Designing a wireless communication system for smart sensor shorts in football

Using lossless data compression and pattern diversity

D.B.J. Burgers

Designing a wireless communication system for smart sensor shorts in football

Using lossless data compression and pattern diversity

by

D.B.J. Burgers

to obtain the degree of Master of Science Embedded Systems
at the Delft University of Technology,
to be defended publicly on Tuesday January 26, 2021 at 10:00 AM.

Student number:	4455347	
Project duration:	February 1, 2020 – January 26, 2021	
Thesis committee:	dr. ir. A. Bossche,	TU Delft, supervisor
	dr. ir. G.J.M. Janssen,	TU Delft
	dr. ir. A. van Genderen	TU Delft
	prof. dr. P.J. French	TU Delft
	Ing. J. Bastemeijer	TU Delft, technical support
	MSc A.S.M Steijlen	TU Delft, daily supervisor

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Abstract

Decreasing injuries in football, is a topic of interest for the KNVB and KNHB. To reach this goal, the use of smart sensor pants is researched. The pants will use five Inertial Measurement Units (IMUs), to measure the angular velocity (dps), linear acceleration (g) and magnetic field (μT), at the lower body of the football player. This data can then be used to develop models for finding injury risk factors that are related to movements. The smart sensor pants will first be used for research purposes, and later on for football players in matches and training sessions. This means that the electronics integrated into the pants, should not be obstructive to football players, and hence be small. On the other hand, the device should be capable of measuring the data and sending it reliable to the sideline. Currently, the system is capable of reading out the IMUs and storing the data on an SD-card, for post-analysis of the data.
As next step, the communication link should be implemented, to make it useful for real time measurement to provide feedback to the football players. The design of an efficient communication system in terms of power, size and reliability, is quite a big task. By looking at similar research to body worn devices, it is noted, that most devices deal with a 10 times lower data rate then the current device. On top of that, it is noted that the absorption of the body is causing problems on the link reliability. Based on these observations, it is decided to focus the research of this Thesis on the reduction of the data load, and the optimization of the transmission part of the smart sensor pants. The Thesis is split in two parts, the first part will show the use of lossless data compression to decrease the amount of data to transmit. The second part will start with showing the benefit of using a patch antenna over a dipole antenna, and continues by showing the benefit of using a dual antenna configuration over a single antenna configuration.
To be more specific, the first part of this Thesis, starts by comparing different lossless data compression algorithms, from which the FELACS algorithm is chosen as most suited. The Golomb-Rice coding of the algorithm is modified, and therefore a modified FELACS (MFELACS) algorithm is used. This algorithm is then implemented on the current hardware, and tested on data from the smart sensor pants in a realistic football scenario. These results show an average compression ratio of 43 %-45 % in the most intensive 5-minutes of a football game, with a minimum of 38 % in an interval of 10 s. To improve the compression algorithm, an adjustment to the MFELACS algorithm is proposed. This adjustment is theoretically tested and shown to outperform the MFELACS algorithm with a higher compression ratio.
In the second part, the use of a dual antenna configuration is discussed, whereby the use of a patch antenna is compared to a dipole antenna. It will be shown, that a dual antenna configuration can significantly improve the signal strength around the player, resulting in an almost isotropic radiation pattern, using pattern diversity. On top of that, this form of pattern diversity is observed to increase the reliability of the link, using switched combining. Moreover, it will be shown that a patch antenna will be more suited for this application, due to the higher gain in the front, and the robustness against interference when placed close to a conducting material.
In summary, the two main contributions of this Thesis, are the reductions in data load, and the testing and verification of the dual patch antenna configuration. These contributions provide the basis for the communication part of the smart sensor pants.

Preface

I would like to thank my supervisors Andre, Annemarijn and Jeroen. Even though we were not able to work on the 15^{th} floor anymore. We were still able to continue the weekly meetings, which were really helpful in discussing results and guiding the project. I really enjoyed working together during this project, and really gained a lot of knowledge on this topic. I would like to also separately thank Jeroen, for providing first aid after spilling a cup of Tea.
Next, I would like to thank everyone who has supported me in providing the means for conducting the experiments. Beginning with the v.v. 's-Gravendeel, which provided me players and a location to perform the experiments for the data compression part. Continued, by my uncle and aunt, who allowed me to do measurements at their farm for the antenna experiments, while providing me with electricity and a canteen to warm up.
Thirdly, I would like to thank my friends for giving support during my Thesis. Even though most contact is nowadays online, it is still nice to sometimes meet outside for a walk. Especially, Fenno for being one of my participants during the antenna measurements. Even though it was very cold standing still for a long time, he did not complain. Also, Dennis helped me a lot by sharing his experience doing his Thesis. Finally, Maurice gave up multiple parts of the day to help me out with multiple shorter experiments.
Fourthly, I would like to thank my girlfriend Marieke. She always supported me during my Thesis and helped me to relieve my stress. She also gave up an entire day in her Christmas holiday to help me with the experiments. And she convinced me in taking a week of in the summer to go camping for my first time, in Belgium and France. Which I really enjoyed and can look back to as a special memory. I would also like to thank her family for their support during my Thesis.
Fifthly, I would like to thank my Family for supporting me. Even though we see each other less, due to COVID-19. They always have time, when I need some help.
Last, but certainly not least, I would like to thank my mother, sister and brothers. Because I had to work a lot at home, I claimed the office downstairs to work, and made a small mess of it with all my stuff, without getting a lot of complaints. They also supported me in keeping a positive mindset, and accompanied me during my breaks, for a walk or lunch.

"The difference between theory and practice is in theory somewhat smaller than in practice."
— *Frank Westphal*

D.B.J. Burgers
Delft, January 2021

Contents

1 Introduction

Football is known for its high rate of muscle injuries. Part of these injuries are overuse injuries, which are the result of large physical demands of the player. Under the Swedish elite football team, 37 % of all injuries were over use injuries. In a more broad study on injury characterisation under 23 of the 50 best European teams, the most occurring injuries were at the lower extremities. [110]. The contribution of injuries in general is investigated in [33] and finds that a player has 2.0 injuries per season on average with most injuries occurring at 17 % at the subtype thigh strain. This means that a team consisting of 25 players has an average of 50 injuries per season. The prevention of these injuries can prevent undesired effects. Starting with the economic savings in rehabilitation costs, going up to €500.000 per month for a professional first team football player in [32]. To continue, in the European cups in male professional football and in the league play, a significant influence on performance is observed due to injuries [42]. Finally, an effect can be seen on the mental well being of an injured football player [7].
The cause of most injuries in football is most likely located at the lower extremes [96]. By measuring the lower extremes, models can be developed for finding risk factors in movements related to injuries. Therefore the KNVB (Koninklijke Nederlandse Voetbal Bond) and KNHB (Koninklijke Nederlandse Hockey Bond) have set up a project for measuring movements at the lower body of football and hockey players. In the beginning of the project, the focus will be on developing the measurement system for football players. Later on, when the measurement system is finished, it will also be used by the KNHB. In the measurement system multiple Inertial Measurement Units (IMUs) are placed at the lower body for measuring the: angular velocity (dps), linear acceleration (g) and magnetic field (μT). Once the data are measured, the movement scientists from the KNVB, doing their PhD at the VU Amsterdam or the University of Groningen, can use this to develop their biomechanical model for finding injury risk factors that are related to movements. The role of the TU Delft is to provide a functional end product, which will be called the ”sensor shorts”. During this process, multiple prototypes are made, which will be used by the movement scientist, for developing their models, these prototypes are referred to as the ”sensor pants”.
The use of IMUs for developing models is widely studied in literature. This research is mainly focusing on the following topics: estimating of biomechanical loads [16, 106]; biomechanics in kicking [20]; and a model estimating the joint kinematics [80, 100], muscle strains and muscle elongation velocities [109]. The overview given, is just a fraction of all analysis performed on human movements, based on IMUs. Moreover, there are also studies using a video reference tracking system, which can be used when all movements can be performed in a lab environment [49, 71]. This system is already available at the KNVB and can be used as a reference to verify the results from IMUs.

The aim of the KNVB and KNHB is to design and build a sensor shorts, which is capable of preventing injuries. In order to do this a measurement system should be available and capable of measuring the movements of the lower body, without obstructing the player. The data from the system should be available real time as input for the model described earlier. The model will be used to give feedback to the player with the advice to either stop or slow down the sport activity. The aim of the project is to use this system during training and in matches.

During the early stages of the project, the use of commercial measurement systems from companies such as Xsens [114] , Vicon [108] and myoMOTION [73] were considered. However, shortcomings such as the size of the device leading to discomfort in wearing and the insufficient measurement range, for high accelerations (> 16 g) and angular velocities (> 2000 dps), have led to partner up with the TU Delft for developing a measurement system meeting these requirements. This partnership also offers the benefits of having access to the raw sensor data (often filtering is applied in commercial systems) and the ability to optimize the design to this specific application.

1.1. Current prototype

The design of the sensor shorts is under the supervision of Annemarijn Steijlen as part of her PhD. In the project, every year a new prototype is developed with extra functionality. In the end, two types of pants will be delivered. One for research and one for commercial usage. During development, the first prototypes are only used for research purposes. Later these requirements can be updated for the final commercial version. This Thesis will focus on the pants used for research. Later on a commercial sensor shorts will be designed, based on the one developed for research.
At the time of writing, the second prototype is already developed. The second prototype in Figure 1.2b and 1.2a is a long thermal pants with a total of five IMUs. With two IMUs located at the upper and lower part of both legs and one at the trunk. To continue, the IMUs are each a combination of a 3-axis magnetometer, gyroscope and accelerometer. With the gyroscope and accelerometer forming one sensor from the company invense. The sensor is called the ICM-20649 with a range of ±4000 dps and ±30 g. The magnetometer is from AsahiKASEI and called the AK8963 with a range of ±4900 µT. The AK8963 is connected to the ICM-20649 using the I2C protocol. Through this coupling the sensor combination works as one sensor with 9 degrees of freedom (DOF) and can be read out using one SPI connection as can be seen in Figure 1.1.

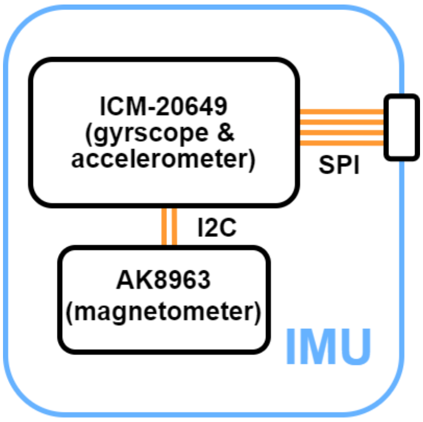
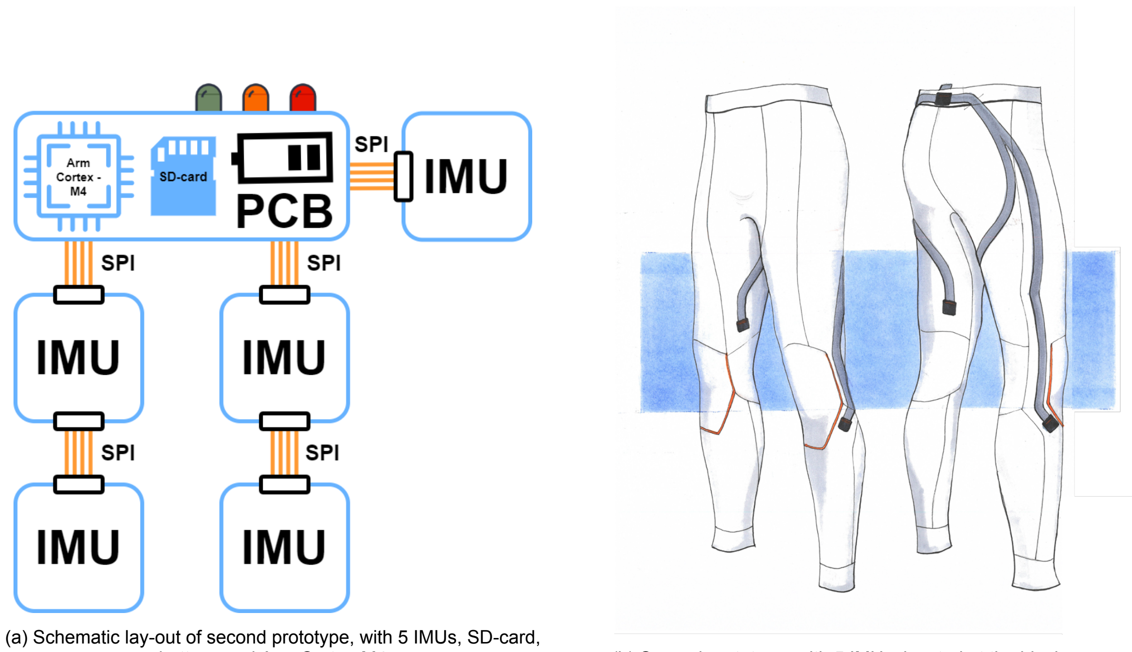


Figure 1.1: IMU internal lay-out



(a) Schematic lay-out of second prototype, with 5 IMUs, SD-card, battery, and Arm Cortex-M4

(b) Second prototype, with 5 IMUs, located at the black squares

Figure 1.2

The five IMU modules are all connected to the main PCB located at the trunk, see Figure 1.2a. The trunk part is selected, because it will introduce the least amount of obstruction when wearing and will most likely cause the least amount of damage in case of falling. The PCB will be responsible for handling power; reading out sensors and storing data on the SD-card. The microcontroller on the PCB is based on the Arm Cortex-M4 core from the STM32F411 Nucleo-64 development board, with the following relevant specifications:

- running at 100 MHz max (adjustable)
- 512 kB of FLASH
- 128 kB of SRAM
- 125 DMIPS (or 1.25 DMIPS / MHz)

A commercial 1350 mA h battery from XQISIT is placed on top of the main PCB. The battery is selected, based on the high capacity and safety certifications. To continue, the real-time operating system FreeRTOS [36] is used. A real-time operating system uses tasks as tool to ensure the response of a system within a specified time constraint. The functionality of a system can be divided into tasks meeting the system requirements. A tasks consists of an activation-, start-, computation- and finishing time. The current prototype is divided in the following tasks:

- readAccGyroTask: this task will read out the accelerometer and gyroscope for all 5 IMUs and passes the values to the storeBuffTask.
- readMagTask: this task will read out the magnetometer from the 5 IMUs and passes the values to the storeBuffTask.
- storeBuffTask: this task will store the IMU readings from the accelerometer, gyroscope, magnetometer into a ping-pong buffer. Each buffer has a size of 4 kB.
- storeSDTask: this task will write the ping or pong buffer when filled to the SD-card.
- initTask: this task will create a new file to be written by the storeSDTask. It will also handle the input from the buttons and provides feedback using the LEDs.

The activation of a task, in the second prototype, can occur at a fixed frequency (periodic task) or when an event occurs (aperiodic task). When a task is activated, a job is created, which will execute based up on the chosen scheduling scheme. Currently a preemptive highest priority scheduling scheme is used. The use of preemption means that a task with a lower priority can be interrupted by a task with a higher priority. The task with the lower priority will continue when the task with the higher priority is finished. This means that the highest priority will always be executed upon activation. If preemption would not be used, the highest priority task would have to wait for a lower priority task started in the past. If this would be used on the second prototype, the storeSDTask, with an execution time of 100 ms, would block the execution of other tasks (such as the readAccGyroTask or readMagTask), during this period of execution. The full implementation of the scheduling scheme can be seen in Figure 1.3, with the black arrows indicating the activation of a task (i.e. job) and the colored boxes show the execution of the task. A few remarks before ending the description of the second prototype. It can be observed that the readMagTask is executed after the storeBuffTask, this will results in a delay of 4 ms but is not of a great concern, due to relative slow variation in signals from the magnetometers and approval from the movement scientist of the KNVB after discussing it. Also, the main concern is to achieve a fixed interval between the readings of the gyroscope and accelerometer. Therefore, it is desirable to store the data from the gyroscope and accelerometer before reading out the magnetometer. Furthermore, it is desired to preempt the readMagTask if it might execute to long.

1.2. Next prototypes

Currently the prototype does not have the ability to send data real time. In order to use the sensor pants as measurement device in providing real time feedback, a wireless communication system should be integrated. The use of wireless communication for sensor devices in sport applications is studied by

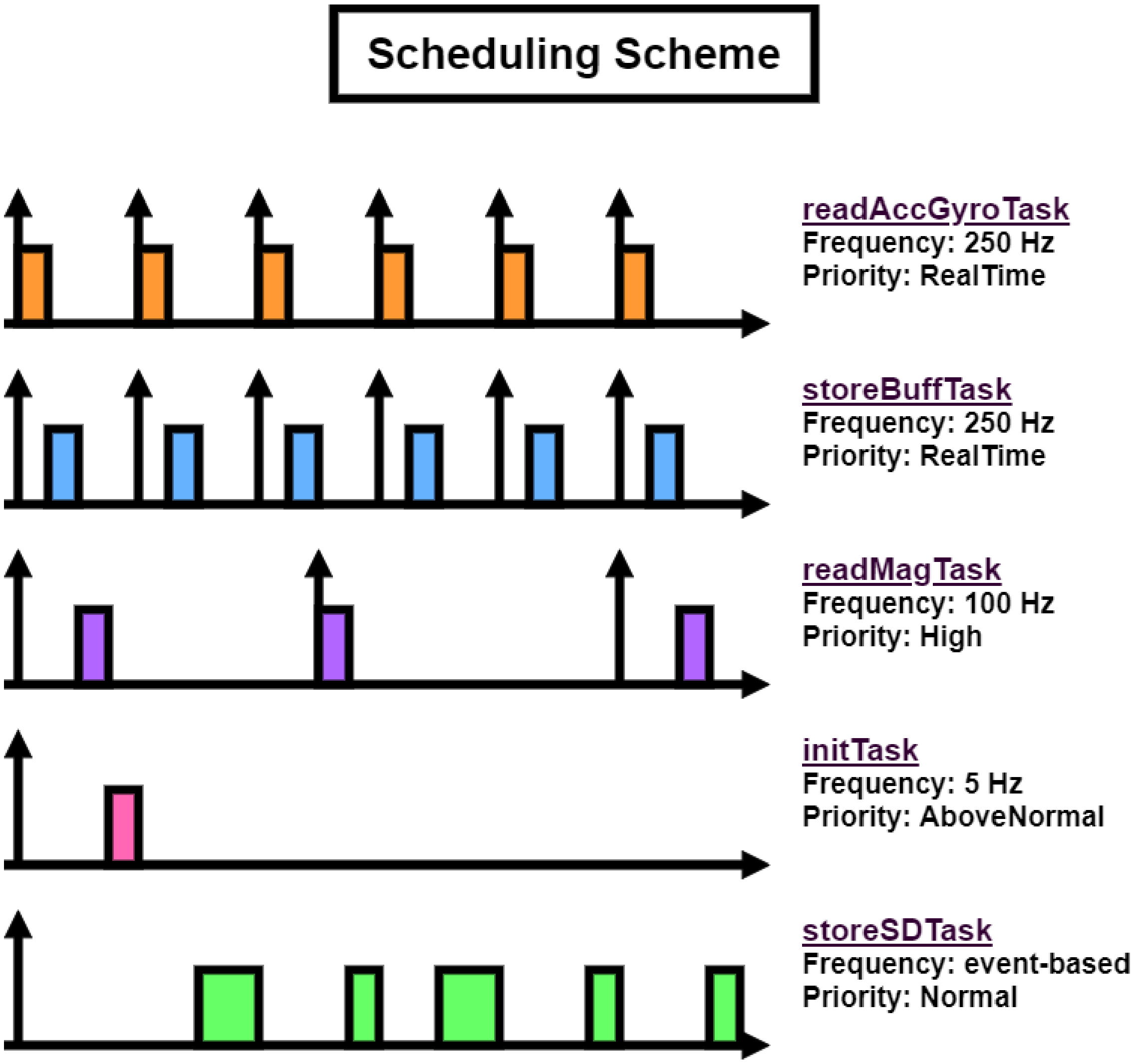


Figure 1.3: Preemptive highest priority scheduling scheme on second prototype

several researchers. A brief summary on the proposed solutions and results will be given, continued by the proposed research questions to partially answer the main research question. Namely, how to get an efficient wireless communication system for the sensor shorts.

1.2.1. Research on wireless communication for sport biofeedback systems

The use of wireless communication systems in wearable sport devices for biofeedback applications often faces design challenges regarding: bit rate, range, reliability and energy. While keeping the device small, preventing any disturbance in sport activity [37]. The demand for a certain bit rate, reliability and transmission range, provide the boundary conditions under which the energy consumption can be optimized. The current prototype uses 5 IMUs, with the gyroscope and accelerometer being sampled at 250 Hz and the magnetometer at 100 Hz. This will lead to an overall load of 144 kbit/s as can be seen in equation 1.1. Whereby the total amount of bits per sample (16 bits total) is multiplied by the amount of IMUs (5 in total), and finally the contributions of each DOF in samples is taken, with 250 samples for the gyroscope and accelerometer (6 DOF in total) and 100 for the magnetometer (3 DOF).

$$\text{data rate} = 16 \times 5 \times (6 \times 250 + 3 \times 100) = 144\,\text{kbit/s} \tag{1.1}$$

When two teams of 11 players would wear the device a rate of 3.168 Mbit/s would be required and potentially more, in case of players on the bench.

In literature the following relevant studies were found and focused on designing and/or implementing communication systems for football applications.

Sivaraman et al. [92] experiments with an athlete wearing a ZigBee-module (MPR2400CA) at 2.4 GHz with eight receiving antennas evenly spaced at the sideline. The aim of the experiment is first to investigate the network topology by periodically broadcasting a message containing a unique identifier and sequence number. Based on the received signals, the network topology can be reconstructed at every time instance. It turns out that the connectivity between players changes second-to-second and the connectivity with the receivers at the side-line is generally poor. In a follow up research in [30], the use of a multi-hop routing protocol is proposed. First the network is flooded for measuring the minimum delay from a player to a receiver at the side-line, as a reference. Next, five routing schemes are implemented and tested, to find the delay and total transmissions required. The tested multi-hop protocols are: 1) single-hop routing, 2) off-line optimal routing, 3) random forwarding, 4) two-copy routing scheme and 5) tunable flooding scheme. The protocols are based on two types of network topologies, the start- and mesh topology. The research concludes that a direct transmissions result in unacceptably high delays. Therefore a tunable flooding-based multi-hop solution is proposed as desired solution. [30]

A similar approach is taken in monitoring a university football club (winners of the japan university football tournament). First an analysis is performed on the radiation pattern of the antenna placed on the waist of the players, followed by placing 6 antennas evenly spaced at the side of the field. Each device sent data over the 920 MHz and 2.4 GHz band simultaneously, with a packet size of 30 B every 10 s. The data is sent using a CSMA (collision sense multiple access) pseudo TDMA (time division multiple access) method, whereby each module uses a unique fixed back-off time. The antennas placed at the sideline, are placed at the different height to examine this effect. It turns out that using the 920 MHz band can reach a packet success rate up to 1, with less then 6 antennas. However, in some cases a packet success rate of < 1 is observed with 6 antennas. When using the 2.4 GHz band, a packet success rate of < 0.8 is observed. [43] In a follow up experiment [44] the effect of data forwarding between the 6 antennas to the main antenna is investigated. The hypothesis is that packet success rate staying below 1, might be caused by packet losses between the side-line antennas, when sending all the data to the base-station. Unfortunately, this turned out not to be the case. Another observation made during the experiment is that defenders and midfielders are having a higher packet loss.

In a later research [45], the focus is put on monitoring and analyzing vital signs of children sporting. Introducing a network with high densities, and low and high dynamics. In order to find a suited wireless data protocol, a football game is used as test case. During testing a device is placed on the back waist of sixteen football players. During the match a 1-hop, 2-hop and 3-hop algorithm is tested, based on the AOVD and CSMA using the 920 MHz band. With on the sideline, eight Data Forwarding Nodes (DFNs) and one base-station. It turns out that using the 3-hop algorithm with eight DFNs, no packet error is observed. However, when the 3-hop algorithm is used without DFNs, the packet error rate increases to 14 %. The trade-off in having multiple hops is the increased network load going up from 6 % up to 30 %.

Another research [88], tries to create a wireless sensor network on a football field, with four receiving antennas evenly spaced at the side. A TDMA algorithm is tested, using a time compact algorithm for handling multiple packets, leading to an increase in the data rate. The measured range of the device is at maximum 60 m using the ZigBee protocol. The system is capable of having multiple players sending data from one IMU at 200 Hz (around 13 kB/s), with a packet loss of 1 % using 5 nodes. The system can handle players joining and leaving the network in game.

In [62] the effect of the body on the radiation pattern is investigated. The experiment is conducted using a receiver mounted around the bicep of the upper arm. The radiation pattern is then measured around the body. It turns out that the radiation in free space follows an inverse square law, while the propagation in the body is drastically reducing the signal strength. This is also observed, when the sender is placed on the belly of a player in [43].

The research discussed above, is mainly focused on developing a communication system based

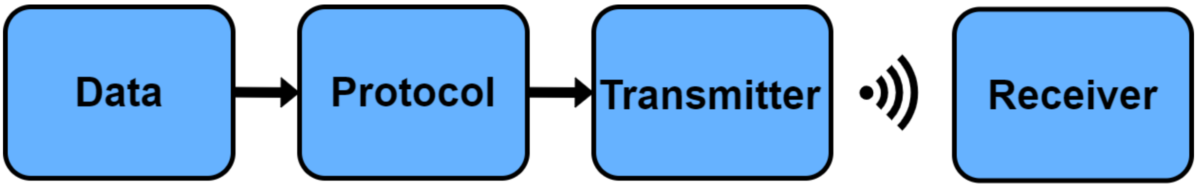


Figure 1.4: Simplified block diagram of different elements in a communication system

on the ZigBee protocol (based on the IEEE 802.15.4 standard). However, the maximum data rate of ZigBee is only 250 kbit/s, while the requirement for the next prototype is 3.168 Mbit/s. It can directly be observed, that the assumed data load in those studies are not meeting the requirements for this project by a factor 10 - 20. It can be noticed, that most research will focus on implementing a multi-hop routing protocol, with the focus on a football match scenario. However, the sensor pants should be usable stand-alone in a match and training scenario. This does not mean that the research is irrelevant, a lot of useful observations can still be taken from the proposed solutions.

To continue, a recent study called "Challenges in wireless communication for connected sensors and wearable devices used in sport biofeedback applications" [60], shows the best suitable protocol for different applications. First a classification is given for different applications. The classification will consider the: functionality (user, instructor or cloud); physical extent of the device (personal, confined or open); type of structure of the sensor deployment (compact or distributed); time constraint (terminal feedback or concurrent feedback) and finally the processing power available. To fit the the sensor pants in this model, the sensor pants will be classified as an *instructor confined distributed terminal* feedback system. To clarify, the data should be send to an *instructor* at the side-line. To continue, the space of operation is known and hence *confined*, with the sensors placed all over the body (*distributed*) and finally the data should be available within a few seconds and thus *terminal*. The paper continues by providing a list of different sensors with estimated data rates. Finally a list of different wireless protocols is provided, listing the maxim bit rate, range and transmission power of each protocol. By setting the required bit rate, range and transmission power for an application one could chose the desired protocol. Based on this table, the IEEE 802.11n, IEEE 802.11af or IEEE 802.11ah could be selected. [60]

To conclude, based on the experimental observations listed above, the following can be noted. The effect of the human body leads to a significant disturbance in the radiation pattern. Due to weak connectivity, the use of a multi-hop routing protocol is proposed. Based on requirements, the currently high data rates, may lead to difficulties in implementing a wireless communication protocol. The aim of this research is to design an efficient wireless communication system for the smart sensor shorts. This means that all elements connected to the wireless communication system should be considered. Therefore a classification of these elements is made and can be seen in Figure 1.4. To begin with, the first block mentions the data input. Currently a lot of research is done on the use of lossless data compression to improve the efficiency of wireless communication systems. Lossless data compression is a technique, in which the size of the data is reduced, by removing redundant data, without losing any information. The use of data compression is already widely used in other applications, think of zipping a computer file or watching a YouTube video. Secondly the protocol is shown, which handles the transport, network and data link layer of the OSI-model. The research summarized above on wireless communication systems in football is mostly focused on testing different protocols. Thirdly, the transmitter is considered, currently a lot of research is being done on implementing antennas in wearables. Finally, the receiver at the side-line can be considered, but the above research already shows that the use of 6 antenna's will most likely be sufficient.

1.2.2. Research questions

Based on the previously classified elements for getting an efficient wireless communication system shown in Figure 1.4. The focus of this Thesis will be on reducing the amount of data to send and improve the transmitter side of the sensor pants. This will then hopefully provide a basis for designing and/or choosing a suitable protocol in future works. In order to achieve an improvement in the data rate and reliability of the transmission side, the following research questions will be answered:

1. What is an optimal lossless data compression algorithm suited for the sensor shorts?
2. What is the optimal transmitter configuration for having a reliable but efficient wireless connection between transmitter and receiver?

The two research questions will be discussed in such a way that the Thesis is split in two parts. One part will discuss the lossless data compression and the other part will discuss the transmitter configuration. Each part will consist of a literature research, design/implementation, and testing and results chapter. Finally a conclusion and discussion is given, whereby the results of both parts will be combined.

Before continuing, a more in depth explanation will be given on why the focus will be first on the data and transmitter part of the communication system and not on the protocol. To begin with, in most similar research it is observed or assumed, that the communication part will be the main energy consumer. Furthermore, it can be seen that data rates are drastically lower in devices used in literature, compared to required for the sensor pants. By combining the observation or assumption that the communication part will be the main energy consumer, given the relatively high data rate, it is expected that data compression will improve the energy consumption significantly. Furthermore, the focus on improving the transmitter side, is based on the observation that the body absorbs radiation. This absorption does not only influence the radiation pattern dramatically, leading to problems in the communication range. It will also decrease the efficiency of the system, due to radiation losses in the body. Therefore the transmitter configuration will be researched for potential improvements to have a reliable connection. In conclusion, by considering data reduction and improvements on the transmitter configuration, most likely a smaller and more reliable device can be made. The reduction in size is caused by the lower power consumption, resulting in a smaller battery. This is desired, due to the current lack of small devices for this application. On top of that, the aim of the sensor pants is to be used commercially in football. This means that the device should not be obstructive to the player and hence be small.
Other potential benefits are the increase in link-reliability, due to the more efficient transmitter configuration and lower data rates. Furthermore, the characterization of the optimal radiation pattern is known and can be used to choose or design a suitable protocol.

1.2.3. Mobile network or local network

The need for doing research on wireless communication in wearable devices, would take another approach if existing mobile networks are used. It should be pointed out, that the device should be usable on a football field no matter the location. This means that the device should be operational in other countries and rural areas where the connectivity of mobile networks might be insufficient or in crowded stadiums (where everyone is using there phone). Also the use of subscriptions to mobile providers, will increase the costs of the device. It is therefore decided to consider only solutions in which the receiver is placed close to the soccer field.

2

State of the art lossless data compression

This chapter will give a brief summary of the literature research regarding lossless data compression in Appendix D. First, the selection criteria are discussed for the compression algorithms. Followed, by some theoretical background in data compression, with an analysis on common used data sets in literature. Next, the main principle of most data compression algorithms is discussed. Followed by a list of lossless data compression algorithms. Finally, it is explained, which algorithm is selected and which research questions will be analyzed in this part of the Thesis.

2.1. Selection Criteria

The main benefit of lossless data compression, is the reduction in data load, resulting in a lower power consumption and an increase in network reliability. To select the most suited compression algorithm, the following parameters are considered: compression ratio (CR); power consumption and computation cycles (CC); memory usage; packet dependency; and the ability to adapt to varying sources. For each parameter, a short explanation will be given on why it is important and how it is evaluated. In Appendix D, the full qualification of the scores varying from $---$ till $+++$ are given.

compression ratio

This parameter can be considered as the most important as explained earlier. The definition of the compression ratio is given by Equation 2.1. A shortcoming of this parameter is the dependency on the used data set, as will be seen in "ability to adapt to varying sources". Luckily, almost all compression algorithms are tested on a common data set. Unfortunately, this data set does not contain really challenging data, resulting in almost similar performances between different algorithms.

$$\text{CR} = 100 \cdot \left(1 - \frac{\text{compressed size}}{\text{uncompressed size}}\right)\% \tag{2.1}$$

power consumption and computation cycles

The parameter computation cycles will not directly be used, but its effect will be taken into consideration for the power consumption. To set a score for the power consumption, first the compression ratio is considered, because every reduction in data will gain energy savings in transmission. Secondly, the amount of computation cycles is considered, lowering the effect in power savings. It is therefore decided, to take the score of the compression ratio as upper bound. And if the algorithm is computationally heavy, this score will be lowered depending on the computation cycles. And finally a score is given for power consumption, based on the compression ratio and computation cycles. It is worth noting, that the estimation of the computation cycles is difficult from a theoretical perspective, and is mainly considered in case of two similar performing algorithms, to give a final verdict.

memory usage

The use of small hardware, will quickly lead to a relatively low amount of memory available. To make sure, that the algorithm can be implemented, the amount of memory required by the algorithm should stay within these bounds. A penalty is given on different levels of memory usage, with a minimum score of $---$, indicating that it can not be implemented. The side effect of high memory usage by the compression algorithm, is the decreased space for implementing other functionality, and also the ability to upgrade the processor to a more efficient one.

packet dependency
Even though, a fully reliable communication channel, without packet losses and errors is desired. This might not be feasible in practice. To make sure, that the data is still decompressable, each packet should be decompressable independently. In practice, an algorithm either has this property or not.

ability to compress varying sources
It is noted, that using the data set coming from the SensorScope (not available at the time of writing), might lead to similar performances. On the other hand, when data sets from volcanic activities are used, it turns out that similar performing algorithms, have dramatically different performances. [64] This has most likely to do with the probability distribution functions (pdf) of the data sets. It might be, that most of the algorithms are using an assumption on the probability distribution function of the data set, which seems to hold for the SensorScope data set, but potentially not for others. A more in depth explanation will be given later on when the data sets are analyzed, and the main principle behind data compression is explained.
Finally, the ability to compress varying sources, even though it might be hard to quantify, might be very important. It turns out, as will be shown later, that the accelerometer and gyroscope have a significantly different probability distribution function compared to the magnetometer. Therefore, requiring an algorithm, capable of handling these different probability distribution functions.

2.2. Data set analysis

Almost all data sets contain redundant information. In this case the data is collected from IMUs and represented by a 16-bit integer for every DOF. Due to this fixed length, redundant information is in the data set. By using Shannon's entropy limit, the maximum redundancy can be calculated for the symbolic representation of the data set. Shannon's entropy limit is given in equation 2.2, with $\mathrm{H}(\mathbf{x})$ being the average number of bits needed to represent one symbol from all N-symbolic representations of $\mathbf{x}$, and $\mathrm{p}(\mathrm{x}_i)$ represents the probability of symbol x_i occurring.

$$\mathrm{H}(\mathbf{x}) = -\sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{p}(\mathrm{x}_i) \log_2(\mathrm{p}(\mathrm{x}_i)) \tag{2.2}$$

The most used data set is coming from SensorScope and is used in: [31, 56–59, 64, 66, 70, 82, 105]. The data set is coming from measurements taken by sensors deployed on three locations: Le Gènèpi Deployment, HES-SO FishNet Deployment and LUCE Deployment. The deployed system consists of a TinyNode node, with a TI MSP430 microcontroller, Xemics XE1205 radio and Sensirion SHT75 sensor module (temperature and humidity sensor). The published data is in the form of a character string. Therefore the data from humidity and temperature is converted back to the original 12 and 14 bit integer respectively, using the datasheet [90]. Based on these raw values, the entropy is calculated for the original signal and the differentiated signal. Whereby the differentiated signal, is simply subtracting two neighbouring points, as shown below:

$$\begin{aligned} \Delta \mathrm{x}(0) &= 0 \\ \Delta \mathrm{x}(\mathrm{i}) &= \mathrm{x}(\mathrm{i}) - \mathrm{x}(\mathrm{i\text{-}1}) \qquad i \in [1, 2, ..] \end{aligned}$$

The calculated entropy for both signals is shown in Table 2.1 and 2.2. It can be seen, that there are differences between the own calculation and the ones in other papers. This can be explained by the following two reasons, first the back conversion might be performed different, because the differences in the temperature data sets are less varying then the relative humidity (the relative humidity requires more steps). Secondly, the data set Temp-LG-20 and RH-LG-20 contain 43059 (used by [105], and also in this Thesis) versus 21523 (used in [59]) samples. This might also indicate, that the data set is updated over time.

Finally a small run is performed with the sensor pants, whereby the entropy is calculated for the signals from the IMUs. Showing a potential compression ratio of 54.7 %. Before ending this section, it is recommended to check the shapes of the probability distribution functions from the differentiated signals in Appendix D in Figure D.3a, D.3b, D.3c, D.3d, D.3e and D.3f, to get a feeling of the relationship between the shapes of the figures and the compression ratio.

Data Set	**Entropy own calculation**	**Entropy based on [56, 59]**	**Entropy based on [67]**
Temp-LU-84	10.0677	10.0677	10.07
Temp-FN-101	10.2609	10.2609	10.26
Temp-LG-20	10.4885	10.2492	10.25
RH-LU-84	10.1061	9.9156	10.08
RH-FN-101	9.8904	9.6070	9.75
RH-LG-20	10.9979	10.7634	10.84

Table 2.1: Entropy data set SensorScope

Data Set	**Entropy own calculation**	**Entropy based on [56, 59]**	**Entropy based on [67]**
ΔTemp-LU-84	4.0469	4.0471	4.05
ΔTemp-FN-101	5.0956	5.0965	5.10
ΔTemp-LG-20	6.8950	6.8178	6.82
ΔRH-LU-84	5.9549	5.7032	5.85
ΔRH-FN-101	5.8746	5.7130	5.84
ΔRH-LG-20	7.8764	7.6052	7.67

Table 2.2: Entropy differentiated data set SensorScope

2.3. Compression Algorithms

Most research regarding lossless data compression for small embedded devices, come up with algorithms using two steps. First the correlations in the signal is removed and secondly the new symbolic representation is encoded. The goal of removing correlations in the signal, is the increased probability of a symbol occurring. These symbols (in this case integer values) can then be encoded using different methods. In general, an optimal encoder (step 2), will assign a lower amount of bits to a frequently occurring symbol and a higher amount of bits to a less frequently occurring symbol. The different combinations of the two steps are summarized in Figure 2.1.

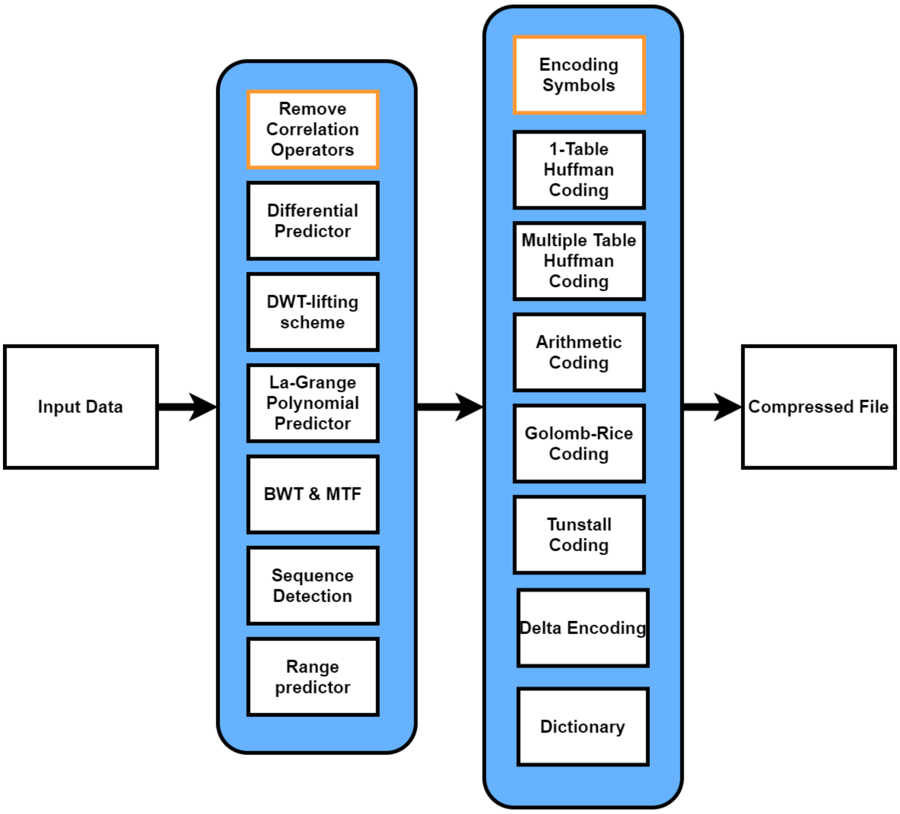


Figure 2.1: Generalized coding scheme

Algorithm	**remove correlation operator**	**encoding symbols**
DWT Lifting Scheme [12]	DWT Lifting Scheme	Delta Encoding
Tunstall [70]	Differential Predictor & Sequence detection	Tunstall coding
FELACS [59]	Differential & Range predictor	Golomb-Rice Coding
S-LZW [82]	Sequence Detection	Dictionary
LEC [67]	Differential predictor	1-Table Huffman coding
S-LEC [64]	Differential & Range predictor	1-Table Huffman coding
Simple algorithm [58]	Differential predictor	2-Table Huffman coding
ALDC [56]	Differential predictor	3-Table Huffman coding
Two-Modal GPC [63, 65]	Differential & Range predictor	1-Table Huffman coding

Table 2.3: summary of methods used in compression algorithms, split into 'remove correlation operator' and 'encoding symbols'

To continue, in Table 2.3, the different data compression algorithms are shown, with the steps taken during compression. Followed by the compression ratios on the SensorScope data set shown in Table 2.4. It is important to note, that almost all algorithms seem to perform similar on the SensorScope data set, but dramatically different on other data sets. Finally, scores are given on the different selection criteria in Table 2.5. Whereby the FELACS (Fast and Efficient Lossless Adaptive Compression Scheme)

algorithm is selected as best choice. This decision is based on the highest score on all 5 criteria. The strength in the FELACS algorithm is mainly coming from its simplistic approach in using Golomb-Rice coding, which are easy to implement on a computer, due to the fast arithmetic operations. On top of that, the range predictor in combination with the differential predictor is expected to give it a very well robustness against ”varying source statistics”.
Because the FELACS algorithm is selected among the other algorithms, it is explained in the next Section 2.4, the other algorithms can be found in Appendix D

	DWT LS [12]	**Tun-stall [70]**	**FE-LACS [59]**	**S-LZW [82]**	**LEC [67]**	**S-LEC [64]**	**Simple algo-rithm Kolo [58]**	**ALDC [56]**	**two modal GPC [63, 65]**
Temp-LU-84	xx.xx	xx.xx	**74.00**	48.99	70.81	72.07	73.48	73.94	70.52
Temp-FN-101	xx.xx	66.12	**67.63**	30.35	65.39	xx.xx	xx.xx	67.48	64.34
Temp-LG-20	xx.xx	54.32	**57.41**	20.02	53.83	54.80	xx.xx	56.90	54.05
RH-LU-84	xx.xx	xx.xx	**66.12**	31.24	62.86	63.71	63.62	65.54	61.95
RH-FN-101	xx.xx	58.20	**67.20**	36.27	62.95	xx.xx	xx.xx	66.33	60.79
RH-LG-20	xx.xx	54.70	**53.85**	21.93	48.67	51.40	xx.xx	52.87	48.82

Table 2.4: Compression ratio (%), based on equation 2.1 of algorithms using the SensorScope dataset

	DWT LS [12]	**Tun-stall [70]**	**FE-LACS [59]**	**S-LZW [82]**	**LEC [67]**	**S-LEC [64]**	**Simple algo-rithm Kolo [58]**	**ALDC [56]**	**two modal GPC [63, 65]**
compression ratio	+	++	**+ + +**	−−	++	+ + +	+ + +	+ + +	++
power consumption	+	−	**+ + +**	−−	++	+ + +	+ + +	++	++
memory usage	+ + +	− − −	**+ + +**	+ + +	+ + +	+ + +	+ + +	+ + +	+ + +
packet dependency	+ + +	− − −	**+ + +**	+ + +	+ + +	+ + +	+ + +	+ + +	+ + +
varying source statistics	++	++	**+ + +**	− − −	−−	++	−	+	−−

Table 2.5: Compression technique selection criteria

2.4. FELACS [59]

The fast and efficient lossless adaptive compression scheme (FELACS) is based on the Golomb-Rice coding. The Golomb-Rice codes are optimal for encoding exponential and geometrical distributions [38]. The compression is done by splitting the data into a fixed and variable part. The fixed part is given by calculating δ mod m, with δ being a non-negative integer in the range [0, m-1]. The variable part is given by $\lfloor \delta / m \rfloor$. A special case, also observed by Rice [79], assumes an exponential distribution on the source data and takes m = 2^k. This gives rise to an implementation using fast arithmetic operations on a computer. The operation $\lfloor \delta / 2^k \rfloor$ is the same as k times shifting δ to the right and δ mod m is calculated by taking the k least significant bits [95]. This methods, splitting the value into a fixed and variable part is called the split simple option for Rice algorithms. Three other proposed methods in rice coding are: fundamental sequence, second extension option and zero block option [79]. The Rice coding is performed on non-negative integers, therefore a conversion is applied on the data set, given in [89, Equation 2.3].

$$d_i = \begin{cases} 2\Delta x_i & 0 \leq \Delta x_i \leq T_i \\ 2|\Delta x_i| - 1 & -T_i \leq \Delta x_i < 0 \\ T_i + |\Delta x_i| & \text{otherwise} \end{cases} \tag{2.3}$$

$$T_i = \min(x_i, 2^N - 1 - x_i)$$

In the FELACS algorithm, each data point can be written as $\ll s_i \mid a_i \gg$, with s_i coming from $\lfloor \delta / m \rfloor$ and a_i from δ mod m, with m = 2^k, and $\delta = d_i$. For data sets with larger numbers, the value of s_i can significantly increase, if a relatively low k is taken. Therefore it is decided to replace this form of Rice-Coding, with the formulas shown in Equation 2.4 and 2.5. It should be noted, that the use of $\lfloor \log_2 (d_i) \rfloor$ might be even fast, due to the fast arithmetic operations used in [67].

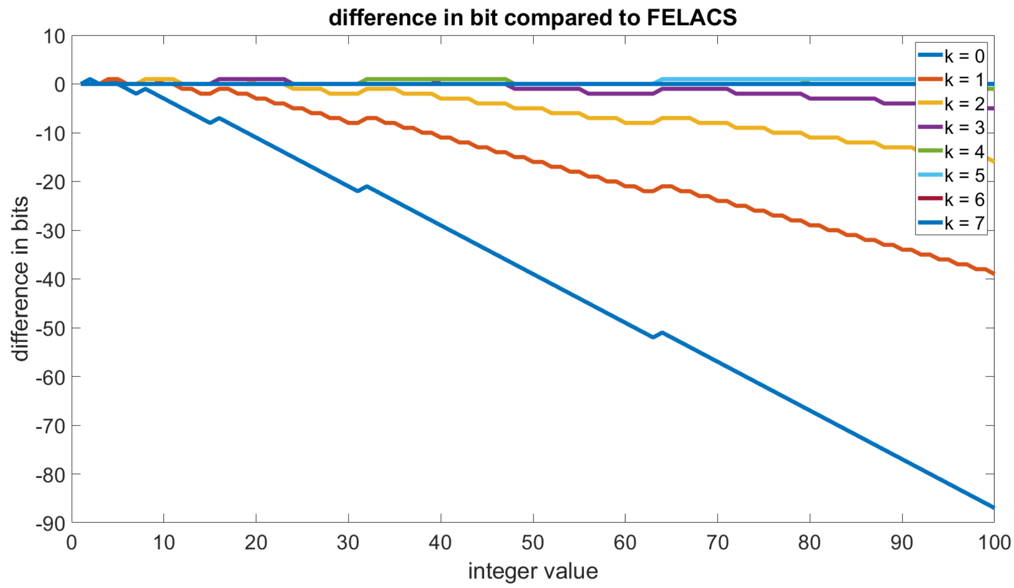


Figure 2.2: difference in bits required to represent an integer value, comparing the modification in Equation 2.4 and 2.5, with the Golomb-Rice codes in the FELACS algorithm

Figure 2.2 shows the difference in bits, between the proposed modification and the original implementation, to represent an integer value. These results show the improvement on representing higher integer values, using this modification. The trade-off in this modification, is payed by an occasionally occurring penalty of 1-bit, in the lower range of integers. However, it is more desired to have a constant compression ratio over the entire time. On top of that, this modification is shown to perform almost similar to Golomb-Rice codes for lower integer values, while being more efficient with higher integer values, increasing the robustness against ”varying source statistics”.

$$s_i = \begin{cases} \lfloor \log_2 (d_i) \rfloor - k^* + 2 & s_i > 1 \\ 1 & \text{otherwise} \end{cases} \quad (2.4)$$

$$a_i = \begin{cases} d_i & s_i = 1 \\ d_i \bmod (2^{s_i + k^* - 2}) & s_i > 1 \end{cases} \quad (2.5)$$

The strength of the original FELACS algorithm is in estimating the optimal k (k*) value for compression. By using an heuristic approach, only 7-bit shift- and J add-operation are required to find the optimal value of k for each block. The value is find by calculating D using [59, Equation 2.6] and choosing the correct value of k based on [59, Table 2.6], whereby J is the amount of data points per block, minus the first point.

$$D = \sum_{i=0}^{J-1} d_i \quad (2.6)$$

Optimum code parameter k*	**D region in bits**
0	$D \leq 2J$
1	$2J < D \leq 4J$
2	$4J < D \leq 8J$
3	$8J < D \leq 16J$
4	$16J < D \leq 32J$
5	$32J < D \leq 64J$
6	$64J < D \leq 128J$
7	$128J < D$

Table 2.6: Decision region for the first 8 optimum coding parameters [59]

A step-by-step example from the original paper will be given, using the MFELACS version. The following data set is given: {5555, 5583, 5548}, from a 14-bit ADC.

