Sensor Based Dairy Cow Estrus Detection

Miika Ihonen

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 16.1.2015

Thesis supervisor:

Prof. Arto Visala

Thesis advisor:

M.Sc. (tech.) Samuli Mäkinen

Aalto University
School of Electrical
Engineering

AALTO-YLIOPISTO
SÄHKÖTEKNIIKAN KORKEAKOULU

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä: Miika Ihonen

Työn nimi: Sensor Based Dairy Cow Estrus Detection

Päivämäärä: 16.1.2015 Kieli: Englanti Sivumäärä: 7+61

Department of Automations and Systems Technology

Professuuri: Jotakin kivaa

Työn valvoja: Prof. Arto Visala

Työn ohjaaja: M.Sc. (tech.) Samuli Mäkinen

Tässä opinnäytetyössä tutkitaan sensoreihin persutuvia lypsylehmän kiimantunnistusmentelmiä. Tutkimusta varten kerättiin kiihtyvyys- ja lämpöanturidataa lehmiltä kaulalle kinntityillä datan keruulaitteilla. Työn lopputuloksena on kolme erilaista kiihtyvyysanturin dataan perustuvaa lgoritmia. Kaikki algoritmit suoritutuivat kiimantunnistuksesta. Kuitenkin passiivisuuden tunnistukseen perustuva algoritmi osottautui kaikista luotettavimmaksi ja varmimmaksi erilaisilla parametreilla. Työn johtopäätöksenä on, että lypsylehmän kiima on tunnistettavissa kiihtyvyysanturiin perustuvalla sensorilaitteella. Kuitenkin varmojen johtopäätösten tekemiseksi, tämän työn tulokset tulisi vielä vahvistaa onnistuneilla siemennyksillä.

Avainsanat: Kiimantarkkailu, Lypsylehmä,
Niska-anturi

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Miika Ihonen

Title: Sensor Based Dairy Cow Estrus Detection

Date: 16.1.2015 Language: English Number of pages: 7+61

Department of Radio Science and Technology

Professorship: Circuit theory

Supervisor: Prof. Arto Visala

Advisor: M.Sc. Samuli Mäkinen

This research studies sensor based dairy cow estrus detection. For the study, we recorded motion and temperature data with a collar sensor. The data was used in algorithm development end evaluation. In result, we developed three different algorithms, all suitable for micro-controller devices. All the developed algorithms succeeded in the estrus detection against a reference system. However, inactivity detection based algorithm was the most reliable and tolerant to different configurations. In conclusion, the estrus is detectable with accelerometer based sensors. However, in order to make secure conclusions, the results of this study should be verified by a successful insemination.

Keywords: Estrus Detection, Dairy Cow,
Collar Sensor

Prologue

I would like to thank you everyone. [42] [11]

Otaniemi, 16.1.2017

Miika S. Ihonen

Contents

Symbols and Abbreviations

Symbols

g	acceleration unit $\approx 9.81[\mathrm{m/s^2}]$
$\mathbf{B}$	magneettivuon tiheys
c	valon nopeus tyhjössä $\approx 3 \times 10^8[\mathrm{m/s}]$
ω_D	Debye-taajuus
ω_latt	hilan keskimääräinen fononitaajuus
$\uparrow$	elektronin spinin suunta ylöspäin
$\downarrow$	elektronin spinin suunta alaspäin

Operators

$\nabla \times \mathbf{A}$	vektorin $\mathbf{A}$ roottori
$\dfrac{\mathrm{d}}{\mathrm{d}t}$	derivaatta muuttujan t suhteen
$\dfrac{\partial}{\partial t}$	osittaisderivaatta muuttujan t suhteen
$\sum_i$	summa indeksin i yli
$\mathbf{A} \cdot \mathbf{B}$	vektorien $\mathbf{A}$ ja $\mathbf{B}$ pistetulo

Abbreviations

MCU	Micro-Controller Unit (or micro-controller)
SD	Secure Digital
SDHC	High Capacity Secure Digital
SDXC	Extended Capacity Secure Digital
SPI	Serial Peripheral Interface bus
$\mathrm{I^2C}$	Inter-Integrated Circuit bus (also IIC)
EEPROM	Erasable Programmable Read-Only Memory
SRAM	Static Random-Access Memory
FIR	Finite Impulse Response
IIR	Infinite Impulse Response
FIFO	First In, First Out
IDE	Integrated Development Environment
FFT	Fast Fourier Transform
RISC	Reduced Instruction Set Computer
CISC	Complex Instruction Set Computer
USB	Universal Serial Bus
ISR	Interrupt Service Routine
I/O	Input/Output
LSB	Least Significant Bit
MSB	Most Significant Bit

