements using TCP, UDP measurements are taken. With this approach, we can compare both modes with respect to throughput. As UDP doesn't require ACK packets to be returned, it generally shows an advantage in throughput over wireless channels compared to TCP [9]. Furthermore, this allows for other measurements to be made, like the measurement of jitter, which has an impact on the received video quality. In order to make the UDP measurement, the UDP target bandwidth of iperf3 is fixed to 50Mb/s, such that the actual bandwidth is always lower. The test script for these measurements can be found in Subsection A.1.2.

4.4.1 Throughput

Here, Figure 4.8 shows the measured throughput rates using a TCP channel. Again, airMAX contributes to a higher available throughput. This is also the case for Figure 4.9, which shows the throughput when sending packet over UDP. Furthermore, the communication over UDP provides higher throughput rates than using a TCP channel.

On the other hand, the throughput is not higher than at comparable distances measured in Section 4.3. Nevertheless, we don't see a sharp drop in throughput like one can find in Figure 4.3 at a distance of 400m.

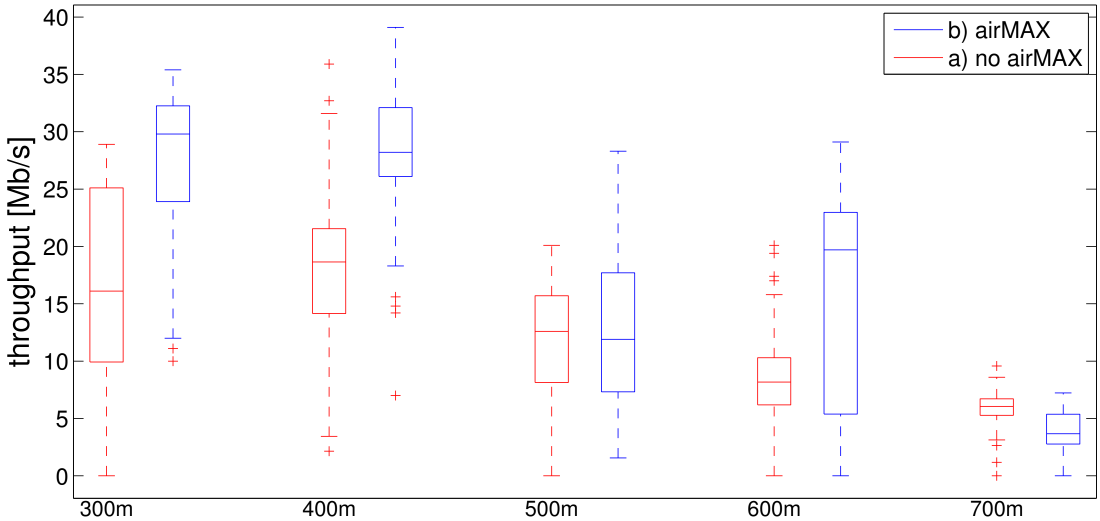


Figure 4.8: TCP throughput with and without airMAX enabled.

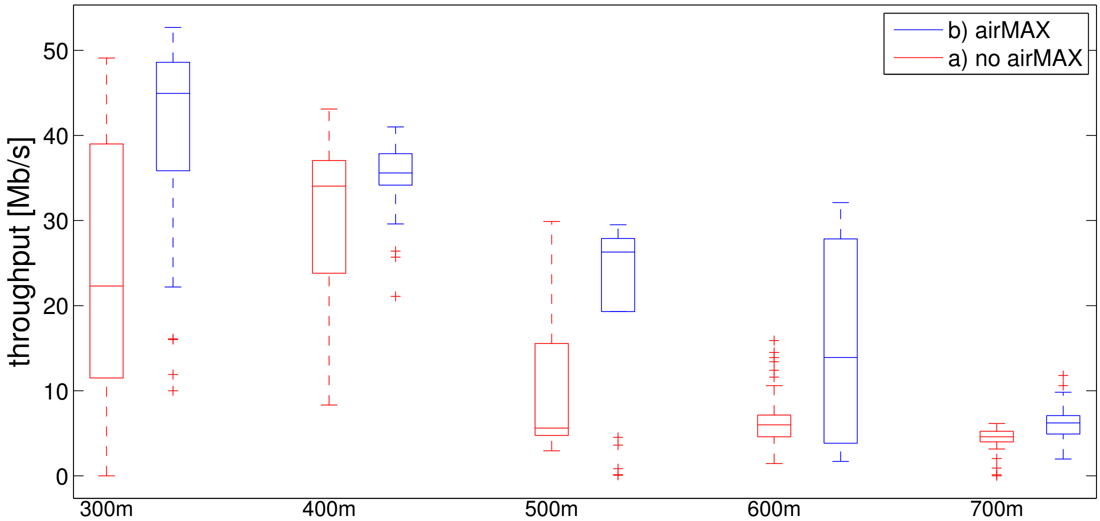


Figure 4.9: UDP throughput with and without airMAX enabled.

Testing with the actual Video Signal

Measurements of the streaming data rate of the video signal using Wireshark showed a required data rate between 600kb/s and 750kb/s when streaming a 432x240px video at 30fps, depending on the content of the transmitted images. Thus, the measured throughput rates are sufficient to provide enough bandwidth for both the video and the telemetry data. Transmitting the video over the wireless link showed the video in full quality at the other end, with only few interruptions from time to time.

4.4.2 Signal Strength and Noise Floor

Figure 4.10 shows the signal and noise power levels for this measurement. Again, the noise floor keeps a constant level. However, while in the measurement shown in Figure 4.7 the signal strength reduces quickly after a distance of 450m, the decrease in signal strength in Figure 4.10 is more moderate, and the phase margin between signal and noise levels is larger. We also observe that the variance in signal strength is higher at larger distances compared to measurements at closer distances.

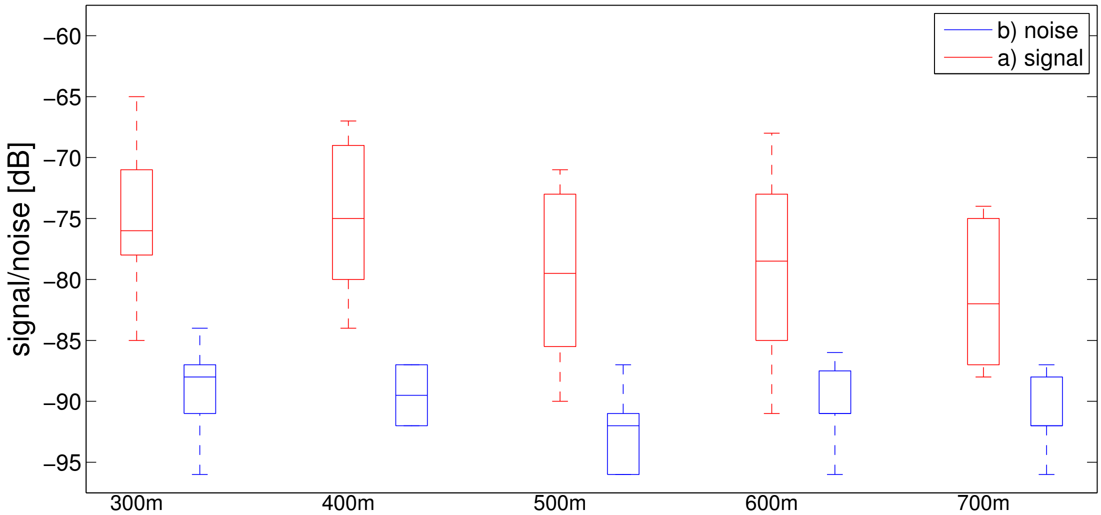


Figure 4.10: Signal & noise levels.

4.4.3 Jitter

As jitter reduces the quality of a streamed video by introducing interruptions in the image, jitter was measured using iperf's UDP mode, as described above. From Figure 4.11 we can see that jitter increases with range, meaning that more interruptions are introduced into the video signal at higher distances. This was also our observation when attaching the camera to the system. While the image quality was still acceptable, interruptions became more frequent.

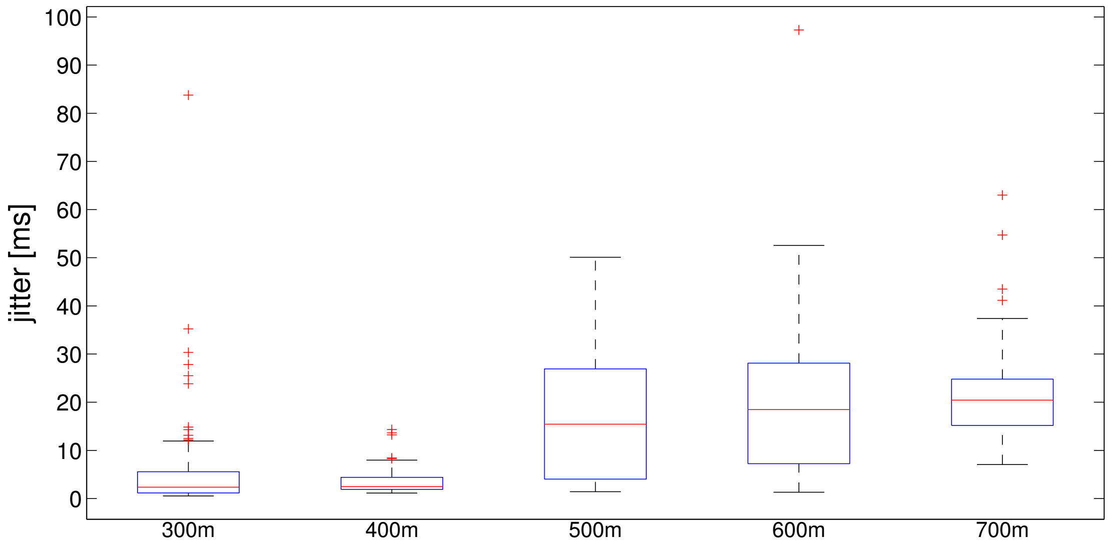


Figure 4.11: Jitter.