1. Calculate Δx_i = {28 , -35}, convert to d_i = {56, 69}, using [59, Equation 2.3].
2. Calculate the decision region, using [59, Equation 2.6]. Giving, 56 + 69 = 125, note that the first value is not used.
3. Based on [59, Table 2.6], the optimum value is k = 5, with J = 2.
4. The values of d_0 and d_1 are given by 0111000 and 001000101 respectively, using Equation 2.4 and 2.5.
5. The following packet is send: 101 (k-value), 01010110110011 (5555), 0111000 (d_0), 001000101 (d_1), and combined: 1010101011011001101110000001000101.

In summary, the modified FELACS (MFELACS) algorithm is expected to achieve high compression ratios, based on the compression results in Table 2.4 (from the original FELACS algorithm) and the expected performances shown for the modified encoding part in Figure 2.2. This is most likely caused by the range predictor k, reducing the representation of s_i in the encoding part. In comparison, most algorithms will only use the combination of a differential predictor and an encoder, this encoder is often based on an assumed static distribution of the differentiated data. In practice, this distribution will not be constant, and can vary a lot depending on the application. To overcome this, the MFELACS algorithm is aware of fluctuations in the magnitude of data points in the same block, and adapt to it using the value k. In other words, the MFELACS algorithm uses a static distribution in encoding, but can handle different distributions from the differential predictor, due to the value k representing s_i more efficiently. Furthermore, the algorithm will use a relatively low amount of computation cycles, without requiring extra memory, resulting in a positive effect on the total power consumption of the final system. Finally, there is no packet dependency, between the compressed blocks.

2.5. Research questions

This subsection will give a brief description of each research question, followed by the actually research question.
The focus of this part of the Thesis is to find a suited compression algorithm for the sensor pants. This begins with validating the compression ratio during different football sessions, and use this data to predict the expected load on the data link. This data can then be used to prove the optimality of the encoding part as a form of heuristic proof.

1. What is the compression ratio during different football sessions?
2. Will the MFELACS algorithm compress data, representative to football scenarios, optimal? More precisely, will s_i follow an exponential distribution of $p(s_i) = 2^{-s_i}$?

Later on, the trade-off between the compression ratio and memory usage (blocksize) is considered, to select an optimal blocksize.

3. What is the trade-off between the amount of data points per packet (increasing the memory usage), and the optimal compression rate?

After all these steps are performed, it is considered to improve the algorithm by replacing the differential predictor by a DWT Lifting scheme in combination with a La-Grange Polynomial interpolation, used in [12].

4. Will the use of a DWT Lifting scheme with a La-Grange Polynomial interpolation, result in a higher compression ratio, compared to a differential predictor?

Finally, it is interesting to measure or predict the extra power added in the current system, when running a data compression algorithm.

5. What is the extra power consumption in the microcontroller, by running the compression algorithm?

It should be noted though, that the optimality of the encoding part of the MFELACS algorithm will be heuristically proven in research question 2, while the ”remove correlation operator” is not proven for optimality. This decision is made, because the encoding part can theoretically be calculated based on Shannon’s entropy limit. While, the optimality of the ”remove correlation operator”, might be less trivial to prove. Therefore an improvement is suggested in research question 4, to replace the differential predictor with a DWT Lifting scheme in combination with a La-Grange Polynomial interpolation.

3 Design lossless data compression

This section will describe the integration of the modified FELACS algorithm in combination with the current hardware. It will first describe the pseudo-code of the algorithm. Followed, by data storing and the integration into the full system, including timing characteristics. Next, some important optimization steps are discussed and finally, the validation of the implementation is discussed.

3.1. Pseudo-code MFELACS

In Algorithm 1, the pseudo-code for the MFELACS algorithm can be seen.

Algorithm 1 MFELACS algorithm

1: **procedure** Compressing one block of data:
2:     **for** all samples **do**
3:         differential operator and store results
4:     **end for**
5:     **for** all samples **do**
6:         apply rice mapping function to convert the differentiated data into a positive range with only positive integers
7:     **end for**
8:     calculate the value D
9:     calculate optimal value for k
10:     save k and the first sample, at the beginning of the compressed data
11:     **for** all samples **do**
12:         calculate s_i
13:         calculate a_i
14:         add $\ll s_i \mid a_i \gg$ to the compressed data
15:     **end for**
16:     return compressed data vector
17: **end procedure**

Line 3: The algorithm starts with taking the differences between the sample points. This differential predictor is already mentioned in D.2.2 and slightly changed to:

$$\Delta x_i = x_{i+1} - x_i \qquad i \in [0, 1, 2, ..]$$

The values for Δx_i are saved in an array with the signed 32-bit integer format, because the maximum value from x is in the range of $[-2^{15}; 2^{15}-1]$, while Δx_i is in the range of $[-2^{16}+1; 2^{16}-1]$. The amount of data points in this array is given by:

$$\text{length}(\Delta x) = \text{length}(x) - 1$$

Line 6: The rice mapping function below is applied to the vector $\Delta\mathbf{x}$:

$$d_i = \begin{cases} 2\Delta x_i & 0 \leq \Delta x_i \leq T_i \\ 2|\Delta x_i| - 1 & -T_i \leq \Delta x_i < 0 \\ T_i + |\Delta x_i| & \text{otherwise} \end{cases}$$

$$T_i = \min(x_i, 2^N - 1 - x_i)$$

Line 8: After the data is converted and stored in d, the value for D is calculated:

$$D = \sum_{i=0}^{J-1} d_i$$

Line 9: Next the optimal value for k is chosen, based on the decision regions in Table 3.1, with the value J equal to the length of d.

Optimum code parameter k*	D region in bits
0	$D \leq 2J$
1	$2J < D \leq 4J$
2	$4J < D \leq 8J$
3	$8J < D \leq 16J$
4	$16J < D \leq 32J$
5	$32J < D \leq 64J$
6	$64J < D \leq 128J$
7	$128J < D$

Table 3.1: Decision region for the first 8 optimum coding parameters [59]

Line 12-15: The values for s_i and a_i can be calculated by:

$$s_i = \begin{cases} \lfloor \log_2 (d_i) \rfloor - k^* + 2 & s_i > 1 \\ 1 & \text{otherwise} \end{cases}$$

$$a_i = \begin{cases} d_i & s_i = 1 \\ d_i - 2^{s_i + k^* - 2} & s_i > 1 \end{cases}$$

To prevent the value of s_i reaching negative values, the lower limit is set to s_i = 1. Finally the values of s_i and a_i are encoded. The value of s_i is encoded based on Table 3.2. The value for a_i is encoded, by representing the decimal value of a_i in k^* + s_i - 2 bits. For example, if a_i = 5, and k^* + s_i - 2 = 4. The binary representation of a_i = 0101 (= 5 in decimal notation). In general, the length of ai, is equal to k* for $s_i \leq 2$, and equal to $s_i + k^* - 2$ for $s_i > 2$.

s_i	binary representation
1	1
2	01
3	001
4	0001
5	00001
6	000001
7	0000001
..	....

Table 3.2: binary representation of s_i

Line 16: Eventually the data is put into the following format:

$\ll k^* \gg \ll$ first sample (16-bit) $\gg \ll s_0 \mid a_0 \gg \ll s_1 \mid a_1 \gg$ .... $\ll s_{end} \mid a_{end} \gg$

This format can also be seen in Section D.3.3, with a numerical example of the MFELACS algorithm. It can be observed, that the amount of ≪ s_0 | a_0 ≫ repetitions, depends on the chosen block size (the block size is the amount of data points in one block). It is discussed earlier, that a high block size results in high memory usage. Therefore, the block size is not fixed in the algorithm and can easily be changed. However, the lowering of the block size results in overhead introduced by the ≪ k^* ≫ ≪ first sample (16-bit) ≫ part.

3.2. System integration

The data compression algorithm is implemented on the PCB with an STM32F411Ret6 running a real time operating system, called FreeRTOS. In the introduction, it is already discussed, that the current system is using a preemptive highest priority scheduling scheme. This means that the dataCompressionTask, should be placed within this constraint. It is decided, to let the dataCompressionTask run, when all sensors are read out and safely stored in a buffer. It is important to compress the data between these tasks, to ensure a fast delivery of compressed data to a wireless communication module. The other tasks, responsible for writing the SD-card and handling buttons to the outside world, are prioritised lower. After the data compression algorithm is implemented in the system, a timing analyses is performed. It turns out that in the worst case scenario, the processor has 44 % of the time, no task to execute. The worst-case timings of the tasks are shown in Figure 3.1. It is observed, that the dataCompressionTask has an execution time of 93 till 169 ms, with an average of 136 ms. If the power consumption of the system is more optimized, the processor would be clocked at a lower frequency, instead of using a sleep state with relatively short intervals of waking up. The use of data compression will result in using 16 % of the processors capacity. This means, that the processor can be clocked lower, but when data compression is used, the new frequency will be 16 MHz higher. This will result in an extra 1.6 mA h of power consumption, based on the given 100 µA/MHz, equal to a 1.6 % share in the current power consumption. This number might be on the higher side, because in most cases it will perform the task faster and leave room for an idle state. Also the effect of preemption is considered, adding a negligible <1 % to the total execution time. For a more detailed system integration, see Appendix C. This also gives an explanation to the fifth research question in Section 2.5.

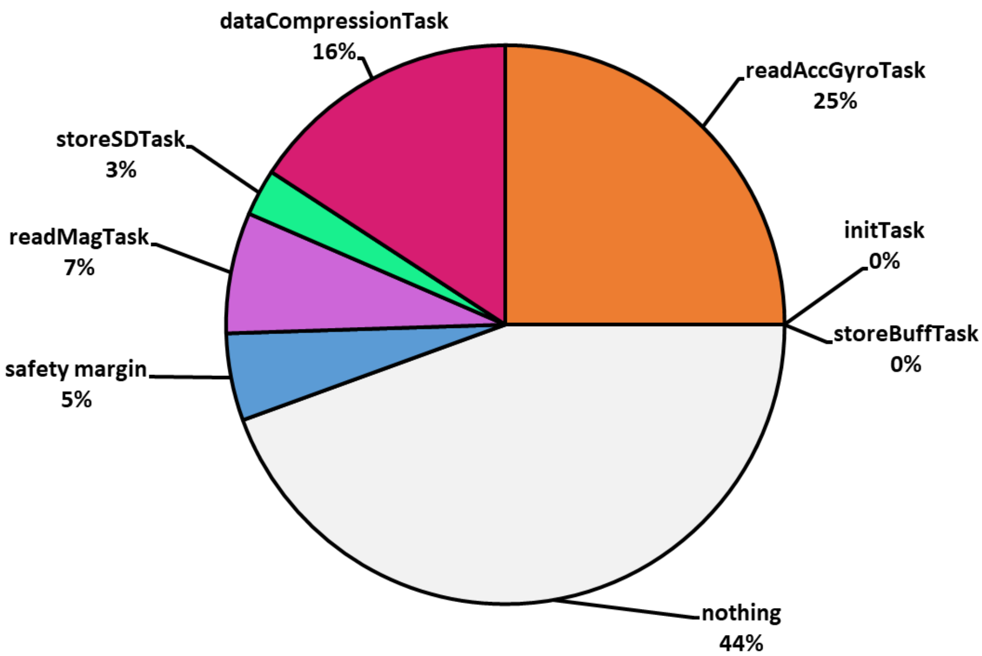


Figure 3.1: Worst case execution time per task per second, microcontroller running at 100 MHz

3.3. Optimization steps

The MFELACS algorithm will run on a microcontroller. Therefore it should be efficient with its resources. In order to guarantee optimal performance in running, the implementation of the algorithm in C/C++ is given in Appendix A. A few abbreviations on important optimization steps in the algorithm will be given. To begin with an efficient implementation for the $\lfloor \log_2(x) \rfloor$ operator is given by the function floorLog2, suited for integers. The strength of this implementation is the use of a right bit-shift operator. The use of a shifting operator is very efficient in C/C++ and is therefore very desired. The use of this shift operator can be seen through the entire algorithm, beginning with the value for omega, which requires the calculation of $2^N - 1 - x_i$. When the operator pow(2,N) is used, the execution time of the entire algorithm is increased by a factor 5, compared to the bit shift equivalent ($1 \ll N$). Also the boundaries shown in Table 3.1, are boundaries, easy to implement using bit shift operators. Finally, the value ai, is calculated by $a_i = x_i - 2^{k^* + si - 2}$. The value for $2^{k^* + si - 2}$, is implemented using ($1 \ll$ (k_optimal + si - 2)). Another, major improvement is the use of the syntax block[i], instead of a block.at(i). Even though, both take the i'th value from an array, the difference in execution time is a factor 1.2, caused by the boundary check using block.at(i) to prevent out of bound errors. Another important aspect to use, is the standard memory allocation in C. This will result in a performance increase of 16 %, compared to the use of the standard push_back() function used in combination with the std::vector. It should be noted, that there should be enough memory allocated, to prevent a memory allocation error, crashing the system.

3.4. Validation

The following steps are taken to verify the implementation of the algorithm and to make the analysis fast and usable. First, the algorithm is implemented on a desktop-PC using the CodeBlock IDE. Next, the files from the SensorScope data set are compressed and decompressed to verify the correctness of the algorithm, these results can be seen in Table 3.3. It can directly be seen that the current implementation is achieving lower compression ratios compared to the original paper. An explanation can be found, when looking at the analyses of the data sets in Section D.2.2. In this analysis, several differences between the data sets are seen, such as the amount of data points and the conversion to the raw integers. Fortunately, the Temp-LU-84 data set, seems to be the same, shown in Section D.2.2, this can also be seen in the identical compression ratios in Table 3.3. If the entropy is calculated from the differentiated signal and converted into a compression ratio value, shown in the column "CR- based on entropy from differentiated signal", only a small difference of at max ±1.18 % in compression ratio is seen, compared to the original implementation in "CR - implementation from Appendix A" in Table 3.4. When the compression ratios from the column "CR-based on entropy from differentiated signal", is compared with the ones from the original paper, "CR - original" in Table 3.5, the differences seem to be odd. Starting with the small differences in compression ratios on the temperature data sets, followed by larger differences of around 3 % till 4 % on the relative humidity data sets. It is therefore concluded, that the data set used in this Thesis is different from the ones used by the authors from the original FELACS algorithm. An explanation for this difference, can be found in the following aspects. First of all, the link to the data sets is not available anymore and could not be found using the waybackmachine. Therefore the data sets were requested from the authors in [105]. Secondly, it could be that the data set is updated over time, explaining the increase in samples. Finally, the conversion to raw integer values for the relative humidity might be performed differently.

As next step, the data files from the sensor pants are taken as input. It is then necessary to remove any redundant data, because the SD-card stores the magnetometer data at a frequency of 250 Hz, instead of 100 Hz. Afterwards the data can be split in 45 columns (5 sensors with 9 DOF), from which every column is split in blocks of 125 samples for the gyroscope and accelerometer, and 50 samples for the magnetometer. These blocks are then compressed using the MFELACS algorithm. During the compression, parameters from the algorithm can be logged in text files and loaded into MATLAB for analysis. Finally, when the algorithm is proven to work correctly, the algorithm is optimized on the PC by checking the execution time after every optimization step. After the optimization process is finished, the algorithm is implemented on the microcontroller and the execution time can be recorded for the algorithm, as shown in Figure 3.1.

data set	CR - original implementation	CR - implementation from Appendix A	difference between implementation original and Appdix A
RH-FN-101	67.20	62.10	+5.10
RH-LG-20	53.85	51.01	+2.84
RH-LU-84	66.12	63.47	+2.65
Temp-FN-101	67.63	67.25	+0.38
Temp-LG-20	57.41	56.28	+1.13
Temp-LU-84	74.00	73.97	+0.03

Table 3.3: comparison between compression ratio (%) from the original algorithm and the implementation in Appendix A

data set	CR - implementation from Appendix A	CR - based on entropy from differentiated signal	difference between implementation from Appendix A and entropy from differentiated signal
RH-FN-101	62.10	63.28	-1.18
RH-LG-20	51.01	50.77	+0.24
RH-LU-84	63.47	62.78	+0.69
Temp-FN-101	67.25	68.15	-0.90
Temp-LG-20	56.28	56.90	-0.62
Temp-LU-84	73.97	74.70	-0.73

Table 3.4: comparison between compression ratio (%) from the implementation in Appendix A and the theoretical entropy

data set	CR - original implementation	CR - based on entropy from differentiated signal	difference between original implementation and entropy from differentiated signal
RH-FN-101	67.20	63.28	+3.92
RH-LG-20	53.85	50.77	+3.08
RH-LU-84	66.12	62.78	+3.34
Temp-FN-101	67.63	68.15	-0.52
Temp-LG-20	57.41	56.90	+0.51
Temp-LU-84	74.00	74.70	-0.70

Table 3.5: comparison between compression ratio (%) from the original algorithm and the theoretical entropy

4

Testing lossless data compression

This chapter will discuss a suitable testing method to verify the research questions stated in Chapter D.3.13:

1. What is the compression ratio during different football sessions?
2. Will the MFELACS algorithm compress data, representative to football scenarios, optimal? More precisely, will s_i follow an exponential distribution of $p(s_i) = 2^{-s_i}$?
3. What is the trade-off between the amount of data points per packet (increasing the memory usage), and the optimal compression rate?
4. Will the use of a DWT Lifting scheme with a La-Grange Polynomial interpolation, result in a higher compression ratio, compared to a differential predictor?
5. What is the extra power consumption in the microcontroller, by running the compression algorithm? (this one is already answered in Section 3.2)

4.1. Compression ratio in football

For testing the compression ratio during different football session, two tests are performed. During the first test, a participant performs one type of movement. At the second test, a training session is performed by the participant. The results of these tests will contribute to answering the first research question.

4.1.1. Experiment 1

For this experiment, two amateur male participants at the age of 23 (participant 1A) and 25 (participant 2A) are assigned to perform one type of movement. The experiment is carried out at the v.v. 's-Gravendeel, on an artificial grass field. Both experiments are approved by the ethical committee from the TU Delft under ID 1237, with the device report shown in Appendix B. The compression ratios given in this section, are the average compression ratios from all sensors, if not specified otherwise.

The following type of movements are performed by the participants, whereby the percentages indicate the physical load experienced by the participant:

- shooting at goal, at 100 %
- vertical jumping, at 50 %
- zig zag running with ball for 50 m, at 80 %
- sideways for 50 m, at 50 %
- sprinting with ball for 50 m, at 100 %

- sprinting for 50 m, at 100 %
- running with ball for 100 m, at 80 %
- running for 100 m, at 80 %
- jogging with ball for 100 m, at 50 %
- jogging for 100 m, at 50 %
- walking with ball for 100 m, at 20 %
- walking for 100 m, at 20 %

The results from these movements are analyzed in the following way. All movements, except the shooting and jumping, show an almost steady compression ratio. These movements are classified as long duration movements ≥ 5 s. The jumping and shooting are classified as short duration movements, because these movements show a peak valley of ≤ 2 s and will then return to a higher compression ratio. By taking the average value of the long duration movements and the average peak values of the short duration movements, the average compression ratio per movement can be classified, as shown in Figure 4.1.

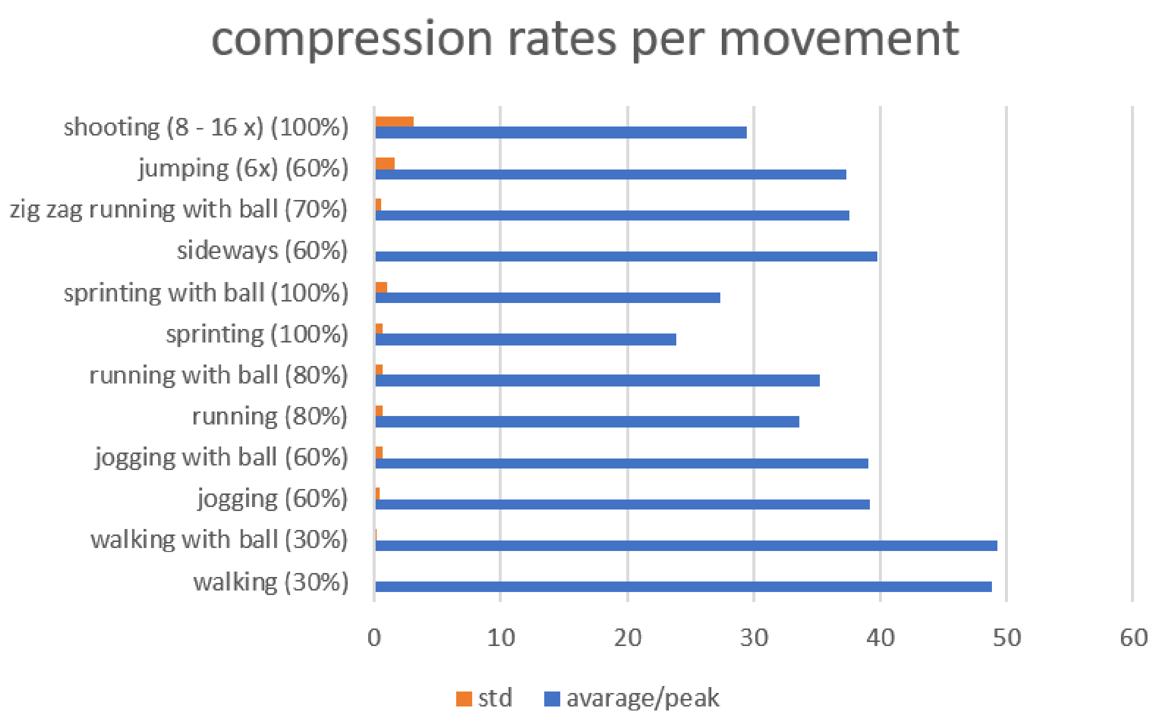


Figure 4.1: compression ratio per movement, blue is the average value of the compression ratio or the peak for jumping and shooting, orange gives the standard deviation, between results from the same movement

It is expected, that a more intensive movement, will result in a lower compression ratio. The lower compression ratios are caused by the higher integer values coming from the sensors. This is nicely illustrated in Figure 4.2 and 4.3. In Figure 4.2, the integer values from the x-axis of the accelerometers located at the lower leg, is showing higher values for more intensive movements. These higher values, directly result in lower compression ratios, shown in Figure 4.3. Another unexpected observation in Figure 4.1, is the compression ratio for shooting and sprinting. It turns out, that sprinting will cause a lower compression ratio than shooting, with a value of 23.9 % ±0.7 %. In summary, the average compression ratio depends on the type of movement being performed, with the lowest compression ratio observed in sprinting.

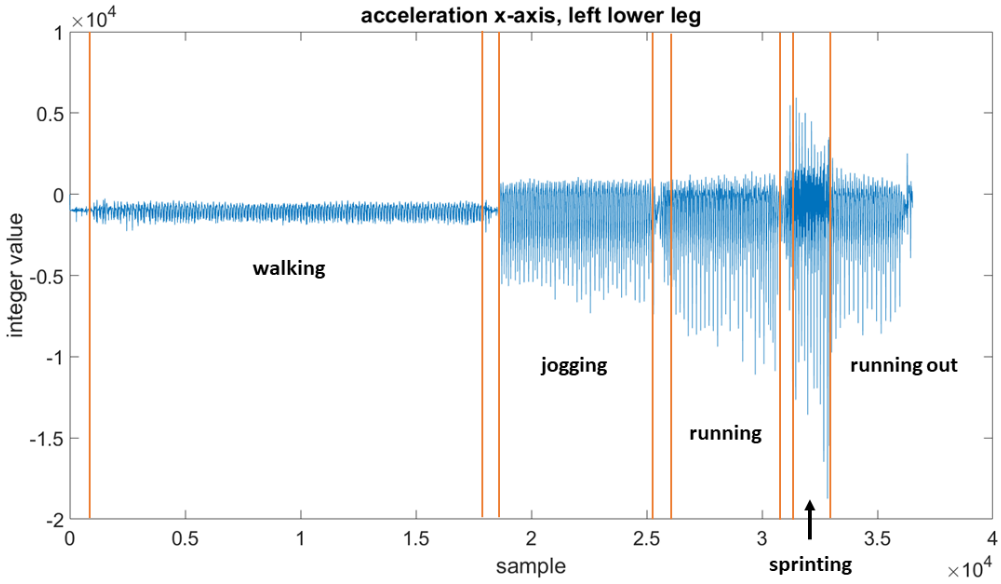


Figure 4.2: Acceleration x-axis left lower leg samples at 250 Hz, the participant will: 1) walk, 2) jog, 3) run, 4) sprint and 5) run out. The small spaces between each movement, are caused by the player standing still and turning at the end of the field.

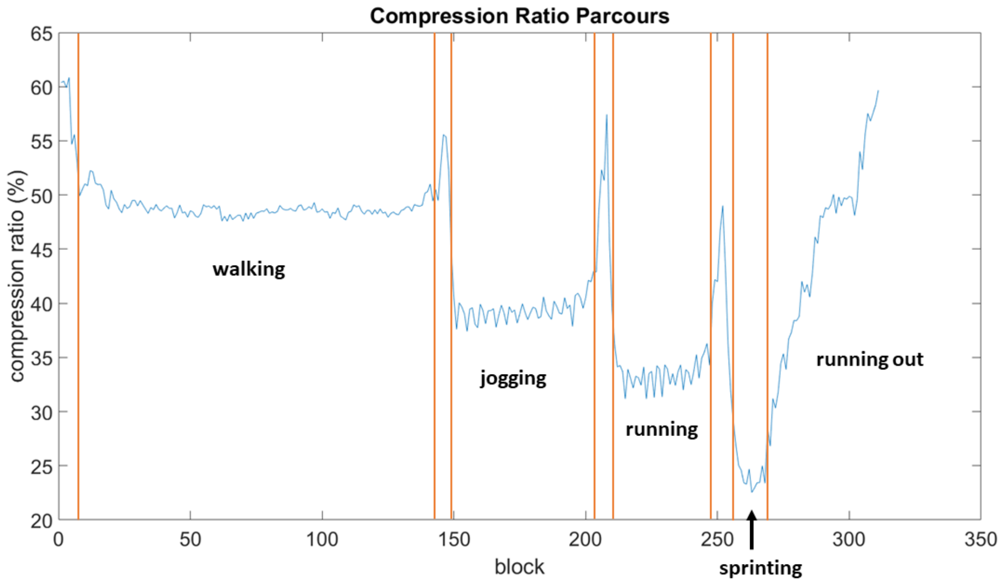


Figure 4.3: Average compression ratio of all sensors, the participant will: 1) walk, 2) jog, 3) run, 4) sprint and 5) run out. The small spaces between each movement, are caused by the player standing still and turning at the end of the field.

4.1.2. Experiment 2

As follow up experiment, 5 male participants are asked from the age of 18 till 25, with 2 participants as amateur football player and 3 from the first team of the v.v. 's-Gravendeel. The amateur participants are classified as participant 1A and 2A, and the "professional" football players are indicated as participant 3P, 4P and 5P. The aim of this experiment is to see, how the compression algorithm will perform in a real football scenario. Therefore a training session is taken from [53]. Whereby the researchers have analyzed the moments with the highest intensity during a football game from 8 elite players. Based on these moments, the amount of time spent on each movement is recorded and multiplied by the velocity, resulting in an average distance covered by a certain movement. These movements are then put into a run, in which the player will perform the same movements as observed in the most intense moment from a match. The run can be seen in [53, Figure 4.4].

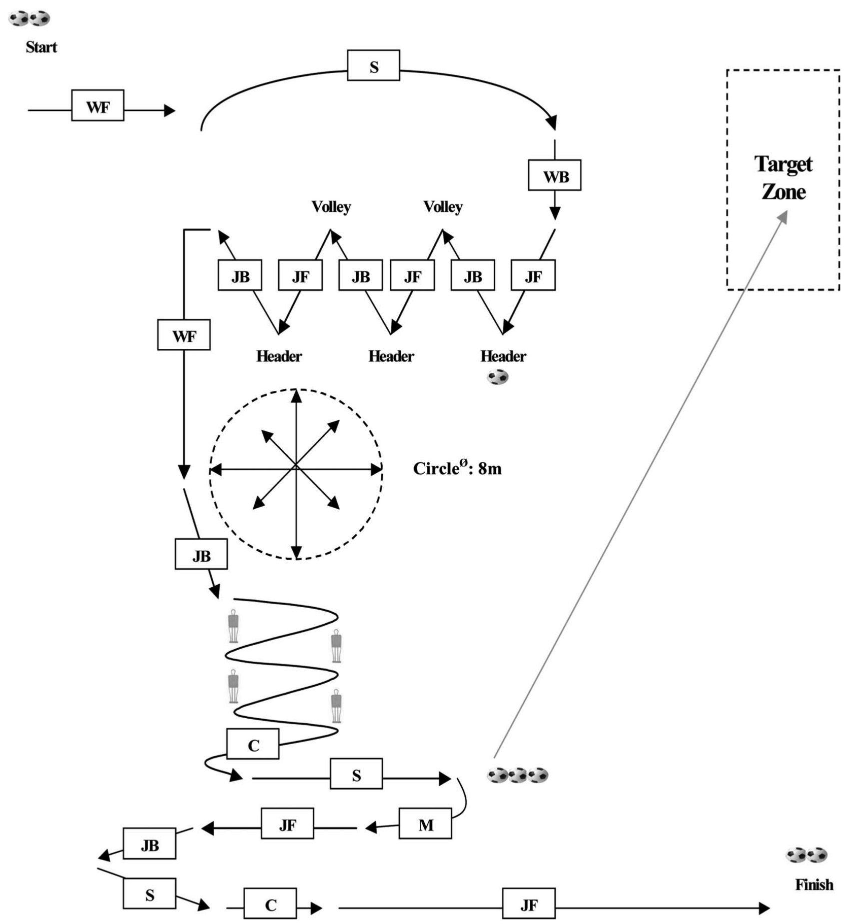


Figure 4.4: A schematic representation of the training session. The player started the drill at the "start" point by taking a throw-in and followed the outlined route (/), which consisted of the following: walk forward (WF), walk backward (WB), jog forward (JF), jog backward (JB), sideways movement (S), cruise (C), and maximal sprinting movement activities (M). [53]

The run is performed two times or more by each participant, and the best two of these runs are taken. Afterwards, the data from all sensors is analyzed, to check for system failures. It turns out that two participants suffered from a small malfunctioning in the sensor pants. Participant 3P, had a failing magnetometer at the lower legs and participant 4P had a disconnected sensor at the right lower leg. This is taken into account in the rest of the data analysis.

The compression ratio from participant 1A, during one run is shown in Figure 4.5. It is hard to see on first sight, whether or not the compression ratios are logical. Therefore, the different movements at each time-instant are shown in Figure 4.6 and numbered. When looking at the first sideways movement (number 2), the compression ratio seem to be fairly constant, this also applies for walking back (number 3) and walking forward (number 1 and 5). This is to be expected based on experiment 1. It is also visible, that shooting (3x, at number 9), has three peak valleys, which was already noticed for the shooting movement. It is interesting to see, that for maximum sprinting (number A) the compression ratio is showing a peak valley, almost similar to shooting. This indicates, that a longer distance (50 m instead of 13 m) similar to experiment 1, will result in a low compression ratio, for a longer duration >2 s.

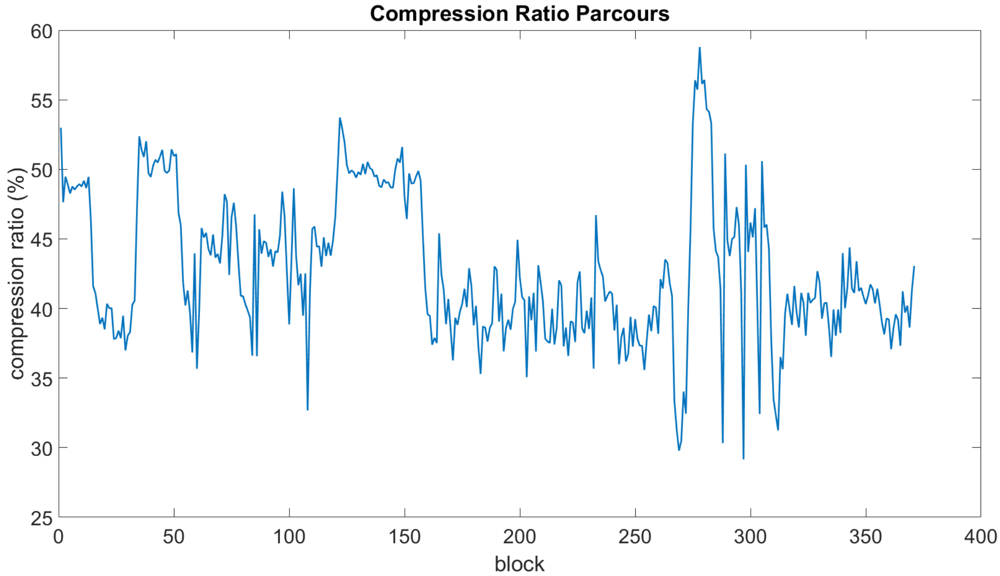


Figure 4.5: average compression ratio from all sensors during the run shown in Figure 4.4

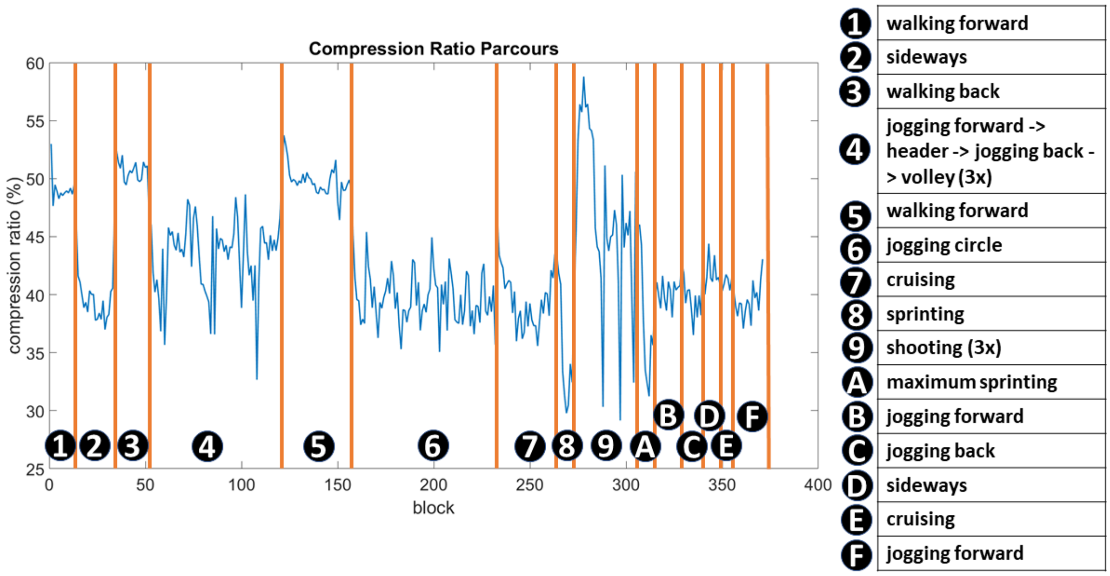


Figure 4.6: average compression ratio from all sensors during the run shown in Figure 4.4, with the classification of each movement

Finally it is interesting to look at the average compression ratios during the training session. It should be noted, that the malfunctioning of the sensors for participant 3P and 4P, have led to not use the average compression ratios from these participants, because it will introduce an offset on the average compression ratio. The result can be seen in Table 4.1. Based on the obtained compression ratios, it can be said, that during the most intensive 4 minutes in a match, the compression ratio will most likely be between 43 % and 45 %.

participant	CR attempt 1	CR attempt 2
1A	42.77	43.19
2A	43.10	43.61
5P	44.68	45.20

Table 4.1: average compression ratio per participant during the training session

4.1.3. Conclusion and discussion compression ratios during football sessions

In conclusion, the compression ratio during a football session, will depend on the intensity of the movement and the duration. The most intensive movement is observed to be sprinting with a compression ratio, at 23 % for ≥5 s. This value can be taken as the maximum load, introduced by one player on the data link. It is worth to note, that the increase in compression ratio, will occur when the mobility of the player is at it highest. Finally, the compression ratio during the most intensive minutes in a match are estimated at 43 % till 45 %, this is a good indication for the average compression ratio in this interval. It is to be expected, that the compression ratio in the less intensive intervals will be higher.

4.2. MFELACS performance

This section will try to answer the second research question: will the MFELACS algorithm compress data, representative to football scenarios, optimal? More precisely, will s_i follow an exponential distribution of $p(s_i) = 2^{-s_i}$?
As discussed earlier in the conclusion and discussion of the literature research in Section D.3.12. The algorithm is using a differential predictor in combination with a range predictor. This means, that the input data is first differentiated and then the parameter k is estimated, to indicate the range in which the data is assumed to lie. If the value of k is estimated correctly, it is assumed, that the value of s_i will follow an exponential distribution given by $p(s_i) = 2^{-s_i}$. The distribution is optimally encoded and shown earlier in Table 3.2. If the distribution of s_i, shows this behaviour for the different data sets, it can be concluded that the algorithm works optimal. It should already be pointed out, that the aim of the research is to show, that the algorithm is working optimal on the data sets. It does not mean, that it is the most optimal algorithm to compress the data set. This would required multiple algorithms being implemented and tested, requiring an enormous amount of time, with most likely, a little bit extra gain. Therefore, the algorithm (FELACS) with the highest potential is slightly modified, implemented, and tested.

4.2.1. Validating data sets

Before the distribution of s_i is given, first an analysis of the data sets from the training session is given. It turns out, that the shapes of the probability distributions look similar between participants, therefore the one from participant 1A is used and given in Figure 4.7a and 4.7b. It is noticeably, that all three sensors follow a different distribution. Surprisingly, the magnetometer has a peak at the value ’0’. This is most likely caused by the sensor itself, it is already decided to replace this generation of the AK8963 with its follow up. For now, the data coming from the magnetometer, seem to behave strange and is most likely caused by not updating a register, resulting in the same read out. It should be pointed out, that these probability distribution functions are already from the differentiated signals.

Based on the differentiated signals, the entropy is calculated. The entropy gives the theoretical limit in encoding this signal. It can be seen in Table 4.2, that the non differentiated signals are harder to compress then the differentiated signals, which is already seen when analyzing data sets in Section D.2.2. It can be seen that the accelerometer and gyroscope are harder to compress, compared with

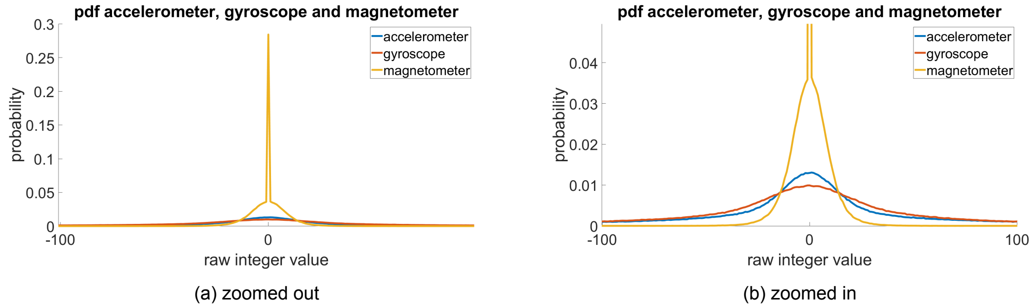


Figure 4.7: probability distribution with 95 % interval of of differentiated signal from accelerometer, gyroscope and magnetometer from participant 1A

the magnetometer. This follows from the distributions shown in Figure 4.7b. Whereby a higher peak at '0', results in a higher compression ratios.

4.2.2. Comparing compression ratio MFELACS versus reference

To measure the performances of the MFELACS algorithm, a suited reference should be taken to compare the results. This reference will be a compression algorithm based on a differential predictor, with an optimal encoder. This reference does not exist in practice, but its performance can be calculated theoretically. Results of this theoretical performance can be seen in Table 4.2. From now on, the term "reference" is used to describe this non existing ideal compression algorithm, using a differential predictor in combination with an optimal encoder.

When the results of MFELACS are compared with the reference, it can be observed, that the compression ratio for the accelerometer and gyroscope are almost identical. Unfortunately, the data from the magnetometers show a significant performance difference. Initially it was assumed, that this gap was caused by the high amount of double readings, resulting in a lot of 0's and a discontinuous probability distribution function. However, as will be seen further on, it is most likely caused by the noisy character of the signal.

Based on the achieved compression ratios, it is very impressive to see the results. Because, the probability distribution function of the signal is not known beforehand, but observed along the way. It is very likely that the accelerometer and gyroscope are profiting from the range predictor, to achieve these near optimal performances, even though, the algorithm introduces some overhead, due to: the first sample being a 16-bit integer, 3-bits for the value of k, and also the extra bits to have a multiple of 8-bits. On the other hand, the compression ratio for the magnetometer is surprisingly less impressive, but still good. In the next subsection, this difference will be explained and also an observation is made, regarding the influence of a noisy characteristic on the compression ratio.

4.2.3. Revision of magnetometer

It is observed earlier, that the magnetometer seems to have a lot of zeros in the differentiated probability distribution function. It is therefore decided to investigate the signals coming from another IMU, the MPU-9250, with an integrated magnetometer. These IMUs were originally used to measure the movements located at the legs, by a system develop by the VU Amsterdam. However, the ICM20649 is now used, for its better performances on the accelerometer and gyroscope in this application. It turns out, that this magnetometer has a smaller spike in its probability distribution function. However, the difference between the compression ratio using MFELACS (66.43 %) and the reference (73.23 %, differential predictor + optimal encoding), results in a larger difference (6.80 %) then originally observed. To explain this difference, the following observations are made to get a better explanation for the lagging performances in the magnetometer. When the differentiated signals from the magnetometer (Figure 4.8a) are compared with the accelerometer (Figure 4.8b) and gyroscope, it can be seen, that the signals from the magnetometer are similar to a noisy signal coming from a normal distribution. On the other hand, the accelerometers are having a better visible pattern, such as sinusoidal shapes caused by a walking movement.

To see the compression ratio of noisy signals, the random number generator from MATLAB is used,

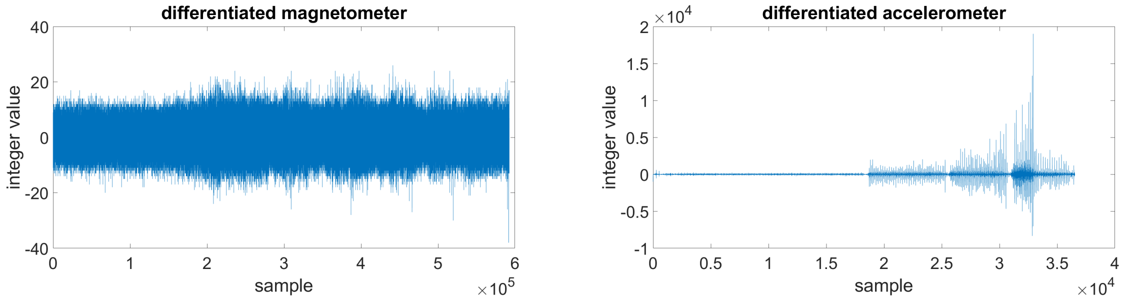
position	acc	gyr	mag	Δacc	Δgyr	Δmag
left upper leg (x)	11.5596	11.6434	7.0741	9.5287	9.1966	4.8276
left upper leg (y)	11.6048	12.0828	8.5736	9.2190	9.8514	5.0420
left upper leg (z)	11.2710	11.7587	8.5122	9.4801	8.4826	5.0012
left lower leg (x)	11.7960	12.0347	8.1102	9.2699	9.3725	4.5412
left lower leg (y)	11.4835	12.2735	8.2416	9.3204	9.1571	4.7704
left lower leg (z)	11.8721	11.3655	8.7039	9.4072	8.0999	5.0154
right upper leg (x)	11.4910	11.6150	6.9521	9.4187	9.0246	4.3627
right upper leg (y)	11.5877	12.0288	8.5921	9.1141	9.7772	4.7118
right upper leg (z)	11.1709	11.7219	8.5455	9.3803	8.4013	4.6830
right lower leg (x)	11.6576	12.0979	8.0955	9.2152	9.2764	4.5573
right lower leg (y)	11.5004	12.1555	8.1068	9.2188	9.1293	4.6834
right lower leg (z)	11.8260	11.3739	8.6407	9.2686	8.0294	4.8875
trunk (x)	11.5351	11.0725	8.0668	8.8182	8.2317	4.2705
trunk (y)	10.9547	10.7179	7.0758	8.5893	8.2354	4.2106
trunk (z)	10.5809	10.6002	8.0164	8.3788	7.7023	4.3316
average	**11.4594**	**11.6361**	**8.0871**	**9.1752**	**8.7978**	**4.6598**
CR (%) - optimal encoding	28.34 %	27.28 %	49.45 %	42.66 %	45.01 %	70.88 %

Table 4.2: average entropy all runs in bits, with the compression ratio theoretically obtained using optimal encoding

position	CR (%) accelerometer		CR (%) gyroscope		CR (%) magnetometer	
	differential predictor + optimal coding	**MFELACS**	**differential predictor + optimal coding**	**MFELACS**	**differential predictor + optimal coding**	**MFELACS**
left upper leg (x)	40.45	39.49	42.52	41.52	69.83	63.99
left upper leg (y)	42.38	41.52	38.43	36.51	68.49	62.46
left upper leg (z)	40.75	36.51	46.98	46.06	68.74	62.76
left lower leg (x)	42.06	41.40	41.42	40.37	71.62	66.18
left lower leg (y)	41.75	41.00	42.77	41.78	70.19	64.68
left lower leg (z)	41.20	40.26	49.38	48.46	68.65	62.94
right upper leg (x)	41.13	40.35	43.60	43.06	72.73	67.31
right upper leg (y)	43.04	42.58	38.89	37.24	70.55	64.95
right upper leg (z)	41.37	40.69	47.49	46.75	70.73	65.81
right lower leg (x)	42.40	41.83	42.02	41.16	71.52	66.12
right lower leg (y)	42.38	41.75	42.94	42.01	70.73	65.32
right lower leg (z)	42.07	41.27	49.82	48.91	69.45	63.84
trunk (x)	44.89	44.51	48.55	47.96	73.31	67.83
trunk (y)	46.32	45.95	48.53	47.74	73.68	68.24
trunk (z)	47.63	47.11	51.86	50.80	72.93	67.44
average	**42.66**	**41.99**	**45.01**	**44.02**	**70.88**	**65.18**

Table 4.3: compression ratio comparison between theoretical differential predictor + optimal coding (reference) versus MFELACS

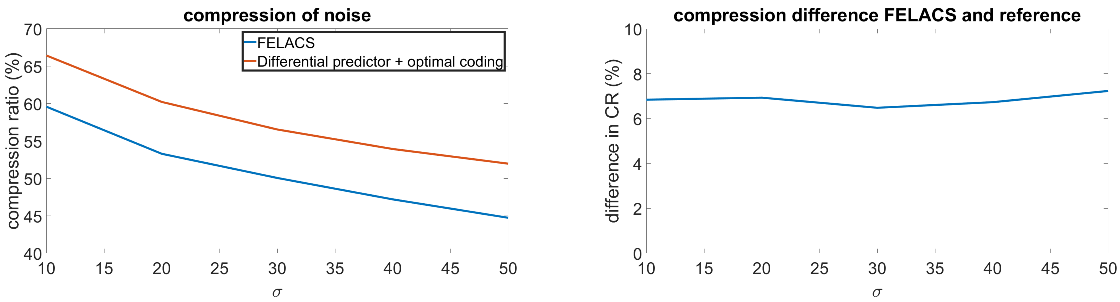
with a normal distribution. The files are then compressed and an interesting observation is made, shown in Figure 4.9a and 4.9b. It turns out, that independently of σ, there is always a difference of approximately 7 % in compression ratio. This same difference is also observed for the magnetometer signal from the MPU-9250. Based on this observation, it is expected, that the range predictor fails for signals with a noisy characteristics, while on the other hand, achieving almost optimal performance on signals, with a relative higher signal to noise ratio.