1 Introduction

People have been using milk as a source of nutrition since the beginning of animal husbandry. The research show that the drinking of milk started 8000 years ago in the present Turkey area. Ever since the dairy farming has spread all over the world. Meanwhile, the spectrum of the available dairy products and fabrication techniques have been increasing. Initially, the headcount of a cattle was small, and it served only a family or a local community. However, since industrialization the farm sizes increased whereas, the number of farms begun to decrease. Same trend has been going on thereafter. Nowadays, strict competition in local and global markets has driven the farming financial in challenges. Consequently, human labor has become an expensive recourse cutting the profitability. Moreover, it has caused the increase of the workload of the farmers. Thus, they have less time to spend with the cattle observing the status of estrus and health issues. Punctual estrus detection is fundamental factor in keeping the calving interval within the optimal range. Extended calving intervals have direct affect in milk yield and the profitability. In Finland, the average calving interval has already exceeded 400 days whereas, 360 days is the optimal and recommended interval. Traditionally, estrus and health issues have been detected sight-wise by a cattle tender. This method is time consuming and considered inefficient. However, such symptoms as lameness are still difficult to detect with sensor-based solutions. Visually, lameness is rather easy to detect. Nevertheless, sensor-based technologies aim to detect health issues before they escalate into sever conditions such as lameness.

Currently, there are numerous technological solutions available as an alternative to human labor. However, they are relatively expensive investments and their actual payback time is difficult to define. Therefore, specially many smaller farms have postponed the use of modern technological aids in cattle monitoring. Additionally, even large farms in developing countries are in the same stage and still rely on human labor and traditional methods. In addition to the high cost and long payback time, the functionalities and the performance level of the solutions can vary significantly. Thus, achieving of complementary solutions are rather challenging. Typically, these commercial solutions offer aids for estrus detection and general health tracking. In fact, they tend to trigger alarms for cattle tender to start inspections instead of providing accurate health status.

Based on this background information, the rigorous detection of estrus for insemination is the most critical issue in nowadays dairy farming. Thus, this study aims to develop and evaluate an effective algorithm for dairy cow estrus detection. Conventionally, cattle tender monitors cattle and detects estrus behavior. That is, a cow being mounted by other cows is considered as secure indication of ongoing estrus. There are additional behaviors giving a hint of ongoing estrus. However, confident detection requires several occasions of these behaviors. By contrast, in sensor-based estrus detection it is more convenient to detect proestrus instead of the actual estrus. The use of accelerometer as the fundamental sensor in these applications is reasonable, hence, the behavior of the cow is very active in this phase. Additionally, detection of proestrus provides the farmer enough time to prepare the insemination before the actual estrus. Unfortunately, cows in tie-stall are not able to be active walking around the cowshed. Thus, only loose-housed cows are considered in this study. Additionally, the results of this study could be adoptable for beef cattle as well as

other species. Nevertheless, these options are not discussed in this study. In addition to the active behavior of the cow, its body temperature rises during the estrus. Therefore, some commercial solutions include body temperature measurement. Similarly, the hardware solution in this study includes a temperature sensor. Nevertheless, the accelerometer is the fundamental component in estrus detection and the temperature sensor is to confirm or question the detection.

The focus of the evaluation is in reliability and punctuality. Therefore, the algorithm shall not trigger false estrus or miss a true estrus either. Additionally, the timing shall remain within reasonable tolerance. Thus, the resulting algorithms shall be able to detect estrus in real-time. In addition to these requirements, the algorithm shall be stand-alone solutions suitable for low-cost micro-controller devices. Therefore, in this study we use common low-cost components in the sensor device hardware configurations. Additionally, the hardware design is rather simple in order to keep the costs low. However, in wearable battery-based solutions the high capacity batteries expensive and their cost may exceed the price of the other hardware. Thus, low energy consumption is considered in the discussions of this study. In addition to the estrus detection, we attempt to review the possibility of behavior monitoring as well. However, the behavior monitoring is a minor topic of this study. Therefore, the results related to behavior monitoring only are discussed only briefly in this study.

Efficient data analysis, algorithm development and testing are an absurd approach. Specially, when the estrus cycle of a cow lasts approximately 21 days and the phases of proestrus and estrus several hours. Therefore, it is essential to use recorded sensor data in algorithm development and in testing as well. Consequently, implementation of suitable data recording software is one of the core topics in this study. Furthermore, properly implemented data recording application gains benefits in possible future studies as well. The data recording application shall be implemented on similar cost-efficient hardware platform as the actual estrus detection algorithm would be. In contrast to software, designing of hardware is not included in this study. However, we will introduce the hardware platform used in this study. Furthermore, we will discuss of its components and their functionalities in reasonable detail. In despite of the requirements of real-time punctuality of the estrus detection algorithms, real-time algorithm testing is excluded from this study. By contrast, the algorithms shall be implemented and tested with appropriate software tool and the results are discussed in the results section. Nevertheless, the convertibility of the algorithms are considered throughout the development process. Furthermore, the actual algorithm implementation for a micro-controller are not discussed in this study.