4.5 Impact of Movement

As mentioned in the introduction of this chapter, tests were conducted to measure the impact of movement when one of both ends of the link is in motion. For this, the omnidirectional antenna was mounted on a car and measurements for throughput, latency, jitter and packet loss were taken for both TCP and UDP connections. As in the previous measurements airMAX has shown to increase throughput, it was enabled in the setup.

Every test run was made with the car moving from the stationary antenna as well as moving towards the antenna. Measurements were taken from a starting velocity of 30km/h and subsequently increased by steps of 15km/h up to a velocity of 75km/h. In order to comply with local traffic law, no measurements at higher velocities were taken. As with increasing velocity, the distance covered in a given amount of time increases, the measurement duration was reduced for higher speeds accordingly in order to keep the results comparable. Also, the measurements were always started and terminated at the same distances from the antenna. The test script for these measurements can be found in Subsection A.1.3.

4.5.1 Throughput

In Figures 4.12 and 4.13 one can see the measured throughput values for both TCP and UDP transmissions. As already observed in the previous section, UDP connections offer higher throughput than TCP connections. Also, the measurements for TCP connections show a higher variance at higher velocities, while this effect is not clearly observable with UDP connections.

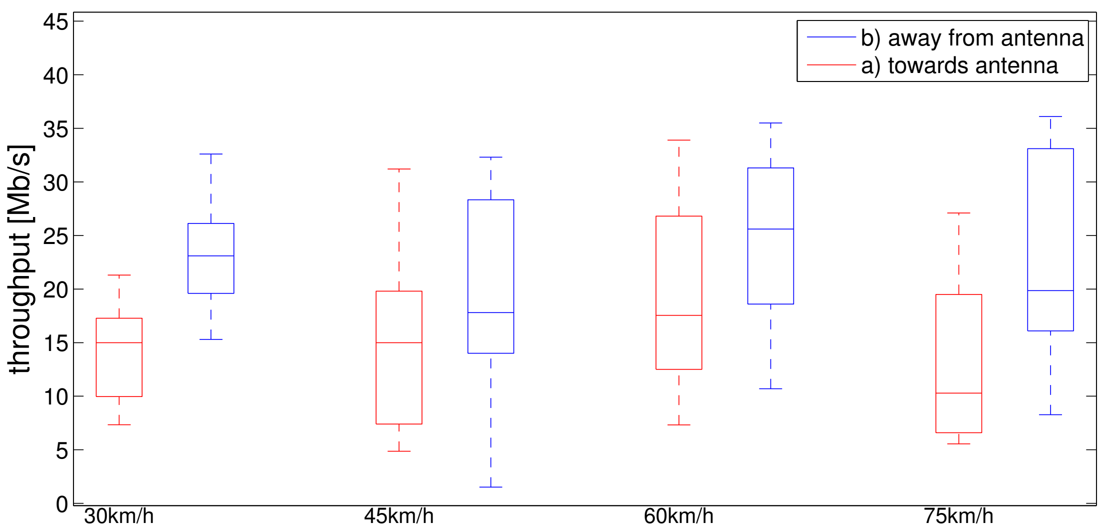


Figure 4.12: TCP throughput when moving towards and away from the antenna.

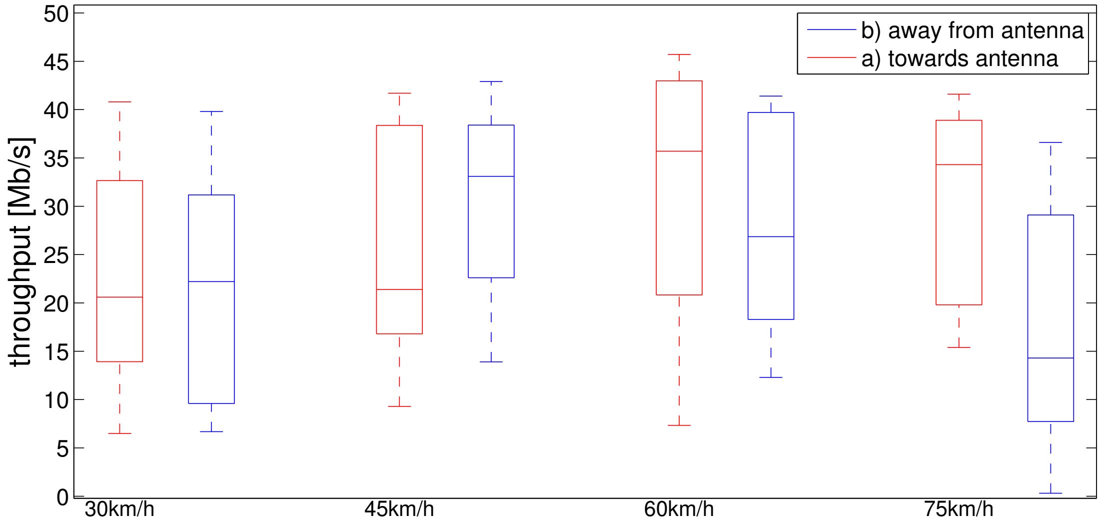


Figure 4.13: UDP throughput moving towards and away from the antenna.

4.5.2 Latency

As can be seen in Figure 4.14, latency is below 5ms for most of the cases, and below 10ms at almost all times. Also, it doesn't increase or decrease with higher velocities. However, from time to time there can be outliers with a duration of more than 10ms. For our application, this is acceptable.

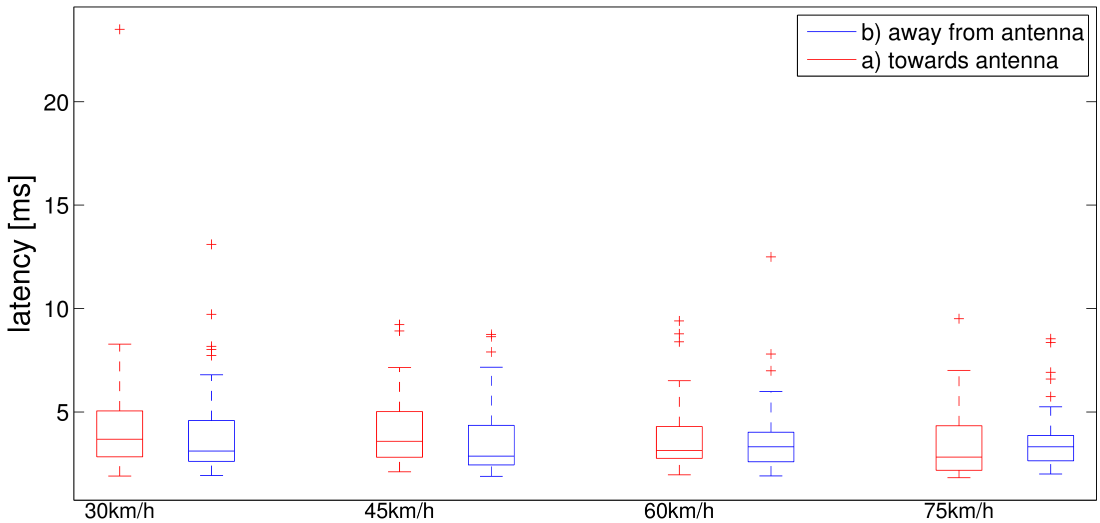


Figure 4.14: Latency when moving towards and away from the antenna.

4.5.3 Jitter

Figure 4.15 shows that jitter is below 10ms most of the time. Only at a velocity of 75km/h and moving away from the antenna, there is a larges amount of time spent on a transmission with longer jitter times. However, the measurement with a movement of the same speed but in opposite direction doesn't follow this pattern, so it cannot be explained with a higher speed alone. Moreover, at almost of the time, jitter is below 15ms, with only few outliers beyond this time.