(a) differentiated signal magnetometer

(b) differentiated signal accelerometer

Figure 4.8: differentiated signal magnetometer and accelerometer



(a) CR using MFELACS versus reference on a noisy signal, coming from a normal distribution

(b) difference in CR using MFELACS versus reference on a noisy signal from a normal distribution

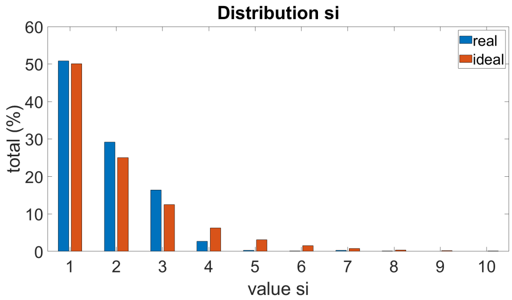
Figure 4.9: absolute and difference in CR using MFELACS versus reference noisy signal, from a normal distribution

4.2.4. Distribution of s_i

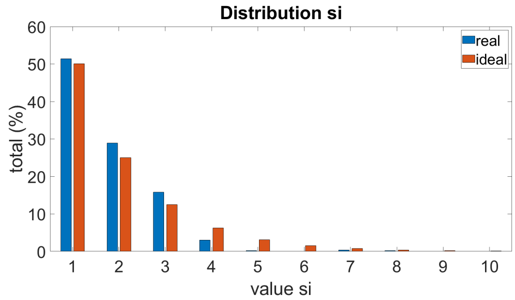
To test the distribution of s_i, the values of s_i are logged during compression and stored to evaluate in MATLAB. The distributions coming from the MFELACS algorithm, are shown next to the theoretical optimal distribution, in Figure 4.10. When looking at Figure 4.10, it is noticeable, that the value 1 up to 3 are slightly higher then expected. On the other hand, the values 4 and higher occur in a lower amount then expected. However, these higher values are expected to occur rarely and the focus is more on the values 1 until 5, which take up most of the space after encoding. To verify the losses of non optimal encoding the distribution, the theoretical optimal coding scheme is compared with the implemented encoding. The entropy is calculated, based on the occurrence of s_i and the optimal amount of bits to represent the data sets, can be seen in Table 4.4. It turns out that encoding s_i ideally, would result in 1.642 bits per sample (or data point), compared to the 1.727 with the current encoder. This means that the use of the proposed encoder for this data set, results in 5.18 % more bits per sample. The total effect of this 5.18 % will be in practice at least half of this value and most likely more, depending on the value of k*, resulting in a higher bit representations for a_i, with s_i having a smaller portion in the total compression size.

s_i value	**s_i occurring in %**	**ideal representation**	**MFELACS encoder**
1	0.533	0.483	0.533
2	0.265	0.507	0.530
3	0.163	0.427	0.490
4	0.032	0.154	0.123
5	0.001	0.013	0.007
6	0.002	0.017	0.010
7	0.003	0.029	0.024
8	0.001	0.011	0.009
total	**1.000**	**1.642**	**1.727**

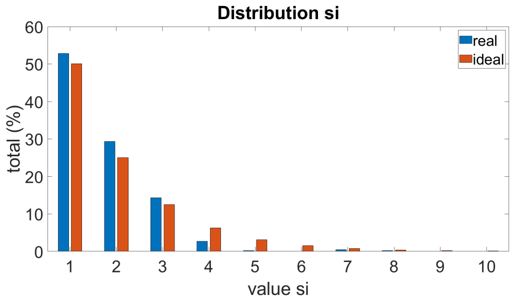
Table 4.4: optimal amount of bits required to represent s_i on average, based on its occurrence, compared to current encoder



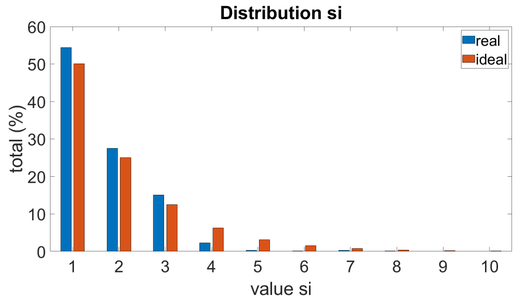
(a) probability distribution of s_i, participant 1A



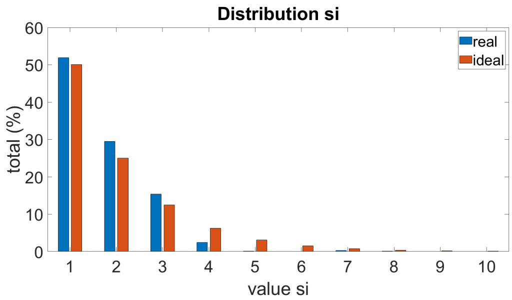
(b) probability distribution of s_i, participant 1A



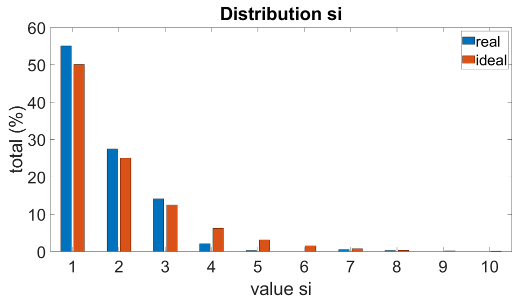
(c) probability distribution of s_i, participant 2A



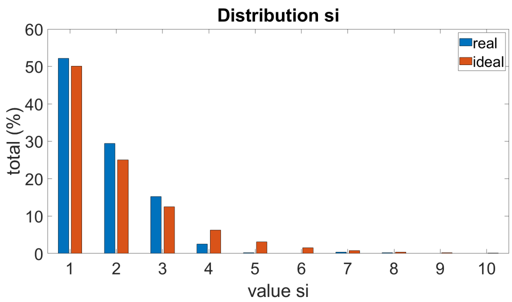
(d) probability distribution of s_i, participant 2A



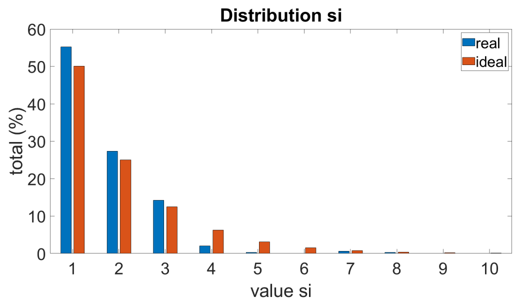
(e) probability distribution of s_i, participant 3P



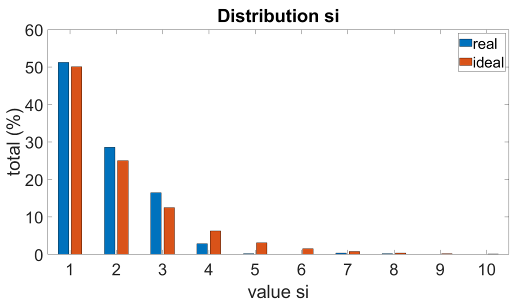
(f) probability distribution of s_i, participant 3P



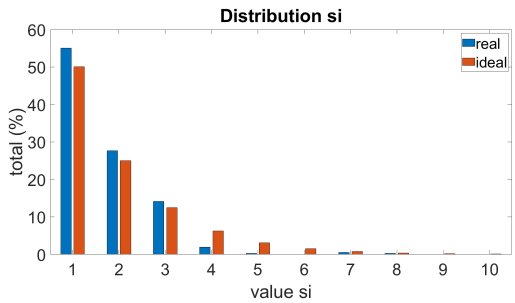
(g) probability distribution of s_i, participant 4P



(h) probability distribution of s_i, participant 4P



(i) probability distribution of s_i, participant 5P



(j) probability distribution of s_i, participant 5P

Figure 4.10: probability distribution function s_i during the training session

4.2.5. Conclusion and discussion of optimal performances MFELACS algorithm

To conclude, the distribution of s_i follows an exponential distribution. Leading to optimal performances when applied to the data set from a training session. This is also observed, when MFELACS is compared with a theoretical reference, using a differential predictor + optimal coding. Only the data sets from the magnetometers, are compressed with non optimal performances. This is most likely caused, by the noise characteristics in the differentiated signal, going into the predictor. Luckily, the magnetometers are only contributing to 16.67 % of the total input data to compress.

4.3. Packet size versus compression ratio

This section will answer the third research question: what is the trade-off between the amount of data points per packet (increasing the memory usage), and the optimal compression rate?
The effect of different block rates on the compression ratio, is performed on the data sets from the participants and the average is shown in Figure 4.11. With a blocksize of: 5, 10, 25, 50, 125, 250, 500, 1000 and 2000. The shape is already observed in the original paper, and seems to have its optimum around 500. However, going from 125 to 250, the effect is negligible and hence a value of 125, would be optimal, considering memory usage.

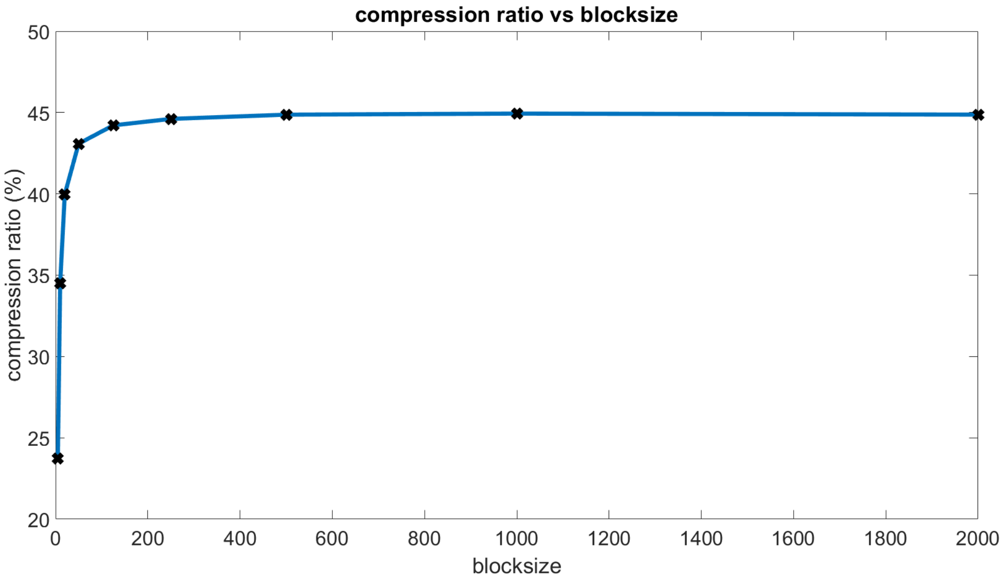
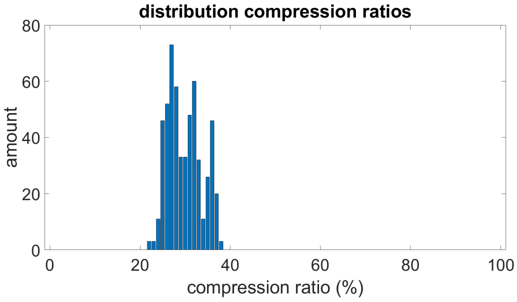


Figure 4.11: Blocksize versus compression ratio

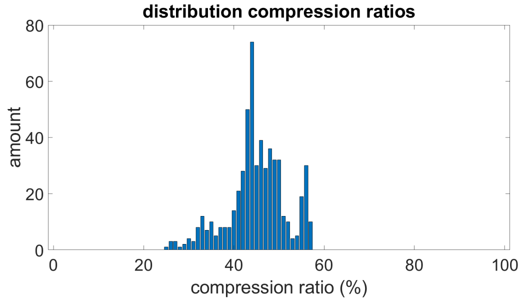
Another interesting observation is made, when the blocksize is increased. If the probability distribution function of the individual blocks is plotted in Figure 4.12a, 4.12b, 4.12c and 4.12d. It can be seen, that increasing the blocksize will result in a higher lowest value in compression ratio, and in increase in the mean value. This observation, shows that short intensive moments, are compensated, by less intensive moments, in a relatively short interval. If the blocksize of 2500 is considered, which is equal to 10 seconds of data. It can be seen, that the worst compression ratio is only 38 %. In practice, there might be a probability of a lower value, but that would be a worst-case scenario, considering the run to be already, very intensive.

Finally, the effect of overhead introduced by sending: the first sample as raw integer; the optimal value k; and the extra bits to have a multiple of 8-bit integers, nicely results in Figure 4.13, showing the effect of this converging to almost none around the value of 125.

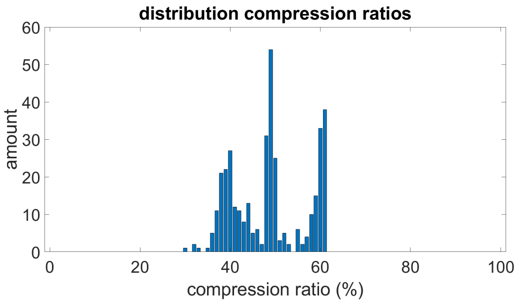
In conclusion, the value 125 will give the best compression ratio, considering memory usage. When higher blocksizes are selected, the lowest obtained compression ratio, turns out to increase. This indicates that it might be safe to use a value of 38 % as lowest value for a period of 10 s, instead of the potentially worst case 23 % earlier seen for sprinting. Also the effect of overhead, is almost negligible at the value 125.



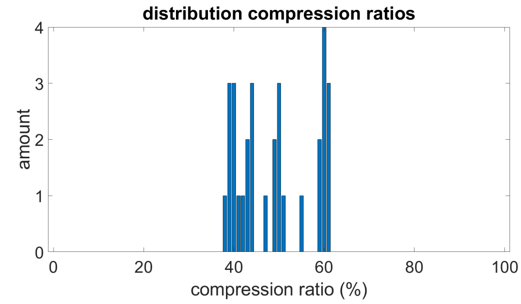
(a) probability distribution of compression ratio per block, with size 5, participant 1A



(b) probability distribution of compression ratio per block, with size 25, participant 1A



(c) probability distribution of compression ratio per block, with size 250, participant 1A



(d) probability distribution of compression ratio per block, with size 2500, participant 1A

Figure 4.12: probability distribution of compression ratio per block, with different size, participant 1A

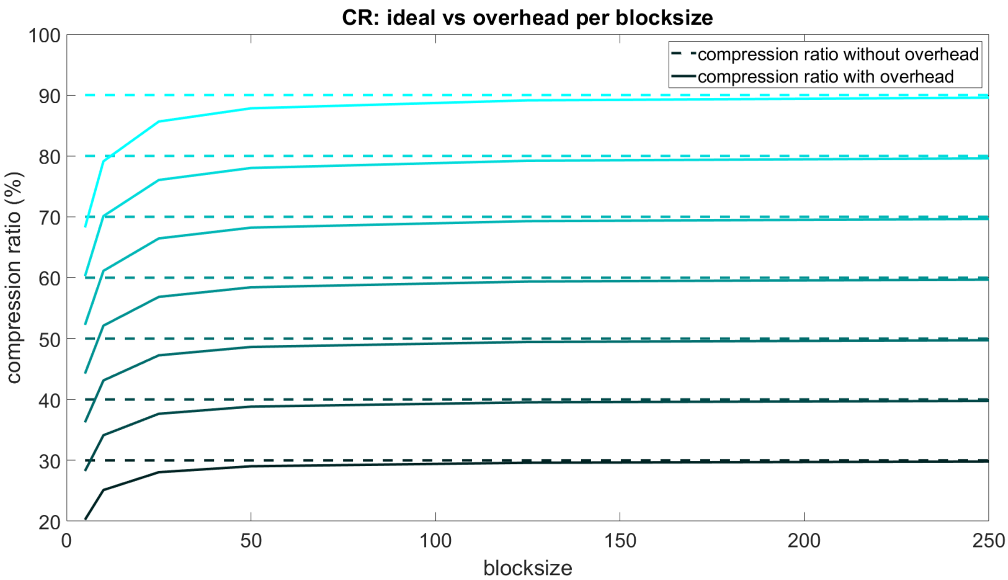


Figure 4.13: Compression ratio with and without overhead per blocksize

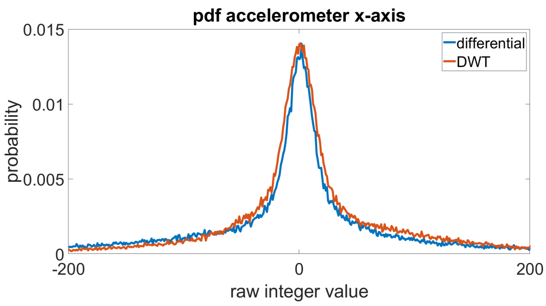
4.4. La-Grange Polynomial predictor

This section will answer the fourth research question: will the use of a DWT Lifting scheme with a La-Grange Polynomial interpolation, result in a higher compression ratio, compared to a differential predictor?

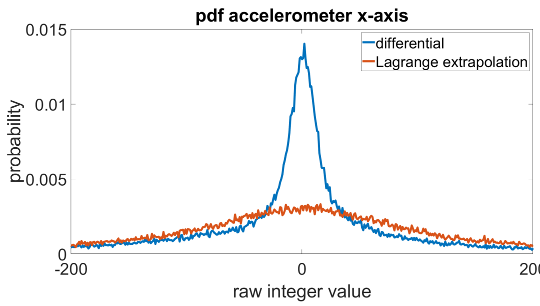
To investigate the performance gain of a DWT Lifting scheme in combination with a La-Grange Polynomial interpolation, the data from the training session are used. The potential advantage of the DWT Lifting scheme + La-Grange Polynomial interpolation is the hopefully better symbolic representation of the data set (e.g. lower entropy), compared to the normally used differential predictor. The original algorithm from [12], is explained in Appendix D.3.1 and recommend to read.

By taking an easy and intuitively to use method, the probability distribution functions of the signals coming from the differential predictor and the DWT Lifting scheme + La-Grange Polynomial interpolation are compared. It is already observed, that a wider graph, will result in a lower compression ratio and a more steep one, in a higher compression ratio. If the La-Grange Polynomial Extrapolation is only used, very poor performances are obtained, compared to the original differential predictor in Figure 4.14b. If the DWT Lifting scheme is used, the performances are almost in the range of the differential predictor, shown in Figure 4.14a. However, a numerical calculation, using the entropy limit, will show that the differential predictor is better than the DWT Lifting scheme. Finally, if the DWT Lifting scheme is combined with the La-Grange Polynomial Interpolation, it also has bad performances, see Figure 4.14c. Note, that the DWT-Lifting scheme uses a La-Grange Polynomial Interpolation, and not the La-Grange Polynomial Extrapolation.

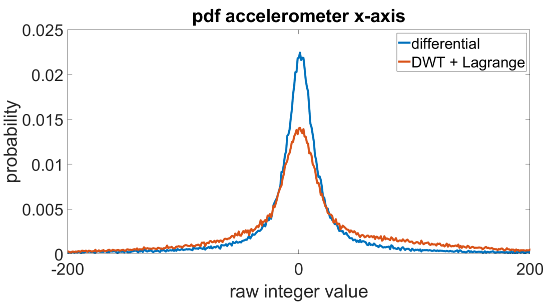
In conclusion, the use of a DWT Lifting scheme + La-Grange Polynomial interpolation proposed in [12], will not increase the performances of the MFELACS algorithm.



(a) probability distribution function of differential predictor and DWT Lifting scheme



(b) probability distribution function of differential predictor and La-Grange Extrapolation



(c) probability distribution function of differential predictor and DWT Lifting scheme with La-Grange Interpolation

Figure 4.14: results DWT and La-Grange interpolation and extrapolation

To get a more satisfactory explanation for the shortcoming of this combination. The effect of bad convergence at the sides of the La-Grange Polynomial interpolation is suspected to cause undesired results. This might also explain the bad performances, with the extrapolation. To overcome the bad convergence at the side, the number of points taken by the La-Grange Polynomial interpolation is increased from 4 to 8. Next it is observed, that the interpolation performs well on the first step of the DWT Lifting scheme, but not anymore on the next steps. Based on these observations, the following new algorithm is proposed, to replace the differential predictor in the MFELACS algorithm. The algorithm is shown in Figure 4.16 and will be explained.

The algorithm first starts with taking the differential values between two neighbouring points, just as the differential predictor does, until the seventh data point x(6) is reached. After this, the differential value between every even point is taken, for example the difference between x(8) - x(6) is stored in s(0). When all the error terms d() and s() are saved, the 8-point La-Grange Polynomial interpolation can be used to predict the uneven values, in the range x(7) up to x(117). The first point x(7) is calculated by taking the even values as input for the La-Grange Polynomial interpolation in Equation 4.1, with **y** = [x(0), x(2), x(4), x(6), x(8), x(10), x(12), x(14)]. The value of interest is P(7), which is calculated by summing up the multiplication of $F_j(x)$ and y_j, where $F_j(x)$ is calculated using Equation 4.2. Hereby the notation x should be read with care, because the values x_j are coming from the vector **x**, where **x** = [0, 2, 4, 6, 8, 10, 12, 14], while x is the input scalar value 7.

$$P(x) = \sum_{j=0}^{n} F_j(x) y_j \tag{4.1}$$

$$F_j(x) = \prod_{k=0, k \neq j}^{n} \frac{x - x_k}{x_j - x_k} \tag{4.2}$$

The difference between the predicted value P(7) and x(7), is stored in p(0). If the entropy of the error terms (d(), s(), and p()) is compared with the ones from the differential predictor, it can be seen that there is a significant improvement on the magnetometers, and a slight improvement on the accelerometers and gyroscopes. These results can be seen in Figure 4.15, whereby the blocksize is taken very large to reduce the effect of the sides (d(0) up to d(11)). The total theoretical improvement on the compression ratio is in absolute number 3.0 %, or differential 6.4 %. This extra gain is at the cost of the extra operations introduced by the La-Grange Polynomial interpolation. Also the effect of using an integer La-Grange Polynomial interpolation, compared to a floating point should be considered. In the current method, the La-Grange Polynomial interpolation is allowed to use floating points in the calculation, only the end result is floored (similar to the fast conversion from floating point to integer). However, floating point operations are more demanding, then integer operations.

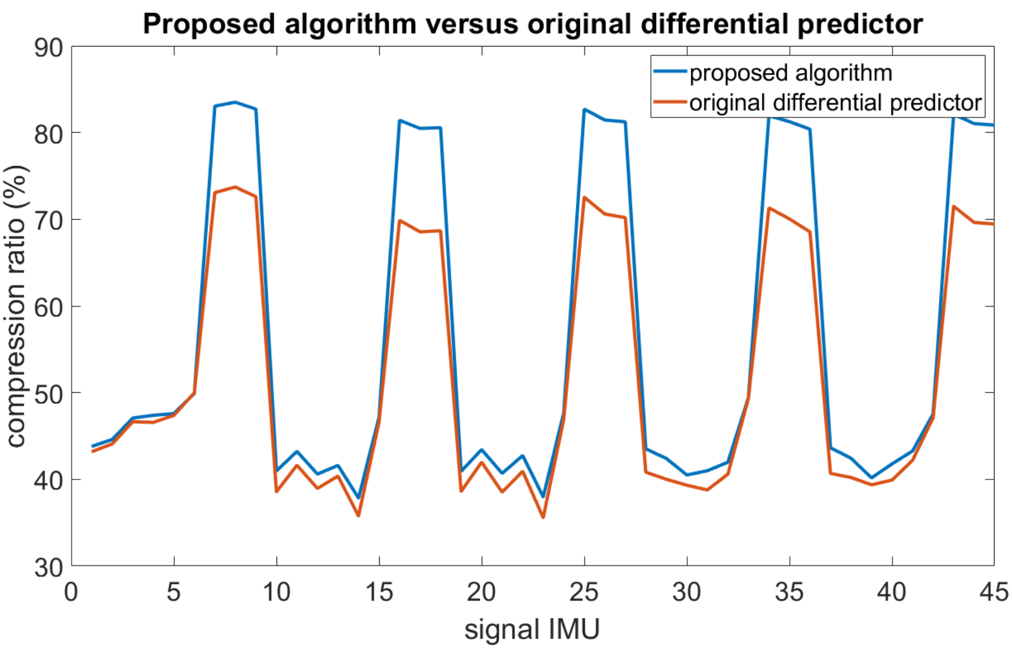


Figure 4.15: Performances of proposed algorithm compared to the original differential predictor, based on a large blocksize. With signal 1-9 coming from the first IMU, (1-3: accelerometer, 4-6: gyroscope and 7-9: magnetometer), repeating for all 5 IMUs

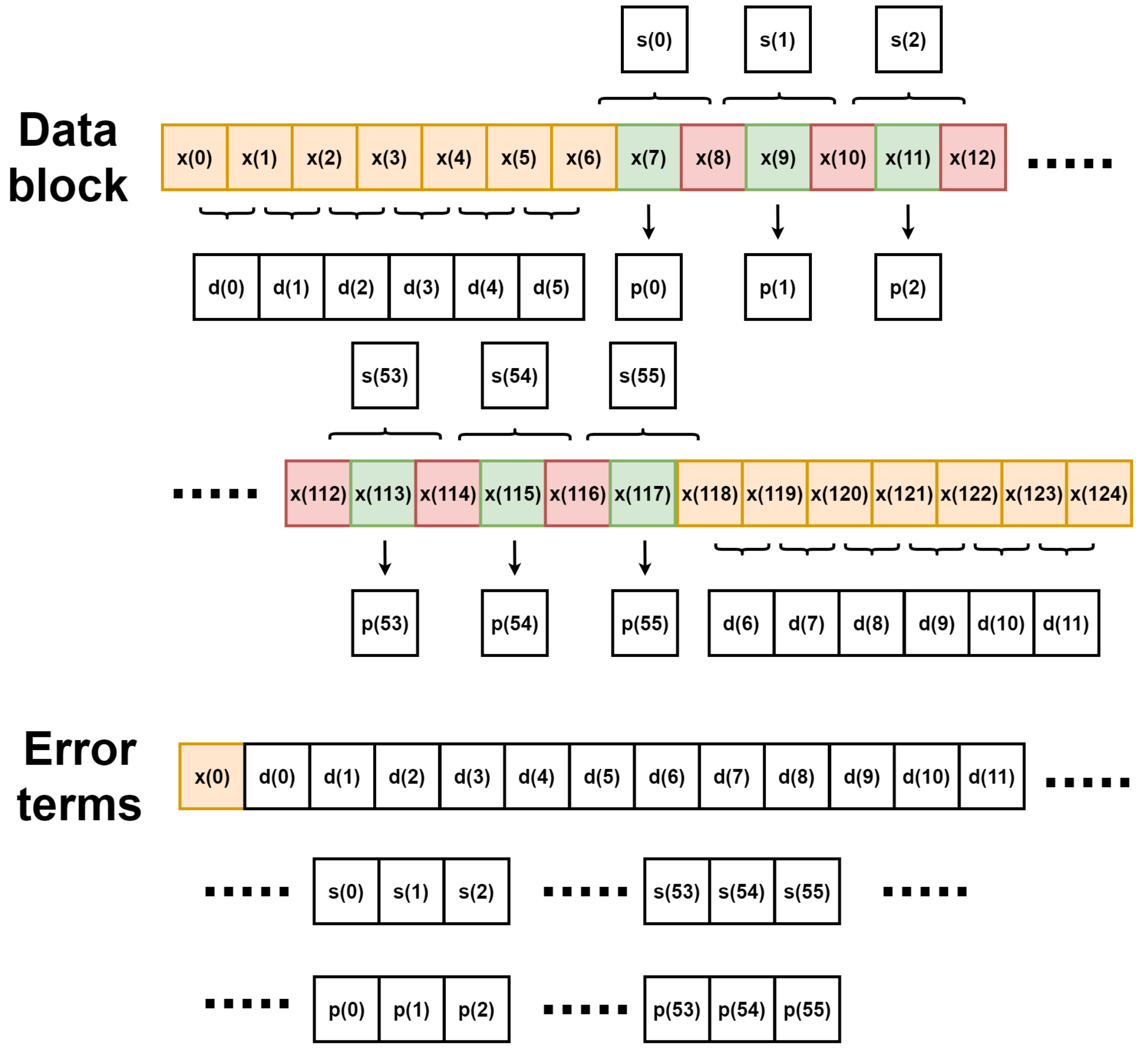


Figure 4.16: Proposed algorithm shown graphically

4.5. Intensity estimation

During the analysis of the compression ratio it was noted, that the compression ratio has a high correlation with the intensity of the movement as shown in Figure 4.17. Even though, more advanced methods might be better off in analyzing data from IMUs. The compression ratio will already be available in the system and might give a simplistic overview of the general intensity during a training. It could also be combined with other models. The reason for high correlation and intensity in the compression ratio, is most likely straightforward. Because, the intensity of a movement is depending on the velocities and accelerations. A higher acceleration and velocity will indicate a more intense movement. Because the data compression algorithm, compresses a smaller data range more efficient than a larger one, these effects will also be visible in the compression ratio. Shortcoming of this method is the possibly lower sensitivity in the more intensive movement. Due the use of a binary representation, only 1-bit is added, when the data range is doubled. Indicating a conversion from an exponential characteristics in the data, into a more linear one in the compression ratio. Another shortcoming, is the classification of 'intensity', being subjective to the definition of the researcher. In this case, the intensity is mainly based on the participants experience of the movement. This could be changed, by looking at the relationship between the speed of the participant versus the compression ratio. Or the maximum duration, each movement can be performed by the participant.

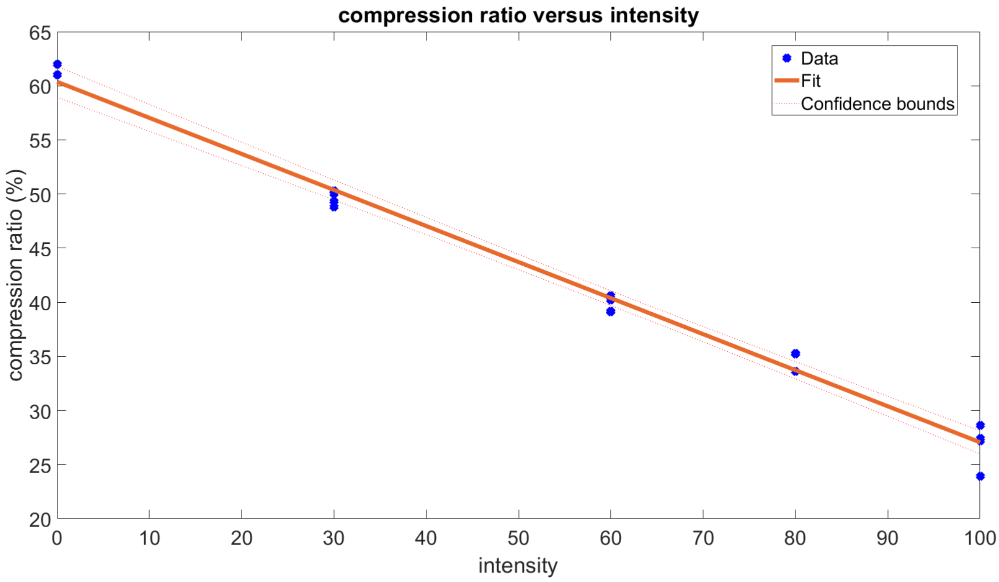


Figure 4.17: Compression ratio versus intensity

5

Conclusion lossless data compression

The following conclusions are made, based on the research questions, stated in Section 2.5. It turns out, that the average compression ratio during the most intensive 5-minutes in a game, is around 43-45 %. With the lowest average compression ratio of 38 % in an interval of 10 seconds. However, the worst case compression ratio for a period of >5 s is observed during sprinting at 23 %, in a more controlled setting.
Next, the algorithm seems to optimally compress data coming from the accelerometers and gyroscopes. Only the magnetometer seem to perform below optimal, due to the noisy character of the differentiated signal. In general, the system compresses the data set near optimal, because the magnetometers will have a lower contribution in the total, while still having good performances. Also the the distribution of s_i, supports the optimality of the algorithm on these data sets.
To continue, the optimal packet size is chosen as 125, whereby the compression ratio seems to saturate, when increasing the blocksize further. This effect is also explained, by the decline of relative overhead on larger packets.
Unfortunately, the use of a DWT Lifting scheme + La-Grange Polynomial did not seem to improve the performance of the MFELACS algorithm. However a new proposed algorithm can theoretically increase the compression ratio in absolute number with 3.0 %, or differential with 6.4 %, at maximum.
Finally, the total power consumption added by the compression algorithm is estimated at 1.6 mA h or 1.6 % in the current system.
After answering the research question, an interesting observation is made between the compression ratio and the intensity, as a linear relationship. However, this relationship is currently questionable, due to the subjective classification. But might be a useful parameter in future work.

6
Discussion lossless data compression

The results obtained using the MFELACS algorithm, are indicating its suited for this application. However, more type of data sets, may increase the reliability of the observed results. Therefore it might be useful during the rest of the project, to also compress data sets from real training sessions. However, this depends on the COVID-19 regulations around football.

The testing and verification of the new proposed algorithm in Section 4.4, might be more interesting from an academic perspective, because it might take a significant amount of time to perform all verification, without the guarantee of getting the extra 3 %. On the other hand, it might turn out that the algorithm might give more improvements in less intensive moments. This is based on the observation that the magnetometers, gained an improvement of 10 %. This high gain on the magnetometer is most likely caused by the slower changing signal, thus indicating the extra potential, when the gyroscope and accelerometer have slower changing signals in the less intensive moments.

Finally, the intensity estimation in Section 4.5 is based on a subjective classification and is presented to the movement scientists as an interesting observation.

7

State of the art dipole and patch antenna in wearables

This chapter will give a brief summary of concept in antennas. Followed by a more in depth explanation of dipole and patch antennas. Continued, by a summary of wearable patch antennas in clothes, and finally explaining the research questions for this part of the Thesis.

7.1. Basics of antennas

An antenna is a device capable of radiating and receiving radio waves. The use of antennas is desired, when no wires can be used for communication. In a basic configuration, a transmitter and receiver have an antenna to transmit and receive a signal, as shown in Figure 7.1.

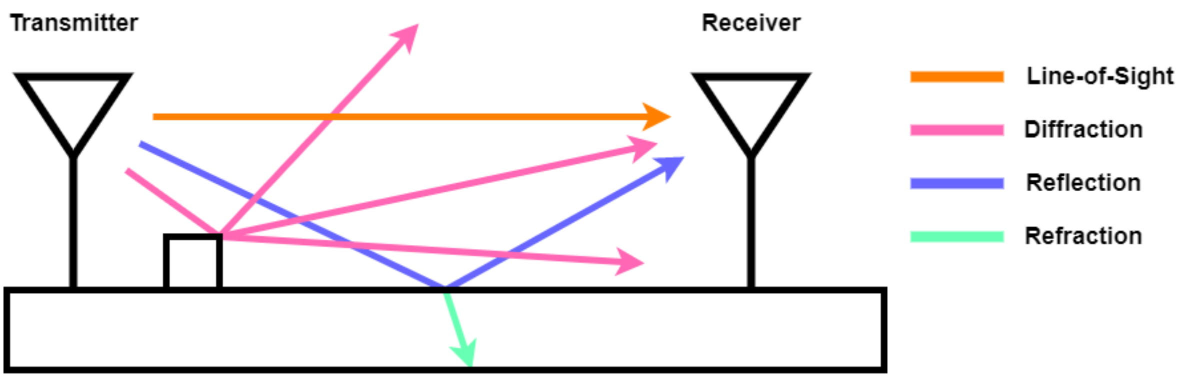


Figure 7.1: basic antenna set-up, using a transmitter and receiver, with: reflection, diffraction, and refraction of signals shown

In an ideal situation, both transmitter and receiver can see each other, without any objects in between, called the line-of-sight (LOS). In practice, objects are all around the antennas, resulting in reflection, refraction and diffraction of the signals, leading to multiple received signals at different time instances. As a consequence, received signals in phase add up and increase the signal strength, while out of phase signals will attenuate the signal leading to dead spots in the communication. To give a better explanation of diffraction, reflection, and refraction, lets first start with diffraction. The effect of diffraction is often occurring, when a wave hits a sharp obstacle and gets diffracted into multiple directions. This can sometimes be beneficial, for example, if an antenna is placed at the top of a mountain trying to communicate with a village lying in the valley region. If there is no LOS, due to rocks, there can still be a connection caused by diffracted signals into the valley. The sharp edges of the rocks, break the wave into different directions, which can be beneficial for the coverage in this case. To continue, reflections are caused by waves hitting a material under a certain angle, and leaving it with the same angle. The amount of waves being reflected by a material, depends on the relative permittivity and conductivity. Materials with a high conductivity are more reflecting, then ones with a lower conductivity. This means for example, that wet ground is more reflective, then dry sand in the dessert. Finally, refraction occurs, when waves travel from one medium to another medium, whereby both media have a different refraction index (η). [35]

The waves mentioned above are electromagnetic waves (EM waves), produced by the alternating current in the antenna. These waves can be described, in the Reactive Near Field Region, Radiative Near Field Region, or Far Field Region, depending on the distance removed from the antenna. For the football application, the Far Field Region will be of main interest and will already be present at 1.6 cm, using a quarter wavelength antenna at 2.4 GHz. This can be seen in [13, Equation 7.1], whereby the Far Field Region depends on the maximum linear dimension of the antenna (D), and the wavelength (λ) of the EM-waves. [13]

$$\text{Far Field Region} > \frac{2D^2}{\lambda} \tag{7.1}$$

In the Far Field Region, the EM waves can be modelled as plane waves, whereby the electric and magnetic field are orthogonal to each other, with the propagation being orthogonal to the electric and magnetic field. The polarization of the antenna is given by the electric field component of these waves, see Figure 7.2. If the electric field is only in one dimension, the antenna is linearly polarized. If it has an electric field component in two dimension, being 90 degrees out of phase with equal magnitude, the antenna is called circular polarized, whereby the rotation determines if the antenna is Right Hand Circularly Polarized (RHCP) or Left Hand Circularly Polarized (LHCP). If one is trying to measure the signal from a RHCP antenna with a linear polarized antenna, there will be a loss of 3 dB on the receiver, because one of the orthogonal components can not be received, due to a polarization mismatch. [13, 19] For the case of a linear polarized antenna as transmitter and receiver, it is important to be aligned along the same direction to have an optimal transfer, and to prevent a mismatch in polarization. If two linear polarized antennas are placed orthogonal to each other no power will be transferred (in practice this is not really the case). [22]

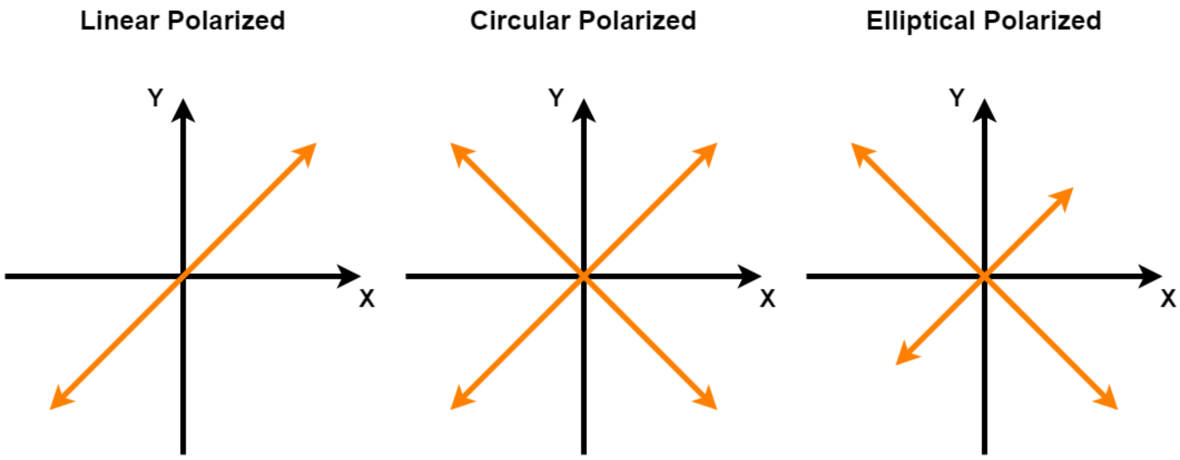


Figure 7.2: antenna polarization, orange line is the E-field

A more advanced configuration to use with antennas, is to introduce multiple transmitters and/or receivers. This is called Multiple In, Multiple Out (MIMO), for multiple transmitters and receivers. There is also Multiple In, Single Out (MISO), or Single In, Multiple Out (SIMO).
The advantage of MIMO is the increase in speed, by splitting the data into parallel streams of data to transmit over the available links. For example, a device has two transmitting antennas, and the receivers has two receiving antennas. The device can now transmit data over both antennas, almost doubling the speed. This mechanism is called spatial multiplexing. Another method is to transmit and receive the signal with multiple antennas to increase robustness and reliability, using Space Time Block Coding (STBC) or maximal ratio combining (MRC). This is also called spatial diversity, and is often used when there is no LOS, and the received signal consists mainly of reflected, diffracted, and refracted

signals. This method is very useful in crowded cities or indoor applications, where there is no LOS and a lot of reflected, diffracted, and refracted signals, from the environment. Another method using multiple antennas, is pattern diversity. This method, combines the radiation patterns of directional antennas to increase the overall radiation pattern, and can overcome weak spots in the connection.
The radiation pattern of an antenna is describing the signal strength, radiated by an antenna, at a certain point in space. In practice, the radiation pattern is often given in the horizontal (azimuth) and vertical (elevation) plane. Using spherical coordinates, the normalized signal strength from each point can be seen. Normally, the signal strength is normalized using the signal strength from an isotropic antenna, which radiates equally in all directions. If the antenna radiates more power in a certain direction, it will have a higher gain compared to the isotropic antenna in that region. This extra gain is at the cost of a lower gain in other regions. This can be seen in Figure 7.3, which has all its power in the main lobe, and hence a high directivity. This is at the cost of a low signal power in the other lobes. Another often used term is the −3 dB-width, which specifies the region between the two point in the main lobe at which the signal power is halved. The lower the −3 dB-width, the higher the directivity of the antenna. [19]

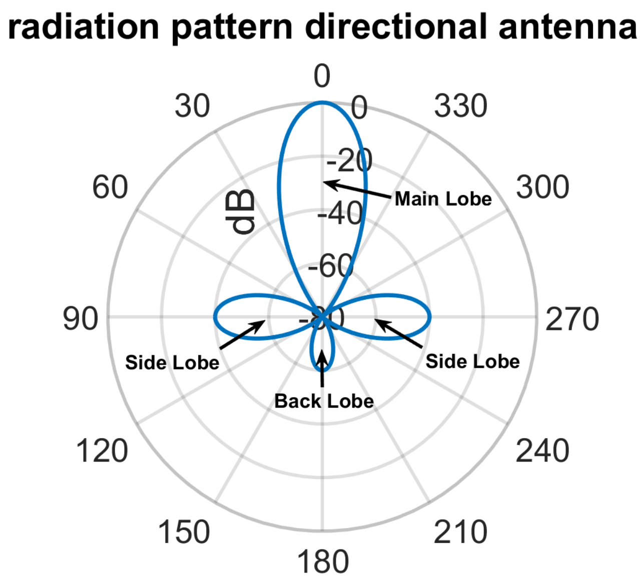


Figure 7.3: radiation pattern directive antenna

The maximum gain at the main lobe, is often referred to as the directivity (D) of an antenna, expressed in dBi. The directivity is calculated using [19, Equation 7.2], with $F(\theta,\phi)$, being the radiation pattern at each point in spherical coordinates, normalized to 1 (0 dB). This formula shows, that a smaller main lobe, will result in a higher directivity.

$$D = \frac{1}{\frac{1}{4\pi}\int_0^{2\pi}\int_0^{\pi} |F(\theta,\phi)|^2 \sin\theta \, d\theta d\phi} \tag{7.2}$$

The radiation pattern of an antenna is the same for transmitting as receiving, this property is called reciprocity. Another property is the front-to-back ratio, which is the ratio between the maximum gain at the main lobe, and the gain in the opposite direction (180 degrees shifted) at the back lobe.
Previously, the use of pattern diversity is discussed, as potential use to improve the radiation pattern, by switching to a better antenna if needed. A simple method to switch between the best antennas is switched combining (SWC), whereby the switching depends on a fixed threshold. This threshold value is important, because a very high threshold can lead to an increase of undesired switching. On the other hand, a low threshold, can decrease the effect of the diversity gain. [115] Another method is selection combining (SC), whereby the best antenna (having the highest SNR) is taken [116]. In practice the channel with the largest signal strength can be selected, by assuming that the noise levels are the same over both channels. The main difference between SWC and SC, is that SWC is only using one antenna to listen to, and switches when it is unreliable, while SC is listening to both antennas, and constantly selects the best one.

7.2. Dipole antenna

A dipole antenna is made of a conducting material, with dimensions capable of producing a standing wave. A common used dipole antenna is the half-wave dipole antenna, which has the length of a half-wavelength ($1/2\lambda$) and can be seen in Figure 7.4.

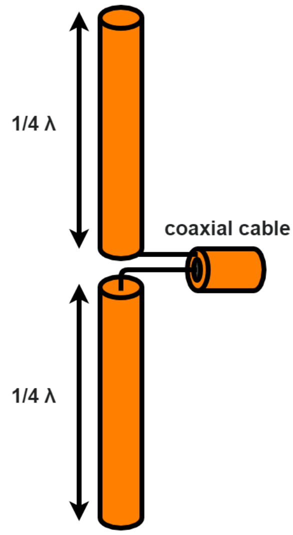


Figure 7.4: half-wave dipole antenna fed by a coaxial cable

The wavelength of the antenna can be calculated using Equation 7.3, with the speed of light c in m/s, and the frequency f in Hz.

$$\lambda = \frac{c}{f} \tag{7.3}$$

The half-wave dipole antenna is fed in the middle using a coaxial feed. In the dipole antenna, the wave pattern of the voltage and current are 90 degrees out of phase, causing the EM-waves to form. To have an optimal power transfer, it is important to have an impedance matching between the load and the antenna. Due to "end effects" and dielectric effects causing a lower propagation speed in the wire, the optimal power transfer is achieved with a length of 0.48λ. The popular use of dipole antennas, is caused by the omnidirectional radiation pattern of the antenna in the elevation plane. This means, that an equal amount of radiation is radiated in the x-y plane of the antenna. The radiation pattern of an antenna is often shown, using an elevation (x-z, vertical) and azimuth (x-y, horizontal) pattern, shown in Figure 7.5, for a vertically placed antenna.

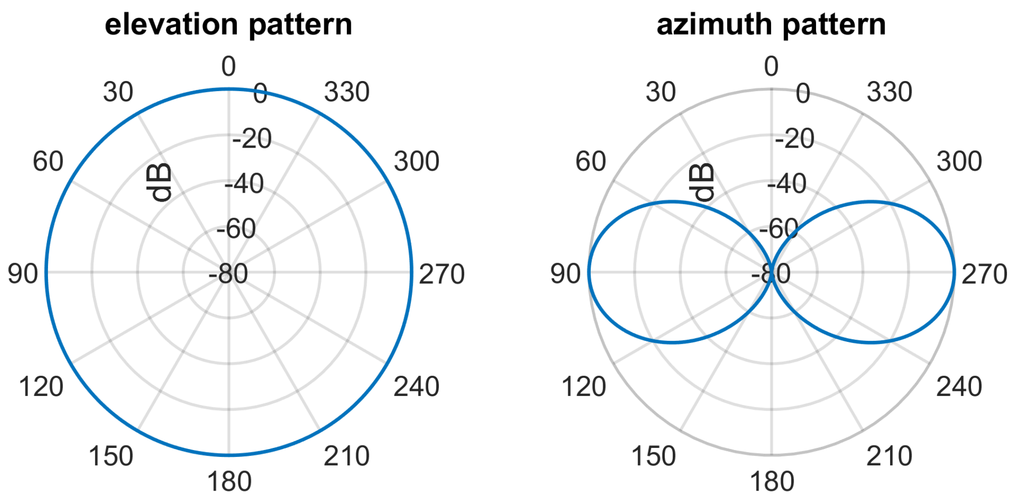
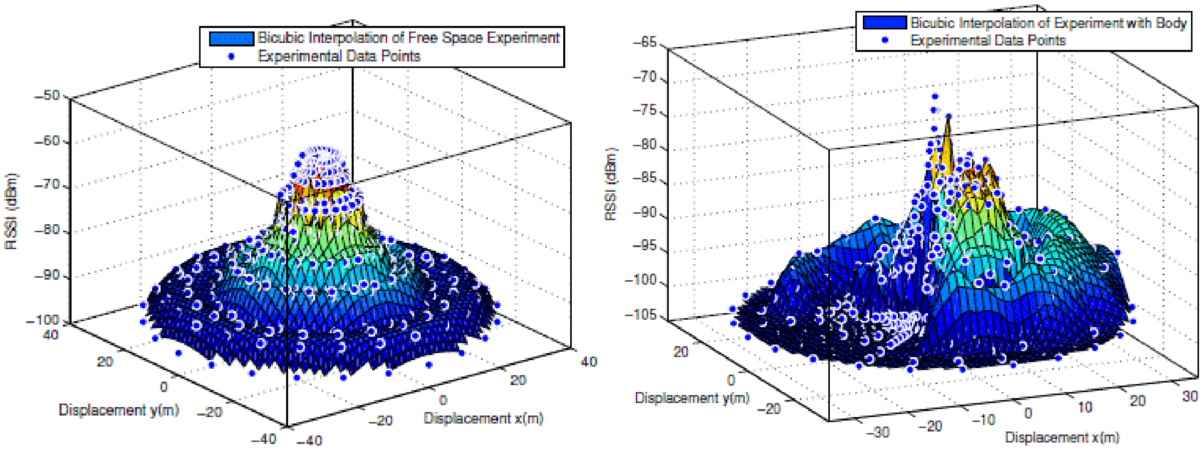


Figure 7.5: radiation pattern of dipole antenna, in azimuth and elevation plane

It is useful to have an equal distribution of the radiation, when the receivers location is not known. This also means, on the other hand, that most of the radiation is lost. When the location of the transmitter and receiver is known, the use of antennas with a higher directivity can be used. For example, a satellite can focus all its power in a certain direction, due to the satellite dish. Another important thing to consider, is the polarization of the antenna. A dipole antenna is linearly polarized, hence transmitter and receiver should be nicely aligned, to have the most optimal transfer. [35]

7.3. Patch antenna

Another type of antenna is the patch antenna, known for its higher directivity in 180°. This property can be beneficial for our application. It is shown in [62] and [43], that the body is absorbing a significant part of the radiation. This can be seen in [62, Figure 7.6a], where the 1/4λ dipole antenna is attached to the right arm of a person and the radiation pattern around the body is shown, the body is located at (x = 0, y = 0). The signal strength is measured in steps of 15° at a fixed distance, this is repeated for different distances, increasing in steps of 1 meter. The measurements are shown with dots, and an interpolated fit is made using these data points. Also baseline is measured, whereby the radiation pattern of the 1/4λ dipole antenna is measured without being attached to the human body in [62, Figure 7.6].