In despite of the real-time punctuality of the resulting algorithms, testing of the algorithms on-site is not in scope of this study. Conversely, the algorithms shall be developed and tested with a suitable computer software. The algorithm development requires real cow data from a dairy farm. Therefore, implementation of a suitable data recording software for a low-cost micro-controller device is one of the core topics in this study.

Kerro menetelmistä.

Onko muita rajauksia?

Puuttuuko jotain muita perusteluja tutkimuksen tekemiselle, valinnoille yms?

2 Background ok

Originally, humans were hunter-gatherers who obtained food by collecting plants and pursuing wild animals. The methods for acquiring food have changed substantially since the beginning of the agriculture. That is, plants and animals are grown centralized in large farms instead of numerous small producers. Actually, the animal husbandry has been estimated to been started more than 10,000 years ago in western Asia. Accordingly, goats were among the first domesticated animals in human history [41]. Thereafter, people have been domesticating other species e.g. cows, sheep and pigs for milk, meat and other animal products. In addition to the quantity of various species, the headcount has been increaing with the population. Furthermore, the industrial revolution has started a trend of growing farm sizes and a loss of smaller farms. Recently, the total headcount of the world has been estimated almost up to 1,000 million heads in 2016 [3].

Nowadays, the cattle breeding is divided into two trends of beef and dairy farming. As a result, the breeds of beef cattle and dairy cattle are considerably different in physics and by nature. The beef breeds are more muscular whereas milk breeds are more tame. Moreover, the scope of beef breeding is in rapid brawn growth, while high milk yield is the target of the dairy cattle breeding. In spite of the same origin of the breeds, they are as distinct as different species. Therefore, the scope of this study is only in dairy farming. Accordingly, the following subsections will discuss briefly of the history and the basics of the modern dairy farming 2.1. Furthermore, we will survey through the life of a cow 2.1.1 and discuss of related issues. Additionally, we will focus in two fundamental cycles in the life of a dairy cow, estrus cycle and lactation cycle 2.1.2. These cycles are directly related to the milk yield and profitability of the dairy farm. In addition to the discussions of the dairy cow, we will take an overview on dairy cow monitoring 2.2.1. The overview includes as well as traditional methods as currently available technological aids. Lastly in the background section, we will survey through the most recent research and studies in dairy cow monitoring 2.2.2. In the survey, we will focus specially in wearable sensor devices and behavior monitoring.

2.1 Dairy Farming ok

As discussed in previously, the origins of animal husbandry are over 10,000 years old. Whereas, the drinking of milk started 8,000 years ago in present Turkey area. Thousand years later, dairy farming started to spread into Europe and thousend yars after that into Africa [41]. Thereafter, dairy farming has spread all over the world. Meanwhile, the variety of dairy products has exploded simultaneously. In result, there are numerous different milk, cheese and yogurt products as well as other dairy refinements. Additionally, dairy products have an essential part in human nutrition nowadays. Furthermore, milk and other dairy products are produced more than ever before. Concurrently, the farm headcounts have been increasing whereas the number of farms has started to decrease. Alone in Finland, there were total of 909,000 cows of which 282,000 milking cows. Meanwhile, 7890 farms delivered milk to milk processing plants in the end of the year 2015. As aresult, the average yield of a farm was 279 thousand liters whereas the average yield of a cow was 8300 liters. Accordingly, the total milk yield in Finland was 2365 million liters in 2015 [22].

Figure 1: Cows in a tie-stall-cowshed [15]. Cows are tied in stall and they are not able to move freely. They also have less space than in a loose-housed cowshed.

Correspondingly in the Unites States of America, there were more than 89 million cows and calves of which more than 9 million were milking cows in 2014. The corresponding cash income of the dairy farms was more than 49,349 million dollars in 2014 in the USA [2].