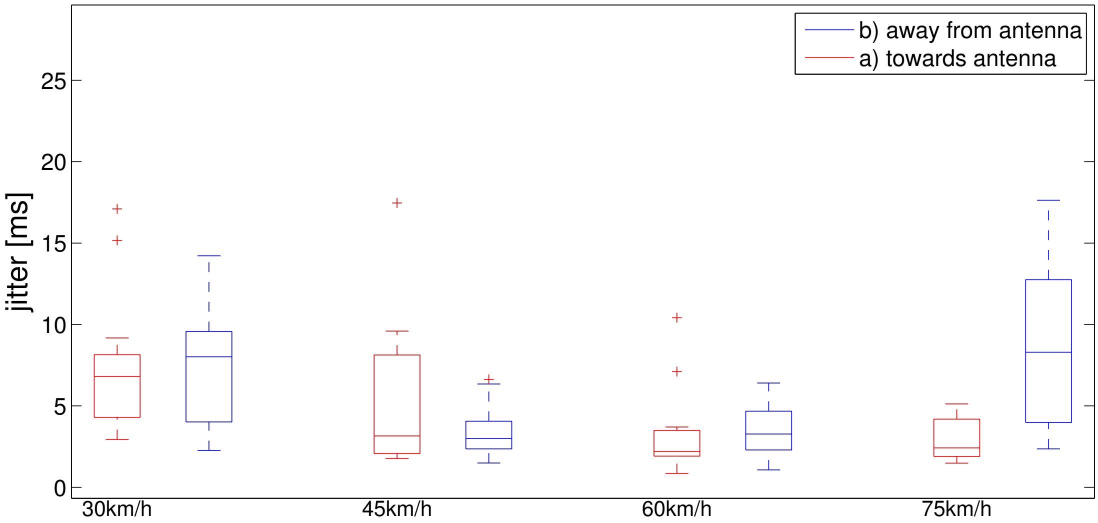


Figure 4.15: Jitter when moving towards and away from the antenna.

4.5.4 Packet Loss

When looking at Figure 4.16 one can see that packet loss keeps below values of 5% for most of the time. However, it slightly increases at higher speeds. Furthermore, some outliers (each single measurement point represents the packet loss measured during one second of the measurement) show high packet loss ratios. This suggests that from time to time the link cannot keep up it's reliability for a short duration. Nevertheless, the number of outliers is small and thus does not harm the overall quality of the link for a long time.

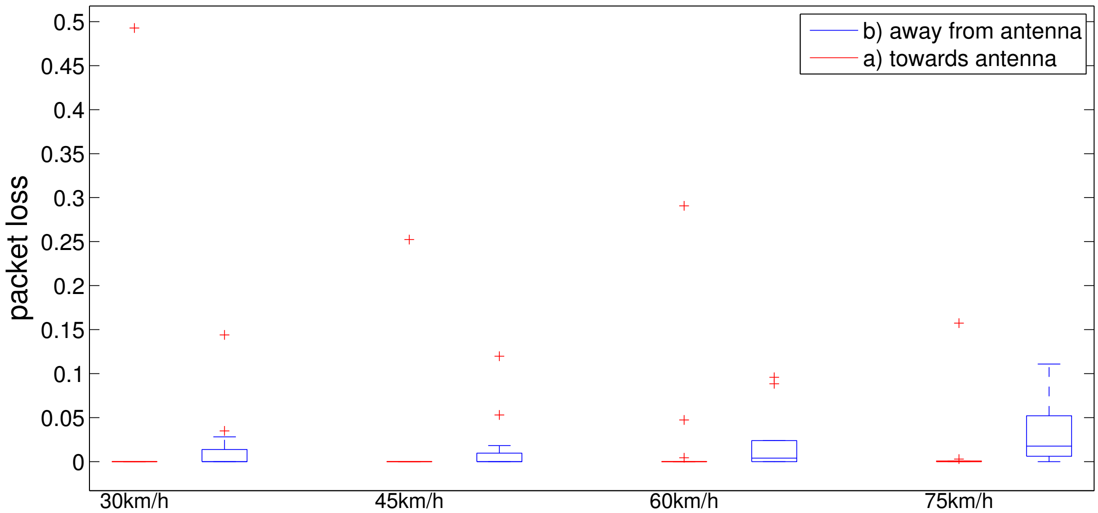


Figure 4.16: Packet loss when moving towards and away from the antenna.

Chapter 5

Conclusion & Future Work

5.1 Conclusion

In this thesis, we implemented a wireless telemetry system for the AMZ Formula Student race car. We first considered the theoretic foundations for the design of a wireless communication system and analysed the previous approach, which directly forwarded CAN packets to a laptop computer. With the results from this analysis, the theoretic background and considering the technical requirements we then designed a new system to handle the telemetry data as well as to provide sufficient bandwidth for a video signal, while keeping the additional weight in the car low.

In Chapter 1, the requirements for the wireless telemetry system were set. The telemetry system has to cover a distance of more than 600m and provide sufficient bandwidth for both the telemetry data and a video signal with a measured data rate of 750kb/s including the overhead of the used network protocols. In order to ensure our system meets the given requirements, we evaluated it under different settings and conditions.

Our design offered 5Mb/s over 600m when mounted to a 1.8m pole and the same possible throughput over a distance of 700m when mounted to a higher pole of 3m, fulfilling the range requirements set and providing sufficient bandwidth for both the telemetry and the video data. Furthermore, we showed that using the proprietary airMAX addition with TDMA yields better throughput rates than the standard CSMA/CA. Lastly, having one of the link's ends in motion didn't reduce the reliability of the link for speeds up to 75km/h.

Moreover, we evaluated different options for a client software to display the gathered data. We then extended this software, the Intecrio Experiment Environment, to meet the requirements of the AMZ team. Using the built-in extension mechanisms, we built widgets to allow for further insight into the dynamics of the race car and increasing the usability of the telemetry system.

5.2 Opportunities for Future Work

In general, there are several possible opportunities for future improvement. Below, we give an overview over possible areas of improvement.

Testing in the Actual Environment

This project was completed before the completion of the larger race car project of this year's competition season. As a result, the system could neither be tested nor deployed on the actual environment. Doing so when the car has been assembled will both give a conclusive answer to whether the system fits the requirements when deployed in the environment it was designed for as well as give hints for possible future improvements of the system.

Further Development of Widgets

Moreover, there are more ideas for further widgets in the Experiment Environment. For example, if in future cars of the team a GPS device would be included, one could imagine a map widget with live updates of the current position.

Increasing Platform independency

The chosen approach was designed to fit into the tool flow used by the members of the team programming the VCU. While this is in line with the goal of easy of use, it also means a dependency of the current VCU system. On the hardware side, the telemetry implementation is relying on the availability of an Ethernet port in the VCU, which, while being given for most of the cases, cannot be guaranteed for all choices of VCU systems.

On the software side, using the Experiment Environment's extension mechanisms means a dependency of the software parts of the telemetry system. This leaves room for improvement and adaptation.

Acknowledgements

My thank goes to Prof. Plattner and Mahdi for letting me do this project at the AMZ as well as for their support and feedback throughout the course of this project. Then, I would like to thank my family and especially my brother Yann and my father Dirk for their helping hands and their patience when conducting the manifold tests in the field. And finally, I'd like to thank the AMZ team for the amazing atmosphere and inspirational spirit, it was a great pleasure to work here.

Appendix A

Scripts

This appendix contains the test scripts for the evaluations of chapter 4. The titles are named after the respective sections in chapter 4.