(a) Interpolated surface fit of Human Body from Experimental data [62]

(b) Interpolated surface fit of Free Space Experimental Data [62]

Figure 7.6: Interpolated surface fit of Human Body and Free Space from Experimental data [62]

The absorption of the body is also expected, based on the dielectric properties of the human body. It can be seen, that the human body blocks around 180° of the direction it radiates into. This could better be used in the other direction in which no losses are introduced by the body. The patch antenna is a very well suited candidate for this application. The principle of a patch antenna is to have a directivity only into the positive part of the z-plane. The basic structure of a patch antenna, is a conducting patch in the shape of a rectangle, with a dielectric placed between the conducting ground plate and the patch, as shown in Figure 7.7. In the basic figure, the patch antenna is fed using a microstrip feed line, this could also be done via the following methods: a coaxial cable feed in the middle, aperture coupling, and proximity coupling. In practice, most commercial patch antennas are feed in the middle. This middle is slightly off-centered, to have the optimal impedance matching. [68, 74]

The radiation of the patch antenna is caused by the fringing fields at the side of the antenna, radiating into free-space. The voltage distribution in the length of the patch antenna is the same as in a dipole antenna, and constant over the full width (in the basic model). Because a ground plane is under the patch antenna, no radiation is leaking to the back of the antenna. In practice, this is only possible, with an infinite ground plane. The length of the patch antenna for a resonance frequency f_{01} can be much smaller then that of a dipole antenna, depending on the substrate (ϵ_r), and ratio between the length (L) and width (W), as can be seen in [51, Equation 7.4]. [51] Whereby, L_{eff} describes the effective length of the patch antenna, which is almost similar to the physical length, the same holds for the effective width W_{eff}. The letter c is used to describe the speed of light and $\gamma = \frac{W_{eq}}{L_{eff}}$. Finally, the m and n are indicating, which TM_{mn} mode is used, later explained in more detail. It worth noting, that the term $\frac{c}{f_{mn}}$

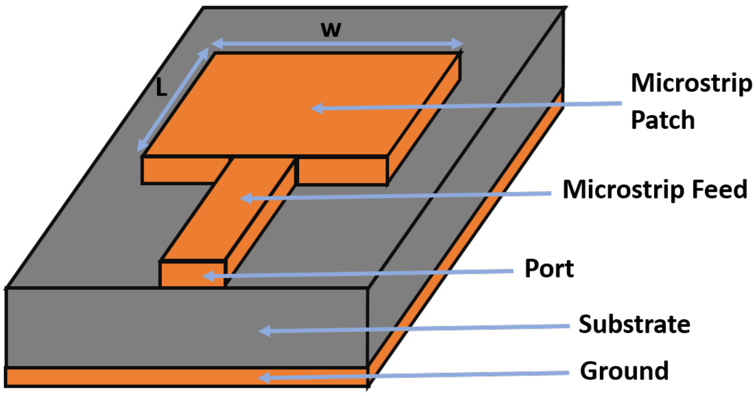


Figure 7.7: Basic patch antenna structure

is the wavelength given in Equation 7.3, but this term is multiplied with $\frac{1}{2\sqrt{\epsilon_r}}$, which leads to a decrease in required length for the patch antenna, when ϵ_r is increased. Also the ratio between the length and width (in γ), can be used to decrease the length at the cost of a higher width.

$$L_{\mathrm{eff}} = \frac{c}{f_{mn}} \frac{1}{2\sqrt{\epsilon_r}} \sqrt{n^2 + \left(\frac{m}{\gamma}\right)^2} \tag{7.4}$$

The radiation characteristics of the patch antenna can be described using three models: 1) Transmission-line model [29, 76], 2) Cavity model [27, 51], and 3) numerical models (based on Maxwell's equations). These models are useful for calculating the dimensions and type of dielectric, to operate in the desired frequency range and to have a desired impedance. They will also provide useful insight, in disturbing factors, influencing the characteristics of the patch antenna, such as the bandwidth and the resonance frequency. It turns out, for patch antennas in clothes, that the influence of the human body, can cause a shift in the resonance frequency. The influence of this shifting can be lowered, by having a broader bandwidth (BW) to operate in. The disadvantage of this, is the lower quality factor (Q-factor) of the antenna, and thus lower efficiency. The Q-factor is given by the ratio between the radiated power and the power in the reactive field [41]. The effect on the bandwidth and the Q-factor can be seen in [28, Equation 7.5], with the voltage standing wave ratio given by ρ. The voltage standing wave ratio (VSWR) is a measure to know the amount of reflected power by the antenna, these reflections can occur, for example, if there is an impedance mismatch. The higher the VSWR, the less efficient the antenna is. [19]

$$\mathrm{BW} = \frac{\rho - 1}{\sqrt{\rho}Q} \tag{7.5}$$

Also the effect on the directivity changes with a wider patch antenna, the wider the antenna, the more narrow the radiation pattern will become (higher D), as can be seen in [28, Equation 7.6]. The term $\left(\frac{w}{\lambda_0}\right)^2$, shows the quadratic relationship, between the directivity and the width. The rest of the parameters are not relevant for this section.

$$D_{01} = \frac{2}{15G_r}\left(\frac{w}{\lambda_0}\right)^2 \left(\frac{\sin(kh)}{kh}\right)^2 \tag{7.6}$$

The cavity model, is an extension on the transmission line model, by taking the higher order TM modes into consideration. The transmission line model only considers the TM_{01} mode, while the cavity model considers the TM_{01}, TM_{10}, TM_{02}, TM_{20}, and so on. The last type of models are numerical models, and mainly interesting for different shapes of patch antennas.
A very well studied phenomena of the patch antenna is the occurrence of surface waves, reducing the overall efficiency. Surface waves are waves that propagate in the transverse direction to the antenna plane. Surface waves are caused by reflected electromagnetic waves, between the dielectric and ground plate. The effect is caused by radiation becoming trapped along the surface of the substrate

[23, 26, 81]. In a substrate the critical angle is given by $\theta_c = \sin^{-1}(\epsilon^{-\frac{1}{2}})$ [23, 26], waves hitting the surface under an angle greater than θ_c will be reflected, and eventually diffracted at the edges of the dielectric material. These reflections cause unwanted radiations and side-lobes and decrease the efficiency of the antenna. A part of the surface waves will be trapped in the material and will lead to losses. The loss of these surface waves increase, when the thickness of the substrate is increased. The effect of the surface waves are deep nulls and ripples in the radiation pattern, an increased back radiation, and gain deterioration [61].
The main advantage of the patch antenna, is the radiation pattern mostly present at the front and not in the back. This will increase the directivity by a factor 3 dB. Another 2 to 3 dB is added by each of the two radiation slots at the sides of the antenna, resulting in a directivity of 8 till 9 dBi.

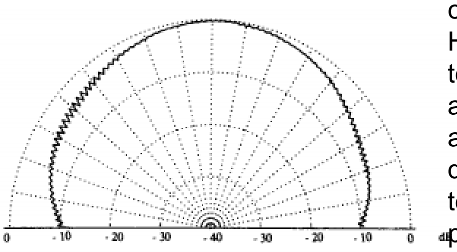


Figure 7.8: patch antenna radiation pattern finite ground plane, based on real-world measurements [107]

A disadvantage of this higher directivity, is the faster decrease in radiation when moving from the front to the back. However, most patch antennas will have a directivity equal to a dipole antenna (which has a directivity of 2 dBi in the azimuth pattern), when 30° removed from the back. A patch antenna with a infinite ground plane would not leak any radiation in the back, and would have all its power radiated to the front, with the highest directivity at 0°. If the ground plane is finite, the patch antenna shown in Figure 7.7, would have the radiation pattern shown in Figure 7.9. It is noticeable, that the radiation in the region 60-90 and 270-300 degrees, has a weaker radiation, compared to an isotropic antenna (the directivity is around 10 dBi). However, in practice, this effect will be much lower as can be seen in real-world measurements in [107, Figure 7.8].

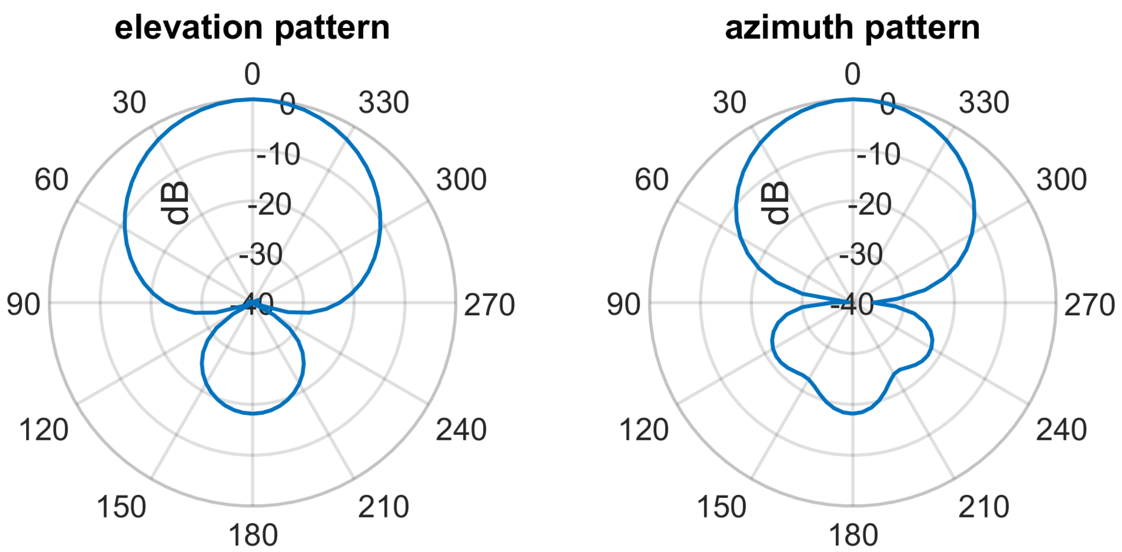


Figure 7.9: radiation pattern of patch antenna, in azimuth and elevation plane, finite ground plane

7.4. Patch antenna in clothes

A widely studied area, is the integration of patch antennas in clothes [46, 47, 54, 72, 85–87, 103, 118]. As mentioned earlier, the radiation in patch antennas is mainly at the front, reducing radiation into the body. This radiation into the body is undesired, due to unnecessary losses and exposure to radiation. Also patch antennas have a structure, which can very well be integrated in clothes. Because the antennas are made by flexible conductive materials, such as copper felt or Zelt, it can have a negligible effect on the wearer. Furthermore, the fabric of clothes can be used as dielectric material, to make the integration more easily. It can also be decided to place the patch antennas on top of the clothes.

Disadvantages of flexible patch antennas, is the impact of bending on the resonance frequency and directivity. It is observed, that the resonance frequency can increase from 2.430 GHz up to 2.550 GHz, with a slight decrease in directivity [87], this is also observed in [118].

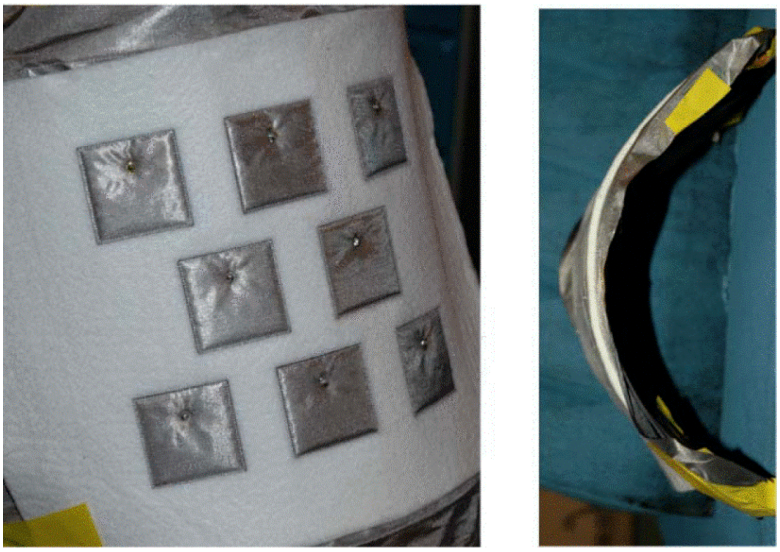


Figure 7.10: Patch antenna integrated in clothes of astronauts [54]

Current applications for these types of antennas are in clothes of astronauts shown in [54, Figure 7.10]. The integration of patch antennas in fire clothes [46] and an antenna in the shape of the Apple logo [72].

To improve efficiency of the antennas, the effect of surface waves can be reduced by having Electronic Band Gap (EBG) surfaces. These surfaces act as a filter on the undesired surface waves. A well known EBG is the Sievenpipers mushroom structure [91] or the split ring resonator (SRR), which can also be added as filter to improve performances of the antenna [10].

7.5. Research questions

It is observed, that the radiation pattern of dipole antennas is highly distorted when placed at the human body. A simple, but effective method would be to use pattern diversity. The main advantage of pattern diversity is to overcome dead spots in the communication link, due to body parts blocking the signal. Also other effects such as fading (caused by refraction, diffraction and reflection) can be solved by a second antenna as form of diversity [50]. The use of pattern diversity over other methods such as MRC or STBC, is due to the high likelihood of having a LOS on at least one of the two antennas, without having to send the signal over both antennas all the time. It should be noted, that the advantage of pattern diversity using switched combining, is mostly found around the switching threshold [93]. This is to be expected, because one of the two antennas can have a to weak signal, resulting in a switch to the other antenna. It is to be expected, in settings using selection combining, that the effect might be observed earlier, because of the constant switching and not the need for a threshold. Next the use of patch antennas in wearables, can reduce the radiation into the body. However, this would be at the cost of a significantly decreased signal strength in a field of at least 180° in the back. This difference can also be noted with dipole antennas. However, dipole antennas will radiate more power into the body, and will therefore have a stronger signal strength in the back, compared to the patch antenna.

To increase robustness in the system, the use of pattern diversity will be investigated. This will help to overcome radiation losses introduced by the body, and the effect of fading (caused by reflection, diffraction and refraction). Next, a patch antenna and dipole antenna will be compared, to see if the lower radiation into the body and higher directivity, can give a significant improvement. The approach taken in this research is more practical, to see if theoretical improvement can also be measured in a more practical setting (as a proof-of-concept). If it turns out that a patch antenna is performing better

than a dipole antenna, it is worth spending time on implementing one from literature. Also the practical effect of pattern diversity can be tested. And finally the performances of the IEEE 802.11n protocol can be seen in the current application, for future research.
This leads to the following research questions for this part of the Thesis:

1. What is the radiation pattern of a single and dual antenna configuration, using patch and dipole antennas?
2. Will pattern diversity overcome the effect of a blocked signal in the dual antenna configuration?

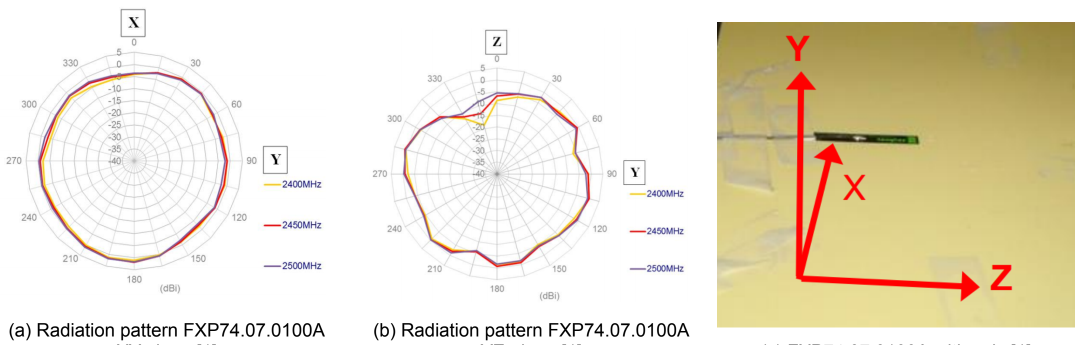
8

Design of antennas in wearable devices

This chapter will deal with design choice made, for designing a proper setup to answer the research questions. First the selection of the antennas is discussed, followed by the WIFI-module and receiver. Next, the integration of the WIFI-module in the current system is explained, with the modification in the pants to fit the antennas.

8.1. Test set-up selection

The most important choice of this part, is the selection of a proper antenna. Nowadays, multiple flexible dipole antennas exist and can directly be placed in wearables, without being obstructive. On the other hand, commercially available patch antennas are still made of non-flexible materials. Because it is observed, that flexible patch antennas in clothes perform almost similar to the non-flexible ones, in terms of radiation pattern and efficiency, it is decided to use non-flexible commercial patch antennas. If patch antennas turn out to perform better, an implementation from literature can be taken. The following flexible dipole antennas from TAOGLAS are considered, because TAOGLAS is known for its expertise in antenna design. The company has a total of 10 different flexible antennas, which are not all suited. After comparing the different flexible antennas, using the data sheets, the FXP74.07.0100A is chosen. This decision is based on the radiation pattern being most equal to an omnidirectional antenna, which could easily be seen by eye. The rest of the antennas had oval looking shapes, big dips, or shapes not even close to being omnidirectional. The radiation pattern of the FXP74.07.0100A is fortunately almost identical to a dipole in the XY-plane shown in [1, Figure 8.1a], and has an efficiency of 50 %, with the size of 47 x 7 x 0.1 mm and weight of 1.2 g. Unfortunately, the orientation in the XY-plane is for practical reasons less desired and therefore the YZ-plane is taken in [1, Figure 8.1b]. This also has an almost isotropic radiation pattern, but has a small dip between 330° and 0°.



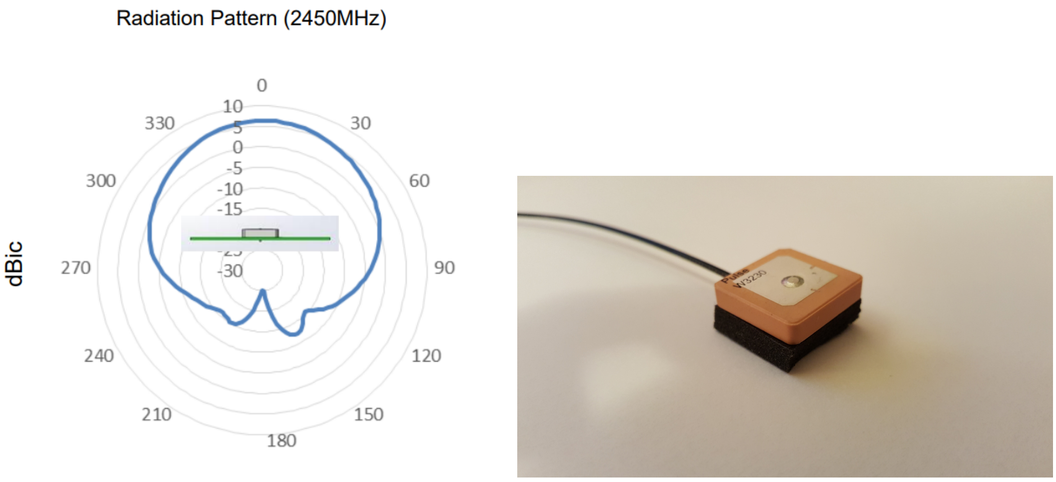
(a) Radiation pattern FXP74.07.0100A XY-plane [1]

(b) Radiation pattern FXP74.07.0100A YZ-plane [1]

(c) FXP74.07.0100A with axis [1]

Figure 8.1: Antenna FXP74.07.0100A

For the patch antenna, the W3230 from PulseLarsen and SXP.25.4.A.08 from Taoglass are considered. Both antennas have a nice radiation pattern and similar efficiencies. However, the antenna from PulseLarsen is chosen as better candidate, with a radiation pattern shown in [2, Figure 8.2a]. The antenna itself is only 18 x 18 x 4 mm, 4.87 g, and 60 % efficient, with the disadvantage of being fed by a coaxial cable slightly off-center, this is solved by adding some foam around it, as can be seen in Figure 8.2b. It is worth noting, that actually the efficiency of the antenna varies from 60 % till 90 % in the frequency band, this might give a different result in received signal strength, depending on the chosen band by the xPico240 module.



(a) Radiation pattern patch antenna W3230 [2]

(b) Patch antenna W3230 with foam around the connector [2]

Figure 8.2: Patch antenna W3230

For the Wi-Fi module, the xPico240 is chosen. This module support the 802.11n protocol, with the ability to use two antennas. The modules transceiver will select the best antenna at any given time, every 300 ms [5]. Unfortunately it not fully clear, how the antenna diversity scheme is implemented. This limitation is accepted, because of the variety of proposed algorithms to choose from, and the expectation to have the highest improvement using two instead of one antenna. It is expected, that the implementation is based on switched combining, due to the selection every 300 ms. Also the module is already adopted in the industry, proving to be a robust product in industrial automation, medical devices, responsive retail and resource management [4]. Moreover, the custom operating system called Lantronix Gateway OS, has an easy to use Software Development Kit (SDK), with a costumer support and forum, to provide support if needed. Based on these advantages, the module is chosen as most suited candidate for fast implementation and testing the hypothesis as a proof-of-concept. Moreover, the module is approximately the size of a coin, being 15 x 25 mm, see [3, Figure 8.3].

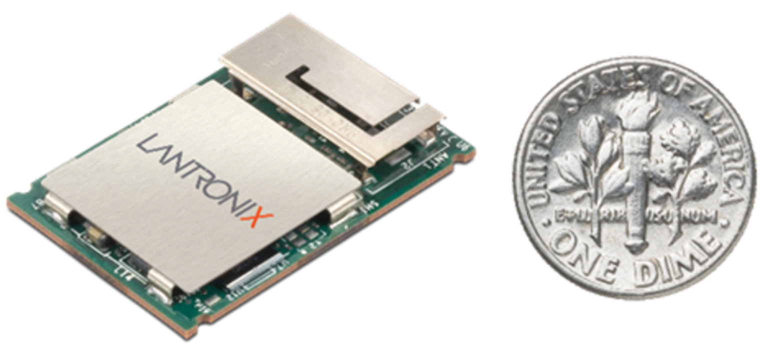


Figure 8.3: xPico240, without external U.FL connectors [3]

For receiving, the EAP-225 is chosen as suitable candidate. The receiver has two dipole antennas attached on the top, which can be replaced by other types of antennas. The advantage of having two antennas at the receiver side, is to have spatial diversity, increasing reliability of the overall system. It also has build in software to monitor: packet- errors, drops, and retries, in a time interval of 5 minutes at minimum. Furthermore, the network activity can be monitored and the RSSI value can be measured from a connected device. The EAP-225 is attached to a wooden structure, to have a minimal interference during measurements, as can be seen in Figure 8.4

Figure 8.4: EAP-225 on wooden structure

9

Testing antenna configuration

This chapter will give an answer to the research questions discussed in Section 7.5. By first describing the experiments being performed, showing the results, and giving a discussion on these results. Finally, an answer is given to the research questions.

9.1. Radiation pattern antennas

This section will discuss the first research question: what is the radiation pattern of a single and dual antenna configuration, using patch and dipole antennas?
First a suited location is chosen to perform the experiment. This location is near a farm, and can be seen in Figure 9.1. The chosen location is almost similar to a football field, because it is an open field, with an equal surface containing soil and grass. It deviates from a football field, due to the water at the side, and the union plants having a lower density then grass. On this location two experiments are performed, the first one measures the radiation pattern of the antennas, and the second one measures the signal strength at different distances.

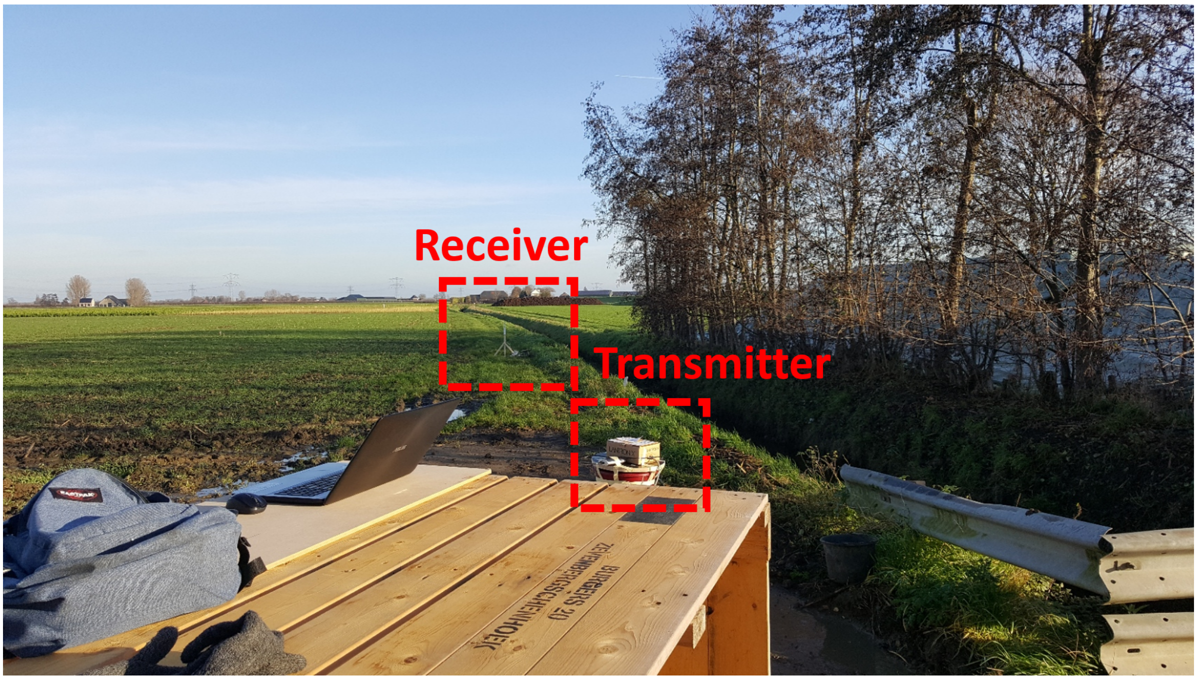
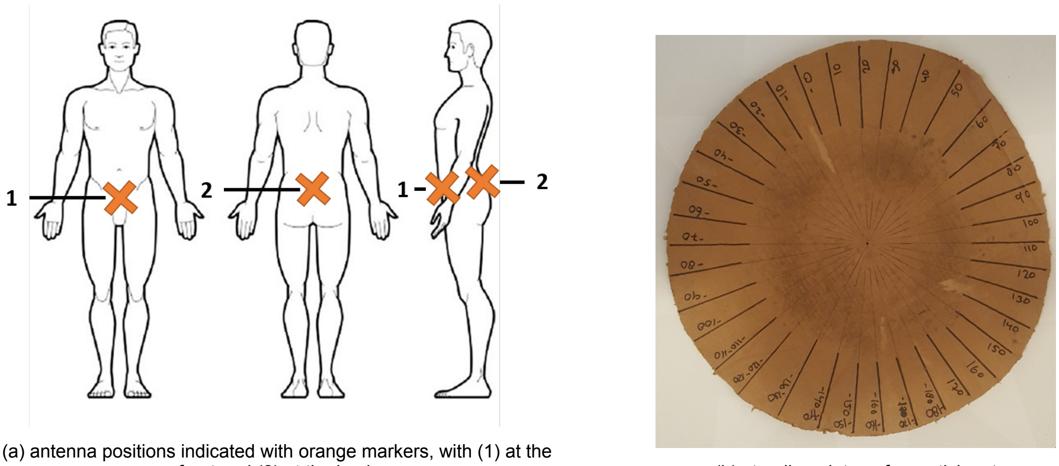


Figure 9.1: test location and set-up

9.1.1. Results measuring radiation pattern

To measure the radiation pattern of the dipole and patch antenna, the antennas are attached to the front and back of the body, at the locations shown in Figure 9.2a. However, due to the cold temperatures at the time of the experiments, one antenna is put on a jacket at the front, and one on jeans at the back. Based on these separate measurements, the radiation pattern for a single and dual antenna configuration can be observed.
To guarantee a relatively fair comparison the following parameters are kept constant: location, day, weather, surroundings and placing at the body. As next step, the polarization between the transmitter and receiver is checked for optimality. For the dipole antenna, a horizontal orientation is taken, because the antenna is placed horizontally on the body. For the patch antennas, a vertical orientation is taken, because this orientation is measured as best performing in the line of sight. From a theoretical perspective, the orientation is irrelevant for a circular polarized antenna, still it is observed, that the vertical orientation is slightly better and therefore chosen. To continue, the weather conditions during the measurements were around 2 °C, with a wind force of 3 (Beaufort-scale), and a wet soil from the last four days of rain. Finally, the variation introduced by the test person is considered, because its hard to fully stand still, and to accurately turn a specific amount of degrees. To compensate for this, the measurement is performed three times for each antenna attached to the back and front. During the measurements, the male participant at the age of 23, length 1.80 m, and weight 77 kg, is standing in the middle of the circle shown in Figure 9.2b, which shows the orientations. To make sure that the test person is nicely aligning with the stripes on the circle, the test person can nicely put his shoes in parallel with these lines. During the experiment, it is decided to make steps of 10° in the clockwise direction, at which three RSSI measurements are taken and averaged. After each circle, a short moment of rest can be taken by the participant to move freely. The average RSSI values after completing a full circle is indicated by 'measurement x', for the rest of this chapter.

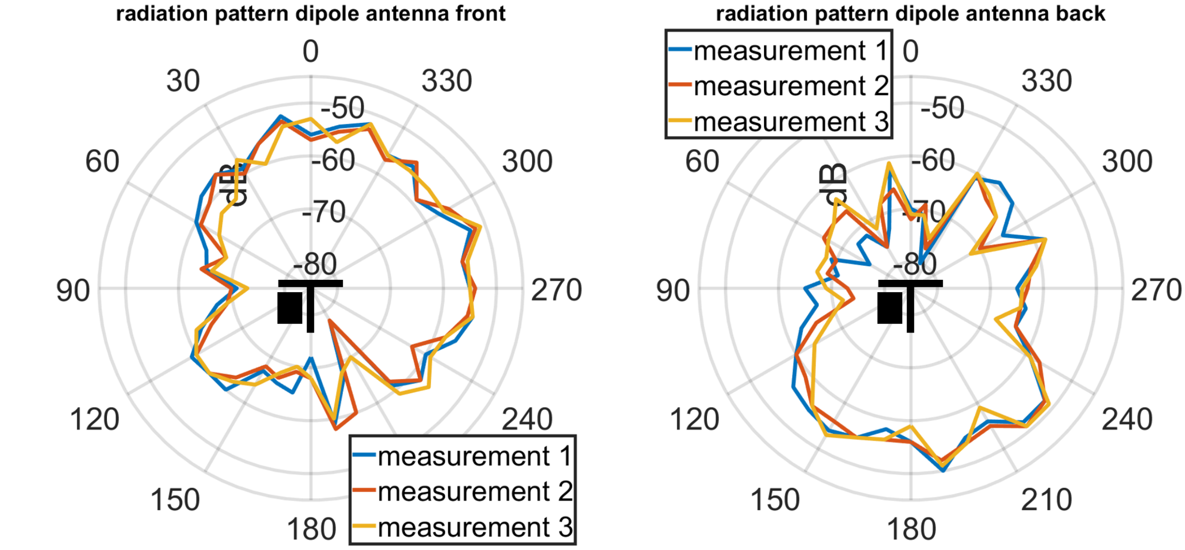


(a) antenna positions indicated with orange markers, with (1) at the front and (2) at the back

(b) standing plateau for participant

Figure 9.2: antenna placement on body, and standing plateau for participant

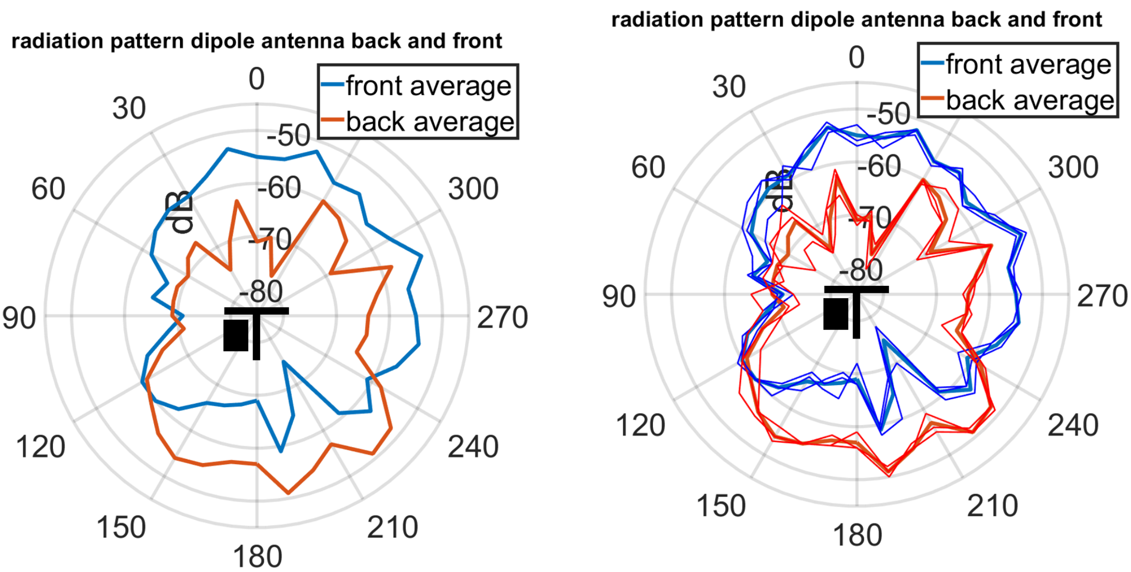
The results of the measurements will be shown and later on discussed in the following order. First the radiation pattern of the dipole antenna is shown, with one placed at the front in Figure 9.3a and one at the back in Figure 9.3b. Next the average value of those three measurements is combined and shown in Figure 9.4a and 9.4b. The same is repeated for the patch antenna, whereby the radiation pattern is measured when the antenna is placed at the front in Figure 9.5a and in the back in Figure 9.5b. Next the average value of those three measurements is combined and shown in Figure 9.6a and 9.6b. As final results, the highest value for the front and back measurement is taken for each orientation step, and plotted against the average value in Figure 9.7a and 9.7b, for the patch and dipole antenna, respectively.



(a) radiation pattern dipole antenna placed on the front, T points to the front at 0°, black box shows the position of the electronics

(b) radiation pattern dipole antenna placed on the back, T points to the front at 0°, black box shows the position of the electronics

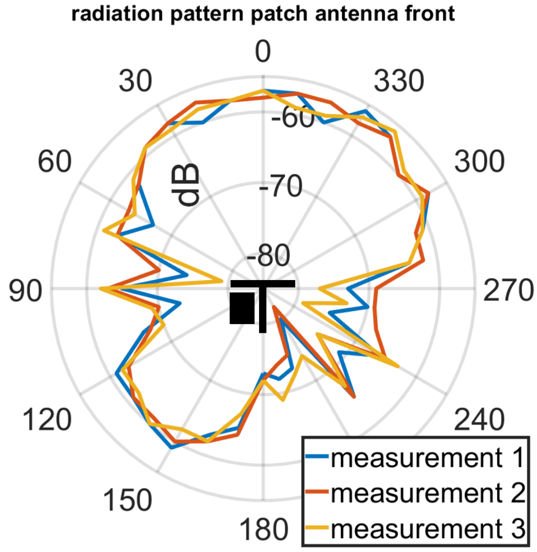
Figure 9.3: radiation pattern, based on three measurements using the dipole antenna, placed on the front and the back



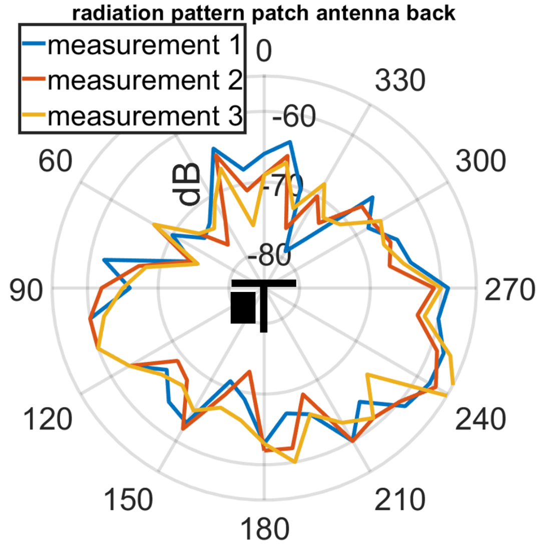
(a) radiation pattern dipole antenna, showing the average value of three measurement, T points to the front at 0°, black box shows the position of the electronics

(b) radiation pattern dipole antenna, showing the average value of three measurement, on top of those three measurements, T points to the front at 0°, black box shows the position of the electronics

Figure 9.4: average radiation pattern in the front and back from the diopole antenna, with (a) showing the average value of the three measurements and (b) showing the average value of the three measurement, on top of the three measurements

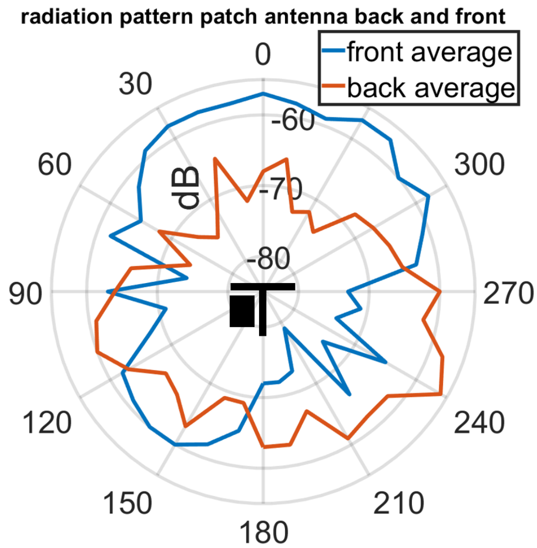


(a) radiation pattern patch antenna placed on the front, T points to the front at 0°, black box shows the position of the electronics

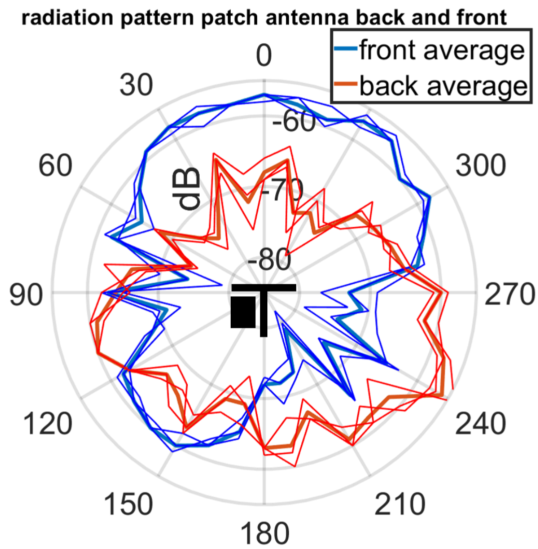


(b) radiation pattern patch antenna placed on the back, T points to the front at 0°, black box shows the position of the electronics

Figure 9.5: radiation pattern, based on three measurements using the patch antenna, placed on the front and the back

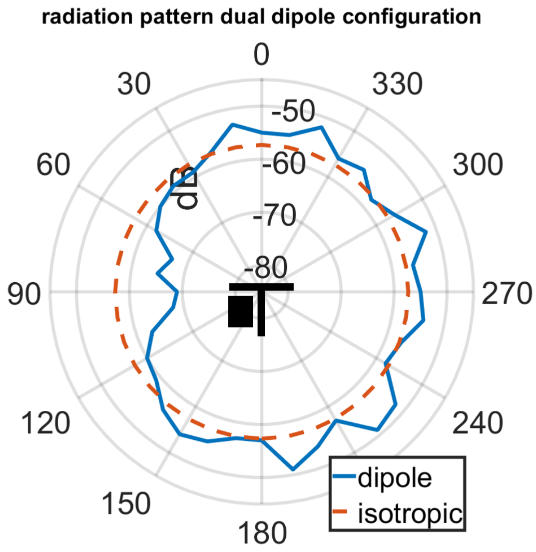


(a) radiation pattern patch antenna, showing the average value of three measurement, T points to the front at 0°, black box shows the position of the electronics

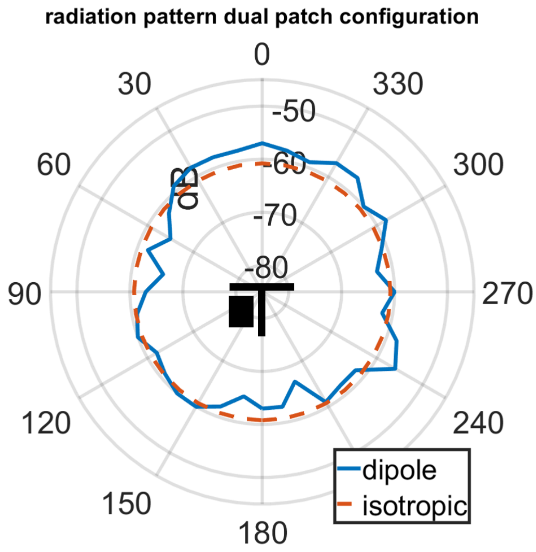


(b) radiation pattern patch antenna, showing the average value of three measurement, on top of those three measurements, T points to the front at 0°, black box shows the position of the electronics

Figure 9.6: average radiation pattern in the front and back from the patch antenna, with (a) showing the average value of the three measurements and (b) showing the average value of the three measurement, on top of the three measurements



(a) ideal radiation pattern dual dipole antenna configuration, with the average value of this radiation pattern, shown as isotropic pattern, T points to the front at 0°, black box shows the position of the electronics



(b) ideal radiation pattern dual patch antenna configuration, with the average value of this radiation pattern, shown as isotropic pattern, T points to the front at 0°, black box shows the position of the electronics

Figure 9.7: best of patch and dipole measurements, with the average value given by the 'isotropic' line

Based on the results from the measurements, a brief discussion will be given on the measured results, beginning with the dipole antennas. In Figure 9.3a and 9.3b, it is visible that the human body is mainly blocking the signal in the back, when the antenna is placed in the front, and vice versa. In Table 9.1, the average difference in gain per region can be seen. It should be made clear, that the classification "front region" and "back region", is used to simplify the text for reading, because the difference in signal strength is observed to be higher for a large number of consecutive steps of 10°, and therefore the word region can be used. The exact location of the regions is given in Table 9.1 and 9.2. It can be seen, that the region in the front is spanning an area of 190°, with an average increase of +9.6 dB, also the back region has an area of 90° in which an average increase of +9.1 dB is observed. These results, show the potential gain of using a dual antenna configuration. The only regions, which are not profiting from the dual antenna set-up are the ones around 90° and 270°. At these points, both antennas will have an almost identical RSSI value, because the body will block the signals equally at this point. More precisely, the radiation will have to go through a part of the belly and the arms.

region	**region width**	**average gain increase**
80°-0° & 250°-360(0)° (front region)	190°	+9.6 dB
130°-240° (back region)	110°	+9.1 dB

Table 9.1: average gain per region dipole antenna

Next the radiation pattern of the patch antenna is measured and shown in Figure 9.5a and 9.5b. It can be observed, that there is an average increase of +10.5 dB in the back, and +10.1 dB in the front, shown in Table 9.2. The front and back region are in this case smaller than the ones from the dipole antenna. Still the front region has a width of 190°, and 110° in the back. The lower back region is mainly caused, by the unexpected side/back lobe shown in Figure 9.5a, which is not seen in the radiation pattern in the datasheet given in [2, Figure 8.2a]. Also the radiation pattern of the antenna placed on the back is deviating from a theoretical perspective, which expects the highest directivity to be at 180°, and not at the sides at 110° and 240°. This effect might be caused by strong reflections at these points, due to surroundings, such as electronics, hands or soil. Still, it is a relatively strong signal and might need some further investigation on the cause.

By combining the best RSSI values from the antennas at the front and back, the combined radiation pattern can be seen in Figure 9.7a and 9.7b, for the patch and dipole antenna, respectively. It is interesting to see that the patch antenna is having an isotropic radiation pattern around -61 dB, with a

region	region width	average gain increase
0°-70° & 280°-360(0)° (front region)	150°	+10.1 dB
180°-270° (back region)	90°	+10.5 dB

Table 9.2: average gain per region patch antenna

standard deviations of 2.7 dB. For the dipole antenna, this is less the case, due to the region around 90°, probably caused by the electronics. Still, the rest of the pattern looks isotropic, with a radius around -57 dB, and a standard deviation of 4.5 dB.
Finally, it is noted that the RSSI values from the dipole antenna are better then the ones from the patch antenna. After measuring the RSSI values at different distances, which will be shown in the next subsection, it is concluded that there might have been a combination of effects leading to the dipole antenna outperforming the patch antenna. These effects, might have been caused by the placement of the antenna on the jacket and jeans, resulting in an angle at which the RSSI is reduced in the vertical plane. This might also explain the behaviour from the patch antenna in the back, because this part of the body has a rounding facing upward. Also the layers from the cotton of the jacket and jeans, result in the antenna being further away from the body, leading to less interference for to the dipole antenna. This extra distance, has a negative effect on the patch antenna, because the body would act as an extension on its ground plane, increasing its radiation to the front.

9.1.2. Results measuring RSSI at different distances

As follow up experiment, it is decided to measure the RSSI at different distances. These measurements are then used to determine a trend-line in the RSSI values using the line-of-sight, to provide a basis for deciding which antenna is most suited for this application. For this experiment, the receiver is moved into the field in steps of 5 meter, at which the RSSI value is measured 5 times. This is performed for two scenarios, one in which the antenna is attached to a peace of wood, and one in which the antenna is attached to a plastic bucket filled with water. The results of the measurements can be seen in Figure 9.8, showing the average RSSI value of the 5 measurements, at each distance.

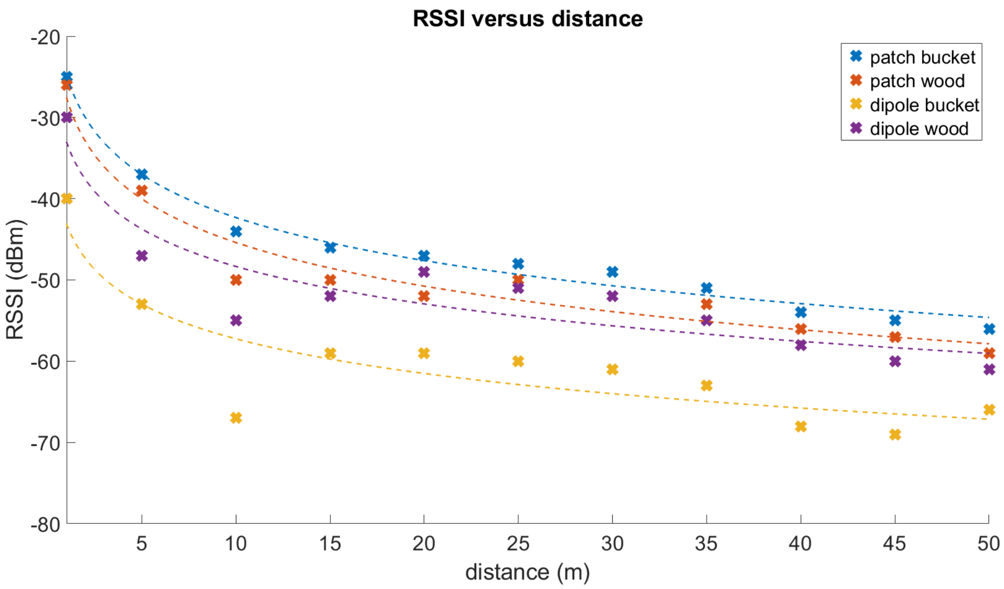


Figure 9.8: RSSI versus distance, with '- - ' being the logarithmic interpolation, and the 'x' showing the average of 5 measurements

As processing tool, the fit function from MATLAB is used to generate a trend-line in the data shown in Figure 9.8, using a logarithmic fit function, shown in Equation 9.1. The shape of this fit function is based on the free-space model, assuming an exponential decay of the signal, whereby 3 dB signal strength is lost every doubling in distance. But due to the effect of reflection, diffraction, and refraction, the decay might be faster or slower.

$$f(x) = a + b*\log(x) \tag{9.1}$$

When looking at Figure 9.8, it can be seen that the patch antenna is outperforming the dipole antenna, for both cases. This can nicely be seen by comparing the trend-lines. The biggest difference occurs, when both are attached to a bucket of water, with a significantly large difference around 15 dB, as can be seen in Figure 9.9, whereby the 'x' is used to represent the differences between two measurements, and the '- -' shows the differences between the logarithmic fit from those two measurements. It is decided to compare the performances of the antennas based on the differences seen in the measured results, and not to much on the differences between the logarithmic fit functions, because it is observed that the logarithmic fit is positively biasing the differences between the patch and dipole antenna. However, the fits will give a nice indication of the decay of the RSSI value, and also confirms that one is outperforming the other.

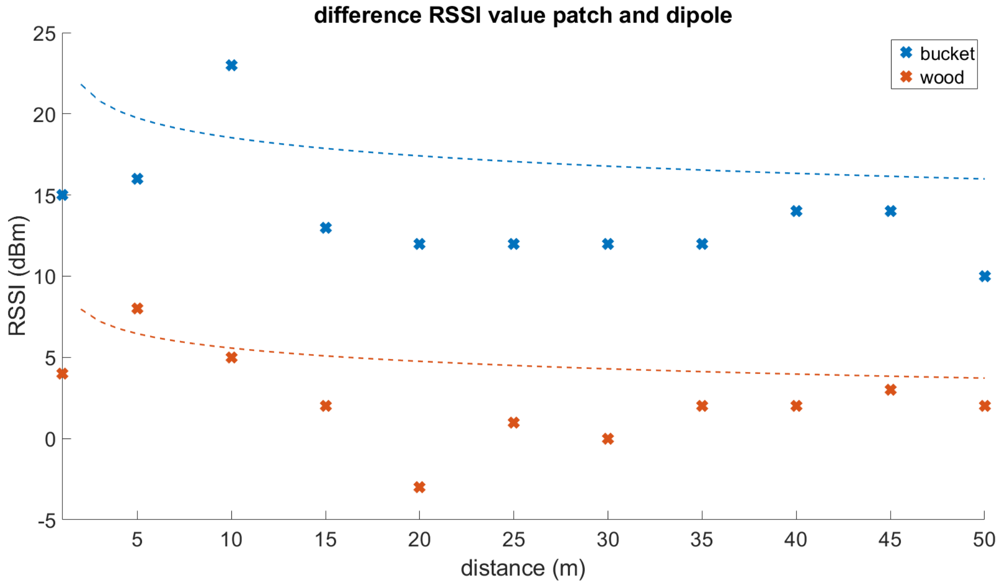


Figure 9.9: difference in RSSI measurement between patch and dipole antenna, whereby the 'x' is the difference taken from the measurements, and '- -' is the difference taken from the logarithmic interpolation

The better performance of the patch antenna is to be expected, because the patch antenna is profiting from the conducting property of the water, extending its ground plane and increasing the radiation to the front. On the other hand, the dipole antenna is suffering from the bucket of water, introducing capacitive effects, shifting the resonance frequency of the antenna, while also introducing a mismatch in impedance. The effect of having a conducting material behind the antenna, is also shown in the data sheet with a metal plate in [1, Appendix E in Figure E.1]. Another factor at play is the long cable length, introducing a lowered efficiency shown in [1, Appendix E in Figure E.2]. On top of that, the radiation pattern around the antenna could be changed, resulting in a weakened field in the measured orientation. When considering all the above mentioned effects, the 10 dB loss in RSSI could be explained.
In practice, the influence of the body is different from a bucket of water, but can give a rough approximation off the conductive properties of the body. When looking at the results in Figure 9.8, it can be concluded that the high sensitivity of the dipole antenna placed nearby a conducting surface, is undesired for the application and can result in a decrease of performance in the communication system. The opposite effect is seen by the patch antenna, which seems to benefit from the conducting properties of the water, by having an increased RSSI of approximately 3 dB to 4 dB. This shows the benefit of using a patch antenna over a dipole antenna. Another important factor to consider is the loss of 3 dB, when receiving a patch antenna with a dipole antenna, due to the mismatch in polarization, slightly reducing the benefits of the patch antenna in this measurements.
In summary, it can be said that an increase of approximately 3 dB, can be seen between the patch and dipole antenna when attached to wood. In the case of higher interference, caused by the bucket filled with water, this difference goes up to around 15 dB. These numbers can theoretically increase with

3 dB, if the patch antenna is received with a circular polarized antenna, instead of a linear polarized antenna. Finally, the differences between the measurements can best be seen in Figure 9.9, whereby it is noticeable that the logarithmic fit is having an offset (e.g. a higher value b, from Equation 9.1) in favour of the patch antennas. Because this fit is slightly biased, the differences discussed between the antennas is based on the measured results.