Originally, cows were wild pasture animals and, afterwards, domesticated by humans. At the beginning of animal husbandry, the people was migrating nomads. Therefore, the cattle traveled with people. Eventually, people started farming and settled in constant regions with their animals. First, the animals were held in yards but then people started to build structures for protection and easier keep of the animals. Next, people started to keep their cattle inside buildings. In linear pottery culture, people and animals lived together inside longhouses. Finally, people started to build separate buildings for theri animals. Special building for keeping cows are called cowsheds. Currently, the cowsheds are divided in two types of tie-stall and loose-housed cowsheds [18]. In general, tie-stall cowsheds are smaller and, therefore, tighter than loose-housed cowsheds. Furthermore, cows are not allowed to move freely in tie-stall. Conversely, In loose-housed cowshed cows are allowed to move freely round the clock. Additionally, they may have free access to pasture in some solutions. Recently, the tie-stall cowsheds have been under critique. That is, the cows are not able to behave as social animals. Additionally, the monitoring and health keeping of the cows is more difficult in tie-stalls. Therefore, most of the new-builds are rather loose-housed than tie-stall cowsheds. In addition, it is considered the cows of being happier and healthier in loose-housed cowsheds.

Figure 2: Cows in a loose-housed-cowshed [34]. Cows are able to move freely and act as social animals.

2.1.1 Dairy Cow

Tähän alkuun vielä jotakin lehmän perustietoja, kuten arvio rotujen määristä. Lehmän keskimääräinen paino, koskeus, pituus jne. Ehkä myös maininta siitä, kuinka paljon lehmä tarvitsee tilaa mm. laskeutumiseen ja ylösnousuun.

Previously, we surveyed through the history of animal husbandry and discussed of the beginning milk producing. Additionally, we introduced such cowsheds as tie-stall and loose-housed cowsheds. Correspondingly, this subsection will debate on the dairy cow itself in general level. Whereas, the subsequent subsections will focus on such milk yield related cycles as estrus and lactation cycles. Inherently, cows are plain and herd animals. Moreover, they live in hierarchy [13]. In large herds they form smaller groups where they do their daily activities such as eat and rest together [13] [18].

Typically, cows lay down approximately from $11\,\mathrm{h}$ to $12\,\mathrm{h}$ every day. Meanwhile, they stand up and change their pose several times [45]. Additionally, they may move their location between haunts and watering places. Thus, in loose-housed cowshed a cow may walk from $400\,\mathrm{m}$ to $800\,\mathrm{m}$ per day [33]. On pasture, their daily walking range may extend to several kilometers [33] [13]. However, cows are very cautious animals. Thus, insecure circumstances such as slippery ground or high steps can reduce their daily range. Furthermore, cows can be easily injured in challenging places. Naturally, injuries effects to their health and consequently to profitability [18]. Nevertheless, Walking enforces the health of them, increases hormonal activity and metabolism [33].

In addition to their normal activity, cows may have exceptional states. Typically, these states become apparent in their behavior. That is, sicknesses and injuries reduces their activity level, whereas, proestrus increases it. Phases of estrus are covered in detail in section 2.1.2.

Tänne vähän lisää tietoa lehmien terveysongelmista, kuten ontumisesta, sorkkahommista ja ruoansulatusvaivoista.

Cow can lick all of its body excluding neck and head. Cows doze standing and sleep lying. The estrus period is approximately 21 days and the estrus lasts from 12 to 16 hours. A bull may detect estrus 2 days before the main estrus. [13]

Access to fresh and clean water is vital. [33]

2.1.2 Lactation and Estrus Cycles

In previous subsections, we discussed of dairy farming and dairy cow in general. Consequently in this section, we will proceed the discussion to lactation cycle. The lactation cycle is emphatically related to the milk yield and thus, the profitability of the farm. The lactation cycle means the period between two calves. Thus, it is also called as calving period. After the calving a cow begins to lactate, which is the the actual purpose of a dairy cow [18]. However, the calves are more or less a secondary product in dairy farming and they are not in scope of this study. Nevertheless, the milk yield if a cow increases in the first weeks after the calving. However, after the first weeks the milk yield begins to decrease as illustrated in picture 3. Consequently, it is not cost-efficient to keep on milking the cow infinitely. Therefore, cyclic calving is preferred in order to maintain the profitability. However, the cow will stop milking before calving which causes a dry period.

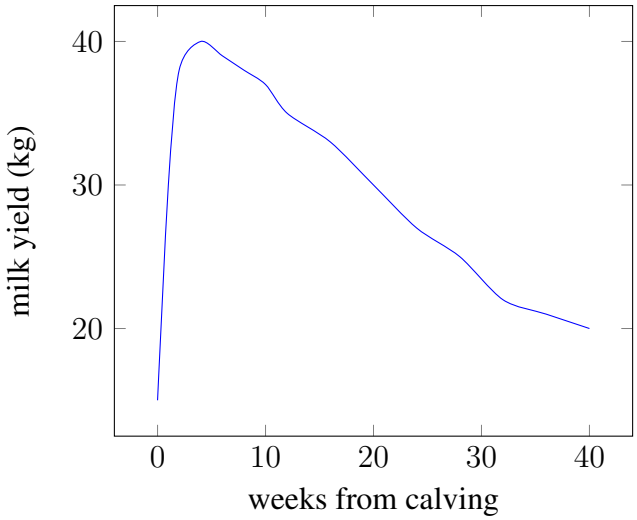

Figure 3: An illustrative lactation curve of a dairy cow representing the realtion between the weeks in milk and the milk yield. After the beginning of the lactation, the milk yield reduces in the function of time [23].