A.1 Test Scripts

A.1.1 Range Evaluation

```
#!/bin/sh


# --------------------------------------------------------------------
# define stuff
# --------------------------------------------------------------------

grimselIP="169.254.202.125"
basestationIP="169.254.202.123"
VCUIP="169.254.202.130"
basestation="amz@$basestationIP"
grimsel="amz@$grimselIP"
id="~/.ssh/amz_rsa" # TODO: supply with -i option
log="log.txt"
config="/tmp/system.cfg"
bpath="/etc/persistent/"
airMAX="radio.1.polling=enabled"
noairMAX="radio.1.polling=disabled"

# getting link layer stats from airOS CLI
ShowParam="
for i in 1 2 3 4 5
do
  echo \"link layer test $i\"
  ip -s link show ath0;
  echo;
  iwconfig;
  sleep 2s;
done

ping $VCUIP -c 5

echo 'logging off from SSH session..';
exit"
```

```
function linkLayerMeasurement() {

  echo '
  reading out grimsel link layer parameters..
  ' >> $log

  ssh -i $id $grimsel "
  $ShowParam
  " >> $log

  echo '
  logged out from SSH


  reading out base station link layer parameters..
  ' >> $log

  ssh -i $id $basestation "
  $ShowParam
  " >> $log

  echo '
  logged out from SSH
  ' >> $log
}

# iperf is a transport layer connectivity testing tool
# https://code.google.com/p/iperf/
function iperf() {

  echo '
  starting iperf testing
  ' >> $log

  iperf3 -c $VCUIP -t 20 >> $log
  sleep 1s
  echo '-----------------------------------------------------------' >> $log
  iperf3 -c $VCUIP -u -t 20 >> $log

  echo '
  iperf test complete

  -----------------------------------------------------------' >> $log
}


# --------------------------------------------------------------------
# handle input options
# --------------------------------------------------------------------

# use the -t option to alter the transmit power of the Picostation
#
# use the -d option to specify the tested distance
# the only thing the -d option does it print the distance into the log
# and into the logfile's name
#
# use the -i option to copy the certificate on the PicoStations
while getopts ":it:d:" option
```

```
do
  case $option in

    i)
      # set up SSH certs, as airOS doesn't hold up to its promises regarding
      # persistent changes
      # found the trick: write changes directly into the into the system.cfg
      # namely (note: the "1" is just an example. keys are simply enumerated):
      # sshd.auth.key.1.comment=username@computername.local
      # sshd.auth.key.1.value=dertatsaechlichekey
      # sshd.auth.key.1.type=ssh-rsa
      # sshd.auth.key.1.status=enabled
      # TODO: implement accordingly

      cat ~/.ssh/amz_rsa.pub | ssh $grimsel "
      mkdir .ssh;
      cat >> /etc/persistent/.ssh/authorized_keys;
      chmod 700 -R .ssh;
      "

      cat ~/.ssh/amz_rsa.pub | ssh $basestation "
      mkdir .ssh;
      cat >> /etc/persistent/.ssh/authorized_keys;
      chmod 700 -R .ssh;
      "
      ;;
    t)
      case ${OPTARG} in
        -*) echo "Option -t requires an argument." >&2
            exit 1
            ;;
        *)
          echo "
----------------------------------------------------------------------------
!!!!!!!!!!!!!!!!!!! changing output power to  ${OPTARG} !!!!!!!!!!!!!!!!!!!!
----------------------------------------------------------------------------
" >> $log

          ssh -i $id $grimsel "
          iwconfig ath0 txpower ${OPTARG}
          " >> $log

          ssh -i $id $basestation "
          iwconfig ath0 txpower ${OPTARG}
          " >> $log
          ;;
        esac
      ;;
    d)
      case ${OPTARG} in
        -*) echo "Option -d requires an argument." >&2
          exit 1
          ;;
        *)
          distance=${OPTARG}
          log=log$distance.txt
          ;;
        esac
```

```
      ;;
    \?)
      echo "Invalid option: -$OPTARG" >&2
      exit 1
      ;;
    :)
      echo "Option -$OPTARG requires an argument." >&2
      exit 1
      ;;
  esac
done


# ---------------------------------------------------------------------
# the actual testing
# ---------------------------------------------------------------------

echo "
----------------------------------------------------------------------
starting test for $distance meters
----------------------------------------------------------------------

" >> $log

echo "
------------------------------------------------------------
Round 1: starting test using initial configuration
------------------------------------------------------------
" >> $log

iperf

linkLayerMeasurement

# rather do this manually than breaking anything irreversably..
# echo "
# ------------------------------------------------------------
# Round 3: testing using airMAX
# ------------------------------------------------------------
# " >> $log

# ssh -i $id $basestation "

# echo'
# altering configuration for airMAX
# ';

# if [ -f $config -a -r $config ]; then
#   cp -f $config $bpath;
#   sed -i "" \"s/$airMAX/$noairMAX/g\" \"$config\";
# else
#   echo \"Error: Cannot read $config\";
# fi

#   cfgmtd -w;
#   save;

# /usr/etc/rc.d/rc.softrestart save
```

```
# " >> $log

# iperf

# linkLayerMeasurement


echo "
----------------------------------------------------------------------
ending test for $distance meters
----------------------------------------------------------------------



" >> $log
```

A.1.2 Range Evaluation with 3m Pole

```
#!/bin/sh


# ---------------------------------------------------------------------
# define stuff
# ---------------------------------------------------------------------

grimselIP="169.254.202.125"
basestationIP="169.254.202.123"
VCUIP="169.254.202.130"
basestation="amz@$basestationIP"
grimsel="amz@$grimselIP"
id="~/.ssh/amz_rsa"
log="log.txt"
config="/tmp/system.cfg"
bpath="/etc/persistent/"
airMAX="radio.1.polling=enabled"
noairMAX="radio.1.polling=disabled"

# getting link layer stats from airOS CLI
ShowParam="
for i in 1 2 3 4 5
do
  echo \"link layer test $i\"
  ip -s link show ath0;
  echo;
  iwconfig;
  sleep 2s;
done

ping $VCUIP -c 5

echo 'logging off from SSH session..';
exit"

function linkLayerMeasurement() {

  echo '
  reading out grimsel link layer parameters..
  ' >> $log
```

```
  ssh -i $id $grimsel "
  $ShowParam
  " >> $log

  echo '
  logged out from SSH


  reading out base station link layer parameters..
  ' >> $log

  ssh -i $id $basestation "
  $ShowParam
  " >> $log

  echo '
  logged out from SSH
  ' >> $log
}

# iperf is a transport layer connectivity testing tool
# https://code.google.com/p/iperf/
function iperf() {

  echo '
  starting iperf testing
  ' >> $log

  iperf3 -c $VCUIP -R -t 30 >> $log
  sleep 1s
  echo '----------------------------------------------------------' >> $log
  iperf3 -c $VCUIP -u -R -b 50M -t 30 >> $log

  echo '
  iperf test complete

  ----------------------------------------------------------' >> $log
}


# -------------------------------------------------------------------
# handle input options
# -------------------------------------------------------------------

# use the -t option to alter the transmit power of the Picostation
#
# use the -d option to specify the tested distance
# the only thing the -d option does it print the distance into the log
# and into the logfile's name
#
# use the -i option to copy the certificate on the PicoStations
while getopts ":it:d:" option
do
  case $option in

    t)
      case ${OPTARG} in
```

```
        -*) echo "Option -t requires an argument." >&2
            exit 1
            ;;
        *)
          echo "
----------------------------------------------------------------------------
!!!!!!!!!!!!!!!!!!!! changing output power to  ${OPTARG} !!!!!!!!!!!!!!!!!!!!
----------------------------------------------------------------------------
" >> $log

          ssh -i $id $grimsel "
          iwconfig ath0 txpower ${OPTARG}
          " >> $log

          ssh -i $id $basestation "
          iwconfig ath0 txpower ${OPTARG}
          " >> $log
          ;;
        esac
      ;;
    d)
      case ${OPTARG} in
        -*) echo "Option -d requires an argument." >&2
          exit 1
          ;;
        *)
          distance=${OPTARG}
          log=log$distance.txt
          ;;
        esac
      ;;
    \?)
      echo "Invalid option: -$OPTARG" >&2
      exit 1
      ;;
    :)
      echo "Option -$OPTARG requires an argument." >&2
      exit 1
      ;;
  esac
done


# ---------------------------------------------------------------------
# the actual testing
# ---------------------------------------------------------------------

echo "
---------------------------------------------------------------------
starting test for $distance meters
---------------------------------------------------------------------

" >> $log

iperf

linkLayerMeasurement
```

```
echo "
---------------------------------------------------------------------
ending test for $distance meters
---------------------------------------------------------------------



" >> $log
```

A.1.3 Impact of Movement

TCP

```
#!/bin/sh


# -------------------------------------------------------------------
# define stuff
# -------------------------------------------------------------------

grimselIP="169.254.202.125"
basestationIP="169.254.202.123"
VCUIP="169.254.202.130"
basestation="amz@$basestationIP"
grimsel="amz@$grimselIP"
id="~/.ssh/amz_rsa" # TODO: supply with -i option
log="log.txt"
config="/tmp/system.cfg"
bpath="/etc/persistent/"
airMAX="radio.1.polling=enabled"
noairMAX="radio.1.polling=disabled"

# getting link layer stats from airOS CLI
ShowParam="
for i in 1 2 3 4 5
do
  echo \"link layer test $i\"
  ip -s link show ath0;
  echo;
  iwconfig;
  sleep 2s;
done

ping $VCUIP -c 5

echo 'logging off from SSH session..';
exit"

function linkLayerMeasurement() {

  echo '
  reading out grimsel link layer parameters..
  ' >> $log

  ssh -i $id $grimsel "
  $ShowParam
  " >> $log

  echo '
```

```
  logged out from SSH


  reading out base station link layer parameters..
  ' >> $log

  ssh -i $id $basestation "
  $ShowParam
  " >> $log

  echo '
  logged out from SSH
  ' >> $log
}

# iperf is a transport layer connectivity testing tool
# https://code.google.com/p/iperf/
function iperf() {

  echo '
  starting iperf testing
  ' >> $log

  iperf3 -c $VCUIP -R -t 10 >> $log
  #sleep 1s
  #echo '------------------------------------------------------------' >> $log
  #iperf3 -c $VCUIP -u -R -b 50M -t 15 >> $log

  echo '
  iperf test complete

  ------------------------------------------------------------' >> $log
}

# --------------------------------------------------------------------
# handle input options
# --------------------------------------------------------------------

# use the -t option to alter the transmit power of the Picostation
#
# use the -d option to specify the tested distance
# the only thing the -d option does it print the distance into the log
# and into the logfile's name
#
# use the -i option to copy the certificate on the PicoStations
while getopts ":it:d:" option
do
  case $option in

    i)
      # set up SSH certs, as airOS doesn't hold up to its promises regarding
      # persistent changes
      # found the trick: write changes directly into the into the system.cfg
      # namely (note: the "1" is just an example. keys are simply enumerated):
      # sshd.auth.key.1.comment=username@computername.local
      # sshd.auth.key.1.value=dertatsaechlichekey
      # sshd.auth.key.1.type=ssh-rsa
      # sshd.auth.key.1.status=enabled
```

```
      # TODO: implement accordingly

      cat ~/.ssh/amz_rsa.pub | ssh $grimsel "
      mkdir .ssh;
      cat >> /etc/persistent/.ssh/authorized_keys;
      chmod 700 -R .ssh;
      "

      cat ~/.ssh/amz_rsa.pub | ssh $basestation "
      mkdir .ssh;
      cat >> /etc/persistent/.ssh/authorized_keys;
      chmod 700 -R .ssh;
      "
      ;;
    t)
      case ${OPTARG} in
        -*) echo "Option -t requires an argument." >&2
            exit 1
            ;;
        *)
          echo "
-----------------------------------------------------------------------------
!!!!!!!!!!!!!!!!!!! changing output power to  ${OPTARG} !!!!!!!!!!!!!!!!!!!!
-----------------------------------------------------------------------------
" >> $log

          ssh -i $id $grimsel "
          iwconfig ath0 txpower ${OPTARG}
          " >> $log

          ssh -i $id $basestation "
          iwconfig ath0 txpower ${OPTARG}
          " >> $log
          ;;
        esac
      ;;
    d)
      case ${OPTARG} in
        -*) echo "Option -d requires an argument." >&2
          exit 1
          ;;
        *)
          distance=${OPTARG}
          log=log$distance.txt
          ;;
        esac
      ;;
    \?)
      echo "Invalid option: -$OPTARG" >&2
      exit 1
      ;;
    :)
      echo "Option -$OPTARG requires an argument." >&2
      exit 1
      ;;
  esac
done
```

```
# -------------------------------------------------------------------
# the actual testing
# -------------------------------------------------------------------

echo "
---------------------------------------------------------------------
starting test for $distance meters
---------------------------------------------------------------------

" >> $log

echo "
-----------------------------------------------------------
Round 1: starting test using initial configuration
-----------------------------------------------------------
" >> $log

echo "start"

iperf

echo "stop"

#linkLayerMeasurement

# rather do this manually than breaking anything irreversably..
# echo "
# -----------------------------------------------------------
# Round 3: testing using airMAX
# -----------------------------------------------------------
# " >> $log

# ssh -i $id $basestation "

# echo'
# altering configuration for airMAX
# ';

# if [ -f $config -a -r $config ]; then
#   cp -f $config $bpath;
#   sed -i "" \"s/$airMAX/$noairMAX/g\" \"$config\";
# else
#   echo \"Error: Cannot read $config\";
# fi

#   cfgmtd -w;
#   save;

# /usr/etc/rc.d/rc.softrestart save

# " >> $log

# iperf

# linkLayerMeasurement
```

```
echo "
-------------------------------------------------------------------
ending test for $distance meters
-------------------------------------------------------------------



" >> $log
```