9.1.3. Two-Ray-Interference model

The variation of the measured RSSI values around the logarithmic fit, are mainly coming from reflection, diffraction, and refraction. To get a better feeling for this effect and to better explain this variation, the following model will be discussed, called the Two-Ray-Interference model [77]. This model is based on the free-space model, which assumes no objects between transmitter and receiver, but adds the strongest reflection from a reflective object to it. This model might give more insight in the current setting, with the wet ground acting as a reflective flat ground plane.
The model starts by describing the length of the LOS path and the length of the reflected path, denoted as d_{los} and d_{ref}, given in Equation 9.2 and 9.3, respectively, whereby the notation is used from [94]. Then it takes the height of the transmitter and receiver into account in h_t and h_r. Based on the d_{los}, d_{ref}, and the wave length, the phase difference (ϕ) can be calculated using Equation 9.4. Next the reflection coefficient (Γ) is calculated in Equation 9.7, using the angle of incident θ_i, and the relative permittivity ϵ_r, and substituting $\sin\theta_i$ and $\cos\theta_i$ using Equation 9.5 and 9.6. Finally the signal power can be calculated using Equation 9.8. For the model itself the following parameters are taken for the main reflection caused by the ground, $h_r = h_t = 1$ m, $\epsilon_r = 25$, and $\lambda = 0.125$ m. For the reflection caused by the ditch, the value $h_r = h_t = 5.3$ m is taken.

$$d_{los} = \sqrt{d^2 + (h_t - h_r)^2} \tag{9.2}$$

$$d_{ref} = \sqrt{d^2 + (h_t + h_r)^2} \tag{9.3}$$

$$\phi = 2\pi \frac{d_{los} - d_{ref}}{\lambda} \tag{9.4}$$

$$\sin\theta_i = \frac{h_t + h_r}{d_{ref}} \tag{9.5}$$

$$\cos\theta_i = \frac{d}{d_{ref}} \tag{9.6}$$

$$\Gamma = \frac{\sin\theta_i - \sqrt{\epsilon_r - \cos^2\theta_i}}{\sin\theta_i + \sqrt{\epsilon_r - \cos^2\theta_i}} \tag{9.7}$$

$$P = 20 \log_{10}\left(4\pi \frac{d}{\lambda} |1 + \Gamma e^{i\phi}|^{-1}\right) \tag{9.8}$$

For the simulation the following two settings are considered. The first situation will only model the reflection caused by the ground plane, resulting in Figure 9.11. The other situation will take the reflection from the ground plane, but also adds on top of this, the reflection from the side of the ditch, resulting in Figure 9.12. The path of the reflections can be seen in Figure 9.10 It can be seen, that the effect of reflections can cause deep dips in the signal strength, and show the potential high influence of the chosen distance to measure. It is to be expected, that small errors made by misplacing the receiver do not cause a consistent benefit for one antenna. But explain the measured RSSI values shown in Figure 9.1 to not follow a logarithmic fit, as would be expected on the free-space model.

9.1.4. Conclusion antenna configuration

Based on the current results, the first research question can be answered:
What is the radiation pattern of a single and dual antenna configuration, using a patch and dipole antenna?
The shape of the radiation pattern for the dual antenna configurations (patch and dipole), can best be seen as an almost isotropic radiation pattern in the horizontal plane. While the single antenna configuration has an almost isotropic radiation pattern in half its plane, and a significantly reduced one in the other half. The potential gain of a dual antenna configuration, is +9.3 dB in a region of 300° for the dual dipole antenna configuration, and +10.3 dB in a region of 240° for the dual patch antenna configuration. As next observation, it is seen that the dipole antenna is more vulnerable for interference caused by conducting materials, while the patch antenna is benefiting from this conducting material as an extension to its ground plane. In case of no interference, the patch antenna performs 3 dB better then the dipole antenna, and 15 dB better when placed on a bucket of water. On top of that, the measured results can theoretically be improved by 3 dB, when a circular polarized receiver would be used. Therefore, it can be concluded, that the use of a patch antenna is desired over a dipole antenna for this application.

Figure 9.10: dominant reflections in current set-up, one via the ground, and one via the ground at the pitch

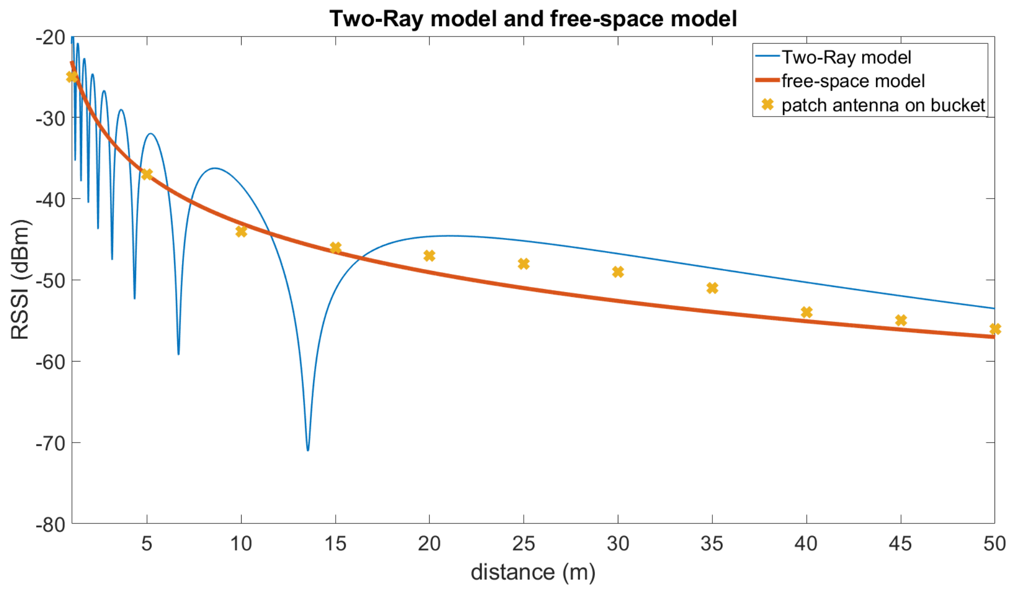


Figure 9.11: Two-Ray-Interference model with one reflection and free-space model

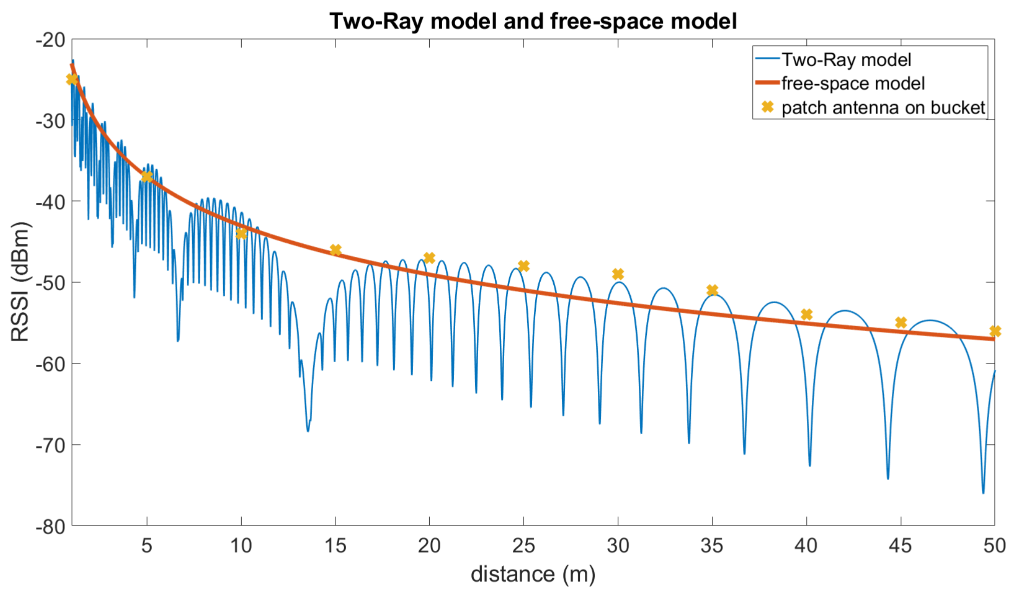
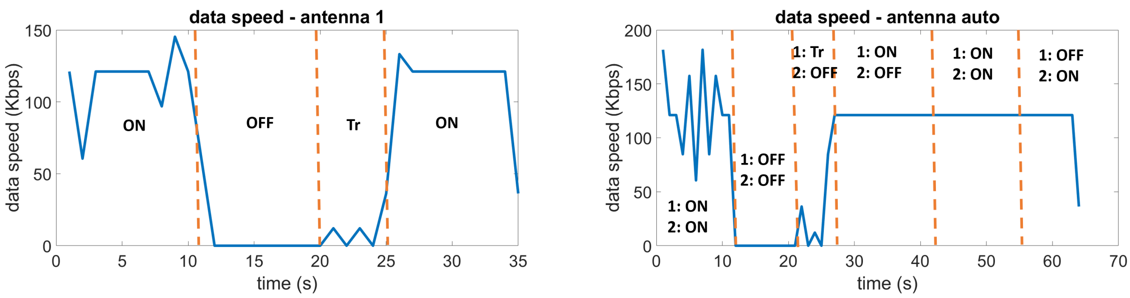


Figure 9.12: Two-Ray-Interference model with two reflections and free-space model

9.2. Pattern diversity using switched combining

This section will answer the second research question: will pattern diversity overcome the effect of a blocked signal in the dual antenna configuration?
Based on the results from the radiation pattern measurements, it is desired to investigate the switching behaviour of the chosen module. This will provide insights in the potential advantages of switching between antennas. First a controlled experiment is performed to verify the switching properties of the module. This is performed by setting up a connection and sending dummy data to the receiver. After 10 seconds, the antenna is disconnected from the module to check the data speed. After another 10 seconds, the module is reconnected. In Figure 9.13a, it can nicely be seen, that the connection drops when the antenna is disconnected and turns back on, when reconnected. As a side note, the region shown with 'Tr', is the interval in which the antenna is reconnected, this takes more time then simply disconnecting, resulting in some small spikes. This experiment confirms that the data rate would drop to zero, when the antenna is disconnected. As follow up experiment, both antennas are connected and the module is set to 'auto', indicating that it will switch to the best antenna. In Figure 9.13b, this switching property can nicely be seen. First both antennas are connected, after 10 seconds both antennas are disconnected, resulting in a dead connection, after another 10 seconds the first antenna is reconnected, and the connection is restored because the module switches to the correct antenna. After another 10 seconds, the second antenna is reconnected, giving the module the choice to either use antenna 1 or 2. If antenna 1 is now disconnected, the module keeps its data speed, indicating it is switched to the second antenna. One potential shortcoming of this experiment is the possibility of already being switched to antenna 2, before antenna 1 is disconnected. However, multiple repetitions of the experiment give the same results, indicating that the switching speed is sufficient enough.



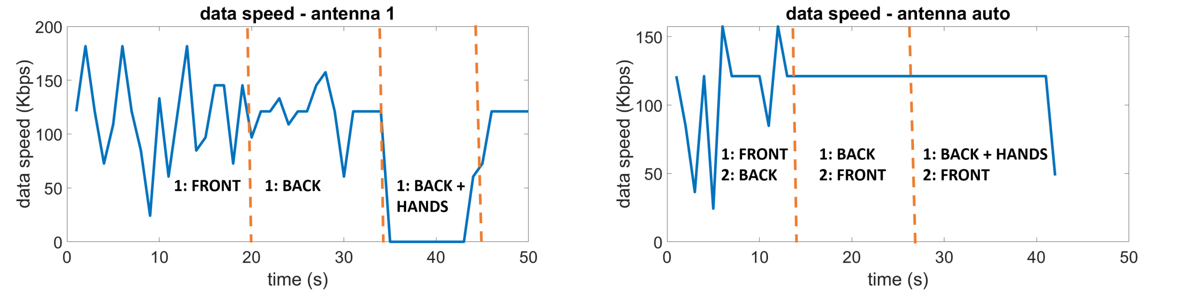
(a) data speed - antenna 1, ON: antenna is connected, OFF: antenna is disconnected, and Tr: antenna is being connected

(b) data speed - antenna auto (1&2), ON: antenna is connected, OFF: antenna is disconnected, and Tr: antenna is being connected

Figure 9.13: testing switching behaviour, by disconnecting antennas

To get a more realistic scenario, the following experiment is conducted to verify the switching behaviour. The transmitter and receiver are placed in such a way, that the connection becomes to weak if a hand is placed before the antenna, shown in Figure 9.14a. The participant is asked to place a hand before the antenna, when the antenna selection is set to 'auto', this can be seen in Figure 9.14b. In this experiment, the participant has a dipole antenna attached to the front and back of the body. Depending on the orientation of the participant, antenna 1 is in the front, or in the back (the body is between transmitter and receiver). It can be seen, that the connection does not drop out, meaning it is nicely switched.

In a follow up experiment, the test person is walking in a straight line with the antenna attached to the front, with the receiver placed towards the back. The test person walks 50 meter forward (F), and after every 10 meter, the person stops and covers the antenna for 10 seconds. At the 50 meter point, the test person turns around and walks back (B), and stops every 10 meter to cover the antenna for 10 seconds. Before the results will be discussed, first a short remark should be given. The xPico240 module has a web-API, which is active during the measurements and will interrupt the system approximately every 5 seconds. Disabling the web-API would be a time consuming process for every experiment, therefore the results have periodic spikes in the measurements. The results from this experiment can be seen in Figure 9.15, whereby it can be seen that the data speed starts to drop to almost zero at a distance of 10 meter, with only one antenna attached to the front of the body. In this orientation the hand and body are blocking the signal, when walking forward. When the person turns around, the effect of the body

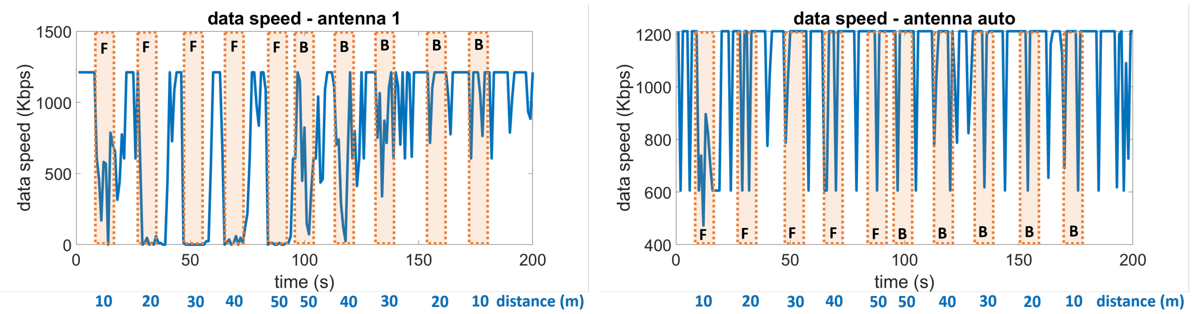


(a) data speed - antenna 1, FRONT: antenna is positioned towards the receiver, and BACK: antenna is positioned opposite to the receiver

(b) data speed - antenna auto, FRONT: antenna is positioned towards the receiver, and BACK: antenna is positioned opposite to the receiver

Figure 9.14: testing switching behaviour, when an obstacle is introduced

disturbing is decreased, only the hand will be the main contributor in blocking the signal. This blocking is visible up to 30 meter, when walking back in Figure 9.15a. When the module is set to selecting the best antenna at any given time, it seems to switch to the correct antenna as would be expected. This can be seen in the range from the axis going from 600 to 1200 Kbps in Figure 9.15b, while the range in Figure 9.15a is from 0 to 1200 Kbps. Based on these measurements, it can be said, that the dual antenna configuration only suffers from the spikes from the web-API and not noticeably from the disturbance introduced by the hand or body.



(a) data speed - antenna auto 1 (at the front), orange boxes are the interval where the hands are placed before the antenna, F = walking forward, B = walking back

(b) data speed - antenna auto (1&2), orange boxes are the interval where the hands are placed before the antenna, F = walking forward, B = walking back

Figure 9.15: measuring speed while walking forward and back, while covering the antenna after every 10 meter

Based on the observations of the switching behaviour from the module, it can be concluded, that antenna blocking can be improved using a dual antenna configuration. This dual antenna configuration is also desired to increase the overall radiation pattern, using two antennas to compensate for weaker spots introduced by the body or other players.

10
Conclusion antenna configuration

Based on the results from the radiation patterns, it is worth using a dual antenna configuration. The increase for the dual dipole configuration is +9.3 dB in a region of 300°, and for the dual patch antenna configuration +10.3 dB in a region of 240°. This leads to an almost isotropic radiation pattern around the player, using pattern diversity. Next, it is measured, that the patch antenna is outperforming the dipole antenna, with 3 dB, when placed on wood, and 15 dB, when placed at a bucket of water. This difference can be increased by 3 dB if a circular polarized receiver is used. Furthermore, these results show the influence of conducting materials on dipole antennas, making them less suited for this application. Moreover, it is observed that the use of a form of switched combining for the dual antenna configuration can be beneficial, when the signal gets blocked by body parts. This leads to the conclusion, that the use of a dual patch antenna configuration, is most suited for this application, among the tested configurations. Finally, it is suspected during the measurements from the radiation patterns, that the placement on the body might be an important factor to consider as well. This involves proper placement on the body, whereby the highest directivity in the vertical plane, is aligned with the receiver.

11 Discussion antenna configuration

The results given in Chapter 10, are to be expected based on theory. Still it is good to mention some shortcomings in the used method, to take into consideration. First of all, the radiation patterns are only measured at a fixed distance, while this pattern can vary at different positions, resulting in other differences in gain at the front and the back region. Next, it is seen that the patch antenna did not outperform the dipole antenna at the fixed distance in the measurements shown in Figure 9.7a and 9.7b. This might have been caused by the rounding of the body, when placed at the back (this is something to consider from a more practical point of view). It could have also been caused by the antenna, not being close to the body, resulting in an advantage for the dipole antenna.
Moreover, the effect of placing the antennas on the exact location on the sensor shorts is not measured, due to the cold weather. By performing this experiment, a better prediction can be given of the influence of the body. Next, the results for the RSSI measurements can be improved by decreasing the step size to 1 meter.
It is also suspected, that the increase in RSSI from the patch antenna placed at a bucket of water, is partially caused, by the decrease in surface waves. This will give an increase in directivity, at the cost of the RSSI at the sides.
Because the current patch antenna is not very well suited for integration, there still needs to be one selected, integrated and tested from literature. However, it is expected that these flexible patch antennas perform similar to the current patch antenna, and its performance can most likely be predicted on measurements shown by the authors, in combination with the measurements from this Thesis. Of course, it is good to be open for other ways to integrate the patch antenna into the sensor shorts.
Finally, the implementation of the antenna switching is not fully known with the current xPico240 module. Even though, it is sufficient to prove the effect of switching or selecting the best antenna, it does limit the assessment of the results. In the previous section, it is decided to use the term switching instead of selection, because the module does only compare the antennas every 300 ms. This 300 ms, is equal to a frequency of 3.33 Hz, which is often seen as a suited switching frequency in switched combining. However, if the module does take the best antenna after every 300 ms, it is more like an implementation from selection combining, because switched combining is most often using a threshold between the antennas, to prevent unnecessary switching. Still the term, switched combining is considered to be the best way of describing it, due to the frequency at which the antennas are compared.

12
Conclusion

The development of smart sensor pants for football players, is essential for measuring the lower body in training sessions and matches, to make more useful models to find risk factors related to injuries. Eventually these smart sensor pants can be used to provide real time feedback, using these models, to prevent injuries. As contribution to this smart sensor pants, this Thesis has focused on researching the reduction in data load, and optimization of the transmission part of the smart sensor pants.

In the first part, the modified FELACS (MFELACS) algorithm is implemented. Next, it is tested on a data set, similar to the most intensive 5-minutes of a football game. The results show an average compression ratio of 43 %-45 %, and a realistic minimum of 38 %, for an interval of 10 s. The performance of the algorithm is shown to be optimal for this application, with a chosen blocksize of 125 samples. Still, there can be a theoretical improvement on the MFELACS algorithm, to improve the compression ratio with 3 %, based on a proposed improvement.

In the second part of the Thesis, the best antenna configuration is researched, as first steps in the design of the smart sensor pants. This involves the comparison between a single and dual antenna configuration, and the comparison between a patch antenna and dipole antenna. Based on these results, it is shown that a patch antenna outperforms a dipole antenna for this application, due to the higher signal strength in the line-of-sight. Furthermore, the dipole antenna loses performance, when placed closely to a conducting surface, while the patch antenna profits from this conducting surface. Finally, the use of a dual antenna configuration is shown to increase reliability, as a form of pattern diversity, resulting in an almost isotropic radiation pattern around the player.

13 Future work

The focus of this Thesis has been on the reduction of the data load, and the optimization of the transmission part of the smart sensor pants. The elements involved in designing this communication system are discussed in the Introduction (Chapter 1) and shown in Figure 1.3. This Figure shows all the elements to take into account, when giving advice for future work in this chapter.

In the first part of this Thesis, it has been shown that the data compression algorithm achieves near optimal performances, however, it is also shown that this can be improved. Theoretically, the performance gain of this improvement is estimated at 3 %, and from a practical perspective not recommend to work on anymore, because there is no certainty in reaching this 3 %. However, from a more academical perspective, it might be an interesting improvement to research, in the field of data compression.

Next, the advantage of patch antennas over dipole antennas is shown in the second part of this Thesis. To make the patch antenna suitable for integration in clothes, a suitable candidate from literature can be selected, implemented and verified.

It could also be decided to take another approach, without the need for flexible patch antennas. This approach can look into spots, where there is not really an obstruction caused by placing the current antennas.

To continue, the current experiments are still lacking results from antennas placed close to the body, as would be the case in the sensor pants. These type of measurements are highly recommended to perform, and can best be done when it is not winter. They could also provide information on the best spot to put the antennas on.

A next thing to consider is the use of a different communication protocols. In this case the IEEE 802.11n is chosen. But, there are more protocols, which can be used. It could also be decided to replace the current WIFI module, with a potential better one (e.g. more suited for this application).

Another, potentially more interesting direction to take, would be the verification of the current set-up on multiple players in the field. These results can then be used to measure, if there are still bottlenecks in the system, or that other parts of the system might need more focus first. These measurements can involve, power consumption, packet errors and retries, time to reconnect, and optimal receiver configuration. Moreover, a first verification can be given on the reliability of the network using the proposed solutions in this Thesis. Based on these results, it can be decided, whether the communication system is sufficient or not. If this turns out to be the case, it might be more interesting to work on other parts of the system.

Finally, it can be considered to replace the receiving antennas at the sideline. It is worth using results from the transmission part, in optimizing the receiving part as well. The new antennas can be circular polarized, to prevent a polarization mismatch. Or more directional, reducing the radiation in the back, increasing the gain in the desired direction.

A

MFELACS C/C++ code

```
#include "MFELACS.h"

int floorLog2(int x)
{
    int ni = 0;
    while(x > 0)
    {
        x = (x >> 1);
        ++ni;
    }
    if(ni > 0)
    {
        --ni;
    }
    return ni;
}

void FELACS (uint8_t* compressed, int* indexComp, std::vector<uint16_t> block,
                int blocksize, int N, std::vector<int>* lengthComprDataColumn)
{
    std::vector<int32_t> diffBlock(blocksize-1);
    std::vector<uint32_t> diffBlockConv(blocksize-1);
    int32_t omega;
    uint16_t k_optimal;
    uint32_t D;
    uint8_t temp;
    uint32_t filledBits;
    int si;
    uint16_t ai;
    uint8_t lengthai;
    int extraBits = 0;

    // store first sample
    compressed[*indexComp] = firstSample;
    ++*indexComp;
    compressed[*indexComp] = (firstSample >> 8);
    ++*indexComp;

    // differential operator
```

```
for(int32_t i = 0 ; i < blocksize - 1 ; ++i)
{
    diffBlock[i] = int32_t(int32_t(block[i + 1]) - int32_t(block[i]));
}

// rice-mapping function
for(int32_t i = 0; i < int32_t(diffBlock.size()) ; ++i)
{
    omega = int32_t(std::min(int32_t(block[i]),
            int32_t((1 << N) - 1 - block[i])));

    if(0 <= diffBlock[i] && diffBlock[i] <= omega)
    {
        diffBlockConv[i] = (diffBlock[i] << 1);
    }
    else if(-omega <= diffBlock[i] && diffBlock[i] < 0)
    {
        if(diffBlock[i] < 0)
        {
            diffBlock[i] = -diffBlock[i];
        }
        diffBlockConv[i] = (uint32_t(diffBlock[i]) << 1) - 1;
    }
    else
    {
        diffBlockConv[i] = omega + uint32_t(abs(diffBlock[i]));
    }

}

// calculate D
D = 0;
for(uint32_t i = 0 ; i < diffBlockConv.size() ; ++i)
{
    D += diffBlockConv[i];
}

// calculate k optimal
if(D < (diffBlockConv.size() << 1))
    k_optimal = 0;
else
{
    for(int k = 1 ; k < 7 ; ++k)
    {
        if((diffBlockConv.size() << k) < D &&
            D <= (diffBlockConv.size() << (k + 1)))
        {
            k_optimal = k;
        }
    }
    if((diffBlockConv.size() << 7) < D)
    {
        k_optimal = 7;
    }
}
```

```
    // save k_optimal with 3 bits
    filledBits = 0;
    temp = k_optimal;
    filledBits += 3;

    // save rest of the data points in structure << si, ai >>
    for(int32_t i = 0 ; i < int32_t(diffBlockConv.size()) ; ++i)
    {
        // log(0) will give an error, thus diffBlockConv should be > 0.
        // next calculate the value of si and store these in compressed
        if(diffBlockConv[i] > 0)
        {
            si = floorLog2(diffBlockConv[i]) - k_optimal + 2;
            if(si < 1)
            {
                si = 1;
            }
            for(int32_t k = 0; k < si ; k++)
            {
                // if si = 3 insert '001', si = 1 insert '1', only '1'
                // has to be inserted, '0' are already initialized
                if(k == (si - 1))
                {
                    temp += (1 << (filledBits % 8));
                }
                filledBits += 1;
                // if filledBits is an equal of 8, it means temp is full
                if(filledBits % 8 == 0)
                {
                    compressed[*indexComp] = temp;
                    ++*indexComp;
                    temp = 0;
                }
            }
        }
        else
        {
            si = 1;
            // si = 1 insert '1'
            temp += (1 << (filledBits % 8));
            filledBits += 1;
            // if filledBits is an equal of 8, it means temp is full
            if(filledBits % 8 == 0)
            {
                compressed[*indexComp] = temp;
                ++*indexComp;
                temp = 0;
            }
        }

        if(si == 1)
        {
            ai = diffBlockConv[i];
        }
```

```
        else
        {
            ai = diffBlockConv[i] - (1 << (k_optimal + si - 2));
        }

        // write ai to compressed
        if(si >= 2 )
        {
            lengthai = k_optimal+si-2;
        }
        else
        {
            lengthai = k_optimal;
        }

        // write residu of bits
        for(uint16_t k = 0 ; k < lengthai ; k++ )
        {
            // check if k's bit of ai is 0 or 1
            if((ai & (1 << k)) != 0)
            {
                temp += (1 << (filledBits % 8));
            }
            filledBits += 1;
            // if filledBits is an equal of 8, it means temp is full
            if(filledBits % 8 == 0)
            {
                compressed[*indexComp] = temp;
                ++*indexComp;
                temp = 0;
            }
        }
    }

    // push last value back if not done and set temp to 0;
    if((filledBits % 8) != 0)
    {
        extraBits += 8 - (filledBits % 8);
        compressed[*indexComp] = temp;
    }
    else
    {
        temp = 0;
    }
    return;
}
```

B
Device report

The device report given below is signed by the AMA advisor Dana Shani-Kadmiel.

Delft University of Technology
INSPECTION REPORT FOR DEVICES TO BE USED IN CONNECTION WITH HUMAN SUBJECT RESEARCH

This report should be completed for every experimental device that is to be used in interaction with humans and that is not CE certified or used in a setting where the CE certification no longer applies[1]. The first part of the report has to be completed by the researcher and/or a responsible technician. Then, the safety officer (AMA – Arbo en milieu adviseur) of the corresponding faculty has to inspect the device and fill in the second part of this form. Please visit https://intranet.tudelft.nl/arbeidsomstandigheden/arbeidsomstandigheden/overzicht-amas/ for more information.

Note that in addition to this, all experiments that involve human subjects have to be approved by the Human Research Ethics Committee of TU Delft. You can find more information on the procedures at http://www.hrec.tudelft.nl/

Device identification (name, location): Movement Measurement system

Configurations inspected[2]: J. Bastemeijer

Type of experiment to be carried out on the device:[3] Movements during physical exercise

Name(s) of applicants(s): Bastiaan Burgers, Annemarijn Steijlen, Andre Bossche, Jeroen Bastemeijer

Job title(s) of applicants(s): MSc student, PhD candidate, Associate Professor, Technician

(Please note that the inspection report should be filled in by a TU Delft employee. In case of a BSc/MSc thesis project, the responsible supervisor has to fill in and sign the inspection report.)

Date: 15-06-2020

Signature(s):

---

1 Modified, altered, used for a purpose not reasonably foreseen in the CE certification

2 If the devices can be used in multiple configurations, otherwise insert NA

3 e.g. driving, flying, VR navigation, physical exercise, ...

Setup summary

Please provide a brief description of the experimental device (functions and components) and the setup in which context it supposed to be used. Please document with pictures where necessary.

More elaborate descriptions should be added as an appendix (see below).

The aim of the measurement system is to measure movements in football exercises. The experiments will be carried out on a football field.

The measurement system
The sensor pants consist of the following parts:

- Commercial tights (figure 1, 2 & 3)
- 5 inertial measurement units with encapsulation (figure 5 & 6)
- a printed circuit board with encapsulation (PCB) (figure 4)
- battery pack (figure 7 & 8)
- a wireless communication module (to be added)

The device contains 5 IMU modules to measure the movements. Data from the IMUs are read out and saved at the main PCB, which contains a microcontroller and an SD card. A wireless communication module will be added in combination with an antenna integrated in the clothes. A power bank (CE marked) is placed on top of the PCB to provide energy.

Figure 1: front side sensor pants

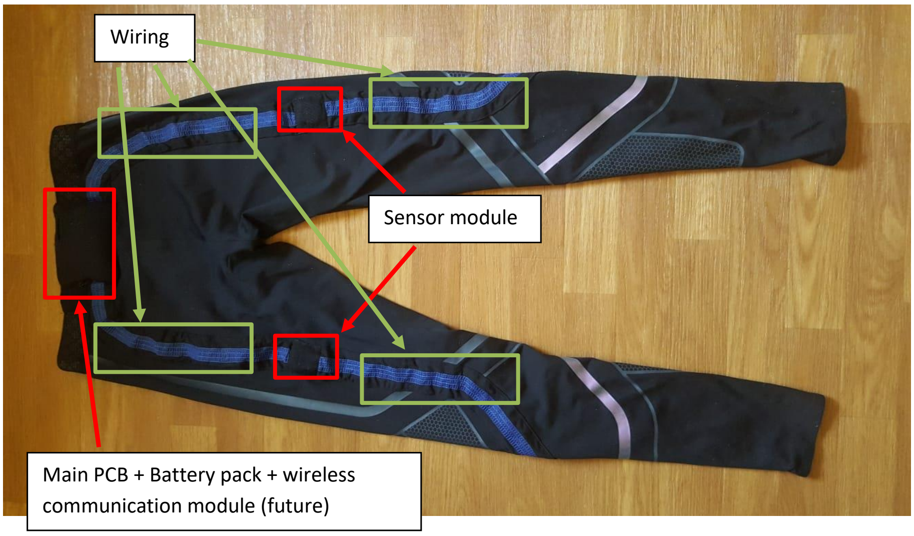


Figure 2: back side sensor pants

Figure 3: Wiring in sensor pants (blue part)

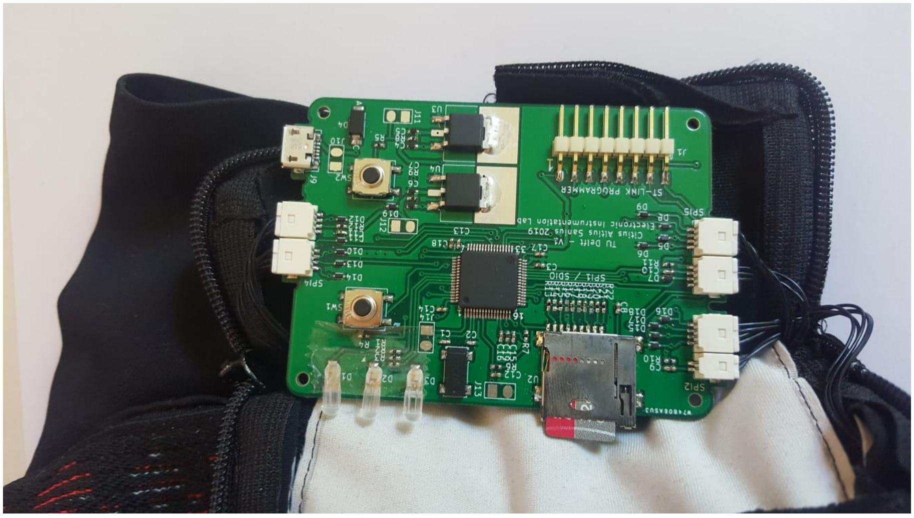


Figure 4: main PCB (when used, it is encapsulated using the same 3D printed encapsulation as can be seen in figure 5)

Figure 5: sensor module with encapsulation

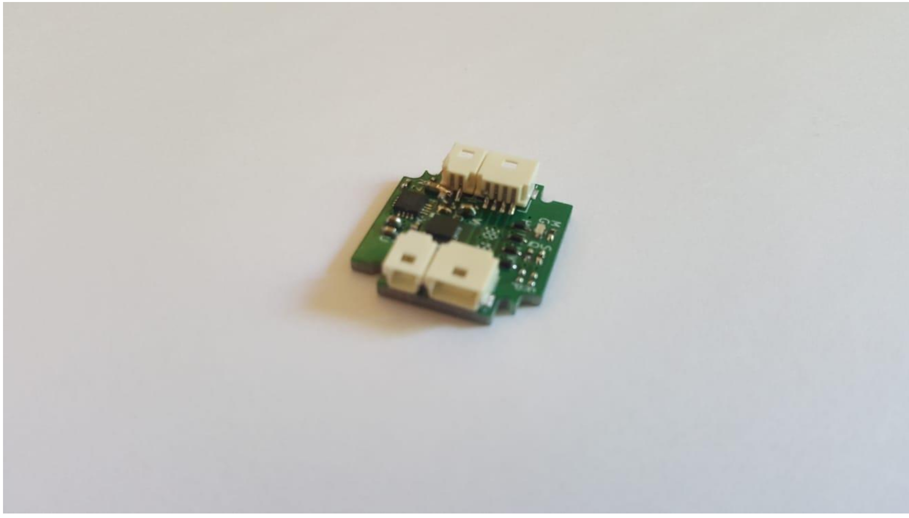


Figure 6: sensor module without encapsulation

Figure 7: back battery (the battery is placed on top of the PCB)

Figure 8: front battery

Risk checklist

Please fill in the following checklist and consider these hazards that are typically present in many research setups. If a hazard is present, please describe how it is dealt with.

Also, mention any other hazards that are present.

Hazard type	Present	Hazard source	Mitigation measures
Mechanical (sharp edges, moving equipment, etc.)	unlikely	Sharp parts of a broken encapsulation might come through the pants and damage the skin.	The encapsulation has rounded corners and protects the wearer against sharp edges from the IMU PCB. It is highly unlikely this hazard will occur. Also, it is more likely the top part of the encapsulation will break an stick out. Not the part closest to the skin.
Electrical	(Very)Unlikely	Failure of the electronic (unlikely) circuit or shorting the battery (very unlikely).	Electronic failure will only lead to loss of data or an unusable device. Shorting the battery leading to a battery exploding or getting fire is very unlikely. A well-protected commercial battery is used. The researcher will be present during the test. The participant and researcher should remove as fast as possible the sensor pants, if there is suspicion of a failing battery. This will most likely be noted, due to the slow build up in heat of the battery, which will most likely be noticed.
Structural failure	Likely	Parts can be become disconnected.	This can only be observed afterwards. But does not influence the participant.
Touch Temperature	Unlikely	Fire caught by the battery	The researcher is always present and will watch the participant. In case of suspicion of a failing battery or smoke. The participant and researcher should remove as fast as possible the sensor pants.
Electromagnetic radiation	Unlikely	High doses of EM-radiation from the antenna.	The radiation will be measured before any participants will be exposed to it.
Ionizing radiation	N.A.		

(Near-)optical radiation (lasers, IR-, UV-, bright visible light sources)	N.A.		
Noise exposure	N.A.		
Materials (flammability, offgassing, etc.)	Unlikely	It can occur that even a certified battery will cause fire.	The researcher is always present and will watch the participant. In case of suspicion of a failing battery or smoke. The participant and researcher should remove as fast as possible the sensor pants.
Chemical processes	N.A.		
Fall/collision risk	Likely	It can occur that a participant falls/collides during the exercises.	It is emphasized to the participant, to do the exercises in their own pace. Also participants doing sports in their spare time are mainly selected and hence better resistant to falling.
Non-contact Injury	unlikely	It can occur that a participant gets an injury during the test	It is emphasized to the participant, to do the exercises in their own pace and that the researcher is most interested in the results from the sensors instead of the sports performance
Other:			
Other:			

Appendices

Here, you may add one or more appendices describing more detailed aspects of your setup or the research procedures.

Device inspection

(to be filled in by the AMA advisor of the corresponding faculty)

Name: Dana Shani-Kadmiel

Faculty: Electrical Engineering Mathematics & Computer Science (EEMC)

The device and its surroundings described above have been inspected. During this inspection I could not detect any extraordinary risks.

(Briefly describe what components have been inspected and to what extent (i.e. visually, mechanical testing, measurements for electrical safety etc.)

The Movement Measurements System was presented and checked by me. The research apparatus is based on a a commercial sport pants (industrial product). The wiring and electrical components are designed to have as little impact as possible on the user during the workout. The risk analysis written by the researcher (pages 7-8 above) covers most of the risks. These risks can be manageable and their rating is low. I would recommend the following additional steps to reduce the risks level even lower:

1) **Lithium battery - the power supply of the system is based on a commercial certified (CE) Lithium battery. It is important to check the integrity (visual inspection) of the battery before each use and also in case of physical impact (hit, fall, etc.). Damaged battery should be replaced.**
2) **Inform users about the risks and how to react in case the battery warms up and/or burns out. It is recommended to prepare a guide/memo for the user (including all relevant risks). Before use, please refer to the user guide (go over the manual with him/her) and practice how to take off the battery or the pants in these cases: overheating and/or smoke/fire are coming out.**
3) **It is recommended that the first tests will be held in a "sterile" environment, meaning to workout as individual (not as part of a group nor in a football match) and in "comfortable" conditions.**

Date: 14-07-2020

Signature:

Inspection valid until[4]: **14-07-2022**

Note: changes to the device or set-up, or use of the device for an experiment type that it was not inspected for require a renewed inspection

4 Indicate validity of the inspection, with a maximum of 3 years

C
More detailed system integration

Before the data can be compressed, the following process in data storing is taken. First the data points coming from the sensors are passed to the buffer. The buffer will then store the data using a ping-ping buffer shown in Figure C.1. The ping-pong buffer is a special form of a circular buffer. The circular buffer can be seen as a circle divided in partitions, with each partition representing a set of data points. The circular buffer works with a read- and write pointer, to manage the data reading and storing. The read pointer, will point to the last read element in the buffer and shift after every read. The write pointer, will point to the last stored data point. To make it more intuitively, imagine the circular buffer as a clock, with every 5 minutes a data point. One of the clock hands will represent the read pointer and the other one will be the write pointer. It can occur that the microprocessor needs to handle other tasks first, the read out of data might be delayed. However, the storing of data continues and may lead to the clock hand (write pointer), overtaking the clock hand for reading out. This will result in the buffer overwriting not yet read out data, or not writing new data.
A circular buffer with only two elements is also called a ping-pong buffer. In this case an element is a larger amount of data points. The read and write pointer can be used, but other methods will work as well. The principle of the ping-pong buffer is relatively easy, one buffer gets filled, while the other is read out. As long as emptying the buffer is faster then filling, no problems will occur. Just as the write-pointer not overtaking the read-pointer. The advantage of the ping-pong buffer is the ease and efficiency of implementing. To begin with, only a few states are required to keep track of filling the buffers as can be seen in Figure C.1. Secondly, the ping-pong buffer will set a realistic deadline for emptying the buffer. If the microcontroller is not capable of reading out and compressing all the data from the ping-pong buffer, it will most likely run at its limits. It can safely be assumed that the filling time is 500 ms, which is a long enough period to spread out variations in the execution times of the different tasks (see Figure 1.3 discussed in Section 1.2.1). The trade-off in this approach, is the potential inefficiency of memory usage. This is solved by clocking the frequency of the microprocessor till the point at which the ping-pong buffer is just emptied in time before it gets filled again. Of course a safety margin is build in, to prevent a buffer overflow. Using this method will then optimize the memory usage.

The data compression algorithm is implemented as a separate task, running at a higher priority then the initTask and storeSDTask. The scheduling scheme is still a highest priority preemptive scheduling scheme as discussed in Section 1.2.1. The updated scheduling scheme, including the data compression task, can be seen in Figure C.2.

To verify the current implementation, the run-time of the different tasks is measured, using the xTaskGetTickCount() function in FreeRTOS. By measuring the start- and end time of the task, the execution time of one job can be calculated. To give an illustration of the basic functions used in FreeRTOS, the code from the implementation of the readAccGyroTask is given below. The first function in the code is vTaskDelayUntil(&xLastWakeTime, xFrequency), which handles the correct execution frequencies of the tasks. The function will periodically create an instance of the task (e.g. a job), which lead to a constant timing between the different activations. This means, that even though the task might not start executing directly, its next activation will be a fixed amount of milliseconds removed from the last activation. This is achieved by the input argument xLastWakeTime, which keeps track of the last activation. The value xFrequency will give the amount of milliseconds, between the activations.

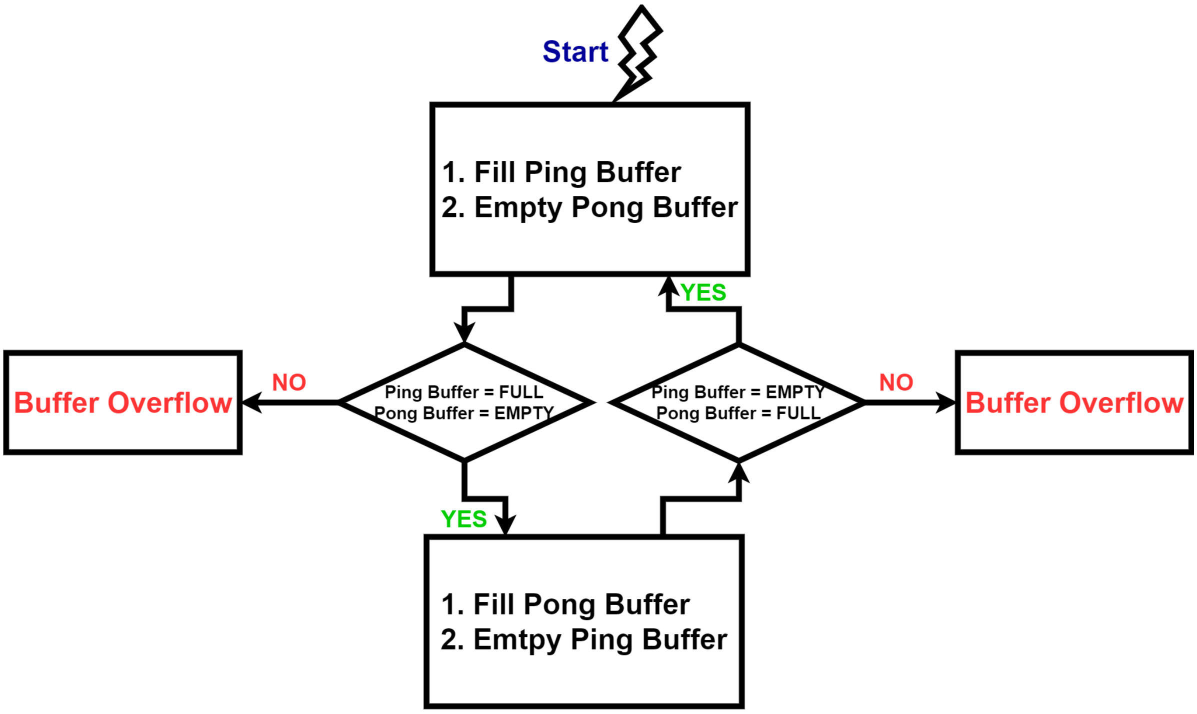


Figure C.1: finite state machine ping-pong buffer

In the code, the xTaskGetTickCount() function is highlighted. By subtracting the end time from the begin time, the differential value, diffAccGyroTask, can be plotted in the Serial Wire Viewer (SWV) Data Trace Timeline Graph from Atollic TrueSTUDIO IDE, shown in Figure C.3. Also the SWV Console can be used to print the values for a more qualitative analysis, shown in Figure C.4.

```
void readAccGyroTask(void const * argument) {
    /* these variables are used to set the frequency of the task.
     * xLastWakeTime: keeps track of when the task is activated for
     *                the last time.
     * xFrequency:    is added to this xLastWakeTime, after which the
     *                task is reactivated and xLastWakeTime is updated.
     * For more information see also the FreeRTOS documentation for
     * this function
     */
    TickType_t xLastWakeTime;
    const TickType_t xFrequency = 4;

    TickType_t begin, end;

    for (;;) {
        vTaskDelayUntil(&xLastWakeTime, xFrequency);
        /* check if startSystem is true, which indicated the SD card
        * is mounted successfully and all sensors are opened
        */
        if (startSystem) {
            /* Read out the acceleration and gyroscope of all 5
             * sensors and finally add a timestamp.
             * The timestamp is used to verify if the task are running on
             * the desired frequency.
             */
            begin = xTaskGetTickCount();
```

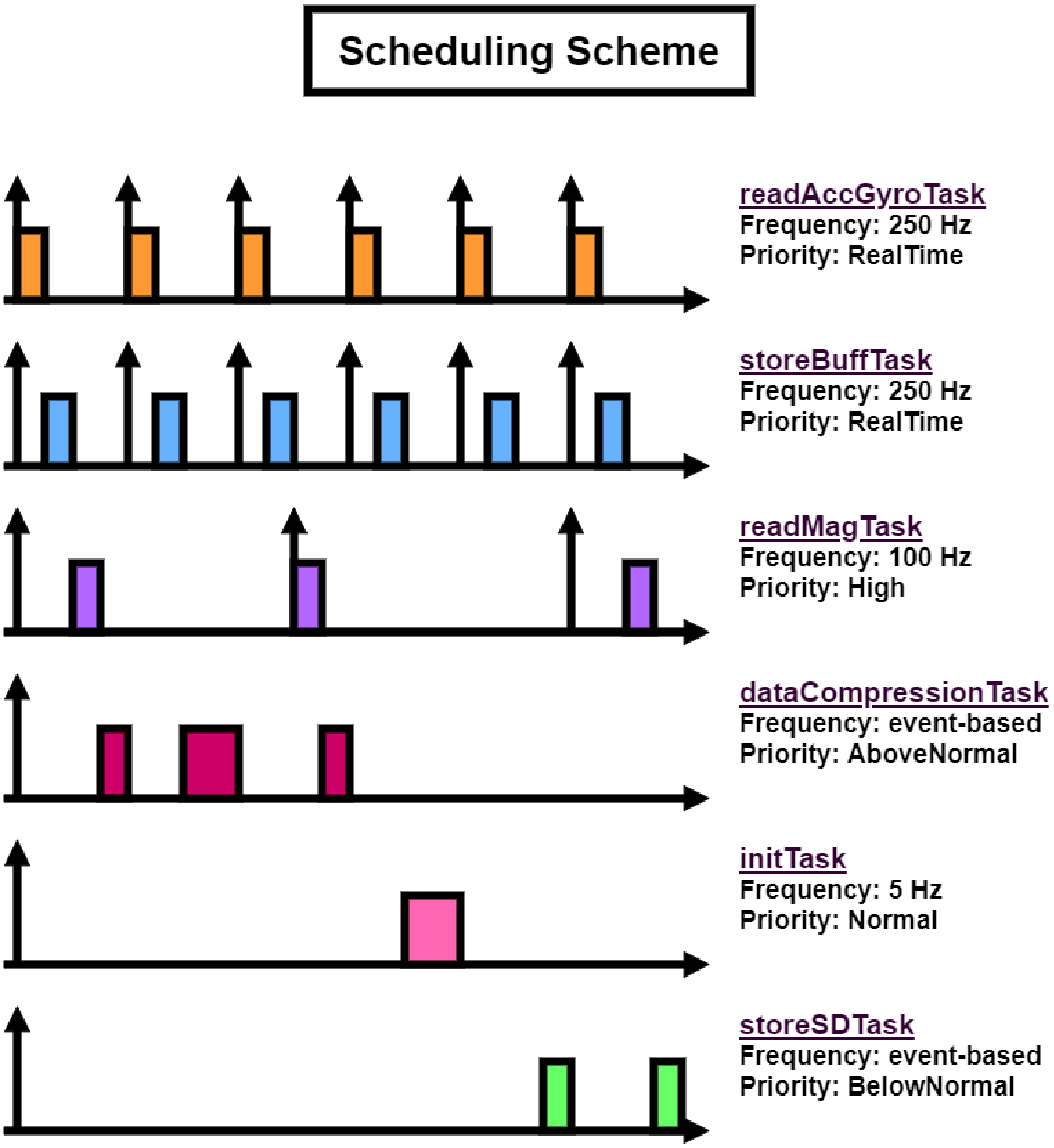


Figure C.2: updated scheduling scheme with data compression

```
sensor_left_A.get_accelerometer(&accGyroData.acc_x_11,
        &accGyroData.acc_y_11, &accGyroData.acc_z_11);
sensor_left_A.get_gyroscope(&accGyroData.gyr_x_11,
        &accGyroData.gyr_y_11, &accGyroData.gyr_z_11);
sensor_left_B.get_accelerometer(&accGyroData.acc_x_12,
        &accGyroData.acc_y_12, &accGyroData.acc_z_12);
sensor_left_B.get_gyroscope(&accGyroData.gyr_x_12,
        &accGyroData.gyr_y_12, &accGyroData.gyr_z_12);
sensor_right_A.get_accelerometer(&accGyroData.acc_x_21,
        &accGyroData.acc_y_21, &accGyroData.acc_z_21);
sensor_right_A.get_gyroscope(&accGyroData.gyr_x_21,
        &accGyroData.gyr_y_21, &accGyroData.gyr_z_21);
sensor_right_B.get_accelerometer(&accGyroData.acc_x_22,
        &accGyroData.acc_y_22, &accGyroData.acc_z_22);
```

```
                sensor_right_B.get_gyroscope(&accGyroData.gyr_x_22,
                        &accGyroData.gyr_y_22, &accGyroData.gyr_z_22);
                sensor_trunk.get_accelerometer(&accGyroData.acc_x_31,
                        &accGyroData.acc_y_31, &accGyroData.acc_z_31);
                sensor_trunk.get_gyroscope(&accGyroData.gyr_x_31,
                        &accGyroData.gyr_y_31, &accGyroData.gyr_z_31);
                accGyroData.timeStamp = xTaskGetTickCount();
                end = xTaskGetTickCount();
                diffAccGyroTask = end - begin;
            }
        }
}
```

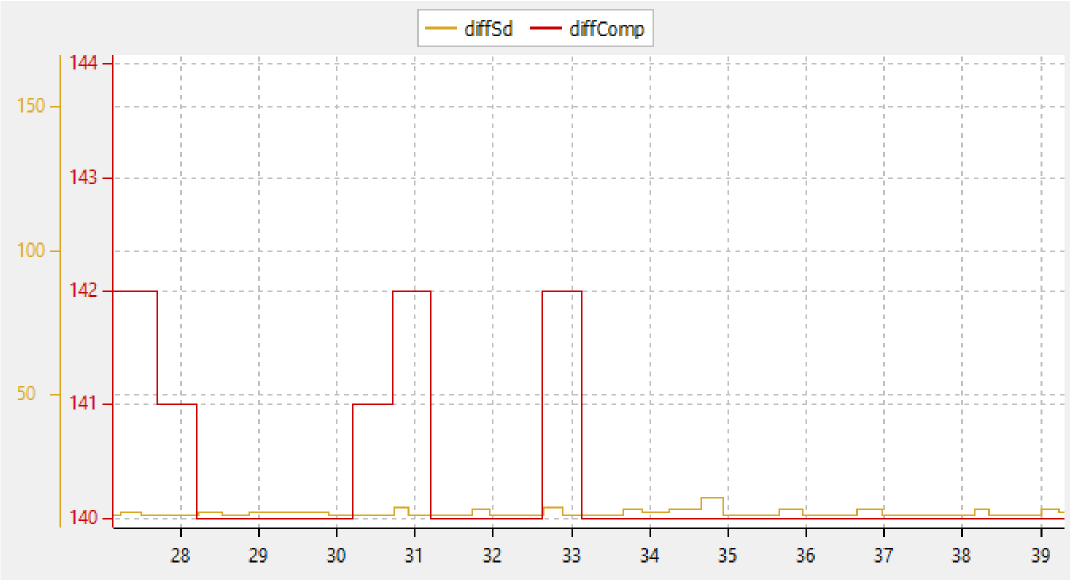


Figure C.3: Atollic TrueStudio SWV Data Trace Timeline Graph

Before showing the results of these measurements, first the shortcomings of this measurement method will be discussed. To start, FreeRTOS only allows the programmer to measure time in the order of milliseconds. This means for example, that a task running at the higher priority, can give execution times of 0 ms or 1 ms. It is very hard to determine the real execution time, because the quantization step is at the same magnitude as the execution time. One could try to solve this, by executing the code inside the task ten times, this will increase the accuracy by using a scaling factor of ten. The timing of the tasks can be seen in Table C.1, whereby the tasks are isolated as much as possible.