This causes a dilemma. That is, too short as well as too long calving interval reduces the total milk yield and cuts the profitability. Is general, it is recommended to keep the calving interval roughly in 360 days. Moreover, the milk yield increases after each calving. Thus, it supports the idea of regular calving [1].

In recent years, the calving interval has been increasing globally. Alone in Finland, the average calving interval has been extended up to 400 days and it has affected to the milk yield. There has been discussions of the root causes for the extended calving interval. In general, whereas the farm sizes have been increasing the cattle tenders have more work and less time for observing the cows. Thus, it has been more difficult for them to detect any estrus behavior and take actions for insemination. Consequently, this trend has forced the farmers to seek technological aids for punctual estrus detection. Thus, the main scope of this study is in developing of estrus detection algorithms for wearable sensor device. The hardware and software designs are intoruced in section 3 and the results are discussed in 4. Additionally, we will survey through of currently available solutions in section 2.2.1.

However, before proceeding to the research of this study, it is necessary to discuss of the estrus cycle of dairy cow. The estrus cycle is the period between ovulations. Actually, the estrus cycle is considered to begin from the ovulation. The normal duration of a cow estrus cycle is approximately 21 ± 3 days. A bull may detect an estrus already 2 days in advance. The estrus lasts from 12 to 16 hours [13]. The estrus is the time frame when the cow is ready to mate or being inseminated. The estrus is also called as standing heat, hence, the cow allows to being mounted by other cows. Traditionally, being mounted by others has been considered as certain sign of ongoing estrus. Before the estrus, cow has a proestrus period. During the proestrus, cow behaves restlessly and usually it attempts to mount other cows. Normally, cows who are not in estrus do not allow to be mounted. Therefore, these are only attempts and are not considered as a sign of estrus. Typically, the duration of proestrus is from 9 to 18 hours [18].

2.2 Health Monitoring and Estrus Detection

The previous subsection 2.1 discussed of the fundamentals of dairy farming. The discussion started form the history of animal husbandry and ended to study of dairy cow itself. The study of dairy cow included basic knowledge of the cow and its environments. Moreover, we discussed of lactation and estrus cycles and their affect on the milk yield and profitability. Additionally, we briefly surveyed through the most common health issues with dairy cow. In continuation to previous discussions, this subsection discusses of different methods for live-stock monitoring. First of the following subsection 2.2.1 surveys through currently used methods and technologies. Correspondingly, the second subsection 2.2.2 discusses of studies of existing technology as well as recent development projects for future solutions.

2.2.1 Current Solutions

Traditionally, live-stock monitoring has been sight-wise task assigned to the cattle tender [18]. However, this method is not considered to be efficient. In the USA alone, the rate of successfully detected estruses has been estimated below $50\,\%$ in large farms. Additionally, this inefficiency leads to annual loss of 800 million dollars for the milk industry [12]. Naturally, the reliability of the observations depends on several such factors as the experience of the cattle tender, availability of time and the amount of cattle. Additionally, even experienced cattle tender might be erroneous with non-familiar cattle. Furthermore, the head count of farms tend to be increasing whereas the number of cattle tenders remaining the same. Consequently, the tenders have less and less time for purely observing the cattle. Therefore, detection of estrus of health issues has become even more difficult. In despite of the challenges, sight-wise observations by the cattle tender are still common monitoring method in small farms and in developing countries [12].

Nowadays, there are several commercial poruducts available for dairy cow monitoring. Most typically, these products are sensor devices attached with a strap to leg [24, 16] or to collar [17, 16, 25] sensor devices are collar or leg devices mounted with straps. [17] [24]. However, there are even tail-attached sensor [25] and in rumen [26] sensors available. In addition to various attachments, the devices have various applications. Some sensors are specialized in calving detection [25] or digesting monitoring [26] only whereas leg and collar sensors provide more wide range of functionalities. Typically, they monitor the cow motion and activity [24, 17, 16, 25]. Moreover, leg sensors can count steps and detect the pose of the cow (standing, lying and walking). Furthermore, the leg and collar sensors are used in estrus detection. Additionally, some devices are capable of monitoring rumination [17] and eating [16]. Most of the devices are planned for wireless data transmission between the sensor device and a computer of farm server [17] [16] [24] [26]. However, some devices require wired USB data transmission [24]. In despite of the technology or the feature being monitored, all the solutions aim to a healthier cow and, thus, improvements in profitability. Others may alert of possible health issues in advance [17, 16, 24] whereas others are guiding to optimize the feeding and digestion [26]. Furthermore, numerous current solutions provide mobile phone interfaces ans most significant alerts are given in SMS messages or in email [17].