UDP

```
#!/bin/sh


# ------------------------------------------------------------------
# define stuff
# ------------------------------------------------------------------

grimselIP="169.254.202.125"
basestationIP="169.254.202.123"
VCUIP="169.254.202.130"
basestation="amz@$basestationIP"
grimsel="amz@$grimselIP"
id="~/.ssh/amz_rsa" # TODO: supply with -i option
log="log.txt"
config="/tmp/system.cfg"
bpath="/etc/persistent/"
airMAX="radio.1.polling=enabled"
noairMAX="radio.1.polling=disabled"

# getting link layer stats from airOS CLI
ShowParam="
for i in 1 2 3 4 5
do
  echo \"link layer test $i\"
  ip -s link show ath0;
  echo;
  iwconfig;
  sleep 2s;
done

ping $VCUIP -c 5

echo 'logging off from SSH session..';
exit"

function linkLayerMeasurement() {

  echo '
  reading out grimsel link layer parameters..
  ' >> $log

  ssh -i $id $grimsel "
  $ShowParam
  " >> $log

  echo '
  logged out from SSH
```

```
  reading out base station link layer parameters..
  ' >> $log

  ssh -i $id $basestation "
  $ShowParam
  " >> $log

  echo '
  logged out from SSH
  ' >> $log
}

# iperf is a transport layer connectivity testing tool
# https://code.google.com/p/iperf/
function iperf() {

  echo '
  starting iperf testing
  ' >> $log

  #iperf3 -c $VCUIP -R -t 15 >> $log
  sleep 1s
  #echo '-----------------------------------------------------------' >> $log
  iperf3 -c $VCUIP -u -R -b 50M -t 15 >> $log

  echo '
  iperf test complete

  -----------------------------------------------------------' >> $log
}

# ------------------------------------------------------------------
# handle input options
# ------------------------------------------------------------------

# use the -t option to alter the transmit power of the Picostation
#
# use the -d option to specify the tested distance
# the only thing the -d option does it print the distance into the log
# and into the logfile's name
#
# use the -i option to copy the certificate on the PicoStations
while getopts ":it:d:" option
do
  case $option in

    i)
      # set up SSH certs, as airOS doesn't hold up to its promises regarding
      # persistent changes
      # found the trick: write changes directly into the into the system.cfg
      # namely (note: the "1" is just an example. keys are simply enumerated):
      # sshd.auth.key.1.comment=username@computername.local
      # sshd.auth.key.1.value=dertatsaechlichekey
      # sshd.auth.key.1.type=ssh-rsa
      # sshd.auth.key.1.status=enabled
      # TODO: implement accordingly
```

```
      cat ~/.ssh/amz_rsa.pub | ssh $grimsel "
      mkdir .ssh;
      cat >> /etc/persistent/.ssh/authorized_keys;
      chmod 700 -R .ssh;
      "

      cat ~/.ssh/amz_rsa.pub | ssh $basestation "
      mkdir .ssh;
      cat >> /etc/persistent/.ssh/authorized_keys;
      chmod 700 -R .ssh;
      "
      ;;
    t)
      case ${OPTARG} in
        -*) echo "Option -t requires an argument." >&2
            exit 1
            ;;
        *)
          echo "
----------------------------------------------------------------------------
!!!!!!!!!!!!!!!!!!! changing output power to  ${OPTARG} !!!!!!!!!!!!!!!!!!!
----------------------------------------------------------------------------
" >> $log

          ssh -i $id $grimsel "
          iwconfig ath0 txpower ${OPTARG}
          " >> $log

          ssh -i $id $basestation "
          iwconfig ath0 txpower ${OPTARG}
          " >> $log
          ;;
        esac
      ;;
    d)
      case ${OPTARG} in
        -*) echo "Option -d requires an argument." >&2
          exit 1
          ;;
        *)
          distance=${OPTARG}
          log=log$distance.txt
          ;;
        esac
      ;;
    \?)
      echo "Invalid option: -$OPTARG" >&2
      exit 1
      ;;
    :)
      echo "Option -$OPTARG requires an argument." >&2
      exit 1
      ;;
  esac
done


# -------------------------------------------------------------------
```

```
# the actual testing
# ---------------------------------------------------------------------

echo "
---------------------------------------------------------------------
starting test for $distance meters
---------------------------------------------------------------------

" >> $log

echo "
-----------------------------------------------------------
Round 1: starting test using initial configuration
-----------------------------------------------------------
" >> $log

echo "start"

iperf

echo "stop"

#linkLayerMeasurement

# rather do this manually than breaking anything irreversably..
# echo "
# -----------------------------------------------------------
# Round 3: testing using airMAX
# -----------------------------------------------------------
# " >> $log

# ssh -i $id $basestation "

# echo'
# altering configuration for airMAX
# ';

# if [ -f $config -a -r $config ]; then
#   cp -f $config $bpath;
#   sed -i "" \"s/$airMAX/$noairMAX/g\" \"$config\";
# else
#   echo \"Error: Cannot read $config\";
# fi

#   cfgmtd -w;
#   save;

# /usr/etc/rc.d/rc.softrestart save

# " >> $log

# iperf

# linkLayerMeasurement


echo "
---------------------------------------------------------------------
```

```
ending test for $distance meters
---------------------------------------------------------------------



" >> $log
```