However, not all task can be isolated, due to some dependencies on other tasks, this is also indicated by ”other active tasks”. Based on the timing characteristics it can be said that the total execution time will be around 473 ms per second, when combining all tasks. However, due to the preemption of certain tasks, the real execution time will be higher. To measure the overhead of preemption, the following method is used. First all tasks are activated and the total time of the storeSDTask is measured at 9 ms for most of the time. However, some spikes of 140 ms are seen, these are to be expected, based on the scheduling scheme. The dataCompressionTask, is executed within 122 ms, when all tasks are active. This means that the storeSDTask, might be interrupted by the dataCompressionTask, leading to a block of 122 ms, resulting in a peak of 140 ms. The storeSDTask has a relatively low execution rate and execution time, therefore it is hard to determine the overhead introduced by preemption. It is decided to use the dataCompressionTask to estimate the overhead introduced by preemption. It is expected that the dataCompressionTask is approximately 30 times interrupted by the readAccGyroTask, during the 122 ms interval, resulting in a contribution of 30 ms of execution time. To continue, the readMagTask will interrupt 12 times, resulting in a contribution of 9 ms of execution time. This means

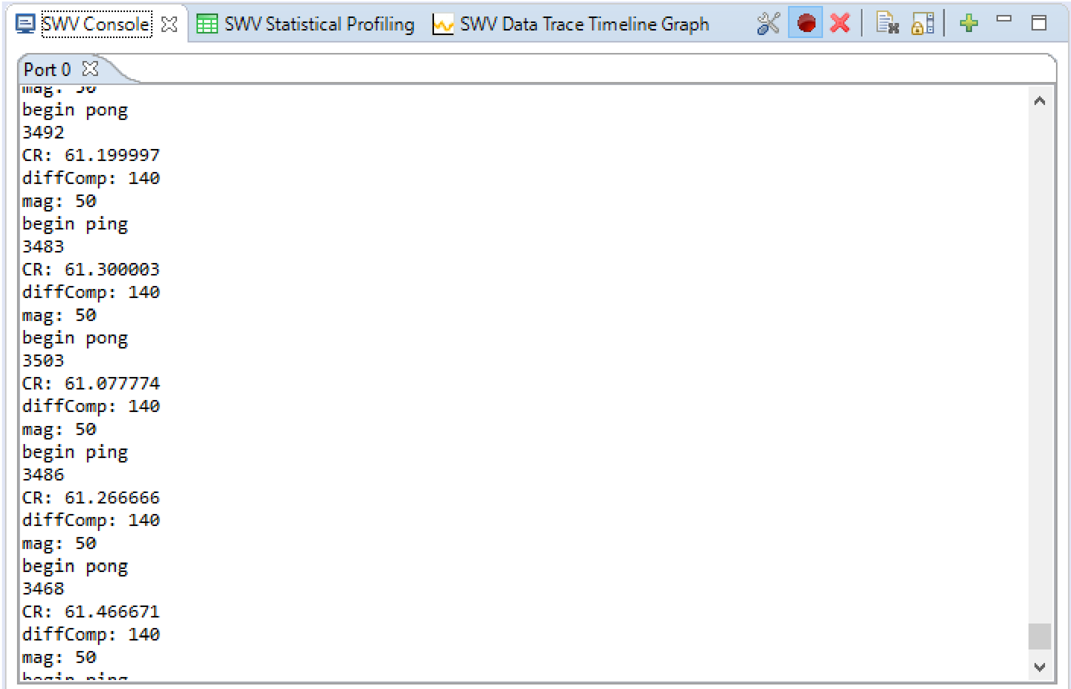


Figure C.4: Atollic TrueStudio SWV Console

Task	**Execution (10x)**	**Execution (1x)**	**other active tasks**	**Task Frequency in hertz**	**Execution time in milliseconds per second**
readAccGyroTask	10	1	initTask	250	250
storeBuffTask	0	0	initTask + storeSDTask	100	0
readMagTask	7	0	initTask	5	70
dataCompressionTask	not necessary	63	initTask + storeBuffTask	2	126
initTask	0	0	–	3	0
storeSDTask	not necessary	9	initTask + storeBuffTask	2	27

Table C.1: timing characteristics of tasks

that the expected amount of execution time, based on the measurements in Table C.1 would result in an execution time of 30 + 9 + 63 = 102 ms. However, the total execution time is 122 ms, this means a total overhead introduced by preemption of 20 ms is introduced, which equals 16 %. Taking this into consideration, the total execution time of all tasks is estimated at 498 ms.
It is noted in a follow up experiment, that the execution time of the dataCompressionTask starts to fluctuate, when the pants is moved. This is to be expected, due to the lower compression ratio and hence an increase in computation cycles. Therefore, the results from above, are reconsidered and adjusted in the following way. The sensor values are read out, but set to zero, this will simulate the same load on the dataCompressionTask, as when the readAccGyroTask and readMagTask would be off (this is also done in the measurements shown in Table C.1). Then the overhead is recalculated, resulting in an overhead of approximately 1 %. Both methods (the wrong and correct one) are given in a more structured way in Table C.2. Based on these results, the extra overhead introduced by preemption is negligible. Also the effect of the debugger increasing the execution time is assumed to be negligible, because it is optimized for these kind of applications, and even used for more time-critical

real-time systems. The execution time of the dataCompressionTask, is measured, with all tasks active. The difference in execution time between the highest and lowest compression ratio is taken as 25 %. Therefore the dataCompressionTask will have a have an execution time of 63 ms and 79 ms. Resulting in a total execution time of 505 ms in a worst case scenario. Taking some extra safety overhead in the execution time, in case of the SD-card having a slightly higher execution time. The total execution time is taken as 555 ms at this point. In summary, the execution time of the whole system can be seen in Figure C.5. It should be pointed out, that the execution times of the tasks, are the highest observed values, this is common practice in real-time systems. On top of this, a small safety margin of 10 % is taken.

method	**execution time dataCompressionTask**	**expected execution time readAccGyroTask**	**expected execution time readMagTask**	**expected execution time initTask**	**expected execution time dataCompressionTask**	**overhead**
real sensor values are taken	122	30	9	0	63	16 %
sensor values are put to zero	93	23	6	0	63	1 %

Table C.2: overhead calculation, wrong and correct method

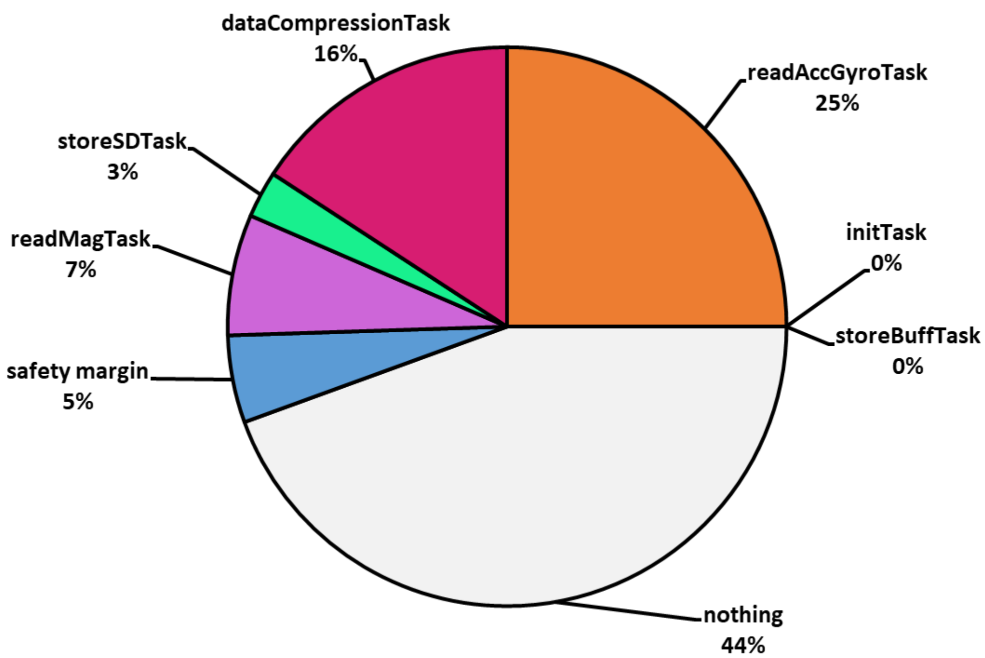


Figure C.5: Execution time per task per second, microcontroller running at 100 MHz

D

Literature Research Lossless Data Compression

This chapter will give an overview of current state of the art lossless data compression algorithms suited for low power embedded devices. First a summary of important parameters will be discussed. Followed by a theoretical background in lossless data compression. Continued by a brief explanation of current state of the art lossless data compression algorithms, whereby important parameters are discussed for each compression algorithm. Finally a summary of the different algorithms is shown and the best suited algorithm will be selected.

D.1. Important parameters

Before lossless data compression will be discussed, first important parameters will be considered. These parameters will be important to select the most suited compression algorithm. This section will discuss the following parameters:

- compression ratio (CR)
- power consumption and computation cycles (CC)
- memory usage
- packet dependency
- ability to adapt to varying source statistics

At the end of this chapter, each parameter will get a score varying from $---$ till $+++$, to make a comparison quantifiable. The requirements for getting a score will be provided in this chapter.

remark

It is observed that a packet loss ratio above 10 % in a network using lossless data compression decreases the efficiency [40]. Therefore the following assumption is made: the link will be reliable enough with almost no packet losses to meet the system requirements. Also the use of lossy compression algorithms is not considered, because the data should not be modified, because the aim of this project is to gather the raw data.

compression ratio

The use of data compression, will most likely reduce the overall power consumption of the system and can result in designing a more efficient wireless protocol. Hence, the compression ratio can be considered as the most important parameter and is calculated using Equation D.1.

$$\text{CR} = 100 \cdot \left(1 - \frac{\text{compressed size}}{\text{uncompressed size}}\right)\% \tag{D.1}$$

To compare different algorithms on compression ratios, the same data set is often used. The use of one data set as benchmark will have the benefit of giving a quantitative comparison between different algorithms. However, as will become clearer further on, it can lead to missing out other aspects of the algorithm. Furthermore, it is observed, that certain algorithms perform similar on one data set, but have a significant different performance on another data set. To compensate for this behaviour, the parameter ”ability to adapt to varying source statistics” is introduced and estimated for each algorithm. The scores obtained for a certain compression ratio can be seen in Table D.1, whereby the term ”max” is used to indicate the maximum compression ratio obtained on one data set, under all algorithms.

criteria	decision
$+++$	max × 95% < CR ≤ max × 100%
$++$	max × 90% < CR ≤ max × 95%
$+$	max × 80% < CR ≤ max × 90%
$-$	max × 60% < CR ≤ max × 80%
$--$	max × 30% < CR ≤ max × 60%
$---$	max × 0% < CR ≤ max × 30%

Table D.1: scores for compression ratio, with max as the highest obtained compression ratio for the same data set

power consumption and computation cycles

The use of lossless data compression in wireless sensor networks can lower the required data rate on the communication channel and lower the energy consumption [78, 82]. It is shown that for saving one byte of data, roughly between four thousand (on a Chipcon CC2420) to two million (on a MaxStream XTend) cycles of computation can be used, leading to the same energy consumption [82]. This means that a lot of computation cycles can be used to save one byte of data, under the condition that the algorithm can finish within the given time limit and processing capacity. However, in [78] it is observed that some of the lossless data compression algorithms do not always lead to better energy gains.
The use of a simplistic model can provide an explanation for the energy saved by compression compared to no compression as can be seen in equation D.2.

$$E_{\text{saved}} = E_{\text{no-comp}} - E_{\text{comp}} \tag{D.2}$$

$$E_{\text{no-comp}} = \text{Data} \times E_{\text{Send-one-Byte}} \tag{D.3}$$

$$E_{\text{comp}} = \text{CR} \times \text{Data} \times E_{\text{Send-one-Byte}} + \text{Data} \times E_{\text{Comp-one-Byte}} \tag{D.4}$$

The calculation of both terms $E_{\text{no-comp}}$ and E_{comp} can show, why it might not always pay off using data compression. First consider the term $E_{\text{no-comp}}$ calculated in Equation D.3. The amount of energy required for transmitting one byte of data ($E_{\text{Send-one-Byte}}$) mainly depends on four things: 1) hardware; 2) distance between sender and receiver; 3) objects/interference between sender and receiver; and 4) communication protocol.

The term $E_{\text{Send-one-Byte}}$ is also used in E_{comp} (Equation D.4), but the total contribution is lowered, due to the compression ratio. However, the effect of this reduced data rate is lowered by the amount of energy required for compressing one byte of data ($E_{\text{Comp-one-Byte}}$), depending on: 1) compression ratio, 2) hardware, and 3) computation cycles.
Therefore, results in exact energy savings based on a simplified model, depend on: hardware, communication protocol, compression ratio, computation cycles, transmission distance, and objects/interference between sender and receiver. The simplified model described above, is a simplification of the real world, but is sufficient enough to show the challenges in exactly determining the power savings of a lossless compression algorithm. Still it is assumed for the sensor pants, that data compression will decrease the overall energy consumption of the system. However, it is difficult to make an estimation at this point. But as discussed earlier in Section 1.2.1, it is observed/assumed to be a significant part in the power consumption of the system. To continue, for sake of simplicity the following will be assumed to compare power savings between compression algorithms:

- If two algorithms have different compression ratios, but almost the same amount of computation cycles and memory usage, the algorithm with the highest compression ratio will most likely have the best power savings.
- If two algorithms have different compression ratios, and different computation cycles and memory usage. The algorithm with the higher compression ratio and significantly higher computation cycles and memory usage, will gain the same score as the algorithm with the lower compression ratio and significantly lower computation cycles and memory usage.

Ideally, relative power savings can be observed when applying different compression algorithms on the same data set using the same platform. However, this is not always the case and sometimes not even studied for all compression algorithms. Therefore an estimation is made on power savings between the different algorithms. As can be seen in Table D.2:

score	criteria
$+++$	CR > maxCR × 95% && CC < minCC × 200%
$++$	CR > maxCR × 90% && CC < minCC × 400%
$+$	CR > maxCR × 80% && CC < minCC × 1000%
$-$	CR > maxCR × 60% && CC < minCC × 2000%
$--$	CR > maxCR × 30% && CC < minCC × 4000%
$---$	CR > maxCR × 0% && CC < minCC × 8000%

Table D.2: scores for power savings. maxCR is the highest obtained compression ratio. minCC is the lowest amount of computation cycles under all algorithms compressing the same data set

Table D.2 is based on the assumptions made above. This means in practice, that two algorithms using approximately the same amount of computation cycles, will have their power savings fully determined by the compression ratio. If the compression ratios are similar, but both algorithms have a significantly different amount of computation cycles, a lower result is given to the algorithm with the higher amount of computation cycles.
Finally an explanation will be given, why the amount of computation cycles is not taken as separate parameter. To begin with, ideally computation cycles are given in terms of million instructions per second (MIPS). This is unfortunately not analyzed and therefore an estimation is made in this Thesis. The estimation can be made based on steps taken in the algorithms and will only be a rough estimation. The amount of operations will determine the execution time of the algorithm on the processor and the power savings due to a lower operating frequency (100 µA/MHz). Because all proposed algorithms are designed for low power embedded systems, the assumption will be made, that all discussed algorithms can be implemented on the current platform. This means, that the algorithm is capable of finishing in time and hence only the advantage of saving power is left, which can be taken into account in the power savings parameter.

memory usage
The term memory usage will be considered separately, because it will not only influence the power savings. But will also be limited by the amount of processes running. In most literature the effect of memory usage from compression algorithm is not considered as a problem, unless certain tree structures or dictionaries have to be stored. In the case of compressing 45 signals coming from five IMUs with 9 DOF, a relatively high amount of data has to be stored in the SRAM. Because new samples have to be stored while the compression algorithm is running, a ping-pong buffer is used, doubling the amount of data to be stored. On top of that, the compressed file has to be stored before sending. This will give a total memory usage of maximal 27 kB. The total memory usage of the other processes in the microcontroller is around 60 kB, leaving 40 kB for the compression algorithm and the wireless communication functionality. Therefore, the memory usage parameter will be considered, taking the amount of memory used by the compression algorithm into account. The memory usage can then be seen as overhead introduced by the algorithm and put into a score shown in Table D.3.

packet dependency
Even though the packet error in the communication link should be zero, it is hard to guarantee in practice

score	criteria
$+++$	< 5 kB
$++$	< 10 kB
$+$	< 15 kB
$-$	< 20 kB
$--$	< 25 kB
$---$	> 25 kB

Table D.3: scores for memory usage, indicating the overhead introduced by the algorithm for the same data set

and might be an invalid assumption. Therefore the compressed packages should be decompressable independently of each other. The scores for packet dependency are dependent and independent, without using the full range of the scoring system as can be seen in Table D.4. This is because all studied algorithms are either introducing dependency or do not introduce it at all.

score	criteria
$+++$	independent
$++$	-
$+$	-
$-$	-
$--$	-
$---$	dependent

Table D.4: scores for packet dependencies for the same data set

ability to compress varying sources

The words used to describe this parameter might not fully grasp its intention. However, this parameter might be an important, but yet hard to quantify one. To begin with, it is observed that different signals measured in nature can have different properties. Some signals tend to change faster then others, leading to different compression results between compression algorithms. It is observed, that some algorithms compress data sets with a lower entropy (explained in Section D.2.1) more efficient than data sets with a higher entropy. This is important to consider, because data from the sensors pants can change more rapidly for running and sprinting, then for walking. The algorithm should be able to handle these differences efficiently. To continue, signals used to compare different algorithms in literature, do not have to be similar to signals coming from IMUs. A more in depth analysis of the data sets will be given later on in Section D.2.2.

The score for this parameter is hard to quantify, because most algorithms are tested on the same data set. It is also observed that most algorithms perform similar on the standard used data set, discussed in Section D.2.1. Fortunately, some algorithms are tested on data sets with faster changing signals, leading to drastically different results. However, this is only researched by a few authors and therefore difficult to compare. Luckily, the quantification of this parameter is still possible as will become clearer after the different algorithms are discussed. For now, the following information is sufficient, the compression algorithm will make an assumption on the distribution function (e.g. exponential, normal, etc.) of the a signal to encode. If one distribution is assumed, the algorithm is very static and a low score will be given. If the algorithm does not assume a static distribution beforehand, it will gain a higher score as can be seen in Table D.5.

score	criteria
$+++$	high amount of distributions
$++$	multiple distributions
$+$	3 static distributions
$-$	2 static distributions
$--$	1 static distribution
$---$	static distribution completely off

Table D.5: scores for ability to compress varying sources

D.2. Theoretical background on data compression

This section will first start with the concept of entropy, which is the main principle behind data compression. Followed by an analysis on data sets used in literature compared to data sets from the sensor pants. Ending with the principle of Huffman and Arithmetic coding, describing the conversion from a data set into an optimal bit representation.

D.2.1. Entropy

Almost all data sets contain redundant information. In this case the data is collected from IMUs and represented by a 16-bit integer for every DOF. Due to this fixed length, redundant information is in the data set. By using Shannon's entropy limit, the maximum redundancy can be calculated for the symbolic representation of the data set. Shannon's entropy limit is given in equation D.5.

$$\mathrm{H}(\mathbf{x}) = -\sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{p}(\mathrm{x}_i)\log_2(\mathrm{p}(\mathrm{x}_i)) \tag{D.5}$$

If Shannon's theorem would be applied on a data set containing only 0's and 1's, the amount of bits required for representing 1-bit (e.g. a '0' or '1'), depends on the probability of a '0' or '1' occurring. In Figure D.1, the entropy of a binary memoryless source is shown. It can be seen that for p(0) = p(1), exactly 1-bit is required for every bit send. In this case, the data contains no redundant information, with the symbolic representation '0' and '1'. To add, this formula is based on the assumption that the source is memory less. Intuitively, when there is a 50/50 change of a '0' or '1' occurring the source is very unpredictable and thus not reducible. On the other hand, when p(0) = 1, it is certain that all characters will be a '0' and hence no bits are required to represent the data. It can be said that the source is fully predictable and has a no element of 'surprise'. [39]

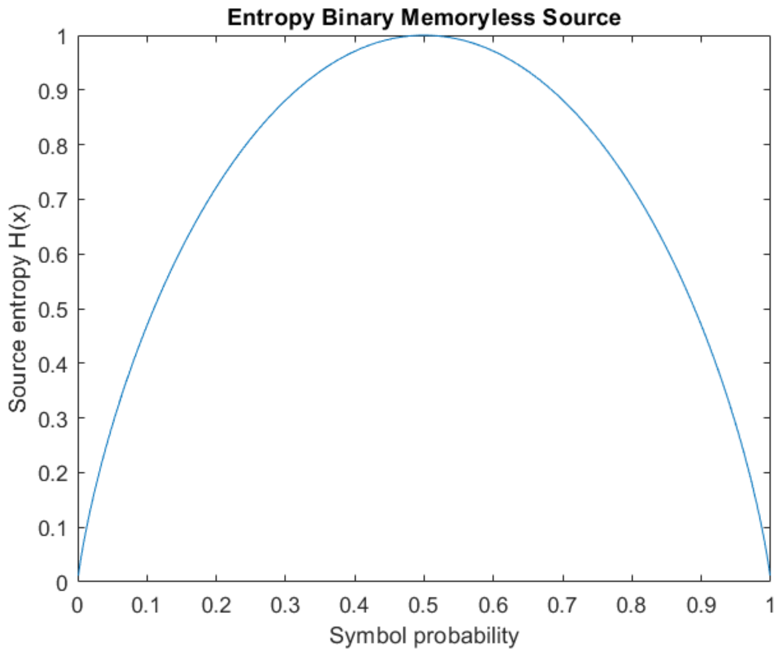


Figure D.1: Entropy of a binary memoryless source in terms of symbol probability

By applying Shannon's theorem on a chosen symbolic representation from a data set, the minimum amount of bits per symbol is obtained and hence the maximum compression size. However, the amount of bits required for representing the data set depends on the pattern recognition applied to the data set. The following example illustrates the importance of pattern recognition on data sets. Consider the following data sequence **010101010101010101010101**. By splitting the data in x_0 = '0' and x_1 = '1', the entropy limit results in an average of 1-bit per symbol. However, it can be seen that the data is

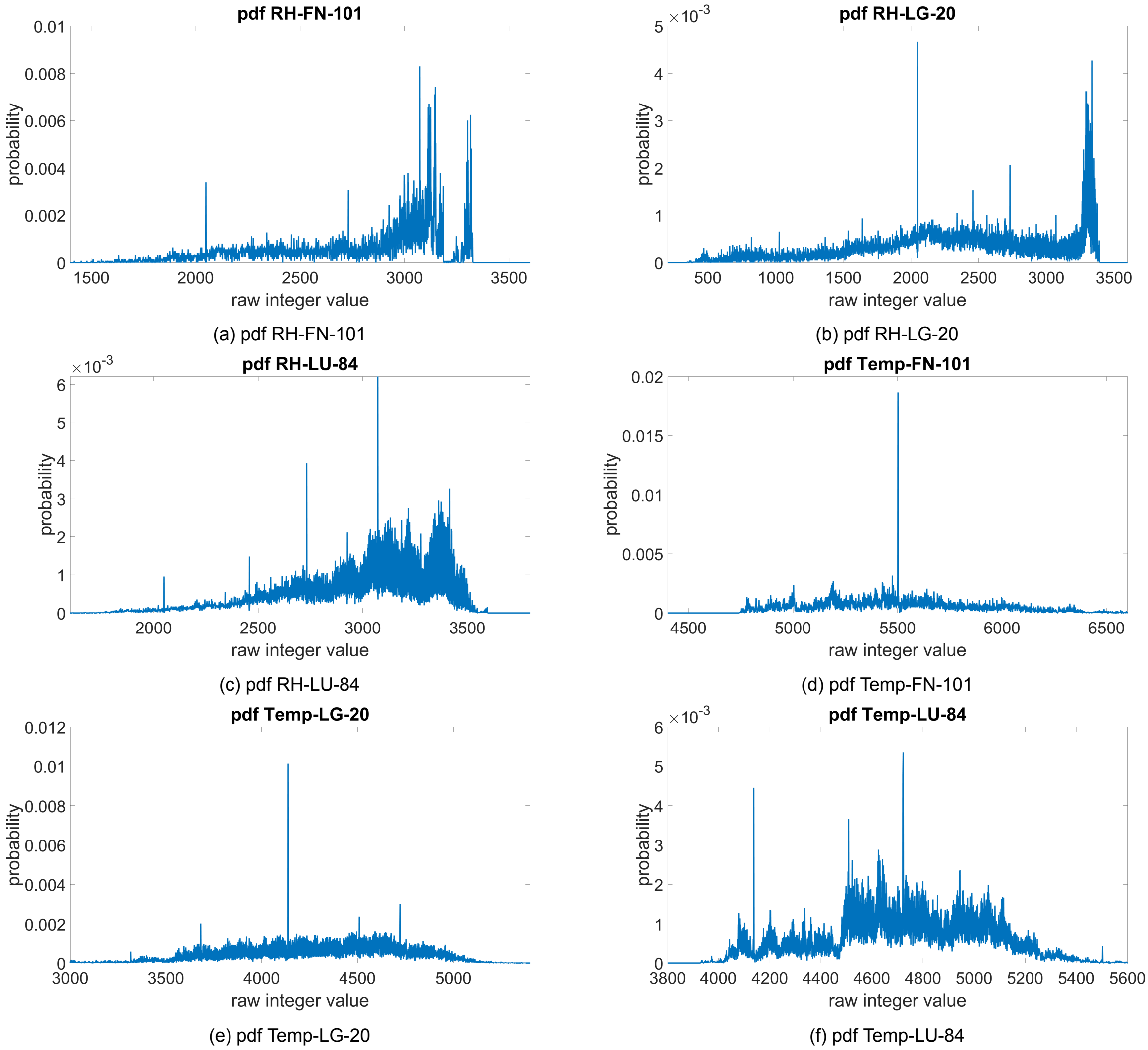
a repetition of the same pattern x_{01} = '01'. If Shannon's entropy theorem is now applied, a representation of zero bits is required. Another method is prediction instead of pattern recognition. For example, Lagrange polynomials can be used to extrapolate or interpolate data from a sensor and predict the next sample. The difference between the next sample and the predicted sample is then saved. The new values are consisting of a smaller range of numbers with a higher probability of occurring. By using Shannons's theorem, a lower amount of bits can now be used to represent the data set.
The strength of data compression techniques lies in the ability to predict or recognise patterns, remove correlations, and use an efficient symbolic representation in such a way that a computer is capable of compressing and decompressing the data file. In the previous example, the recognition of '01' leads to capturing the correlation between the '0' and '1' (always occurring after each other). By choosing the symbolic representation '01', the correlation in the signal is fully used and the entropy is dropped to 0 bits per symbol. In [34] (and references used in [34]) show that data compression algorithms will introduce overhead on top of Shannon's entropy limit. The overhead will depend on the inefficiencies in representing a certain symbolic representation. For example, the value 50 occurs more often then the value 30, however the value 50 is represented with 5 bits, while 30 uses only 3 bits. Other shortcomings are caused by the limitation of using integers. Take for example two symbols, which are most optimal compressed using 2.8 and 2.2 bits, this is not feasible using integers and hence a 2- and 3-bit representation is chosen.

D.2.2. Data sets from literature and sensor pants

The use of a common data set is to make a fair comparison between different compression algorithms in literature. Some papers have deviated from this common data set and used their own data. While others have included the common data set and added other types of data sets. In this section the most frequently used data set is discussed and also other types of data sets. These data sets are then compared to data sets obtained from the sensor pants during football.
To begin, the data set coming from SensorScope is the most common used in literature [31, 56–59, 64, 66, 70, 82, 105], the sensors are deployed on three locations: Le Gènèpi Deployment, HES-SO FishNet Deployment and LUCE Deployment. The deployed system is a TinyNode node, consisting of a TI MSP430 microcontroller, a Xemics XE1205 radio and a Sensirion SHT75 sensor module [15]. The published data is converted to a character string. Therefore the data from humidity and temperature is converted back to the original 12 and 14 bit respectively, according to the datasheet [90]. Some papers [9, 83] do not use the raw data for compression and will therefore not be considered in the comparison between the different compression algorithms.
Unfortunately, links to the SensorScope project and the TinyNode module are not available anymore. Also the data set is not available anymore online, fortunately the first author of [105] was able to provide the data set. Not only the SensorScope data set is used, but also other data sets. In [12] measurements from own equipment are used. In [102] the temperature data from three locations on the Squannacook river, Nashua river Watershed are used. In [25] data from real-world, public domain benchmarks from long-running environmental WSN deployments and seismic data are used.
To analyze the data sets, a matlab script is made, which converts the SenorScope data set into the correct 12 and 14 bit representation as described earlier. Next, the probability distribution functions are calculated and can be seen in Figure D.2a, D.2b, D.2c, D.2d, D.2e and D.2f.

In table D.6 the entropy is calculated based on the probability distribution functions. It can be seen that there is a significantly small difference between the calculated entropy for Temp-LG-20, RH-LU-84, RH-FN-101 and RH-LG-20, compared to the entropy calculated in [56, 59] and [67]. The difference in Temp-LG-20 and RH-LG-20 is most likely caused by the difference in samples used, 43059 (provided by [105]) versus 21523 (used in [59] and [67]). Also a difference can be seen between [56, 59] and [67], it is possible that the data sets are slightly updated over time. To continue, the small difference in relative humidity is unclear. By using the data sheet [90], the inverse equation D.6 for calculating the original raw 12-bit data (raw_h) is derived. The inverse equation is not given in [56, 59, 67], but the authors claim to have used the inverted versions of the conversion functions.

$$
\begin{aligned}
\mathrm{a} &= \mathrm{t1} \cdot (\mathrm{T_{oc}} - 25) + \mathrm{c1} - \mathrm{HR_{percentage}}(\mathrm{i}) \\
\mathrm{b} &= \mathrm{t2} \cdot (\mathrm{T_{oc}} - 25) + \mathrm{c2} \\
\mathrm{c} &= \mathrm{c3}
\end{aligned}
$$



(a) pdf RH-FN-101
(b) pdf RH-LG-20
(c) pdf RH-LU-84
(d) pdf Temp-FN-101
(e) pdf Temp-LG-20
(f) pdf Temp-LU-84

Figure D.2: probability distribution function data set SensorScope

$$\text{raw\_h(i)} = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad i \in [1, 2, ..] \tag{D.6}$$

Data Set	Entropy own calculation	Entropy based on [56, 59]	Entropy based on [67]
Temp-LU-84	10.0677	10.0677	10.07
Temp-FN-101	10.2609	10.2609	10.26
Temp-LG-20	10.4885	10.2492	10.25
RH-LU-84	10.1061	9.9156	10.08
RH-FN-101	9.8904	9.6070	9.75
RH-LG-20	10.9979	10.7634	10.84

Table D.6: Entropy data set SensorScope

Data Set	Entropy own calculation	Entropy based on [56, 59]	Entropy based on [67]
ΔTemp-LU-84	4.0469	4.0471	4.05
ΔTemp-FN-101	5.0956	5.0965	5.10
ΔTemp-LG-20	6.8950	6.8178	6.82
ΔRH-LU-84	5.9549	5.7032	5.85
ΔRH-FN-101	5.8746	5.7130	5.84
ΔRH-LG-20	7.8764	7.6052	7.67

Table D.7: Entropy data set SensorScope

Even though not all entropy calculations give the same results for the data sets. The Temp-LU-84 and Temp-FN-101 data set give the same results, indicating a high likelihood of correct applying Shannon's entropy theorem on the data sets. If any algorithm would be tested on the SensorScope data set, it is likely that a small difference is introduced in the back-conversion to raw data.
To continue, the use of a predictor is discussed earlier and seen as a potential mean to improve compression ratios. In case of natural signals, there is a relative high correlation between neighbouring points. This can be partially removed by differentiating the signal using (assuming the first sample is zero):

$$\Delta x(0) = 0$$
$$\Delta x(i) = x(i) - x(i\text{-}1) \qquad i \in [1, 2, ..]$$

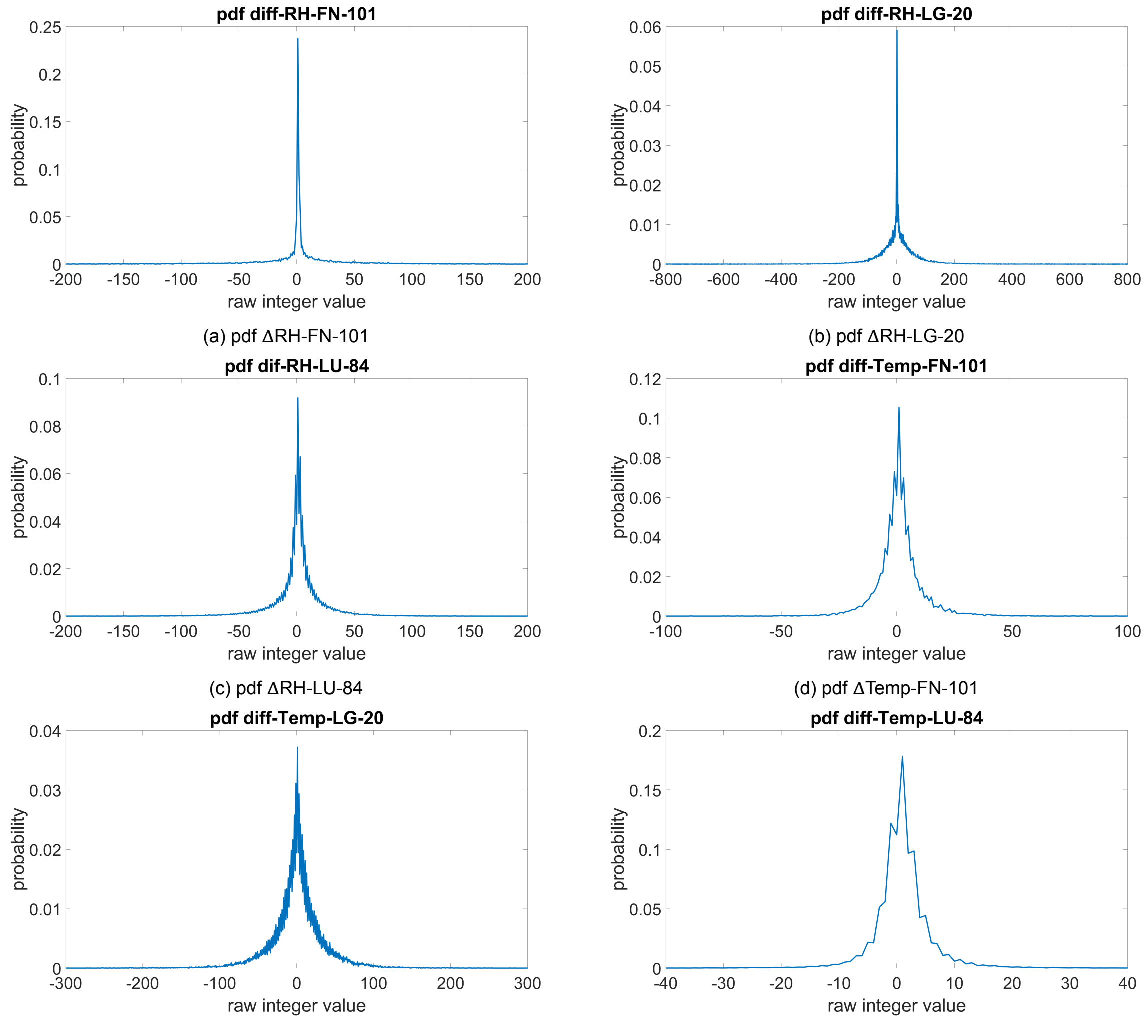
It can be observed in Figure D.3a, D.3b, D.3c, D.3d, D.3e and D.3f that the probability distribution functions of the differentiated signals have a lower variance, with a more similar shape. When the entropy is calculated, a significantly lower entropy can be seen in Table D.7. It should be noted again, that some differences are observed between the entropy calculations for ΔTemp-LG-20, ΔRH-LU-84, ΔRH-FN-101 and ΔRH-LG-20.

If the entropy is calculated for the data sets obtained using the current prototype given in Table D.8. It can be seen that the magnetometer has a high potential in data reduction. Considering the lower entropy limit, the data speed can theoretically be reduced from 144 kbit/s to 65.3 kbit/s, with a data compression ratio of 54.7 %.

In order to code the different symbols occurring in the differentiated signal, the use of Huffman coding [48] and Arithmetic coding [21] will be discussed in the next section. Afterwards different schemes are proposed for data compression. The following algorithms will be discussed: DWT lifting scheme [12], Tunstall based algorithm [70], FELACS [59], SLZW [82], LEC [67], S-LEC [64], Simple algorithm - Kolo [58], ALDC [56], Two-Modal Transmission GPC [63, 65], median predictor based algorithm [69] and BWT & MTF [24]. Finally a conclusion is given comparing the different techniques.

D.2.3. Huffman coding [48]

In order to represent the symbols obtained as efficiently as possible, Huffman proposes a method to achieve a minimum-redundancy code, also called the 'optimum' code. To achieve this, five restrictions



(a) pdf ΔRH-FN-101 (b) pdf ΔRH-LG-20

(c) pdf ΔRH-LU-84 (d) pdf ΔTemp-FN-101

(e) pdf ΔTemp-LG-20 (f) pdf ΔTemp-LU-84

Figure D.3: probability distribution function differentiated data set SensorScope

are given on the symbol coding and summed up below (the restrictions are taken from the original paper, with some extra abbreviation):

1. Each symbol is represented with a unique set of coding digits. (e.g. ’00’ can not be used to represent the symbol ’a’ and ’b’).
2. The symbols are represented in such a way, that it does not require any additional information to know the start of a symbol, as long as the starting point is known from the message.
3. $L(1) \leq L(2) \leq ... \leq L(N-i) = L(N)$. [48] The length of the most occurring symbol $L(1)$ should be smaller or equal to the second most occurring symbol $L(2)$. Also the last symbol should be equal to the second-last.
4. At least two symbols of the same length (L(N)) have the same representation, except for their final digits, with a maxim of D, which equals the maximum amount of symbols. (always both options are coded, for the Huffman tree, which will be explained in more details)
5. All combinations of digits with length $L(N)$-1 digits, must have a function to represent a symbol, or to partially represent the beginning of a symbol. (This leads to using all bits before adding a new one, for example, a = ’1’, b = ’01’, this means that L(b) - 1 = 1, which indicates that all combinations should be used to represent the first bit in the symbols a and b, this hold because ’0’ and ’1’, are both used)

position	acc	gyr	mag	Δacc	Δgyr	Δmag
left upper leg (x)	10.7493	10.7566	7.1627	8.0059	8.2416	4.8337
left upper leg (y)	10.1671	11.4393	8.6093	7.7525	8.7003	4.9614
left upper leg (z)	10.2295	11.1944	8.4559	7.8079	7.9998	4.9282
left lower leg (x)	10.4155	11.4580	8.3209	7.8373	8.5052	4.4736
left lower leg (y)	10.5525	11.5604	8.1369	7.8113	8.4801	4.4396
left lower leg (z)	11.0003	10.4577	8.7423	7.7531	7.7096	4.6396
right upper leg (x)	10.6346	10.8150	7.0169	7.9091	7.6751	4.2556
right upper leg (y)	10.1409	11.3498	8.5170	7.6679	8.5379	4.4575
right upper leg (z)	10.1497	11.0092	8.4193	7.6657	7.6957	4.4056
right lower leg (x)	10.3241	11.0870	8.3220	7.6259	8.2730	4.3847
right lower leg (y)	10.5546	11.4885	7.8525	7.8253	8.0038	4.4481
right lower leg (z)	10.9744	10.4965	8.6538	7.7628	7.2958	4.5643
trunk (x)	10.2039	10.6392	8.0756	7.5260	7.3298	4.2158
trunk (y)	9.9645	9.7830	7.2222	7.4236	7.2670	4.1717
trunk (z)	9.8777	9.8591	8.2788	7.1426	7.0043	4.2001
average	**10.3959**	**10.8929**	**8.1192**	**7.7011**	**7.9146**	**4.4920**

Table D.8: Entropy measurements current prototype

Based on these restriction a binary tree can be created, using the method in Figure D.5. This method, starts by taking the lowest two probabilities and sums them up, in this case 0.01 and 0.03. The sum of these is 0.04, and this number is added to the list of probabilities, and the numbers 0.01 and 0.03 are removed. From this new updated list, the lowest two probabilities are taken again and processed similar. Eventually all probabilities are grouped and a tree can be constructed based on this process. This tree is called the Huffman coding tree and can be seen in [48, Figure D.4]. In this tree, the leafs are the probabilities from the original message shown in Figure D.5. By walking through the tree and concatenating the '0' and '1' from top to bottom, the results obtained in [48, Table D.9] are observed, with P(i) the probability of symbol 'i' occurring and L(i) the length of symbol 'i' in bits.

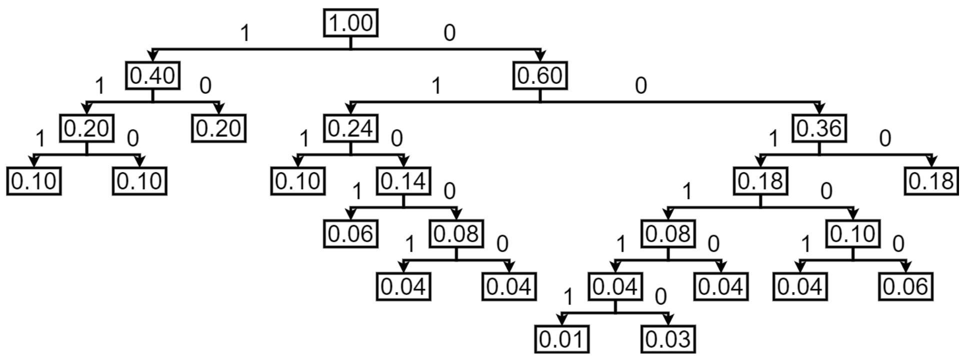


Figure D.4: Huffman coding tree obtained after following the optimum binary coding procedure in Figure D.5

An interesting observation of the Huffman tree are the new probabilities after splitting a node in Figure D.4. These probabilities tend to approximate half the value of its higher (previous) node. For example the first node '1.00', splits into '0.40' and '0.60'. The second node '0.40' splits into '0.20' and '0.20'. This property is to be expected, because every bit should represent a 50 % of occurring. In Figure D.1, discussed in Section D.2.1, the maximum entropy is obtained if P(0) = P(1). The same principle holds for the Huffman tree, if every nodes splits into an equal probability of occurring, the maximum entropy is reached. Hence, the Huffman coding procedure exhibits the property of optimally encoding probability distribution functions of a set of symbols. However, it can be observed that not all nodes split into an equal probability, this gives room for improvement. By representing each symbol as a binary code, it is impossible to have a more optimum solution then the Huffman method. To overcome this problem, the use of arithmetic coding is introduced.