2.2.2 Research and Studies

In former subsection we discussed of currently available technological aids for dairy cow monitoring. Different products provide different functionalities from rumination and digestion monitoring to motion tracking and estrus detection. Similarly in this section, we survey various researches and studies for dairy cow monitoring. In contrast to the previous section, the solutions discussed here are not yet commercially available products. Nevertheless, some studies have been using commercially available devices in their research. In this subsection, our scope will be specially in solutions that are not available in current product range such as behavior monitoring. However, studies of the estrus detection are in our interest as it is the main topic of this study as well.

Most typically in these researches, the sensor devices are based on various accelerometers [27] [44] [19] [32] or pedometers [12]. Additionally, such alternative approaches as intravaginal probe has been tested for estrus detection [5] [4]. In despite most of the solutions are accelerometer-based, their approach to the topic is quite different. The most complex of these algorithm aims to full behavior detection with support vector machines [27]. They used an accelerometer based sensor attached to collar of the cow. Their overall performance of the multi-class model was $78\,\%$ precision. However, The precision with lying down and standing up activities were poor. Correspondingly, another research studied use of simple decision tree instead of support vector machines of hidden Markov Models [44]. In this study, they used various time windows from 1 to 10 minutes. In result, the 1 minute window provided better results than 10 minute window. Additionally, this method was barely equally goon in results as the vector machine algorithms.

Another study used a commercially available leg sensor for estrus detection [19]. From the leg they were able to detect whether the cow was lying, standing or moving. The overall sensitivity in estrus detection was $88.9\,\%$ adn error rate $5.9\,\%$. Also they used 1 minute sample periods. In this study, the detected estruses were controlled by success of insemination. In their conclusions, they suggested this method could only support existing methods and the reliability is not high enough for standalone solution. Yet another research used pedometers for a step-count based estrus detection [12]. In this method they were able to improve the visual estrus detection rate up to $84.2\,\%$. Also in this study, the estrus detection was verified by the success of insemination. Either this solution was standalone, hence the purpose was use it together with the sight-wise estrus detection. In addition to these accelerometer-based behavior and estrus detection, one research studied lameness detection as attaching one accelerometer to each limb [32]. Their ground idea was sample the accelerometer data from each limb at $25\,\mathrm{Hz}$ and use wavelet analysis in order to detect asymmetry, which could be considered as lameness.

In contrast to these more traditional accelerometer-based solution, another researche studied the use of intravaginal probe for estrus detection [4] [5]. The probe was designed to be inserted inter vagina and transmit the temperature and conductivity data wirelessly. The idea for this study was to use non-motion based sensors. Therefore, this solution could be used also in tie-stall cowshed. The research showed some promising results but lacked of reliability. In their sequel study they used two sizes of the intravaginal probes, $160\,\mathrm{mm}$ and $120\,\mathrm{mm}$ probes. However, neither of these were successful. The larger probe caused bleeding during the research, whereas the smaller started to rotate and even ejected the

vagina. Nevertheless, they were able to found some correlation between the intravaginal temperature as well as the electrical conductivity and estrus.

In behavior based cow monitoring, the sensor devise attempts to recognize on or more of the following acts:

- *Standing* is a state where the cow stand on all of its legs and stays still.
- *Lying* is state when the cow lies down on it side. Cows spend half of their day lying down.
- *Ruminating* is an act where the cow chews the feed from its rumen in order to digest fibers etc.
- *Feeding* means the act when the cow is eating feed.
- *Normal walking* when the cow is in health and walks normally.
- *Lame walking* when the cow has health issues in its legs or claw and its walking is cautions on asymmetrical.
- *Lying down* means when the cow is standing but changes its state into lying.
- *Standing up* is when the cow was lying but stands up.

. The behavior monitoring provides beneficial information for the farmer about the health status as well as the mix of the feed.

3 Research ok

In previous section 2, we discussed of the essential backgrounds of this study. The discussion started form the beginning of animal husbandry and dairy cow in modern farming. The discussion ended in introduction of currently used methods and survey of recent research and studies. Moreover, the background section focused in milk yield related estrus and lactation cycles in present day farming. In conclusion of the background, there is a certain need for an efficient solutions in dairy cow monitoring. Accordingly, wearable wireless sensor devices are currently the most promising option due their overall performance and availability. Respectively, the target of this study is to develop and evaluate convenient estrus detection algorithms for wearable wireless sensor devices. Therefore, this section discusses of required tools and supportive methods in data recording and algorithm development. Additionally, the data recording software in this study are created from a scratch. Thus, it is necessary to introduce the related software implementations in this study.