Latency

```
#!/bin/sh


# -------------------------------------------------------------------
# define stuff
# -------------------------------------------------------------------

grimselIP="169.254.202.125"
basestationIP="169.254.202.123"
VCUIP="169.254.202.130"
basestation="amz@$basestationIP"
grimsel="amz@$grimselIP"
id="~/.ssh/amz_rsa" # TODO: supply with -i option
log="log.txt"
config="/tmp/system.cfg"
bpath="/etc/persistent/"
airMAX="radio.1.polling=enabled"
noairMAX="radio.1.polling=disabled"

# getting link layer stats from airOS CLI
ShowParam="
for i in 1 2 3 4 5
do
  echo \"link layer test $i\"
  ip -s link show ath0;
  echo;
  iwconfig;
  sleep 2s;
done

ping $VCUIP -c 5

echo 'logging off from SSH session..';
exit"

function linkLayerMeasurement() {

  echo '
  reading out grimsel link layer parameters..
  ' >> $log

  ssh -i $id $grimsel "
  $ShowParam
  " >> $log

  echo '
  logged out from SSH


  reading out base station link layer parameters..
```

```
  ' >> $log

  ssh -i $id $basestation "
  $ShowParam
  " >> $log

  echo '
  logged out from SSH
  ' >> $log
}

function latency() {
  ping $VCUIP -A -w 10

}

# iperf is a transport layer connectivity testing tool
# https://code.google.com/p/iperf/
function iperf() {

  echo '
  starting iperf testing
  ' >> $log

  #iperf3 -c $VCUIP -R -t 15 >> $log
  sleep 1s
  #echo '-----------------------------------------------------------' >> $log
  iperf3 -c $VCUIP -u -R -b 50M -t 15 >> $log

  echo '
  iperf test complete

  -----------------------------------------------------------' >> $log
}

# -------------------------------------------------------------------
# handle input options
# -------------------------------------------------------------------

# use the -t option to alter the transmit power of the Picostation
#
# use the -d option to specify the tested distance
# the only thing the -d option does it print the distance into the log
# and into the logfile's name
#
# use the -i option to copy the certificate on the PicoStations
while getopts ":it:d:" option
do
  case $option in

    i)
      # set up SSH certs, as airOS doesn't hold up to its promises regarding
      # persistent changes
      # found the trick: write changes directly into the into the system.cfg
      # namely (note: the "1" is just an example. keys are simply enumerated):
      # sshd.auth.key.1.comment=username@computername.local
      # sshd.auth.key.1.value=dertatsaechlichekey
      # sshd.auth.key.1.type=ssh-rsa
```

```
      # sshd.auth.key.1.status=enabled
      # TODO: implement accordingly

      cat ~/.ssh/amz_rsa.pub | ssh $grimsel "
      mkdir .ssh;
      cat >> /etc/persistent/.ssh/authorized_keys;
      chmod 700 -R .ssh;
      "

      cat ~/.ssh/amz_rsa.pub | ssh $basestation "
      mkdir .ssh;
      cat >> /etc/persistent/.ssh/authorized_keys;
      chmod 700 -R .ssh;
      "
      ;;
    t)
      case ${OPTARG} in
        -*) echo "Option -t requires an argument." >&2
            exit 1
            ;;
        *)
          echo "
----------------------------------------------------------------------------
!!!!!!!!!!!!!!!!!!! changing output power to  ${OPTARG} !!!!!!!!!!!!!!!!!!!
----------------------------------------------------------------------------
" >> $log

          ssh -i $id $grimsel "
          iwconfig ath0 txpower ${OPTARG}
          " >> $log

          ssh -i $id $basestation "
          iwconfig ath0 txpower ${OPTARG}
          " >> $log
          ;;
        esac
      ;;
    d)
      case ${OPTARG} in
        -*) echo "Option -d requires an argument." >&2
          exit 1
          ;;
        *)
          distance=${OPTARG}
          log=log$distance.txt
          ;;
        esac
      ;;
    \?)
      echo "Invalid option: -$OPTARG" >&2
      exit 1
      ;;
    :)
      echo "Option -$OPTARG requires an argument." >&2
      exit 1
      ;;
  esac
done
```

```
# ---------------------------------------------------------------------
# the actual testing
# ---------------------------------------------------------------------

echo "
----------------------------------------------------------------------
starting test for $distance meters
----------------------------------------------------------------------

" >> $log

echo "
------------------------------------------------------------
Round 1: starting test using initial configuration
------------------------------------------------------------
" >> $log

echo "start"

latency >> $log

echo "stop"

#linkLayerMeasurement

# rather do this manually than breaking anything irreversably..
# echo "
# ------------------------------------------------------------
# Round 3: testing using airMAX
# ------------------------------------------------------------
# " >> $log

# ssh -i $id $basestation "

# echo'
# altering configuration for airMAX
# ';

# if [ -f $config -a -r $config ]; then
#   cp -f $config $bpath;
#   sed -i "" \"s/$airMAX/$noairMAX/g\" \"$config\";
# else
#   echo \"Error: Cannot read $config\";
# fi

#   cfgmtd -w;
#   save;

# /usr/etc/rc.d/rc.softrestart save

# " >> $log

# iperf

# linkLayerMeasurement
```

```
echo "
----------------------------------------------------------------------
ending test for $distance meters
----------------------------------------------------------------------



" >> $log
```

Appendix B

Datasheets

This appendix contains the data sheets from both last year's as well as this year's telemetry hardware. The following two pages contain the datasheet for the module of last year's car, the Avisaro WLAN Cube, while the third page contains the datasheet for this year's hardware, the Ubiquity PicoStation M2 HP. Both modules are described in more detail in chapter 3.

Avisaro WLAN Cube 2.0 (SD) W23766

Mit CAN-Schnittstelle

Der WLAN Cube verbindet eine CAN-Schnittstelle mit einem WLAN Netzwerk. Die Daten können von dieser direkt am PC eingelesen und weiter verarbeitet werden. Das Device eignet sich auch als WLAN Relais-Station zur Überbrückung längerer Reichweiten. Der Cube besteht aus einem wetter- und staubfestem Gehäuse.

- Gepufferte Echtzeituhr für zeitgesteuerte Befehle
- Scriptprogrammierung zur individuellen Anpassung
- Senden und Empfangen von Daten
- Wendung als Relais zur Überbrückung von längeren Reichweiten

Funktionsweise: "Script-Programmierung"

Die Script Programmierung eignet sich um den Logger an individuelle Anwendungen anzupassen, wie zum Beispiel die Ringspeicherung, Zeitstempel etc. Viele fertige Scripts sind verfügbar und müssen lediglich geladen werden. Als Standard ist die direkte Informationsweitergabe eingestellt (WC1). Eigene Scripts können in der BASIC ähnlichen Sprache realisiert werden. Das Device kann so z.B. selbständig Sensoren abfragen, die Daten aufarbeiten und weitergeben.

Konfiguration

Die Konfiguration des Device erfolgt über die Web-Administratoren-Seite, über die Angaben zur Schnittstelle, wie Baudrate, Verhalten bei Start, etc. gemacht werden. Alle Konfigurationen bleiben auch bei Stromausfall erhalten.

Um die Seite zu öffnen, gehen Sie wie folgt vor:
1. Schalten Sie das Gerät ein und verbinden Sie Ihren Computer mit dem WLAN.
SSID: Avisaro
(Kanal: 11,Modus: Ad-hoc (ohne Access Point), Verschlüsselung: keine)
Stellen Sie sicher, dass Ihr Computer nicht mit einem anderen WLAN oder LAN Netzwerk verbunden ist. Avisaro Cube verfügt über einen DHCP-Server und wird Ihr Computer automatisch eine IP-Adresse zuweisen. Wenn Sie eine Adresse manuell vergeben, stellen Sie sicher, dass sie in dem gleichen Adressbereich liegt.
2. Öffnen Sie alle Browser und rufen Sie die IP-Adresse 192.168.0.74. In ausgewählten Fällen wie eine WLAN-Brücke mit zwei Geräten finden Sie möglicherweise unterschiedliche IP-Adressen auf dem Lieferschein.
3. Melden Sie sich mit dem Benutzernamen: 'Admin' und dem anfänglichen Kennwort: '1234'. Aus Sicherheitsgründen sollten Sie das Kennwort ggfls ändern. Achtung: Achten Sie darauf, von dem neuen Login-Parameter. Verlorene Passwörter können nicht ersetzt werden und die Box muss an Avisaro gesendet werden.

WLAN

802.11 b/g,
WPA + WEP
Reichweite-outdoor: max 250 m
Reichweite-indoor: 30-50 m

CAN Schnittstelle

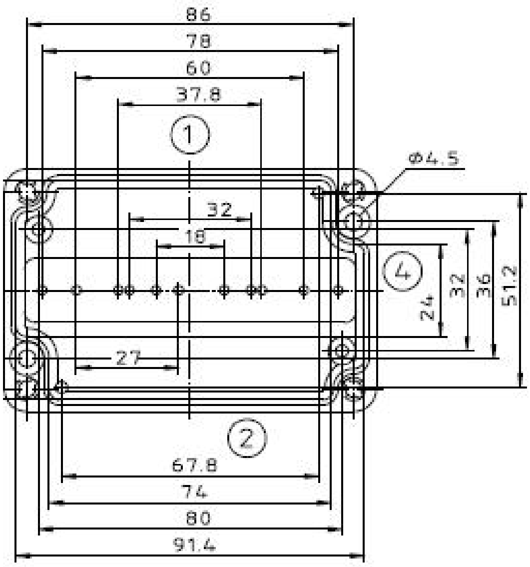
eine CAN-Schnittstelle
Nachrichtenformat 2.0A und 2.0B
"Listen Only" modus möglich
Baudrate bis 1 Mbit/s, Abschlusswiderstand schaltbar

Elektrische Eigenschaften

Versorgungsspannung: 6V - 32V, Verbrauch: ~0.9 W, Verpolungschutz, Power Save möglich. Standzeit der Echtzeituhr: 1 Monat. Ladezeit: 48h

Mechanische Eigenschaften

Abmessungen: 98 x 64 x 34 mm
Gewicht: 166 g. Schutzklasse: IP66 (wetterfest).
Temperatur: -30°C - 85°C



Anschlusstyp: Anschlussklemme CAN

Das Anschlusskabel wird durch die wetterfeste Öffnung ins Gehäuse geführt. Dort werden die Kabelenden mit Schraubklemmen angeschlossen.