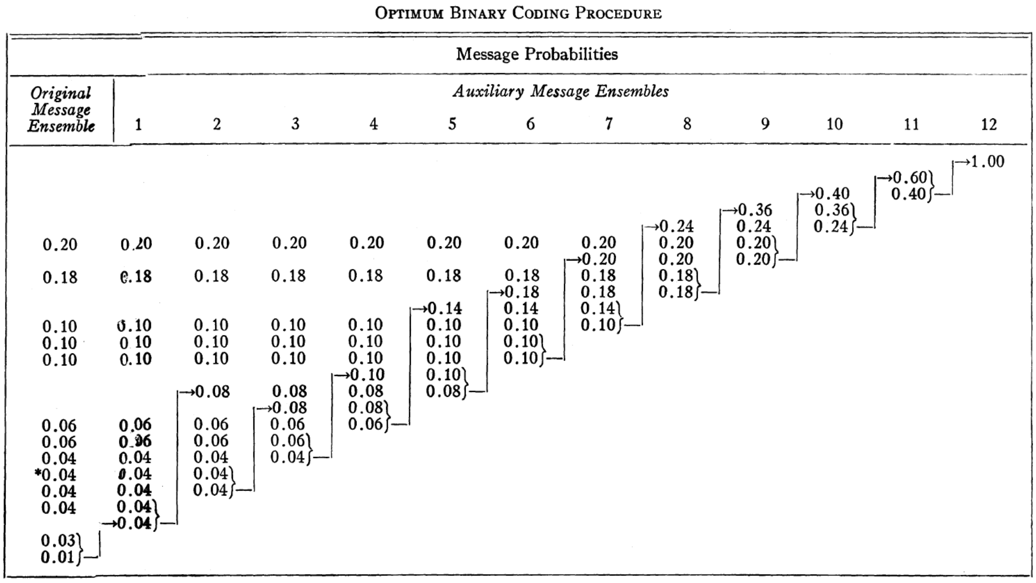


Figure D.5: Optimum binary Coding Procedure [48]

i	P(i)	L(i)	P(i)L(i)	Code
1	0.20	2	0.40	10
2	0.18	3	0.54	000
3	0.10	3	0.30	011
4	0.10	3	0.30	110
5	0.10	3	0.30	111
6	0.06	4	0.24	0101
7	0.06	5	0.30	00100
8	0.04	5	0.20	00101
9	0.04	5	0.20	01000
10	0.04	5	0.20	01001
11	0.04	5	0.20	00110
12	0.03	6	0.18	001110
13	0.01	6	0.06	001111

Table D.9: Results of Optimum Binary Coding Procedure, with P(i) as the probability of i occurring and L(i) the representation of i in bits [48]

D.2.4. Arithmetic coding [21]

Another method to represent a symbolic distribution is to use arithmetic coding, which does not have the constraint of a symbol being encoded by a whole number of bits. The algorithm seems to almost reach the entropy limit in compression. The principle is straightforward and can be implemented on a computer, without using a floating-point operation. The arithmetic coding will use intervals as decision regions to know which symbol is encoded. First the algorithm will perform the following steps, it will start with a low and high bound, with the values 0 and 1, respectively. It will then update these lower and higher values for every symbol, eventually, a number lying in the final interval is stored. To put this more mathematically, consider an alphabet with {a_1, a_2, .. , a_n}, each character has a probability of occurring, given by P(a_i). The value K(a_i) represents the cumulative distribution of the symbol. Take the example in [21] using the symbols {a,b,c,d} with probabilities P = {1/2, 1/4, 1/8, 1/8} and K = {0, 1/2, 3/4, 5/6}. By applying [21, Equation D.7 and D.8] the lower and higher bound are calculated. In the first iteration the symbol 'a' is taken, leading to the interval 0 till 0.5. In the next iteration the symbol 'b' is taken. For this iteration, the lower and higher bound are calculated using the probabilities given in P. This can graphically be seen as splitting the interval 0 till 0.5 in partitions equal to their probability of occurring, as can be seen in [21, Figure D.6]. By taking the interval belonging to the symbol, the updated interval can be calculated, until all symbols are used. If this method is performed on the following input string: {a, b, a, a, b, c, d, a}, the calculated intervals can be seen in [21, Figure D.6]. The final interval [0.2723388672, 0.2723999024) is then represented by any number in this sequence, for example 0.27234. It can be seen that arithmetic coding stores an entire input string into one fractional number. Previously, it is mentioned, that it can also perform these operations without floating-point (fractional) arithmetic. This method proves to work and is especially useful for smaller systems lacking a dedicated floating-point unit. The basic principle stays the same, a 32-bit integer range [0, 2^{32}) is used to represent the ranges instead of a real number [0,1).

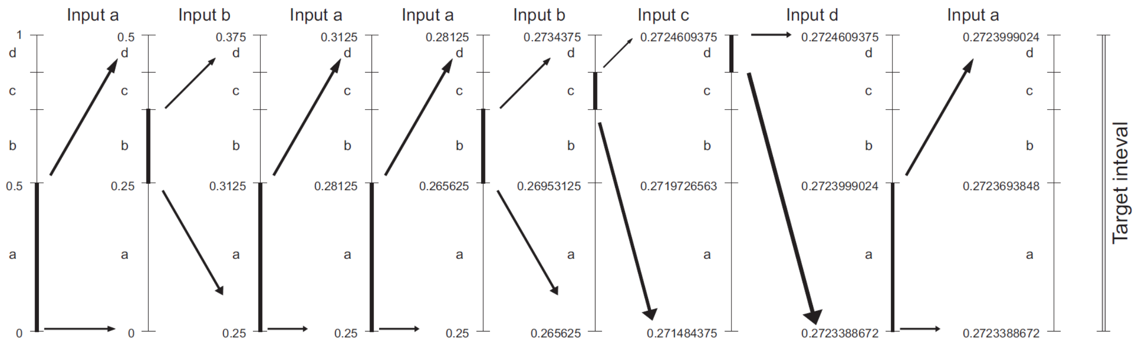


Figure D.6: Arithmetic coding example [21]

$$\text{low} = \text{low} + K(a_{j-1}) \cdot (\text{high} - \text{low}) \tag{D.7}$$

$$\text{high} = \text{low} + K(a_j) \cdot (\text{high} - \text{low}) \tag{D.8}$$

D.2.5. Huffman and Arithmetic coding practical performance

An important remark to keep in mind is that arithmetic and Huffman coding make an assumption on the probability distribution function of the data set. A mismatch between this prediction and the data set, will result in lower performances in compression. The compression algorithms, discussed later on, will sometimes use Huffman encoding. Good performance are observed, when using a Huffman table, with approximately the same probability distribution function as the source, as will be seen in the LEC algorithm [67]. However, when this is not the case, the performance significantly drops as is observed with the LEC algorithm in [64]. This will be taken into account in the ”ability to compress varying sources” parameter, described in Section D.1.

D.3. Compression algorithms

This section will give a list of different lossless compression algorithms, suited for low power embedded systems. Each algorithm is shortly explained and a score is given to the parameters described in Section D.1. In Section D.2.3 and D.2.4, the use of Huffman and Arithmetic coding is explained to provide a basis in lossless data compression, but is just a fraction of all compression methods available. In this section, new concepts will be introduced. These new concepts will be explained alongside the algorithm.

D.3.1. DWT Lifting Scheme [12]

A Discrete Wavelet Transformation (DWT) is used to transform redundant samples in the spatial domain to decorrelated coefficients in the time-frequency domain [12]. By making use of a ”lifting” procedure introduced in [97, 98], the mean and differences are used to compute the DWT coefficients. This procedure is as follows, the data set is split in even values {x(0), x(2), ..} and odd values {x(1), x(3), ..}. The difference is calculated between the even and odd values and stored in a vector d_{j+1}, the j denotes the iteration. After one iteration two vectors are saved, as can be seen in [12, 52, Equation D.9]. When there is a high correlation between the neighbour points, the values stored in d_{j+1} will be low, leading to higher compression ratios. In [12, 52, Equation D.10] the even elements are replaced by taking the average of the data, smoothing the input for the next iteration. In the final step, the new data set is created by taking s_{j+1} for the next iteration. The process is repeated n times on a data set with length 2^n. Leading to a vector with differential coefficients, being concatenated as: $[d_1, d_2, \dots, d_n]$, with lengths $[2^n/2^1, 2^n/2^2, \dots, 2^n/2^n]$, and a vector s_n containing the average value of all samples. A graphical representation of the iterations can be seen in [8, Figure D.7].

$$d_{j+1} = x[2n+1]_j - P \cdot x[2n]_j \tag{D.9}$$

$$s_{j+1} = x[2n]_j + U \cdot d_{j+1} \tag{D.10}$$

$$x_{j+1} = s_{j+1} \tag{D.11}$$

The decompression operation is given by inverting the compression procedure, as can be seen in [12, 52, Equation D.12, D.13 and D.14].

$$x[2n]_j = x[2n]_{j+1} - U \cdot d_{j+1} \tag{D.12}$$

$$x[2n+1]_j = x[2n+1]_{j+1} + P \cdot x[2n]_j \tag{D.13}$$

$$x_j = \mathrm{merge}(x[2n+1]_j, x[2n]_j) \tag{D.14}$$

In [12, 52, Equation D.9] D.9, the P is a predictor value, in the original LiftingWiSe algorithm [8] the value is taken as P = 1 and hence the d_j is the difference between two neighbour points. The value for U in [12, 52, Equation D.10] is taken as U = 1/2, in order to calculate the average value between two points, which converges after n iterations into s_n being the average value of the signal. By replacing $P \cdot x[2n]_j$ with an n-point La-Grange interpolating polynomial given in [12, 52, Equation D.15 and D.16], better results are obtained for n = 4, due to more accurate predictions of the signal.

$$P(x) = \sum_{j=0}^{n} F_j(x) y_j \tag{D.15}$$

$$F_j(x) = \prod_{k=0, k \neq j}^{n} \frac{x - x_k}{x_j - x_k} \tag{D.16}$$

The data result is compressed using Delta Encoding, which encodes the data with a fixed amount of bits. The value of this changes constantly and therefore, the value is frequently updated. During each update, an update message is send, introducing some overhead. Because the algorithm is lossless, an extra message is send, to prevent losses, when a sample point is outside the delta bit range. To continue, in [8] it also noted that the multiplication with U = 1/2, can lead to loosing the least significant bit, but this effect will be neglected. The algorithm is compared to the Discrete Cosine Transform, LiftingWiSe [8] and simple Delta encoding [11] algorithms, using the Shimmer3 GSR+ data set. It can be observed, that the DWT Lifting scheme with a Lagrange polynomial, outperforms the other algorithms.

Unfortunately, the algorithm does not make use of the SensorScope data set, making it difficult to give a compression rate score. To get an indication of the performance, the use of a DWT + Lagrange interpolation is used on the SensorScope. However, the use of the Lagrange interpolation turned out to slightly increase the values of d_j, indicating a decrease in performance.

From a theoretical perspective, the DWT lifting scheme will most likely perform better than just taking the difference between two points. However, results on the SensorScope data set, already show worse

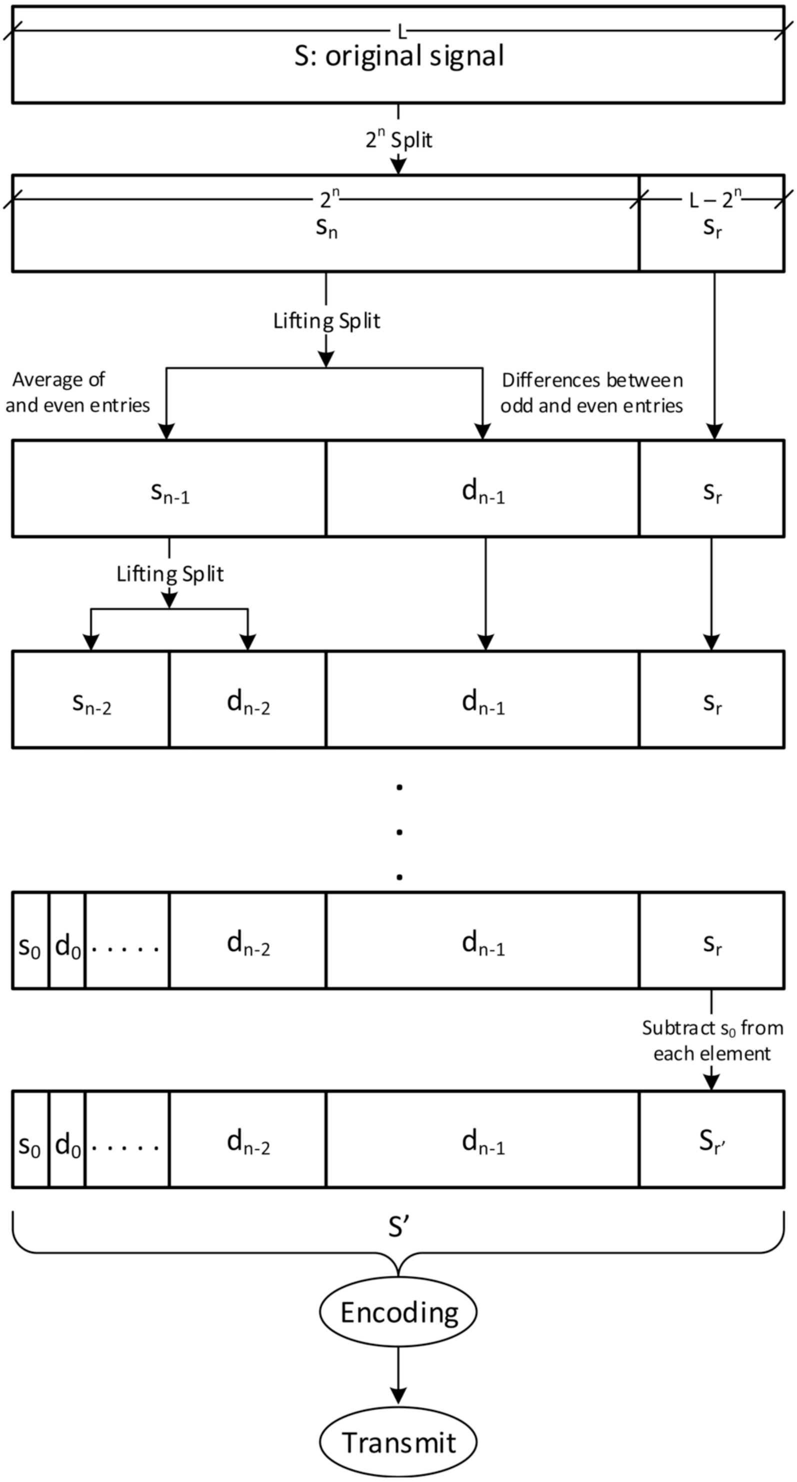


Figure D.7: Lifting Scheme Procedure [8]

performance, then expected. On top of that, the use of Delta Encoding will most likely be insufficient compared to other methods later discussed, resulting in a score of + on compression ratio. The use of only 12 adding or subtracting operations and 9 multiplications per data point to get the vector S' ([8, Figure D.7]) , is considered as a + on power consumption. It should be noted, that the conversion to a more efficient symbolic representation, either using a fixed amount of bits or applying Huffman coding. Will be assumed to use approximately the same amount of operations. The extra memory usage can

parameter	score	explanation
compression ratio	+	DWT + Lagrange polynomial interpolation + Delta Encoding
power consumption	+	low computation cycles, hence CR determines the score
memory usage	+ + +	no extra memory required for the algorithm
packet dependency	+ + +	no packet dependency
varying statistics	++	DWT lifting scheme removes correlations for handling different distributions and the Lagrange polynomial is used as predictor

Table D.10: scores for DWT Lifting Scheme

be neglected, giving a + + + score. By sending S', only this data will be lost. The next package can still be decompressed, scoring a + + +, indicating no packet dependencies. Finally, the use of DWT lifting scheme can get the most efficient representation of different source statistics and can therefore handle multiple distributions, even though the performance seem to fall short in practice. The use of the Lagrange polynomial could increase performances, but is observed to decrease the performance. Hence a score of ++ is obtained. A summary of the mentioned scores is given in Table D.10.

D.3.2. Tunstall Based Algorithm [70]

The Entropy based Adaptive Lossless data compression algorithm (AELDC) is shown in Figure D.8.

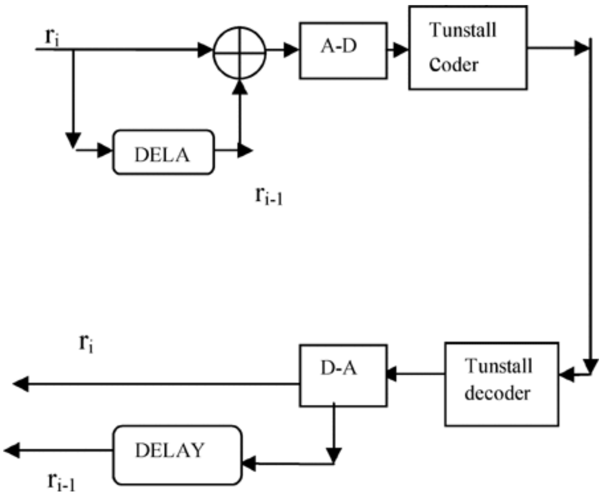


Figure D.8: AELDC block diagram [70]

The AELDC algorithm is making use of the Tunstall coder, which is originally explained in [104] and a brief summary is given in [89]. Tunstall codes uses a fixed code length in contrast to many other compression techniques, where a variable codeword is used. The n-bit Tunstall codes are created based on analyzing a sequence of N different characters. The different characters are first organized based on their probability. In the next iteration, the entry with the highest probability is taken and replaced with N new entries, each entry consists of the removed element plus another character. In every iteration the codebook will increase by (N - 1). Therefore K iterations can be performed for a dictionary of size 2^n as can be seen in [89, Equation D.17]. An example of a 3-bit Tunstall code, applied to a memoryless source with the symbols {A, B, C}, is given in [89, Table D.11]. In Table D.12 the amount of bits per iteration can be seen, calculated using equation D.18. With x being the amount of bits per character and char(x_i) the amount of characters in the i's element of sequence x (e.g. for iteration 3, x_7 = AAC). To continue, P(x_j | $\forall x_i \in x_j$) means the probability of x_j, if x_i has all its elements in x_j (e.g. take x_j=x_6=AAB and x_i=x_1=B from iteration 3, then all elements from x_1 are in x_6). In the first iteration only 3 sequences are present, hence 2 bits are sufficient representing one sequence. In the second iteration 5 sequences lead to an increase to 3 bits per sequence, still a decrease in bit per character can already be seen. The calculated entropy limit, compared to the performance obtained using a custom made Simulator in Matlab, verifies equation D.18. Note, that a sequence consist of one or multiple characters. In the last iteration the best performance is obtained, caused by the detection of long sequences occurring relative frequent. In the AELDC algorithm it is not stated, which value of n is used for the n-bit Tunstall codes. The algorithm also required to send the code book alongside the compressed data, for the decompression. The algorithm is compared with the S-LZW [82] and LEC [67] algorithm (discussed later on), showing similar to better performance on the SensorScope data set.

$$\mathrm{N} + \mathrm{K} \cdot (\mathrm{N} - 1) \leq 2^{\mathrm{n}} \tag{D.17}$$

Iteration 1	
Sequence	Probability
A	0.60
B	0.30
C	0.10
Iteration 2	
Sequence	Probability
B	0.30
C	0.10
AA	0.36
AB	0.18
AC	0.06
Iteration 3	
Sequence	bit representation
B	000
C	001
AB	010
AC	011
AAA	100
AAB	101
AAC	110

Table D.11: A 3-bit Tunstall code example, given in [89]

$$x = \sum_{i=1}^{n} \Big(\mathrm{P}(\mathrm{x_i}) - \sum_{j=i+1}^{n} \mathrm{P}(\mathrm{x_j} \mid \forall \mathrm{x_i} \in \mathrm{x_j}) \Big) \frac{2^{\mathrm{n}}}{\mathrm{char}(\mathrm{x_i})} \tag{D.18}$$

Iteration	bits per character Matlab simulation	Entropy Limit
1	2	1.29
2	1.87	1.77
3	1.53	1.50

Table D.12: Tunstall based efficiency

The compression ratio of the Tunstall algorithm is similar to that of the highest obtained, except for the RH-FN-101 data set. Hence a score of $++$ is given. The need for updating the tree after every set of two or three symbols, will result in a decreasing score for power consumption to $-$. The use of an approximately 15- or 17-bit Tunstall coder to achieve the given compression ratios is assumed. Because the data range is from 12 till 14 bits, indicating N = 14. Hence the input of two till three symbols, indicates the use of 3 iterations. The saving of 2^{17} entries with 2.5 symbols of 16 bit, will result in 655 kB of extra memory usage. This will results in a score of $---$ on memory usage. Because the code book is updated on the run, the loss of a packet can result in a mismatch between the code book used on the compression and decompression side. Fortunately, this will give a $++$ score on the ability to compress varying source statistics. It should be noted, that the Tunstall coding scheme can also be used static, by making an assumption on the probability distribution. This will then lead to no packet dependencies, but a very bad ability to compress varying source statistics. The summary of the scores can be seen in Table D.13.

parameter	score	explanation
compression ratio	++	Tunstall Encoding
power consumption	−	extensive tree search
memory usage	− − −	655 kB to store code book
packet dependency	− − −	packet dependency
varying statistics	++	code book updates based on input data

Table D.13: scores for Tunstall Based Algorithm

D.3.3. FELACS [59]

The fast and efficient lossless adaptive compression scheme (FELACS) is based on the Golomb-Rice coding. The Golomb-Rice codes are optimal for encoding exponential and geometrical distributions [38]. The compression is done by splitting the data into a fixed and variable part. The fixed part is given by calculating δ mod m, with δ being a non-negative integer in the range [0, m-1]. The variable part is given by $\lfloor \delta/\mathrm{m} \rfloor$. A special case, also observed by Rice [79], assumes an exponential distribution on the source data and takes m = 2^k. This gives rise to an implementation using fast arithmetic operations on a computer. The operation $\lfloor \delta/2^k \rfloor$ is the same as k times shifting δ to the right and δ mod m is calculated by taking the k least significant bits [95]. This methods, splitting the value into a fixed and variable part is called the split simple option for Rice algorithms. Three other proposed methods in rice coding are: fundamental sequence, second extension option and zero block option [79]. The Rice coding is performed on non-negative integers, therefore a conversion is applied on the data set, given in [89, Equation D.19].

$$\mathrm{d_i} = \begin{cases} 2\Delta \mathrm{x_i} & 0 \le \Delta \mathrm{x_i} \le \mathrm{T_i} \\ 2|\Delta \mathrm{x_i}| - 1 & -\mathrm{T_i} \le \Delta \mathrm{x_i} < 0 \\ \mathrm{T_i} + |\Delta \mathrm{x_i}| & \text{otherwise} \end{cases} \tag{D.19}$$

$$\mathrm{T_i} = \min(\mathrm{x_i}, 2^{\mathrm{N}} - 1 - \mathrm{x_i})$$

In the FELACS algorithm, each data point can be written as $\ll \mathrm{s_i} \mid \mathrm{a_i} \gg$, with s_i coming from $\lfloor \delta/\mathrm{m} \rfloor$ and a_i from δ mod m, with m = 2^k, and $\delta = \mathrm{d_i}$. The variable k, is important to estimate correctly, if the value is to low, the bits for representing s_i will significantly increase. If the value is to high, the bits for representing a_i will increase. The optimal value will have the optimum balance between the amount of bits to represent s_i and a_i.

Hence the strength of the FELACS algorithm is in estimating the optimal k (k*) value for compression. By using an heuristic approach, only 7-bit shift- and J add-operation are required to find the optimal value of k for each block. The value is find by calculating D using [59, Equation D.20] and choosing the correct value of k based on [59, Table D.14], whereby J is the amount of data points per block, minus the first point.

$$\mathrm{D} = \sum_{\mathrm{i}=0}^{\mathrm{J-1}} \mathrm{d_i} \tag{D.20}$$

Optimum code parameter k*	D region in bits
0	D ≤ 2J
1	2J < D ≤ 4J
2	4J < D ≤ 8J
3	8J < D ≤ 16J
4	16J < D ≤ 32J
5	32J < D ≤ 64J
6	64J < D ≤ 128J
7	128J < D

Table D.14: Decision region for the first 8 optimum coding parameters [59]

A step-by-step example from the original paper [59] will be given more briefly. The following data set is given: {5555, 5583, 5548}, from a 14-bit ADC.

1. Calculate d_i = {28 , -35}, convert to δ_i = {56, 69}, using [59, Equation D.19].
2. Calculate the decision region, using [59, Equation D.20]. Giving, 56 + 69 = 125, note that the first value is not used.
3. Based on [59, Table D.14], the optimum value is k = 5, with J = 2.
4. The values of d_0 and d_1 are given by 0111000 and 00100101 respectively.
5. The following packet is send: 101 (k-value), 01010110110011 (5555), 0111000 (d_0), 00100101 (d_1), and combined: 101010101101100110111000000100101.

As can be seen the data is compressed from 42 bit to 33 bit, by increasing the block size the compression ratio will also increase. The algorithm is tested on the SensorScope data set and compared with SLZW [82], mLEC [57], Simple algorithm [58] and ALDC [56]. The algorithm shows slightly better performances compared to the mentioned algorithms scoring a + + + on compression ratio. Due to the low amount of operations required to calculate D and k, it will get a + + + on power consumption. It does not require any extra memory, giving a + + + on memory usage. Because every packet contains the first sample, there is no packet dependency, hence a + + + is given. Finally the ability to adapt to varying source statistics is considered. The use of the value k compensate for different ranges of data. This will result in a significant decrease in the value for representing the magnitude of the number. Especially in the case of IMU data, the values can vary significantly in magnitude. Hence, the algorithm is assumed to handle multiple types of distributions, scoring a + + + on ability to compress varying sources. The scoring is summarized in Table D.15

parameter	**score**	**explanation**
compression ratio	+ + +	Golomb-Rice coding, with estimating k
power consumption	+ + +	calculating D requires 7-shift and N add-operations, s_i requires one shift operation and a_i one and operation
memory usage	+ + +	no memory required
packet dependency	+ + +	no packet dependency
varying statistics	+ + +	the estimating of k, increase the adaptability to different sources

Table D.15: scores for FELACS

D.3.4. S-LZW [82]

The sensor-Lempel-Ziv-Welch (S-LZW) algorithm is based on the famous LZW algorithm [111]. The LZW algorithm tries to find repetitions in the data set and stores them more effectively, if the repetition occurs frequently. The repetitions are stored in a dictionary, which is created by both sender and receiver, hence no dictionary has to be send over with the compressed data. The algorithm will execute as shown in [82, Algorithm 2] and an example is given in [82, Table D.16]. To better understand the steps taken from encoded string to output stream, the following is important. The variable ω is encoded, based on the current dictionary. After every iteration a new entry is added in the dictionary. The value given in the output stream is based on the decimal representation of the letter 'A' in the ASCII system. It can be observed that the total amount of symbols ranging from 0 till 255. This means that a new entry in the dictionary, starts from 256, introducing 1 extra bit to represent the original symbols. The entry can store up to 768 entries before a new bit is required to represent the data. The decompression will not be explained such exhaustive. By reading the string as {65 256 65 66 257 66 67 67} for decompression. It can be seen that the first 65 will be decompressed as an 'A'. Afterwards the algorithm will detect the sequence with a value of 256 (9-bit), which indicates the first entry of the dictionary should be taken. The dictionary entry can be reconstructed taking the first string (normal character or from dictionary) before and after the new entry. For the case of 256, this would mean taking {65, 65} from {**65** 256 **65** 66 257 66 67 67}, which is 'AA'.

Algorithm 2 Step: LZW algorithm [111]

```
ω = {}                                              ▷ initialize as empty
procedure Step:
    Read next input character K
    if no such K (input exhausted) then
        code(ω) → output
        EXIT                                        ▷ end of sequence
    end if
    if combination {ω, K} is in dictionary then
        {ω, K} → ω
        repeat Step
    else if combination {ω, K} is not in dictionary then
        code(ω) → output
        add code → dictionary {ω, K}
        K → ω
        repeat Step
    end if
end procedure
```

Input Stream: **AAAABAAABCC**

Encoded String	**Output Stream**	**New Dictionary Entry**
A	65	256 - AA
AA	65 256	257 - AAA
A	65 256 65	258 - AB
B	65 256 65 66	259 - BA
AAA	65 256 65 66 257	260 - AAAB
B	65 256 65 66 257 66	261 - BC
C	65 256 65 66 257 66 67	262 - CC
C	65 256 65 66 257 66 67 67	EXIT

Table D.16: Example of LZW compression from [82]

In the proposed S-LZW the dictionary is limited to 512 entries, limiting the amount of memory needed and making the algorithm suited for smaller embedded systems. The algorithm tries to find patterns frequently occurring, therefore data sets with a higher entropy have been observed to benefit less from the algorithm. Another problem occurring on data sets with higher entropy is the growing dictionary problem. Due to the limitation on the dictionary size, the last part of the data set might not fit in the dictionary anymore. As a consequence, patterns occurring in the beginning of the data set might not occur in the last part of the data set, making the dictionary irrelevant.
The authors in [82] came up with the idea of adding a mini-cache to the dictionary, storing the last used data. Because data in a short time-interval are more likely to have the same pattern. This method leads to an increase of performance in the data set from SensorScope. However, the other data sets (Great Duck Island [99], ZebraNet [117] and Calgary Corpus Geo [17]), did not seem to gain any improvements. The mini-cache seems to be effective in data sets with more repetitions. Therefore a Burrows-Wheeler Transform (BWT) [24] is used to organize data in such a way that it contains runs of characters (this algorithm will later be explained in more depth in Section D.3.10). The combination of a BWT with a Move-to-Front (MTF) operator, an often used combination, is shown to decrease performance on the four data sets. However, using the BWT in combination with the mini-cache leads to better performances for all data sets expect the Calgary Corpus Geo [17]. Another approach used is the Structured Transpose (ST), whereby the data is assumed to have the same most significant bytes (MSBs) in a row for multiple samples. By placing the data in a matrix and transposing it, a structure of repeating MSBs occurs. Leading to better performance using the mini-cache in combination with the ST. Also the use of Run Length Encoding (RLE) is considered, whereby a run of more then 2 characters

is replaced by two times the character and an integer with the value of the amount of extra characters. This turns out to be not effective in combination with the S-LZW algorithm. The paper concludes with a summary of different configurations for the S-LZW algorithm in different wireless sensor network structures. The achieved data compression on the SensorScope data sets, are achieved without applying a conversion to the original 14-bit and 12-bit representation. Therefore the compression achieved on the raw sensor data obtained in [67] using S-LZW are considered, and show significantly lower performance compared other algorithms, leading to a $--$ for compression ratio. The use of a simple dictionary leads to just a simple search in the dictionary, hence the power consumption is given a $--$, due to the bad compression ratio. The memory usage of the 512 entries in the dictionary is estimated at 2.618 kB, hence a score of $+++$ is given. There is also no packet dependency, because every packet will have its own new dictionary created. Finally, the ability to adapt to varying source statistics can be observed to completely fail. This is observed in certain data sets, resulting in full dictionaries, not efficiently used. Hence a score of $---$ is given. The overall scoring is given in Table D.17.

parameter	**score**	**explanation**
compression ratio	$--$	using the Lempel-Ziv-Welch algorithm turns out not to be efficient on sensor data
power consumption	$--$	the low compression ratio results in a relatively high power consumption
memory usage	$+++$	a total of 2.618 kB is required for the dictionary
packet dependency	$+++$	no packet dependency
varying statistics	$---$	the algorithm is based on the recognition of reoccurring patterns, this is observed to fail, partially to the dictionary reaching its limit and data sets not living up to the assumption of reoccurring patterns

Table D.17: scores for S-LZW

D.3.5. LEC [67], ([66])

The LEC algorithm is first published by Marcelloni and Veechio in [66] under the name, a simple algorithm for data compression in wireless sensor networks. A year later, a more complete paper is published in [67], naming the algorithm LEC. As the first publication suggests, the algorithm is simple and only requires a few steps. First the input data is differentiated using [67, Equation D.21], with x_{-1} taken as the central value of the 2^R (R can be the precision of the ADC) possible discrete values. The differentiated signal is then represented by two parts $\ll s_i \mid a_i \gg$. The first part s_i is coded using Huffman coding, as can be seen in [67, Table D.18], whereby each value of s_i depends on the value d_i. The Huffman coding is based on the JPEG algorithm for compressing digital images [75]. The value n_i is needed for a_i as can be seen later on. It should be noted first, that the formula is adapted from the original paper, the original paper describes $n_i = \lceil \log_2(\mid d_i \mid \rceil$ and gives a pseudo-code description of $\lceil \log_2(\mid d_i \mid) \rceil$, which is mathematically speaking equal to $\lceil \log_2(\mid d_i \mid +1) \rceil$, therefore the mathematical representation is used here for n_i.

$$d_i = x_i - x_{i-1} \tag{D.21}$$

$$n_i = \lceil \log_2(\mid d_i \mid +1) \rceil \tag{D.22}$$

Next the value a_i is of variable length and represented as follows:

1. if $d_i > 0$, a_i is the representation of the n_i lower-order bits of d_i in two's complement [67].
2. if $d_i < 0$, a_i is the representation of the n_i lower-order bits of (d_i - 1) in two's complement [67].
3. if $d_i = 0$, a_i is not represented [67].

Note, that the LEC algorithm also discusses non 2's complement representations, due to the sensors used in the pants using 2's complement notation, this is not discussed. To continue, an example is given in Table D.19. The decompression is easily performed by checking for the unique Huffman code to find s_i, from s_i the length of a_i can be found as shown in Table D.19, which is equal to n_i.

The algorithm is tested on the following data sets: SensorScope (including seismic data), Seismic data from OhioSeis Ditial Seismographic Station (link offline) and ECG recording from MIT-BIH Arrhytmia Database [6]. The use of one fixed Huffman table is also considered by the authors, using the

n_i	s_i	d_i
0	00	0
1	010	-1,+1
2	011	-3,-2,+2,+3
3	100	-7,. . . ,-4,+4,. . . ,+7
4	101	-15,. . . ,-8,+8,. . . ,+15
5	110	-31,. . . ,-16,+16,. . . ,+31
6	1110	-63,. . . ,-32,+32,. . . ,+63
7	11110	-127,. . . ,-64,+64,. . . ,+127
8	111110	-255,. . . ,-128,+128,. . . ,+255
9	1111110	-511,. . . ,-256,+256,. . . ,+511
10	11111110	-1023,. . . ,-512,+512,. . . ,+1023
11	111111110	-2047,. . . ,-1024,+1024,. . . ,+2047
12	1111111110	-4095,. . . ,-2048,+2048,. . . ,+4095
13	11111111110	-8191,. . . ,-4096,+4096,. . . ,+8191
14	111111111110	-16383,. . . ,-8192,+8192,. . . ,+16383

Table D.18: Huffman variable length codes [66]

d_i	n_i	s_i	a_i	$\ll s_i \mid a_i \gg$
0	0	00	-	00
20	5	110	10100	11010100
100	7	11110	1100100	111101100100
-314	9	1111110	011000101	1111110011000101
15030	14	111111111110	11101010110110	11111111111011101010110110

Table D.19: Example of LEC algorithm

semi-adaptive Huffman coding [84]. To continue, the algorithm looks similar to the FELACS [59] algorithm, which also splits the data into two parts. The difference is in the encoding of the s_i part, using Rice-coding for FELACS and Huffman coding for LEC. It can be observed that the Exponential-Golomb code of order 0 in FELACS is just a form of Huffman coding [101]. In terms of ciompression ratio both algorithms score well, with FELACS, performing noticeably better. In [67] LEC is compared with S-LZW [82], Gzip, Bzip2, Rar, classical Huffman encoding and classical arithmetic encoding (discussed in [14, 55]). At that time, those algorithms (except S-LZW) are to large in terms of computational cycles and memory usage and therefore unsuited for low power embedded systems. However, it is observed, that those algorithms outperform LEC significantly in terms of compression ratio, with Bzip2 best performing.

The LEC algorithm shows good performances on compression ratios, it will be given a $++$. As the name suggest, a very few steps are required for computing the values s_i and a_i, therefore a $++$ is given to power consumption. The storing of Table D.18 will only take a few bytes to store, hence a $+++$ is given on memory usage. There is also no packet dependency, due to sending the first sample at the beginning of the packet. The ability to adapt to varying sources is given a $--$, due to the static assumption on the distribution of s_i. A summary of the scores is given in Table D.20.

parameter	score	explanation
compression ratio	$++$	the splitting into $\ll s_i \mid a_i \gg$ works quite well on the sensor data sets
power consumption	$++$	few steps to obtain $\ll s_i \mid a_i \gg$
memory usage	$+++$	a little memory required for Table D.18
packet dependency	$+++$	no packet dependency
varying statistics	$--$	the static distribution of s_i will most likely result in significantly worse performance on faster changing signals

Table D.20: scores for LEC

D.3.6. S-LEC [64]

The S-LEC algorithm, is based on the LEC algorithm [67]. The authors in [64] are trying to improve the LEC algorithm, by challenging the differential predictor ([67, Equation D.21]). The use of the differential predictor makes the underlying assumption of residue independence, because the temporal correlation is already removed by the differential predictor. By adding 2-bits to provide context information about the current and previous sample, more temporal correlation can be used. The method is as follows, a compressed bit is represented by $\ll h_i \mid s_i \mid a_i \gg$ instead of $\ll s_i \mid a_i \gg$ (this notation is different from the notation used in [64], to comply with notation used in the original LEC algorithm). By using the value h_i, the algorithm is able to distinguish in which range the value s_i will be. The ranges are given in [64, Table D.21] and it can be seen, that if the value of s_i falls in the intervals represented by $[n_{i-1}-1, n_{i-1}, n_{i-1}+1]$, given in [64, Table D.18], leads to the shorter representation $\ll h_i \mid a_i \gg$ instead of $\ll h_i \mid s_i \mid a_i \gg$. This saves bits needed for representing s_i, when $n_i \geq 1$. However, overhead is present when $h_i = 11$, because s_i is sent with the extra bits of h_i. To lower, this effect (e.g. $h_i = 11$), the following is proposed. In case of $n_i > n_{i-1}$, the value of s_i can be represented with fewer bits, by splitting the data into three clusters: $C_1 = \{n_i \mid i = 0, 1, 2, 3\}$, $C_2 = \{n_i \mid i = 4, 5\}$ and $C_3 = \{n_i \mid i = 6, ..., N\}$, with N the number of bits used by the ADC. Because the value of n_{i-1} and h_i is known, it is logical that a value of $s_i = 4$ is infeasible if $s_{i-1} = 4$. Therefore it can be assumed that the original $s_i = 8$, is represented more efficiently, due to the information on the values n_{i-1} and h_i. The value for s_i would fall under cluster C_2 and can hence save 2 bits in representing s_i. An example of the algorithm can be seen in Table D.22.

h_i	Context Information	
00	$s_i = s_{i-1}$	same group
01	$$s_i = \begin{cases} s(n_{i-1}-1), & n_{i-1} \geq 1 \\ s(2), & n_{i-1} = 0 \end{cases}$$	neighbouring group
10	$$s_i = \begin{cases} s(n_{i-1}+1), & n_{i-1} < N \\ s(N-2), & n_{i-1} = N \end{cases}$$	neighbouring group
11	s_i can not be omitted in codeword, and $\ll h_i \mid s_i \mid a_i \gg$ is required	otherwise

Table D.21: sequential coding in S-LEC [64]

value	h_i	s_i	a_i	n_i
9	– (not needed, for first sample)	101	1001	4
128	11	1110 (n_{i-1} = 4 and n_i = 8, hence $C_i = 4$, thus s_i reduces from 111110 to 1110)	10000000	8
-130	00 ($s_i = s_{i-1}$, also implies $n_i = n_{i-1}$)	–	01111101	8
16	11	110	10000	5
-32	10 (n_{i-1} = 5 and n_i = 6, thus h_i = 10)	–	011111	6

Table D.22: Example S-LEC on {9, 128, -130, 16, -32}

S-LEC is compared with LEC [67] and S-LZW [82], using data sets from SensorScope and Volcanic monitoring [112, 113]. The Volcanic data set 2005-08-11_03.36.40 (link offline) is sampled at 100 Hz with an ADC of 24 bits. Due to the highly dynamic data from volcanic eruptions, data compression is more challenging. It is observed that S-LEC performs slightly better on the SensorScope data sets. However, on the volcanic data sets, the S-LEC algorithm outperforms the LEC dramatically. This failing

is most likely caused by the assumption made on the distribution. The assumed distribution by LEC is a Laplace-like distribution, due to the use of Rice coding [84]. The SensorScope data sets, satisfies this assumption, however the volcanic data set does not seem to satisfy this property. The lack of the LEC algorithm in adapting to varying source statistics was already discusses in Section D.3.5.
The S-LEC performs very well in terms of data compression, resulting in a score of $+++$. Due to the still simplistic approach in representing the data in $\ll h_i \mid s_i \mid a_i \gg$, which can easily be determined by a few operations, gives a score of $+++$ on power consumption. The algorithm does not require much memory to store the Huffman coding table, getting a $+++$ on memory usage. To continue, there is no packet dependency, because the algorithm sends the first raw sample at the beginning of each packet. Finally the algorithm is assumed to handle different source statics quite well. It is already shown to perform better on multiple data sets. However, data sets with low values of s_i or fast changing values of s_i, can still lead to extra overhead by h_i and even perform worse than the original LEC algorithm. These cases will most likely be rare and therefore a $++$ is given. The summary of the scores can be seen in Table D.23.

parameter	score	explanation
compression ratio	$+++$	the splitting into $\ll h_i \mid s_i \mid a_i \gg$ works very well on the sensor data sets
power consumption	$+++$	few steps to obtain $\ll h_i \mid s_i \mid a_i \gg$
memory usage	$+++$	a little memory required for Table D.18
packet dependency	$+++$	no packet dependency
varying statistics	$++$	the ability to adapt on the range of s_i improves performance for multiple data sets

Table D.23: scores for S-LEC

D.3.7. Simple algorithm - Kolo [58]

The Simple algorithm proposed by Kolo et al. [57] is also using Huffman coding tabbles. By making use of the same principle as the LEC algorithm proposed in [66, 67]. First the derivative is calculates as shown in [67, Equation D.21]. Next, two scenarios are implemented and tested. One in which a Huffman coding table is used ([58, Table D.25], optimized for WSNs based on the entropy encoder in the LEC algorithm [67]). As an improvement, the use of two different Huffman coding tables is used. Both tables assume a different geometrical distribution of the source data. The first will assume medium and high correlation in the data ([58, Table D.26]). The other will assume low to medium correlation ([58, Table D.27]). By combining the two Huffman coding schemes an adaptive Huffman compression technique is introduced. In the header of the packet a '0' or '1' is used to indicate which Huffman table is used to code the packet. The algorithm is performed on data from SensorScope (Temp-LU-84 and RH-LU-84), soil temperature measurements (link offline) and seismic data (link offline). The algorithm is tested on different block sizes varying from 1 till 256 (a block is referred to as a sample point of 16 bit). The best performing block size is taken and compared with the LEC algorithm. It is interesting to see, that the adaptive Huffman coding using a block size of three, already outperforms the LEC algorithm. The use of one static Huffman coding table ([58, Table D.25]) already shows a slightly better performance compared to LEC. But, the use of adaptive Huffman coding is preferred.
The scores observed by the simple algorithm are very good for the SensorScope data set, leading to a score of $+++$. The use of multiple Huffman coding, will lead to extra computation cycles, compared to LEC. However, this will double the amount of computation cycles approximately, and can therefore still obtain a $+++$ on power consumption. The need for storing the two Huffman tables will introduce a small amount of extra overhead, giving a $+++$ on memory usage. Also each packet can be decompressed separately, because the first sample is send as raw value. Because an assumption is made on 2 types of distributions, the ability to adapt to varying source statistics will get a $-$. The summary of the scores can be seen in Table D.24.

D.3.8. ALDC [56]

The adaptive lossless data compression (ALDC), is based on the same principle as discussed in the Simple algorithm - Kolo [58]. The algorithm performs the following steps, shown in [56, Figure D.9]:

parameter	score	explanation
compression ratio	+ + +	The use of two different Huffman codes performs well on the SensorScope data set
power consumption	+ + +	double computation cycles compared to LEC
memory usage	+ + +	a little memory required for Table D.25 and D.26
packet dependency	+ + +	no packet dependency
varying statistics	–	2-static distributions are assumed

Table D.24: scores for Simple algorithm

n_i	s_i	d_i
0	100	0
1	110	-1,+1
2	00	-3,-2,+2,+3
3	111	-7,. . . ,-4,+4,. . . ,+7
4	101	-15,. . . ,-8,+8,. . . ,+15
5	010	-31,. . . ,-16,+16,. . . ,+31
6	0111	-63,. . . ,-32,+32,. . . ,+63
7	01101	-127,. . . ,-64,+64,. . . ,+127
8	011001	-255,. . . ,-128,+128,. . . ,+255
9	0110001	-511,. . . ,-256,+256,. . . ,+511
10	01100001	-1023,. . . ,-512,+512,. . . ,+1023
11	011000001	-2047,. . . ,-1024,+1024,. . . ,+2047
12	01100000000	-4095,. . . ,-2048,+2048,. . . ,+4095
13	01100000001	-8191,. . . ,-4096,+4096,. . . ,+8191
14	01100000010	-16383,. . . ,-8192,+8192,. . . ,+16383

Table D.25: Huffman Coding Table 1 [58]

n_i	s_i	d_i
0	00	0
1	01	-1,+1
2	11	-3,-2,+2,+3
3	101	-7,. . . ,-4,+4,. . . ,+7
4	1001	-15,. . . ,-8,+8,. . . ,+15
5	10001	-31,. . . ,-16,+16,. . . ,+31
6	100001	-63,. . . ,-32,+32,. . . ,+63
7	1000001	-127,. . . ,-64,+64,. . . ,+127
8	10000001	-255,. . . ,-128,+128,. . . ,+255
9	1000000000	-511,. . . ,-256,+256,. . . ,+511
10	10000000010	-1023,. . . ,-512,+512,. . . ,+1023
11	10000000011	-2047,. . . ,-1024,+1024,. . . ,+2047
12	10000000100	-4095,. . . ,-2048,+2048,. . . ,+4095
13	10000000101	-8191,. . . ,-4096,+4096,. . . ,+8191
14	10000000110	-16383,. . . ,-8192,+8192,. . . ,+16383

Table D.26: Huffman Coding Table 2 [58]

1. Calculate the differences using equation D.21.

2. Select Code option 1 (2-Huffman Table, [56, Table D.29 and D.30]) Table or Code option 2 (3-Huffman Table, [56, Table D.29, D.30 and D.31]), based on fixed decision regions.

3. Encode using the best Table option.

The selection of Code option 1 or 2 is based on a heuristic approach. The values of a block of n-data points is summed up. If the summation is between 3n and 12n, Code option 2 is selected, otherwise

n_i	s_i	d_i
0	1101111	0
1	11010	-1,+1
2	1100	-3,-2,+2,+3
3	011	-7,. . . ,-4,+4,. . . ,+7
4	111	-15,. . . ,-8,+8,. . . ,+15
5	10	-31,. . . ,-16,+16,. . . ,+31
6	00	-63,. . . ,-32,+32,. . . ,+63
7	010	-127,. . . ,-64,+64,. . . ,+127
8	110110	-255,. . . ,-128,+128,. . . ,+255
9	110111011	-511,. . . ,-256,+256,. . . ,+511
10	110111001	-1023,. . . ,-512,+512,. . . ,+1023
11	1101110101	-2047,. . . ,-1024,+1024,. . . ,+2047
12	1101110100	-4095,. . . ,-2048,+2048,. . . ,+4095
13	1101110000	-8191,. . . ,-4096,+4096,. . . ,+8191
14	11011100011	-16383,. . . ,-8192,+8192,. . . ,+16383

Table D.27: Huffman Coding Table 3 [58]

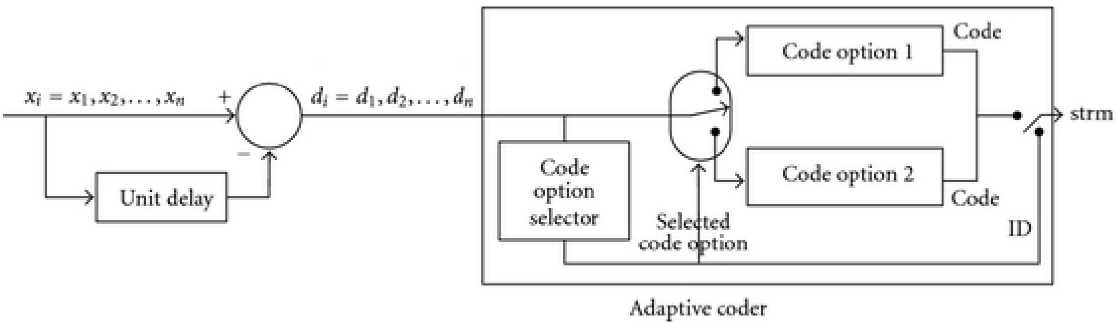


Figure D.9: Functional block diagram of ALDC algorithm using the decision regions approach [56].

Code option 1. To evaluate the effectiveness of the decision region, a brute-force approach is used, whereby the optimal Huffman Table is calculated for all data packets of size n. It turns out that the decision region approach performs similar to the brute force approach. The encoding, is the same as described in the LEC algorithm. To give an example from [56], using the ALDC on x = {8202, 8202, 8202, 8201, 8202, 8202, 8202, 8202, 8208}, performing the following steps:

1. The differential signal is calculated: d = {10, 0 , 0, -1, 1, 0, 0, 6}.
2. The sum of the block is calculated summing the absolute values of the elements in x: 10 + 0 + 0 + 1 + 1 + 0 + 0 + 6 = 18.
3. The region 3n to 12n is given by 24 to 96, for n = 8. Hence the value 18 falls not within this region, thus Code option 1 is selected.
4. Table A, shows the best encoding performances and is therefore used for encoding the string, and a '0' is added to tell the decoder Code option 1 is selected. Another '0' is added to tell that Table A is used to encode.
5. The final result is given by:
   "0 0 1001 1010 00 00 01 0 01 1 00 00 101 110"
   Whereby, the blue and purple parts represent s_i and a_i, respectively.