In this study, the research section has been divided in three subsections. The first subsections 3.1 discusses of data recording. The data records are essential component in algorithm development and evaluation. In this study we will not use already existing data. Thus, we will implement required software as well as record the data for further employment. The subsection introduces the hardware design for the data recording. The hardware discussion covers the main hardware components in reasonable detail. In addition to the hardware design, we discuss of the communication protocols between the main hardware components. Respectively, we introduce the principle work flow of the data recording software. Whereas, the hardware of this study was already existing, the software is self-implemented. Thus, we will discuss of the implementation more in detail. The second subsection 3.2 introduces a set of data processing methods applied on the recorded data. The set of methods consists of basic statistics as well as various digital signal processing tools. In this study, we utilize only the basic statistical analysis and they are introduced briefly. In contrast, the digital signal processing tools are covered in more detailed discussion. Lastly in this section 3.3, we develop three different algorithms for dairy cow estrus detection. All these algorithms have their base in the discussed methods. However, each of the algorithms have their own approach to estrus detection. Nevertheless, all the algorithms detects rather the proestrus than the actual estrus 2.1.2. However, the end of proestrus indicates the beginning of the estrus. Therefore, this kind of approach is highly applicable. Additionally, it provides the cattle tender more time to prepare for the required actions.

3.1 Data Recording ok

Above, we described the research section of this study in general level. As already stated, this subsection will discuss of the data recording process. The data records are essential components in algorithm development and evaluation. The discussions in this subsection will start from the introduction of data recording hardware. The hardware design in this study is an existing hardware setup. Therefore, we will discuss of the main components and functionalities only superficially. Next, we will discuss of the data recording software.

The software for this study is created from a scratch. Therefore, the software design is discussed particularly. In this study, we recorded data in three different occasions of which, the first occasion with a different software implementation. Consequently, we introduce two different software implementation. Lastly in the data recording subsection, we will describe the actual data recording procedure of all the occasions.

3.1.1 Hardware ok

As stated previously, the data recording hardware in this study is an already existing prototype of a dairy cow sensor device. Originally, the hardware consisted of micro-controller unit (MCU), accelerometer and thermometer. However, it lacked of large enough storage memory for data recording. Thus, the hardware was enhanced with a Secure Digital (SD) memory card slot and with a SD memory card for this study. Otherwise, the hardware design remained the same during the data recording processes. In spite of the mentioned customization, the hardware design is not in the scope of this study. Therefore, the following is rather a description of the hardware than a comprehensive discussion of the design itself. In addition to hardware components, we briefly discuss of the communication between the components. In general, the main hardware components brief descriptions are as follows.

- *Atmel ATmega32u4* is a high perforamance low power 8-bit micro-controller. It is designed for optimizing power consumption versus processing performance. It contains 135 powerful Reduced Instruction Set Computer (RISC) architecture instructions of which most executable in a single clock cycle. [10]

- *Bosch Sensortec BMA222E* is an accelerometer with on-chip motion triggered interrupt controller. Thus, it enables motion-based applications even without utilizing a micro-controller. It is capable of measuring acceleration in three perpendicular axises. BMA222E is designed for various consumer products from game controllers to pedometers. It is small sized and low power consuming. Therefore, it is suitable for mobile battery-powered solutions. [11]

- *Texas Instruments TMP112* is a high-accuracy, low-power, digital temperature sensor. It is designed for various applications from portable and battery-powered solutions to general temperature merasurements in industrial controls. [42]

- *Secure Digital (SD)* memory card is specially designed to meet the security, capacity and performance requirements in newly emerging consumer electronic devices. The standard capacity of SD memory card is up to $2\,\mathrm{GB}$. However, High Capacity SD (SDHC) extends the maximum capacity to $32\,\mathrm{GB}$ and Extended Capacity SD (SDXC) up to $2\,\mathrm{TB}$. [37]. In this study, the maximum capacity of SDHC card shall be considered as the maximum available memory capacity. This restriction shall be taken into considerations while designing the sensor software. Moreover, the memory card shall be capable of recording one estrus cycle at minimum (approximately 21 days).