1. Signal Ground (GND)
2. Internal use (*)
3. not connected
4. CAN-L (Low)
5. CAN-H (High)
6. not connected
7. Supply Voltage (6-32V)
8. Supply Ground (GND)

(*) This pin provides 5V power supply to power sensors placed inside the Cube

DIP Switches
1. CAN Terminating Resistor
2. not connected
3. not connected

Bestellnummern

W23766: Logger Cube mit CAN Schnittstelle

Aus der Produktserie W:
W21766: Logger Cube mit RS232 Schnittstelle
W22766: Logger Cube mit RS485 Schnittstelle

Produktserie M: Datenlogger ohne WLAN-Funktionalität
Produktserie C: Datenlogger mit WLAN-Funktionalität

Kontakt

Avisaro AG
Grosser Kolonnenweg 18 /D1
30163 Hannover, Germany
Tel.: +49 (0)511 780 93 90
Fax,: +49 (0)511 353 196 24
E-Mail: info@avisaro.com Web: www.avisaro.com

Specifications

System Information	
Processor Specs	Atheros MIPS 24KC, 400 MHz
Memory	32 MB SDRAM, 8 MB Flash
Networking Interface	(1) 10/100 Ethernet Port

Regulatory/Compliance Information	
Wireless Approvals	FCC Part 15.247, IC RS210, CE
RoHS Compliance	Yes

Physical/Electrical/Environmental	
Dimensions	136 x 20 x 39 mm
Weight	0.1 kg
Enclosure Characteristics	Outdoor UV Stabilized Plastic
Mounting	Wall or Pole Mounting Kit (Included)
Antenna Connector	External RP-SMA
Antenna USA EU	 (1) External, 5 dBi Omni Antenna (Included) (1) External, 2 dBi Omni Antenna (Included)
Operating Frequency	2412-2462 MHz
Range Indoor Outdoor	 Up to 200 m Up to 500 m
Max. Power Consumption	8 W
Power Supply (PoE)	15V, 0.8A Power Adapter (Included)
Power Method	Passive Power over Ethernet (Pairs 4, 5+; 7, 8 Return)
Operating Temperature	-20 to 70° C
Operating Humidity	5 to 95% Condensing
Shock & Vibration	ETSI300-019-1.4

Output Power: 28 dBm

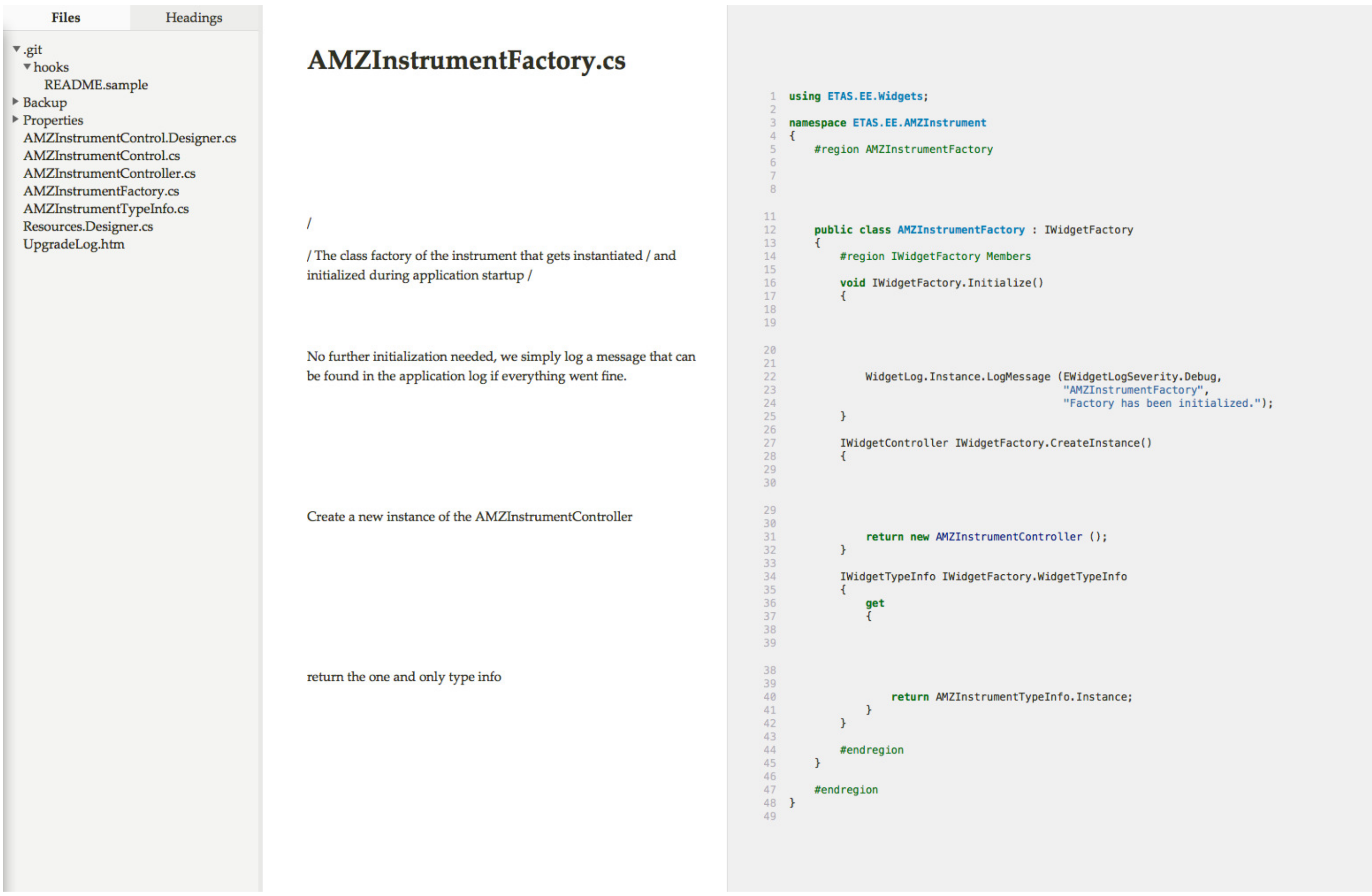
2.4 GHz TX POWER SPECIFICATIONS				2.4 GHz RX POWER SPECIFICATIONS			
	Data Rate/MCS	Avg. TX	Tolerance		Data Rate/MCS	Sensitivity	Tolerance
11b/g	1-24 Mbps	28 dBm	± 2 dB	11b/g	1-24 Mbps	-97 dBm	± 2 dB
	36 Mbps	27 dBm	± 2 dB		36 Mbps	-80 dBm	± 2 dB
	48 Mbps	26 dBm	± 2 dB		48 Mbps	-77 dBm	± 2 dB
	54 Mbps	24 dBm	± 2 dB		54 Mbps	-75 dBm	± 2 dB
11n/airMAX	MCS0	28 dBm	± 2 dB	airMAX	MCS0	-96 dBm	± 2 dB
	MCS1	28 dBm	± 2 dB		MCS1	-95 dBm	± 2 dB
	MCS2	28 dBm	± 2 dB		MCS2	-92 dBm	± 2 dB
	MCS3	28 dBm	± 2 dB		MCS3	-90 dBm	± 2 dB
	MCS4	27 dBm	± 2 dB		MCS4	-86 dBm	± 2 dB
	MCS5	25 dBm	± 2 dB		MCS5	-83 dBm	± 2 dB
	MCS6	24 dBm	± 2 dB		MCS6	-77 dBm	± 2 dB
	MCS7	23 dBm	± 2 dB		MCS7	-74 dBm	± 2 dB

All specifications in this document are subject to change without notice.