The ALDC algorithm is tested on the SensorScope and OhioSeis Ohio Seismic data set. The compression ratio is very good on the SensorScope data set, resulting in a + + +. The amount of computation cycles is most likely a factor 2,5 till 3,5 higher than the LEC algorithm. This will result in a score of ++ in power consumption. The saving of the three Huffman tables will only introduce a little bit of extra memory, hence a + + + is given on memory usage. Finally, the ability to adapt to varying source statistics, is given a +, due to the use of 3-static distributions. The scores are summarized in Table D.28.

parameter	**score**	**explanation**
compression ratio	+ + +	The use of three different Huffman codes performs well on the SensorScope data set
power consumption	++	2.5 till 3.5 more computation cycles compared to LEC
memory usage	+ + +	a little memory required for Table D.29, D.30 and D.31
packet dependency	+ + +	no packet dependency
varying statistics	+	3-static distributions are assumed

Table D.28: scores for ALDC

n_i	s_i	d_i
0	00	0
1	01	-1,+1
2	11	-3,-2,+2,+3
3	101	-7,. . . ,-4,+4,. . . ,+7
4	1001	-15,. . . ,-8,+8,. . . ,+15
5	10001	-31,. . . ,-16,+16,. . . ,+31
6	100001	-63,. . . ,-32,+32,. . . ,+63
7	1000001	-127,. . . ,-64,+64,. . . ,+127
8	10000001	-255,. . . ,-128,+128,. . . ,+255
9	1000000000	-511,. . . ,-256,+256,. . . ,+511
10	10000000010	-1023,. . . ,-512,+512,. . . ,+1023
11	10000000011	-2047,. . . ,-1024,+1024,. . . ,+2047
12	10000000100	-4095,. . . ,-2048,+2048,. . . ,+4095
13	10000000101	-8191,. . . ,-4096,+4096,. . . ,+8191
14	10000000110	-16383,. . . ,-8192,+8192,. . . ,+16383

Table D.29: Huffman variable length codes Table A [56]

n_i	s_i	d_i
0	1101111	0
1	11010	-1,+1
2	1100	-3,-2,+2,+3
3	011	-7,. . . ,-4,+4,. . . ,+7
4	111	-15,. . . ,-8,+8,. . . ,+15
5	10	-31,. . . ,-16,+16,. . . ,+31
6	00	-63,. . . ,-32,+32,. . . ,+63
7	010	-127,. . . ,-64,+64,. . . ,+127
8	110110	-255,. . . ,-128,+128,. . . ,+255
9	110111011	-511,. . . ,-256,+256,. . . ,+511
10	110111001	-1023,. . . ,-512,+512,. . . ,+1023
11	1101110101	-2047,. . . ,-1024,+1024,. . . ,+2047
12	1101110100	-4095,. . . ,-2048,+2048,. . . ,+4095
13	1101110000	-8191,. . . ,-4096,+4096,. . . ,+8191
14	11011100011	-16383,. . . ,-8192,+8192,. . . ,+16383

Table D.30: Huffman variable length codes Table B [56]

n_i	s_i	d_i
0	1001	0
1	101	-1,+1
2	00	-3,-2,+2,+3
3	01	-7,. . . ,-4,+4,. . . ,+7
4	11	-15,. . . ,-8,+8,. . . ,+15
5	10001	-31,. . . ,-16,+16,. . . ,+31
6	100001	-63,. . . ,-32,+32,. . . ,+63
7	1000001	-127,. . . ,-64,+64,. . . ,+127
8	10000001	-255,. . . ,-128,+128,. . . ,+255
9	1000000000	-511,. . . ,-256,+256,. . . ,+511
10	10000000010	-1023,. . . ,-512,+512,. . . ,+1023
11	10000000011	-2047,. . . ,-1024,+1024,. . . ,+2047
12	10000000100	-4095,. . . ,-2048,+2048,. . . ,+4095
13	10000000101	-8191,. . . ,-4096,+4096,. . . ,+8191
14	10000000110	-16383,. . . ,-8192,+8192,. . . ,+16383

Table D.31: Huffman variable length codes Table C [56]

D.3.9. Two-Modal Transmission GPC [63, 65]

In [65] it is observed that distributions of sensors usually exhibits 'long tails' and often exhibit a Laplace distribution instead of a geometrical distribution. In order to use this properties efficiently in lossless compression a two-modal transmission scheme is proposed. The algorithm works as follows, a region [-R,R] is selected, in which the symbols are represented using an alphabet of size M+2, whereby M symbols represent the positive integer values of the data and 2 extra characters are used to indicate a minus sign and a raw data point, as will be explained later on. The points outside the [-R,R] interval, will not be encoded using the alphabet with size M+2, but will be send as raw data. In order to distinguish between an encoded- and raw data point, a *placeholder* symbol is used. The *placeholder* tells if the next point should be read as a raw data point or as a compressed one. These raw data points can be send concatenated to the by alphabet encoded string, or just inserted after the *placeholder* character. For negative values, the use of a minus character is required, therefore an alphabet with M symbols, requires M+2 symbols. This is needed for representing the integer values between [0,R] and the overhead of the *placeholder* and minus symbol. Take for example the following input string {**5**, **10**, **6**, **12**, **7**}, with a decimal alphabet (i. e. M = 10) and R = 9. This will give the following result [65]:

{ S(**5**), S(*placeholder*), S(**6**), S(*placeholder*), S(**7**), raw(**10**), raw(**12**) }

with the raw data points concatenated at the end of the data string. It can also be chosen to use [65]:

{ S(**5**), S(*placeholder*), raw(**10**), S(**6**), S(*placeholder*), raw(**12**), S(**7**) }

with the raw data after the *placeholder* symbol. The raw(x) represents the original binary representation of x, and S(y) represents the compressed value of y.
The effectiveness of the algorithm can be calculated theoretically and an optimization can be performed using these formulas to find the optimal M and R. All formulas can be find in the original paper. Tho, a small summary of the approach will be given. First the Laplacian distribution is considered given by [65, Equation D.23].

$$f(x) = \frac{1}{2}(1 + \text{sgn}(x)(1 - \exp\left(\frac{-\mid x \mid}{b}\right)) \tag{D.23}$$

Based on this distribution the new distribution of the symbols in the range [-R,R] can be calculated. Using [65, Equation D.24] the change of the symbol s occurring due to residue r is calculated. For example, M = 10 and R = 20, leads to 2 symbols as representation of the signal, just like the decimal number 20 would take 2 symbols, e.g. a '2' and '0'. Take residue 5, this is represented as follows: rep(5) = S(0)S(5), the length of the symbols for representing the signal is given by [65, Equation D.25], with length l for positive values and l + 1 for negative values, due to the minus symbol.

$$c_s^r = (f(r + 0.5) - f(r - 0.5))N_s(\text{rep}(r)) \tag{D.24}$$

$$I = \lfloor \log_M R \rfloor + 1 \tag{D.25}$$

Based on this the new probability distribution can be calculated from the samples in the region [-R,R]. These will most likely have a better defined Laplacian distribution. Also the probability of symbol *placeholder* occurring can be calculated, as the summation of the probabilities of all samples lying outside the interval [-R,R] in the distribution as shown in [65, Equation D.26].

$$P(S(\textit{placeholder})) = 1 - (f(R + 0.5) - f(-R - 0.5)) \tag{D.26}$$

Based on these formulas the compression ratios are calculated for a fixed M and variable R and vice versa. Leading to heuristic observations for quickly estimating the optimum value of M and R. All samples fall within [-3σ, 3σ], thus R ≤ 3σ [65]. All local maxima are obtained at R = M^k -1, whereby k is an integer starting at 1 and ending when R is larger than 3σ. The algorithm is tested on a real world data set (link offline). The data set is converted back to the original 12-bit representation. The distribution of the data set is estimated on 15 days out of 11 months of data. By calculating the mean square error between the real data and estimated distribution function, the best approximation for the data set using a Laplace distribution is observed. Finally, the optimal values for M and R are calculated, as explained earlier. It is observed, that the compression ratio between predicted and achieved, closely match for the optimal value of M and R. One side note, the model is not fully accurate in the entire subspace of M and R. In [63] the two modal transmission scheme is extended to a unified compression scheme, capable of lossless and lossy compression. The algorithm is also implemented, taking the [-R,R] over the entire interval of the ADC, to make a fair and straightforward comparison with the LEC, the same differential predictor is used. The results obtained are the same as the LEC algorithm, because the same distribution is assumed. In terms of compression ratio the algorithm will score a $+$. The amount of computation cycles will most likely be similar to the LEC and will therefore score a $+$ on power consumption. There is also a small amount of memory required for storing the Huffman table, hence a score of $+++$ is given. There is also no packet dependency, just as the LEC algorithm. Finally, the use of optimizing a data set to match the sensor data, will be the same as using 1-static distribution and hence a score of $--$ is given. Even though, the method for estimating a good distribution is provided, this can lead to potentially over fitting the distribution. A summary of the scores will be given on Table D.32.

parameter	**score**	**explanation**
compression ratio	$++$	same as LEC
power consumption	$++$	same as LEC
memory usage	$+++$	a little memory required for storing one Huffman table
packet dependency	$+++$	no packet dependency
varying statistics	$--$	1-static distributions is optimized

Table D.32: scores for ALDC

D.3.10. Median predictor [69]

The median predictor based algorithm [69], uses the same Huffman codes as the LEC algorithm. However, a median predictor on three samples (x) is used to calculate the differentiated signal. The median predictor will take the median, minimum and maximum of the three samples. It will then categorize the next sample in one of the following four intervals {-2^{n-1}, **min(x)**}, {min(x), **median(x)**}, {**median(x)**, max(x)} and {**max(x)**,2^{n-1} -1}. Based on this categorization, it will calculate the difference between the new point and the point represented in bold, according to the interval the new point belongs to. The use of the median predictor is interesting to discuss, the encoding of the values is performed using a static Huffman table and is already shown to be ineffective. Therefore this algorithm will not be taken into the comparison, due to the lack of proper testing on the SensorScope data set and the low estimated potential of good compression the sensor pants data set.

D.3.11. BWT & MTF [24]

The Burrow Wheeler Transform (BWT) is introduced in 1994 in [24], to sort data more effectively before compressing. The principle is as follows. A data array of size N is taken as input. The data array ('abraca') is first rotated in [24, Table D.33] and secondly sorted in lexicographic order in [24, Table D.34].

row	
0	abraca
1	aabrac
2	caabra
3	acaabr
4	racaab
5	bracaa

Table D.33: BWT rotation operation [24]

row	
0	aabrac
1	abraca
2	acaabr
3	bracaa
4	caabra
5	racaab

Table D.34: BWT lexicographic operation [24]

The transformation pair (L,I) is saved, whereby L is the last colum of the transformation in [24, Table D.34] (L = 'caraab') and I is the row number in which to original word is placed (I = 1). The decompression is performed by first creating vector F, which is the first column in [24, Table D.34]. This column is reconstructed, by placing L in lexicographical order. As next step, a transformation vector T is derived from F and L. The values in T can be found by walking through all elements of L and taking the corresponding row in F. This can be seen in [24, Table D.35]. It can be seen that the element 'c' in row 0 of L can be found in row 4 of F. The 'a' in row 1 in L, can first be found in row 0 in F. The second 'a' in L in row 3, can be found in the second 'a' in F in row 1. Finally, the vector T can be used to reconstruct the original word S, using the stored value I. By performing the [24, algorithm 3]. The algorithm applied can be seen in Table D.36. The use of a BWT in lossless data compression, can be beneficial. After a transformation, the output string L will most likely have the property of grouping the same characters. This is caused by the following effect. Consider the word 'the', If the word frequently occurs, multiple lines starting with 'he' are introduced, after performing the rotation and lexicographical operations. Because the part 'he' start with a 'T', a sequence of 'T's will be in L. The BWT will lead to more sequences, containing the same character, which is beneficial for compression. As a follow up step, the use of a move-to-front (MTF) coding [18] is discussed. The MTF coding will start with the alphabet Y, containing the characters present in the source, in lexicographical order. The MTF coding will iterate through all characters present in the input string (in this case L). For every character, the number of characters occurring before the current character in the alphabet, is stored in R, and the current character is placed first in the updated alphabet Y. Take for example L = 'caraab', with alphabet Y = {'a', 'b', 'c', 'r'}. By applying MTF coding, R = {2 1 3 1 0 3}. In the first iteration 'c' has two characters before itself in the alphabet, 'a' and 'b'. The value '2' is stored in R and the alphabet is updated to Y = {'c', 'a', 'b', 'r'}. It can be seen, that if characters occur frequently after each other, this will lead to a high amount of '0' in the vector R and compression can be very efficiently. The BWT transformation works best for larger values of N, several of thousands characters are needed. More importantly for this research, the transformation can also be used for non-text inputs. The use of a BWT is investigated in the S-LZW algorithm in Section D.17. But did not seem to gain much for the use of a Lempel-Ziv-Welch based algorithm. However, this does not mean that it can not be used in combination with other types of encoders, such as Huffman coding.

row	L	F	T
0	c	a	4
1	a	a	0
2	r	a	5
3	a	b	1
4	a	c	2
5	b	r	3

Table D.35: BWT construction T vector [24]

Algorithm 3 Reconstruction S [24]

```
S[N] = L[I]
for i = 0, ... , N-1 do
    index = T[I]
    for j = 0, ... , i do
        index = T[index]
        ++j;
    end for
    S[N-1-i] = L[index]
    ++i
end for
```

i	index	L[index]	S
-	-	-	a
0	0 (T[1])	c	ca
1	0,4	a	aca
2	0,4,2	r	raca
3	0,4,2,5	b	braca
4	0,4,2,5,3	a	braca

Table D.36: Reconstruct S, based on Algorithm 3 [24]

D.3.12. Discussion & Conclusion

The compression algorithms discussed in this chapter, are based on different principles. These principles can be categorized under the term: 'prediction operator' or 'encoder'. These to terms can be seen in all algorithms so far. The first part removes correlations in the signal (prediction operator) and the second part encodes this more compact symbols (encoder). This general classification can be seen in Figure D.10. This general classification gives a feeling in which steps compression algorithms make. Also a list of the algorithms is shown in Table D.37, with the 'remove correlation operator' and 'encoding symbols'. A slight note should be given regarding the term 'range predictor', because this is mentioned for three algorithms, while all algorithms use this in a different way. However, these methods can all be considered as a form of range prediction. The same holds for 'sequence detection', given to the S-LZW and Tunstall algorithm. By comparing both mechanisms, a similarity can be seen. Because, the S-LZW, will detect a sequence of symbols and saves these in the dictionary to compress more efficiently on the next occurrence. The Tunstall algorithm, will save the probability of each symbol based on the past sequence. The probabilities of the symbols is then used to create frequently occurring sequences and represent them more efficiently.

The scores for the different algorithms on the SensorScope data set are summarized in Table D.38. Based on these results, scores are given to the compression ratios in the previous sections. When putting all scores together in Table D.39, a decision can be made for the most optimal algorithm. It can be seen that the S-LEC, ALDC and FELACS algorithm, are the top three best performing algorithms, under the selection criteria. However, the FELACS algorithm is selected as potentially best candidate for compression. Because IMUs produce signals with a very slow and fast changing rate, the algorithm needs to handle this efficiently. Because FELACS has shown to obtain already the highest compression

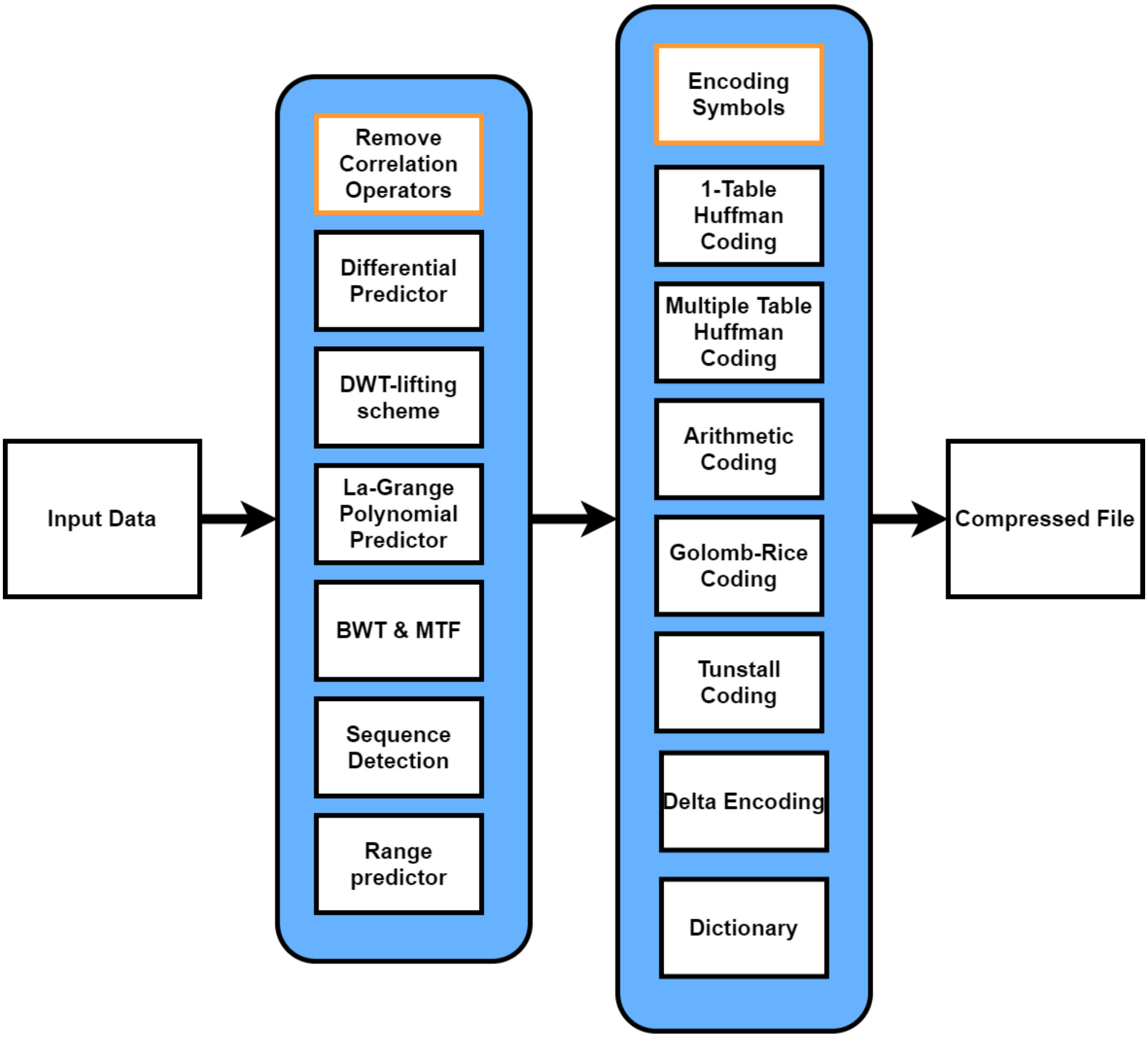


Figure D.10: Generalized coding scheme

Algorithm	**remove correlation operator**	**encoding symbols**
DWT Lifting Scheme	DWT Lifting Scheme	Delta Encoding
Tunstall	Differential Predictor & Sequence detection	Tunstall coding
FELACS	Differential & Range predictor	Golomb-Rice Coding
S-LZW	Sequence Detection	Dictionary
LEC	Differential predictor	1-Table Huffman coding
S-LEC	Differential & Range predictor	1-Table Huffman coding
Simple algorithm	Differential predictor	2-Table Huffman coding
ALDC	Differential predictor	3-Table Huffman coding
Two-Modal	Differential & Range predictor	1-Table Huffman coding

Table D.37: summary of methods used in compression algorithms, split into 'remove correlation operator' and 'encoding symbols'

ratio, with having a good power consumption and memory usage, and without packet dependency. The handling of varying source statistics is assumed to be the best for the FELACS algorithm. The S-LEC and ALDC are also good candidates for the application, but the selection of one algorithm is sufficient enough.

Some remarks before continuing to the next chapter. The compression ratios on the SensorScope data seem to be similar between the algorithms, except the S-LZW. It is important to keep in mind, that there is a significant performance difference if other data sets are used. Unfortunately, these data sets are not taken into the standard data sets and therefore making a comparison more difficult. It is already observed that data from IMUs in football applications can lead to rapidly changing signals. If these parts of the data are compressed at a low compression rate, the whole advantage gets lost. For example, an athlete starts running for a period of 10 seconds. During this movement, higher values coming from the differential predictor are passed to the encoder. In algorithms with a more static assumption on the distribution, results with a low or even negative compression ratio will be given. If an algorithm, takes into account that the range of numbers can be in a higher range, this will lead to an increase in the compression ratio during these moments. The FELACS algorithm will compensate for this, by increasing the value of k. Because the FELACS algorithm uses a differential predictor, it might be sufficient to replace the differential predictor by the DWT Lifting Scheme with a Lagrange interpolation, to improve performance.

	DWT LS [12]	**Tun-stall [70]**	**FE-LACS [59]**	**S-LZW [82]**	**LEC [67]**	**S-LEC [64]**	**Simple algo-rithm Kolo [58]**	**ALDC [56]**	**two modal GPC [63, 65]**
Temp-LU-84	xx.xx	xx.xx	**74.00**	48.99	70.81	72.07	73.48	73.94	70.52
Temp-FN-101	xx.xx	66.12	**67.63**	30.35	65.39	xx.xx	xx.xx	67.48	64.34
Temp-LG-20	xx.xx	54.32	**57.41**	20.02	53.83	54.80	xx.xx	56.90	54.05
RH-LU-84	xx.xx	xx.xx	**66.12**	31.24	62.86	63.71	63.62	65.54	61.95
RH-FN-101	xx.xx	58.20	**67.20**	36.27	62.95	xx.xx	xx.xx	66.33	60.79
RH-LG-20	xx.xx	54.70	**53.85**	21.93	48.67	51.40	xx.xx	52.87	48.82

Table D.38: Compression ratio (%), based on equation D.1 of algorithms using the SensorScope dataset

	DWT LS [12]	**Tun-stall [70]**	**FE-LACS [59]**	**S-LZW [82]**	**LEC [67]**	**S-LEC [64]**	**Simple algo-rithm Kolo [58]**	**ALDC [56]**	**two modal GPC [63, 65]**
compression ratio	$+$	$++$	$\mathbf{+++}$	$--$	$++$	$+++$	$+++$	$+++$	$++$
power consumption	$+$	$-$	$\mathbf{+++}$	$--$	$++$	$+++$	$+++$	$++$	$++$
memory usage	$+++$	$---$	$\mathbf{+++}$	$+++$	$+++$	$+++$	$+++$	$+++$	$+++$
packet dependency	$+++$	$---$	$\mathbf{+++}$	$+++$	$+++$	$+++$	$+++$	$+++$	$+++$
varying source statistics	$++$	$++$	$\mathbf{+++}$	$---$	$--$	$++$	$-$	$+$	$--$

Table D.39: Compression technique selection criteria

D.3.13. Research questions

This subsection will give a brief description of each research question, followed by the actually research question.

The focus of this part of the Thesis is to find a suited compression algorithm for smart sensor pants. This begins with validating the compression ratio during different football sessions, and use this data for predicting the expected load on the data link. This data can then be used to prove the optimality of the encoding part as a form of heuristic proof.

1. What is the compression ratio during different football sessions?
2. Will the FELACS algorithm compress data, representative to football scenarios, optimal? More precisely, will s_i follow an exponential distribution of $p(s_i) = 2^{-s_i}$?

Later on, the trade-off between the compression ratio and memory usage (blocksize) is considered, to select an optimal blocksize.

3. What is the trade-off between the amount of data points per packet (increasing the memory usage), and the optimal compression rate?

After all these steps are performed, it is considered to improve the algorithm by replacing the differential predictor by a DWT Lifting scheme in combination with a La-Grange Polynomial interpolation, used in [12].

4. Will the use of a DWT Lifting scheme with a La-Grange Polynomial interpolation, result in a higher compression ratio, compared to a differential predictor?

Finally, it is interesting to measure or predict the extra power added in the current system, when running a data compression algorithm.

5. What is the extra power consumption in the microcontroller, by running the compression algorithm?

It should be noted though, that the optimality of the encoding part of the FELACS algorithm will be heuristically proven in research question 2, while the ”remove correlation operator” is not proven for optimality. This decision is made, because the encoding part can theoretically be calculated based on Shannon’s entropy limit. While, the optimality of the ”remove correlation operator”, might be less trivial to prove. Therefore an improvement is suggested in research question 4, to replace the differential predictor with a DWT Lifting scheme in combination with a La-Grange Polynomial interpolation.

E

Extra Figures

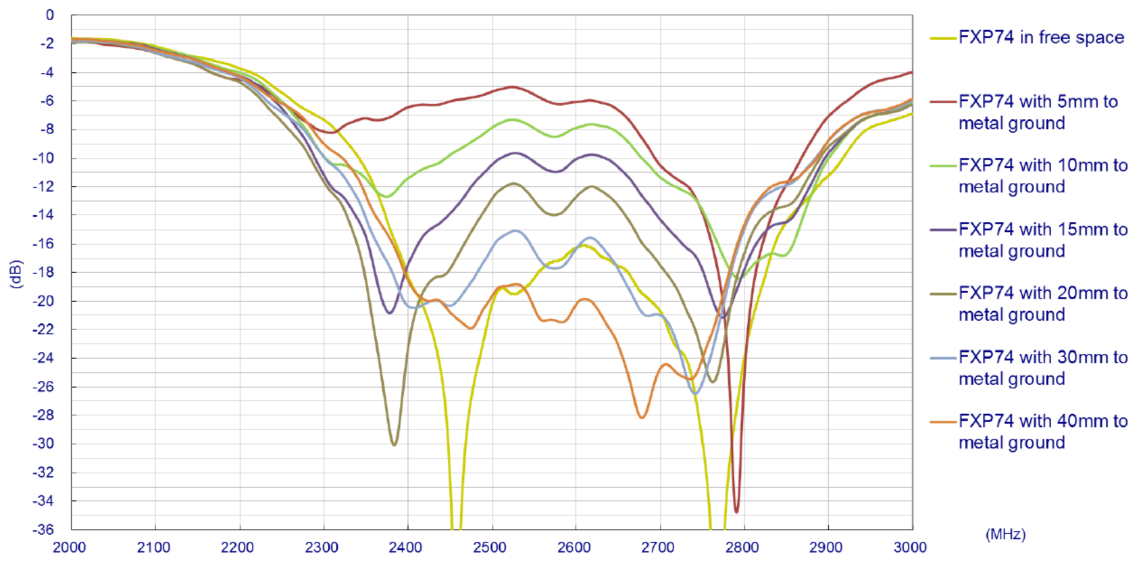


Figure E.1: influence of metal plate on return loss dipole antenna [1]

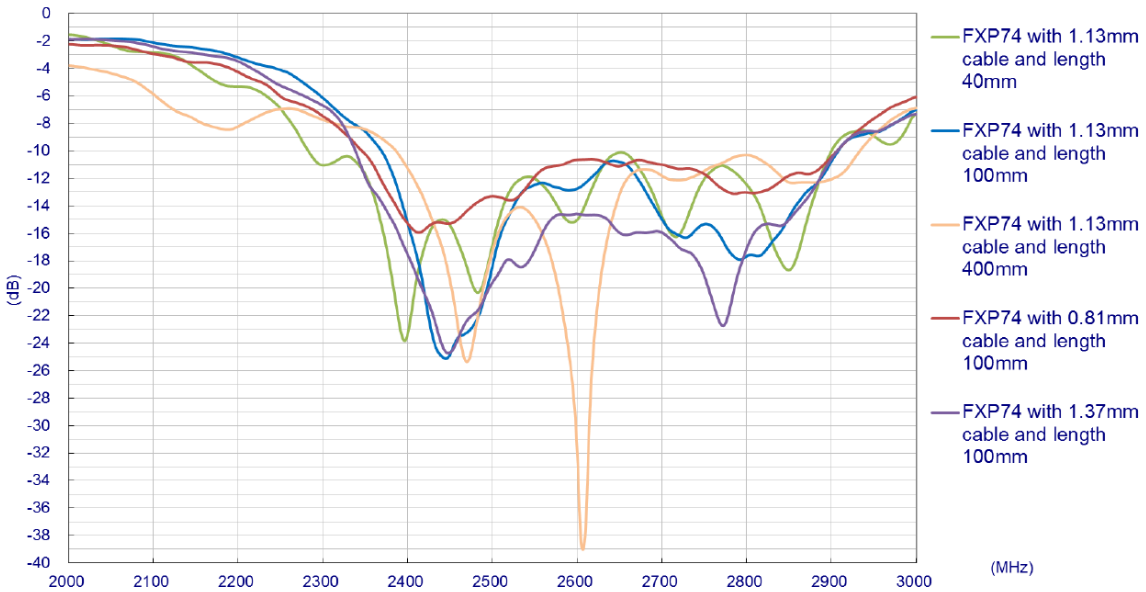


Figure E.2: influence of cable length on return loss dipole antenna [1]

Bibliography

[1] Data sheet fxp74.07.0100a. URL https://nl.mouser.com/datasheet/2/398/FXP74.07.0100A-1508455.pdf.

[2] Data sheet pulselarsen w3230. URL https://nl.mouser.com/datasheet/2/336/W3230-1375653.pdf.

[3] Image xpico240, . URL https://docs.lantronix.com/products/xpico-200/ug/4.2/overview/.

[4] xpico 240 series: Embedded wi-fi iot gateway, . URL https://www.lantronix.com/products/xpico-240/.

[5] xpico240 radio data sheet, . URL https://docs.lantronix.com/products/xpico-200/ug/4.2/network-ifaces/#radio-configuration-settings.

[6] Ecg dataset, 2005. URL https://www.physionet.org/content/mitdb/1.0.0/.

[7] Will Abbott, Thomas E Brownlee, Liam D Harper, Robert J Naughton, Andy Richardson, and Tom Clifford. A season long investigation into the effects of injury, match selection and training load on mental wellbeing in professional under 23 soccer players: A team case study. *European journal of sport science*, 19(9):1250–1256, 2019.

[8] Emad Aboelela. Liftingwise: A lifting-based efficient data processing technique in wireless sensor networks. *Sensors*, 14(8):14567–14585, 2014.

[9] H. Anwer Basha, S. Arivalagan, P. Sudhakar, and R. P. Narmadha. A new deterministic code allocation technique for data compression in wireless sensor networks. In *International Journal of Recent Technology and Engineering (IJRTE)*, pages 80–86. blue Eyes Intelligence Engineering & Sciences Publication, 2019.

[10] Chirag Arora, Shyam Sundar Pattnaik, and Rudra Narayan Baral. Srr inspired microstrip patch antenna array. *Progress In Electromagnetics Research*, 58:89–96, 2015.

[11] Saad Arrabi and John Lach. Adaptive lossless compression in wireless body sensor networks. In *Proceedings of the Fourth International Conference on Body Area Networks*, pages 1–8, 2009.

[12] Joseph Azar, Rony Darazi, Carol Habib, Abdallah Makhoul, and Jacques Demerjian. Using dwt lifting scheme for lossless data compression in wireless body sensor networks. In *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pages 1465–1470. IEEE, 2018.

[13] Constantine A Balanis. *Antenna theory: analysis and design*. John wiley & sons, 2016.

[14] Kenneth C Barr and Krste Asanović. Energy-aware lossless data compression. *ACM Transactions on Computer Systems (TOCS)*, 24(3):250–291, 2006.

[15] G Barrenetxea, J Mezzo, H Dubois-Ferriere, O Couach, M Krichane, M Tromp, H Huwald, M Vetterli, M Parlanges, and J Selker. Sensorcope: A urban environmental monitoring network. *AGUFM*, 2006:H51D–0513, 2006.

[16] Bram Bastiaansen, Michel Brink, Riemer Vegter, and Koen Lemmink. Smart sensor shorts: a novel imu based method to continuously assess the biomechanical training-and match load in team sports. *Crossing Borders in Research on Sport and Physical Activity*, pages 61–65, 2019.

[17] Timothy C Bell, John G Cleary, and Ian H Witten. *Text compression*. Prentice-Hall, Inc., 1990.

[18] Jon Louis Bentley, Daniel D Sleator, Robert E Tarjan, and Victor K Wei. A locally adaptive data compression scheme. *Communications of the ACM*, 29(4):320–330, 1986.

[19] Peter Joseph Bevelacqua. *Antenna Arrays: Performance Limits And Geometry*. PhD thesis, Arizona state university, 2008.

[20] Stephanie Blair, Grant Duthie, Sam Robertson, William Hopkins, and Kevin Ball. Concurrent validation of an inertial measurement system to quantify kicking biomechanics in four football codes. *Journal of biomechanics*, 73:24–32, 2018.

[21] Eric Bodden, Malte Clasen, and Joachim Kneis. Arithmetic coding revealed - a guided tour from theory to praxis. 05 2007.

[22] Steven M Bowers, Amirreza Safaripour, and Ali Hajimiri. Dynamic polarization control. *IEEE Journal of Solid-State Circuits*, 50(5):1224–1236, 2015.

[23] ER Brown, CD Parker, and Eli Yablonovitch. Radiation properties of a planar antenna on a photonic-crystal substrate. *JOSA B*, 10(2):404–407, 1993.

[24] Michael Burrows and David J Wheeler. A block-sorting lossless data compression algorithm. 1994.

[25] Rachel Cardell-Oliver, Stefan Böttcher, and Christof Hübner. Data-aware, resource-aware, lossless compression for sensor networks. In *European Conference on Wireless Sensor Networks*, pages 83–98. Springer, 2013.

[26] Robert E Collin. *Field theory of guided waves*, volume 5. John Wiley & Sons, 1990.

[27] AG Derneryd and AG Lind. Cavity model of the rectangular microstrip antenna. In *Proc. Printed Circuit Antenna Tech. Workshop*, pages 12–1, 1979.

[28] Anders Derneryd and A Lind. Extended analysis of rectangular microstrip resonator antennas. *IEEE transactions on antennas and propagation*, 27(6):846–849, 1979.

[29] M Deshpande and M Bailey. Input impedance of microstrip antennas. *IEEE Transactions on antennas and propagation*, 30(4):645–650, 1982.

[30] Ashay Dhamdhere, Hao Chen, Alex Kurusingal, Vijay Sivaraman, and Alison Burdett. Experiments with wireless sensor networks for real-time athlete monitoring. In *IEEE Local Computer Network Conference*, pages 938–945. IEEE, 2010.

[31] P Dolas and D Ghosh. Distributed compressive data gathering framework for correlated data in wireless sensor networks. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 10(1-6):153–158, 2018.

[32] Jan Ekstrand. Keeping your top players on the pitch: the key to football medicine at a professional level, 2013.

[33] Jan Ekstrand, Martin Hägglund, and Markus Waldén. Injury incidence and injury patterns in professional football: the uefa injury study. *British journal of sports medicine*, 45(7):553–558, 2011.

[34] M. Falahatgar, A. Jafarpour, A. Orlitsky, V. Pichapati, and A. T. Suresh. Universal compression of power-law distributions. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 2001–2005, 2015.

[35] Shahin Farahani. *ZigBee wireless networks and transceivers*. Newnes, 2011.

[36] FreeRTOS. Freertos website. URL `https://www.freertos.org/`. (last accessed: 28-09-2020).

[37] Miguel Garcia, Angel Catalá, Jaime Lloret, and Joel JPC Rodrigues. A wireless sensor network for soccer team monitoring. In *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, pages 1–6. IEEE, 2011.

[38] Solomon Golomb. Run-length encodings (corresp.). *IEEE transactions on information theory*, 12 (3):399–401, 1966.

[39] Ali Grami. Chapter 9 - information theory. In Ali Grami, editor, *Introduction to Digital Communications*, pages 377 – 408. Academic Press, Boston, 2016. ISBN 978-0-12-407682-2. doi: https://doi.org/10.1016/B978-0-12-407682-2.00009-0. URL http://www.sciencedirect.com/science/article/pii/B9780124076822000090.

[40] Alexandre Guitton, Niki Trigoni, and Sven Helmer. Fault-tolerant compression algorithms for delay-sensitive sensor networks with unreliable links. In *International Conference on Distributed Computing in Sensor Systems*, pages 190–203. Springer, 2008.

[41] Mats Gustafsson and Sven Nordebo. Bandwidth, q factor, and resonance models of antennas. *Progress in Electromagnetics Research*, 62:1–20, 2006.

[42] Martin Hägglund, Markus Waldén, Henrik Magnusson, Karolina Kristenson, Håkan Bengtsson, and Jan Ekstrand. Injuries affect team performance negatively in professional football: an 11-year follow-up of the uefa champions league injury study. *British journal of sports medicine*, 47 (12):738–742, 2013.

[43] Shinsuke Hara, Tetsuo Tsujioka, Toui Kanda, Hajime Nakamura, Takashi Kawabata, Kenji Watanabe, Masanao Ise, Noa Arime, and Hiroyuki Okuhata. Development of a real-time vital data collection system from players during a football game. In *2013 IEEE 15th International Conference on e-Health Networking, Applications and Services (Healthcom 2013)*, pages 409–413. IEEE, 2013.

[44] Shinsuke Hara, Kouhei Tezuka, Tetsuo Tsujioka, Hajime Nakamura, Takashi Kawabata, Kenji Watanabe, Masanao Ise, Noa Arime, and Hiroyuki Okuhata. Performance evaluation of packet forwarding methods in real-time vital data collection for players during a football game. In *2014 8th International Symposium on Medical Information and Communication Technology (ISMICT)*, pages 1–5. IEEE, 2014.

[45] Shinsuke Hara, Hiroyuki Yomo, Ryusuke Miyamoto, Yasutaka Kawamoto, Hiroyuki Okuhata, Takashi Kawabata, and Hajime Nakamura. Challenges in real-time vital signs monitoring for persons during exercises. *International Journal of Wireless Information Networks*, 24(2):91–108, 2017.

[46] Carla Hertleer, Hendrik Rogier, Luigi Vallozzi, and Lieva Van Langenhove. A textile antenna for off-body communication integrated into protective clothing for firefighters. *IEEE Transactions on Antennas and Propagation*, 57(4):919–925, 2009.

[47] Shih-Hsun Hsu and Kai Chang. Ultra-thin cpw-fed rectangular slot antenna for uwb applications. In *2006 IEEE Antennas and Propagation Society International Symposium*, pages 2587–2590. IEEE, 2006.

[48] David A Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

[49] Koichiro Inoue, Hiroyuki Nunome, Thorsten Sterzing, Hironari Shinkai, and Yasuo Ikegami. Dynamics of the support leg in soccer instep kicking. *Journal of Sports Sciences*, 32(11):1023–1032, 2014.

[50] William C Jakes and Donald C Cox. *Microwave mobile communications*. Wiley-IEEE Press, 1994.

[51] James R James, Peter S Hall, et al. *Handbook of microstrip antennas*, volume 1. IET, 1989.

[52] Arne Jensen and Anders la Cour-Harbo. *Ripples in mathematics: the discrete wavelet transform*. Springer Science & Business Media, 2001.

[53] David M Kelly, Warren Gregson, Thomas Reilly, and Barry Drust. The development of a soccer-specific training drill for elite-level players. *The Journal of Strength & Conditioning Research*, 27 (4):938–943, 2013.

[54] Timothy F Kennedy, Patrick W Fink, Andrew W Chu, Nathan J Champagne, Gregory Y Lin, and Michael A Khayat. Body-worn e-textile antennas: the good, the low-mass, and the conformal. *IEEE Transactions on Antennas and Propagation*, 57(4):910–918, 2009.

[55] Naoto Kimura and Shahram Latifi. A survey on data compression in wireless sensor networks. In *International Conference on Information Technology: Coding and Computing (ITCC'05)-Volume II*, volume 2, pages 8–13. IEEE, 2005.

[56] Jonathan Gana Kolo, S Anandan Shanmugam, David Wee Gin Lim, Li-Minn Ang, and Kah Phooi Seng. An adaptive lossless data compression scheme for wireless sensor networks. *Journal of Sensors*, 2012, 2012.

[57] Jonathan Gana Kolo, Li-Minn Ang, Kah Phooi Seng, and SRS Prabaharan. Performance comparison of data compression algorithms for environmental monitoring wireless sensor networks. *International journal of computer applications in technology*, 46(1):65–75, 2013.

[58] Jonathan Gana Kolo, Li-Minn Ang, S Anandan Shanmugam, David Wee Gin Lim, and Kah Phooi Seng. A simple data compression algorithm for wireless sensor networks. In *Soft computing models in industrial and environmental applications*, pages 327–336. Springer, 2013.

[59] Jonathan Gana Kolo, S Anandan Shanmugam, David Wee Gin Lim, and Li-Minn Ang. Fast and efficient lossless adaptive compression scheme for wireless sensor networks. *Computers & Electrical Engineering*, 41:275–287, 2015.

[60] Anton Kos, Veljko Milutinović, and Anton Umek. Challenges in wireless communication for connected sensors and wearable devices used in sport biofeedback applications. *Future generation computer systems*, 92:582–592, 2019.

[61] Peter Kovacs and Tomáš Urbanec. Electromagnetic band gap structures: Practical tips and advice for antenna engineers. *Radioengineering*, 21(1), 2012.

[62] Alexander Kurusingal, Ashay Dhamdhere, and Vijay Sivaraman. Modeling signal strength of body-worn devices. In *IEEE Local Computer Network Conference*, pages 244–247. IEEE, 2010.

[63] Yao Liang. Efficient temporal compression in wireless sensor networks. In *2011 IEEE 36th Conference on Local Computer Networks*, pages 466–474. IEEE, 2011.

[64] Yao Liang and Yimei Li. An efficient and robust data compression algorithm in wireless sensor networks. *IEEE Communications Letters*, 18(3):439–442, 2014.

[65] Yao Liang and Wei Peng. Minimizing energy consumptions in wireless sensor networks via two-modal transmission. *ACM SIGCOMM Computer Communication Review*, 40(1):12–18, 2010.

[66] Francesco Marcelloni and Massimo Vecchio. A simple algorithm for data compression in wireless sensor networks. *IEEE communications letters*, 12(6):411–413, 2008.

[67] Francesco Marcelloni and Massimo Vecchio. An efficient lossless compression algorithm for tiny nodes of monitoring wireless sensor networks. *the computer journal*, 52(8):969–987, 2009.

[68] NT Markad, RD Kanphade, and DG Wakade. Design of cavity model microstrip patch antenna. *Computer Engineering and Intelligent Systems*, 6(4):1–13, 2015.

[69] Ashish K Maurya and Dinesh Singh. Median predictor based data compression algorithm for wireless sensor network. *International Journal of Computer Applications*, 975:8887, 2011.

[70] Shabana Mehfuz and Usha Tiwari. A tunstall based lossless compression algorithm for wireless sensor networks. In *2015 Annual IEEE India Conference (INDICON)*, pages 1–4. IEEE, 2015.

[71] Farrokh F Mohammadzadeh, Shijing Liu, Kyle A Bond, and Chang S Nam. Feasibility of a wearable, sensor-based motion tracking system. *Procedia Manufacturing*, 3:192–199, 2015.

[72] Giuseppina Monti, Laura Corchia, and Luciano Tarricone. Logo antenna on textile materials. In *2014 44th European Microwave Conference*, pages 516–519. IEEE, 2014.

[73] myoMotion. myomotion research pro imu. URL https://www.noraxon.com/our-products/research-pro-imu/#1541097779421-89a192e6-7d8d. (last accessed: 26-03-2020).

[74] Vasiliki Paraforou. Design and full-wave analysis of supershaped patch antennas. 2013.

[75] William B Pennebaker and Joan L Mitchell. *JPEG: Still image data compression standard*. Springer Science & Business Media, 1992.

[76] Haul Pues and Antoine Van de Capelle. Accurate transmission-line model for the rectangular microstrip antenna. In *IEE Proceedings H (Microwaves, Optics and Antennas)*, volume 131, pages 334–340. IET, 1984.

[77] Theodore S Rappaport et al. *Wireless communications: principles and practice*, volume 2. prentice hall PTR New Jersey, 1996.

[78] Andreas Reinhardt, Delphine Christin, Matthias Hollick, and Ralf Steinmetz. On the energy efficiency of lossless data compression in wireless sensor networks. In *2009 IEEE 34th Conference on Local Computer Networks*, pages 873–880. IEEE, 2009.

[79] Robert F Rice. Some practical universal noiseless coding techniques. 1979.

[80] Daniel Roetenberg, Per J Slycke, and Peter H Veltink. Ambulatory position and orientation tracking fusing magnetic and inertial sensing. *IEEE Transactions on Biomedical Engineering*, 54(5): 883–890, 2007.

[81] Manidipa Roy and Ashok Mittal. Surface wave suppression in lhcp microstrip patch antenna embedded on textured pin subsrtate. *Progress In Electromagnetics Research*, 89:171–180, 2019.

[82] Christopher M Sadler and Margaret Martonosi. Data compression algorithms for energy-constrained devices in delay tolerant networks. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 265–278, 2006.

[83] Abdeldjalil Saidani, Jianwen Xiang, and Deloula Mansouri. A new lossless compression scheme for wsns using rle algorithm. In *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pages 1–6. IEEE, 2019.

[84] David Salomon. *Data Compression: The Complete Reference*. Number 4. Springer, London, UK, 2006.

[85] Pekka Salonen and L Hurme. A novel fabric wlan antenna for wearable applications. In *IEEE Antennas and Propagation Society International Symposium. Digest. Held in conjunction with: USNC/CNC/URSI North American Radio Sci. Meeting (Cat. No. 03CH37450)*, volume 2, pages 700–703. IEEE, 2003.

[86] Pekka Salonen, Yahya Rahmat-Samii, Heli Hurme, and Markku Kivikoski. Dual-band wearable textile antenna. In *IEEE Antennas and Propagation Society Symposium, 2004.*, volume 1, pages 463–466. IEEE, 2004.

[87] S Sankaralingam and Bhaskar Gupta. Development of textile antennas for body wearable applications and investigations on their performance under bent conditions. *Progress In Electromagnetics Research*, 22:53–71, 2010.

[88] Subramanian Rama Sankaranarayanan. Wireless sensor platform for sporting applications, 2011.

[89] Khalid Sayood. *Introduction to data compression*. Morgan Kaufmann, 2006.

[90] SENSIRION. Sht1x / sht7x humidity & temperature senso, 2007. URL `https://www.mouser.com/datasheet/2/682/\Sensirion_Humidity_SHT7x_Datasheet_V5\protect\discretionary{\char\hyphenchar\font}{}{}469726.pdf`.

[91] Dan Sievenpiper, Lijun Zhang, Romulo FJ Broas, Nicholas G Alexopolous, and Eli Yablonovitch. High-impedance electromagnetic surfaces with a forbidden frequency band. *IEEE Transactions on Microwave Theory and techniques*, 47(11):2059–2074, 1999.

[92] Vijay Sivaraman, Sarthak Grover, Alexander Kurusingal, Ashay Dhamdhere, and Alison Burdett. Experimental study of mobility in the soccer field with application to real-time athlete monitoring. In *2010 IEEE 6th International Conference on Wireless and Mobile Computing, Networking and Communications*, pages 337–345. IEEE, 2010.

[93] David B Smith. Multiple branch switched antenna diversity in rayleigh and rician fading channels. In *IEEE 54th Vehicular Technology Conference. VTC Fall 2001. Proceedings (Cat. No. 01CH37211)*, volume 3, pages 1799–1803. IEEE, 2001.

[94] Christoph Sommer, Stefan Joerer, and Falko Dressler. On the applicability of two-ray path loss models for vehicular network simulation. In *2012 IEEE Vehicular Networking Conference (VNC)*, pages 64–69. IEEE, 2012.

[95] Roman Starosolski and Władysław Skarbek. Modified golomb-rice codes for lossless compression of medical images. In *Proceedings of International Conference on E-health in Common Europe, Cracow, Poland*, pages 423–37, 2003.

[96] Janine H Stubbe, Anne-Marie MC van Beijsterveldt, Sissi van der Knaap, Jasper Stege, Evert A Verhagen, Willem Van Mechelen, and Frank JG Backx. Injuries in professional male soccer players in the netherlands: a prospective cohort study. *Journal of athletic training*, 50(2):211–216, 2015.

[97] Wim Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and computational harmonic analysis*, 3(2):186–200, 1996.

[98] Wim Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM journal on mathematical analysis*, 29(2):511–546, 1998.

[99] Robert Szewczyk, Alan Mainwaring, Joseph Polastre, John Anderson, and David Culler. An analysis of a large scale habitat monitoring application. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 214–226, 2004.

[100] Wolfgang Teufl, Markus Miezal, Bertram Taetz, Michael Fröhlich, and Gabriele Bleser. Validity of inertial sensor based 3d joint kinematics of static and dynamic sport and physiotherapy specific movements. *PloS one*, 14(2), 2019.

[101] Jukka Teuhola. A compression method for clustered bit-vectors. *Information processing letters*, 7(6):308–311, 1978.

[102] C Tharini and P Vanaja Ranjan. Design of modified adaptive huffman data compression algorithm for wireless sensor network. *Journal of Computer Science*, 5(6):466, 2009.

[103] Anneleen Tronquo, Hendrik Rogier, Carla Hertleer, and Lieva Van Langenhove. Robust planar textile antenna for wireless body lans operating in 2.45 ghz ism band. *Electronics letters*, 42(3): 142–143, 2006.

[104] Brian Parker Tunstall. *Synthesis of noiseless compression codes*. PhD thesis, Georgia Institute of Technology, 1967.

[105] J Uthayakumar, T Vengattaraman, and P Dhavachelvan. A new lossless neighborhood indexing sequence (nis) algorithm for data compression in wireless sensor networks. *Ad Hoc Networks*, 83:149–157, 2019.

[106] J.F.J.A.(Jasper) van Zon. Validation of an inertial measurement unit system: Estimation of biomechanical workload in soccer. July 2019.

[107] Patrick Vaudon, Thierry Aubreton, Philippe Dufrane, and Bernard Jecko. Influence of the ground plane structure on the radiation pattern of microstrip antennas. In *Annales des télécommunications*, volume 48, pages 319–329. Springer, 1993.

[108] Vicon. Introducing blue trident. URL `https://www.vicon.com/hardware/blue-trident/`. (last accessed: 26-03-2020).

[109] S. S Vinasthamby. Smart sensor shorts:prevention of hamstring injuries in professional and recreational football athletes by analyzing and monitoring kinematic data. June 2019.

[110] Markus Waldén, Martin Hägglund, and Jan Ekstrand. Injuries in swedish elite football—a prospective study on injury definitions, risk for injury and injury pattern during 2001. *Scandinavian journal of medicine & science in sports*, 15(2):118–125, 2005.

[111] Terry A. Welch. A technique for high-performance data compression. *Computer*, (6):8–19, 1984.

[112] Geoff Werner-Allen, Konrad Lorincz, Jeff Johnson, Jonathan Lees, and Matt Welsh. Fidelity and yield in a volcano monitoring sensor network. In *Proceedings of the 7th symposium on Operating systems design and implementation*, pages 381–396, 2006.

[113] Geoffrey Werner-Allen, Konrad Lorincz, Mario Ruiz, Omar Marcillo, Jeff Johnson, Jonathan Lees, and Matt Welsh. Deploying a wireless sensor network on an active volcano. *IEEE internet computing*, 10(2):18–25, 2006.

[114] Xsens. Wearable sensor platform. URL `https://www.xsens.com/xsens-dot`. (last accessed: 26-03-2020).

[115] Michel Daoud Yacoub. *Foundations of mobile radio engineering*. CRC press, 1993.

[116] Lin Yue. Analysis of generalized selection combining techniques. In *VTC2000-Spring. 2000 IEEE 51st Vehicular Technology Conference Proceedings (Cat. No. 00CH37026)*, volume 2, pages 1191–1195. IEEE, 2000.

[117] Pei Zhang, Christopher M Sadler, Stephen A Lyon, and Margaret Martonosi. Hardware design experiences in zebranet. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 227–238, 2004.

[118] Shaozhen Zhu and Richard Langley. Dual-band wearable textile antenna on an ebg substrate. *IEEE transactions on Antennas and Propagation*, 57(4):926–935, 2009.