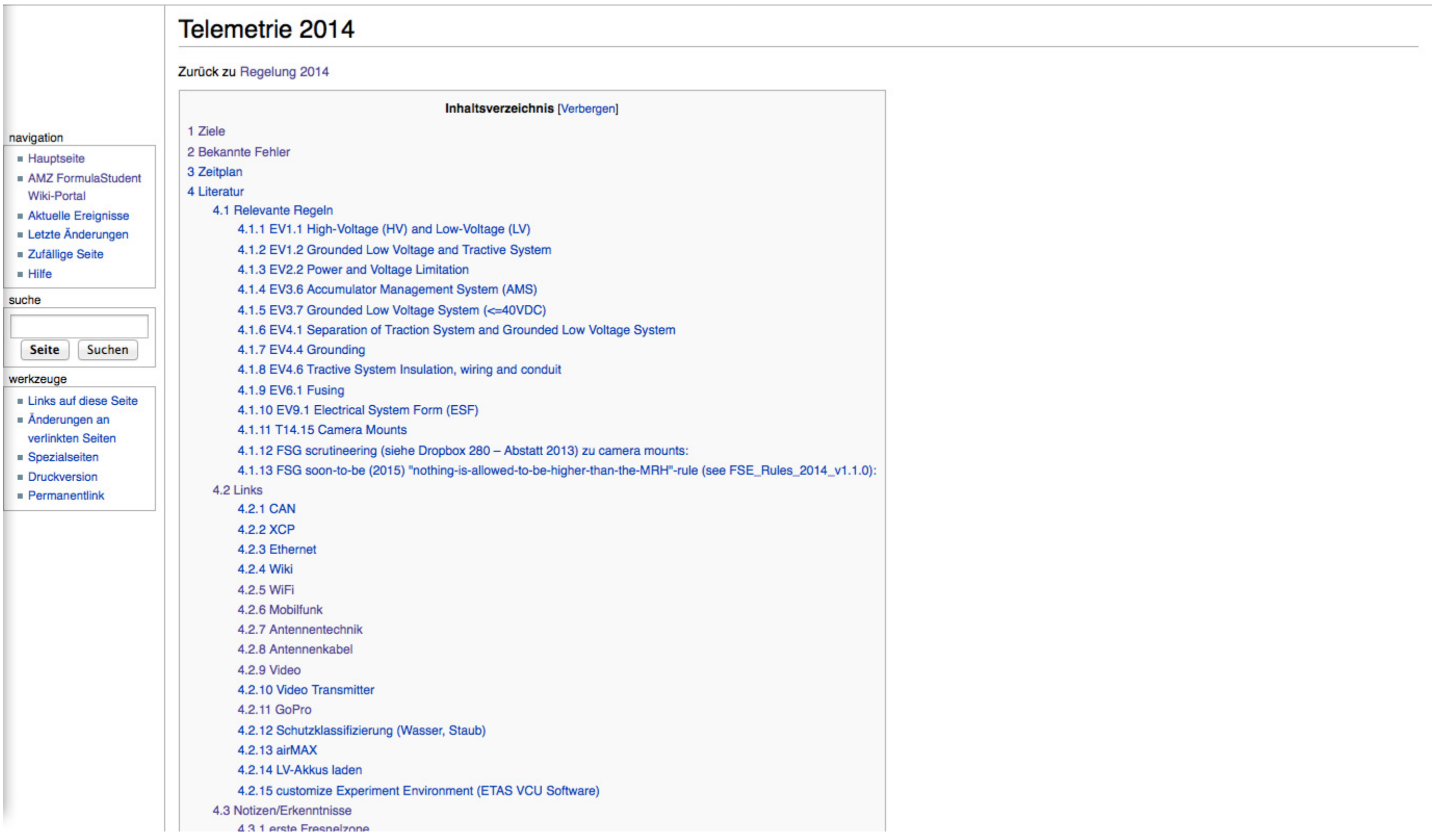
Appendix C

Example Documentation

C.1 docker.js



C.2 AMZ Wiki



navigation
- Hauptseite
- AMZ FormulaStudent Wiki-Portal
- Aktuelle Ereignisse
- Letzte Änderungen
- Zufällige Seite
- Hilfe

suche

Seite Suchen

werkzeuge
- Links auf diese Seite
- Änderungen an verlinkten Seiten
- Spezialseiten
- Druckversion
- Permanentlink

Telemetrie 2014

Zurück zu Regelung 2014

Inhaltsverzeichnis [Verbergen]

1 Ziele
2 Bekannte Fehler
3 Zeitplan
4 Literatur
4.1 Relevante Regeln
4.1.1 EV1.1 High-Voltage (HV) and Low-Voltage (LV)
4.1.2 EV1.2 Grounded Low Voltage and Tractive System
4.1.3 EV2.2 Power and Voltage Limitation
4.1.4 EV3.6 Accumulator Management System (AMS)
4.1.5 EV3.7 Grounded Low Voltage System (<=40VDC)
4.1.6 EV4.1 Separation of Traction System and Grounded Low Voltage System
4.1.7 EV4.4 Grounding
4.1.8 EV4.6 Tractive System Insulation, wiring and conduit
4.1.9 EV6.1 Fusing
4.1.10 EV9.1 Electrical System Form (ESF)
4.1.11 T14.15 Camera Mounts
4.1.12 FSG scrutineering (siehe Dropbox 280 – Abstatt 2013) zu camera mounts:
4.1.13 FSG soon-to-be (2015) "nothing-is-allowed-to-be-higher-than-the-MRH"-rule (see FSE_Rules_2014_v1.1.0):
4.2 Links
4.2.1 CAN
4.2.2 XCP
4.2.3 Ethernet
4.2.4 Wiki
4.2.5 WiFi
4.2.6 Mobilfunk
4.2.7 Antennentechnik
4.2.8 Antennenkabel
4.2.9 Video
4.2.10 Video Transmitter
4.2.11 GoPro
4.2.12 Schutzklassifizierung (Wasser, Staub)
4.2.13 airMAX
4.2.14 LV-Akkus laden
4.2.15 customize Experiment Environment (ETAS VCU Software)
4.3 Notizen/Erkenntnisse
4.3.1 erste Fresnelzone

Bibliography

[1] Data Acquisition System for a 600cc Formula SAE Race Car,
Khan, S., Sonti, S., IEEE International Conference on Vehicular Electronics and Safety (ICVES), 2009,
Retrieved from *http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5400316*

[2] Formula SAE Racecar Online-Monitoring System Based on ZigBee Wireless Technology,
Ma, X. T.,Wang H.,Shi L., Measurement & Control Technology, CNKI, 2009,
Retrieved from *http://en.cnki.com.cn/Article_en/CJFDTotal-IKJS200908008.htm*

[3] Automotive data acquisition system – FST (Master's thesis),
David R. Copeto, Technical University of Lisbon, Portugal, October 2009,
Retrieved from *https://fenix.tecnico.ulisboa.pt/downloadFile/2589867774288/*

[4] Telemetrie im Formula Student Rennwagen auf Basis von CAN Bus, Datenspeicherung und Wireless LAN Technologien (Bachelor thesis),
Sebastian Haase, Hochschule für Angewandte Wissenschaften Hamburg, Germany, 23.09.2007,
Retrieved from *http://www.haw-hamburg.de/fileadmin/user_upload/TI-I/Bilder/Projekte/Faust/Arbeiten/2007/2007Ba_-_sebastian_haase_Telemetrie_im_Formula_Student_Rennwagen_auf_Basis_von_CAN_Bus__Datenspeicherung_und_Wireless_LAN_Technologien.pdf*

[5] Formula SAE Telemetry Collection Unit,
Hadaller, D., Li, H., Sung, A., University of Waterloo, CA, US, 19.12.2004,
Retrieved from *http://www.flamenet.ca/uw/courses/cs856_f04/proj/sae-report11.pdf*

[6] The Complete Design of an Engine Management System for RIT's Formula SAE Team,
Mancini, B., Greene, B., Fiederlein, E., Doelger, G., Tung, K., Bloom, A.
Multi-Disciplinary Engineering Design Conference, Rochester Institute of Technology, Rochester, New York, 2007
Retrieved from *http://edge.rit.edu/edge/P07222/public/P07222ECUConferencePaper_Rev1.03.doc*

[7] Network optimization for wireless microwave backhaul (PHD thesis),
Napoleão Nepomuceno, Université de Nice, Sophia Antipolis, France, Universidade Federal do Ceará, 17.12.2010,
Retrieved from *http://www-sop.inria.fr/members/Napoleao-Vieira.Nepomuceno/thesis.pdf*

[8] Microwave Transmission Networks: Planning, Design and Deployment, 2nd Edition
Harvey Lehpamer, 2010, New York, McGraw Hill

[9] TCP and UDP Performance over a Wireless LAN ,
Xylomenos, G, Polyzos, G. C.
INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, (pp. 439 - 446 vol.2), 1999
Retrieved from *http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=751376*

[10] A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi,
Lee, J. S., Su, Y. W., Shen, C. C.
3rd Annual Conference of the IEEE Industrial Electronics Society (IECON), (pp. 46 - 51),

2007
Retrieved from *http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4460126&tag=1*

[11] IEEE Standards Association, IEEE 802.15™: WIRELESS PERSONAL AREA NETWORKS (PANs)
http://standards.ieee.org/about/get/802/802.15.html
20.02.2014

[12] Avisaro WLAN Cube
http://www.avisaro.com/tl/wlan-cube-20.html
29.01.2014

[13] Ubiquity PicoStation M2 HP
http://www.ubnt.com/airmax#picostationm
29.01.2014

[14] 2014 Formula SAE® Rules
http://students.sae.org/cds/formulaseries/rules/2014_fsae_rules.pdf,
21.01.2014

[15] 2014 Formula Student Electric Rules 2014 version 1.0
http://www.formulastudent.de/uploads/media/FSE_Rules_2014_v1.0.pdf,
10.02.2014

[16] ASAM, XCP Specification
http://www.asam.net/nc/home/standards/standard-detail.html?tx_rbwbmasamstandards_pi1[showUid]=519,
20.02.2014

[17] CCP Specification
http://automotivetechis.files.wordpress.com/2012/06/ccp211.pdf,
20.02.2014

[18] Vector CANape
http://vector.com/vi_canape_en.html?markierung=canape,
20.02.2014

[19] MSDN, Windows Forms
msdn.microsoft.com/en-us/library/dd30h2yb(v=vs.110).aspx,
11.02.2014

[20] AMZ wiki
http://wiki.amzracing.ch,
20.03.2014

[21] docker.js
http://jbt.github.io/docker/src/docker.js.html,
20.02.2014

[22] MSDN, PropertyGrid Class
http://msdn.microsoft.com/en-us/library/system.windows.forms.propertygrid%28v=vs.110%29.aspx,
18.02.2014

[23] MSDN, Getting the Most Out of the .NET Framework PropertyGrid Control
http://msdn.microsoft.com/en-us/library/aa302326.aspx,
18.02.2014

[24] MSDN, GDI+
http://msdn.microsoft.com/en-us/library/windows/desktop/ms533798%28v=vs.85%29.aspx,
18.02.2014

[25] MSDN, WebBrowser Class
http://msdn.microsoft.com/en-us/library/system.windows.forms.webbrowser%28v=vs.110%29.aspx,
18.02.2014