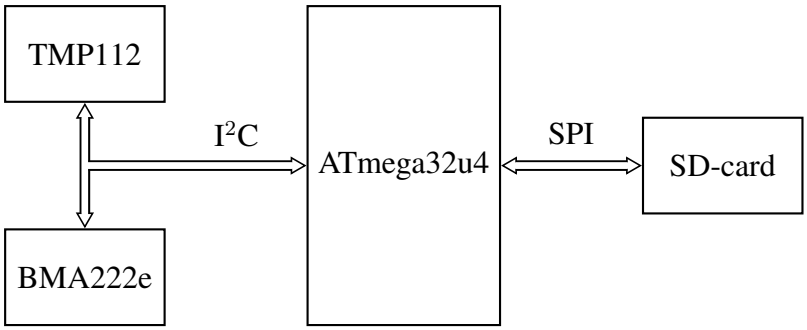

Figure 4: The principal block diagram of the sensor device. The temperature sensor and accelerometer are connected to the micro-controller via the same I^{2}C-bus, whereas, SD-card is connected via SPI-bus. Power connections nor USB are not represented in this figure.

In addition to the hardware components, the hardware configurations utilizes two serial interfaces, *inter-inegrated circuit (I^{2}C) bus* and *serial peripheral interface (SPI)*. They are explained briefly in the following.

- *Inter-Inegrated Circuit (I^{2}C) bus* which is some times referred as *Two-Wire Interface (TWI)* is a serial communication interface developed by Philips Semiconductor. The first version of the I^{2}C was released in 1982. The design of it is rather simple hence, it requires only only two bidirectional open-drain lines, Serial Data Line (SDA) and Serial Clock Line (SCL), with pull-up resistors [43].

- *Serial peripheral interface (SPI)* [39] is a more complex serial interface than the (I^{2}C) bus. It requires at minimum of three parallel wires in three-wire mode [11] *signal select* ($\bar{SS}$), *serial clock* (SCK) and *serial data input/output* (SDI). However, in normal four-wire mode the data input and output are in separate lines, *master out, slave in* (MOSI) and *master in, slave out* (MISO).

As we have now introduced the main components of the sensor device, the communication principles of the device are represented in figure 4. As shown in the picture, SD memory card utilizes SPI serial interface whereas temperature sensor and accelerometer are connected into same I^{2}C bus. Naturally, all of these hardware components are assigned to their specific tasks with respect to their functional description. Therefore, their essential functionalities are discussed in the following topics.

Micro-controller ok

The *micro-controller unit (MCU)* is the core component of the sensors device in this study. That is, the micro-controller is responsible of execution of the software flow whenever the device is powered. Typically, the first executable task in the software flow is the system initialization. Normally, system is initialized once in software flow and each time after resetting or starting the micro-controller. During the initialization, the micro-controller sets up its control register according to the configurations in its program memory. These

registers contains the configurations for the *general purpose input output* (GPIO) pin configurations as well as timers, serial interfaces and other featured functionalities. In addition to self configuration, the micro-controller initializes connected hardware via serial interfaces according to the configurations in the software memory. Next, after initialization the micro-controller begins the execution of the actual software flow. Typically, this flow is a repetitive software loop including variable tasks and events. The hardware initialization and configurations as well as the application tasks and events are discussed more the software section 3.1.2.

As discussed earlier, Atmel ATmega32u4 [10] is the micro-controller unit of the sensor device in this study. It is a high performance, low power micro-controller for various applications. Therefore, we will discuss of its features and functionalities more in detail. In spite of the wide range of its features, only the most fundamental properties are covered in the following discussion.

- $32\,\mathrm{kB}$ *of In-System Self-Programmable Flash memory* is the memory space for the actual program storage. Furthermore, the memory space is divided into two sections, Boot program Section and application Program section.
- *USB 2.0 Full-Speed/Low-speed Device Module* provides interface to write on the in-system self-programmable flash memory of the controller. Thus, it enables uploading the application software without external USB module. Additionally, the interface contributes serial communication between computer and the device.
- *General I/O* consists of 26 programmable I/O lines. These lines can be set as input or output separately. Furthermore, specific input pins can be configured as interrupt pins. Typically, interrupts are utilized for triggering events in the software flow. Additionally, interrupts are also a required feature in serial communication events.
- *Interrupts* are occasions that usually set an interrupt flag. Next, the *interrupt service routine (ISR)* corresponding the interrupt flag is executed with respect to its priority. Optionally, it is possible to execute a task directly in the interrupt. However, it temporarily disables other interrupts, hence, it is not recommended. Furthermore, there are internal and external interrupts. The source of internal interrupts are inside the micro-controller itself, as timers. Correspondingly, the source of external interrupts are external devices such as sensors or memory storages. Additionally, the interrupt can be enabled individually. The main advantage of utilizing interrupts is to execute tasks on-demand instead of periodically.
- *Watchdog Timer (WDT)* of ATmega32U4 employs a separate on-chip $128\,\mathrm{kHz}$ oscillator for creating timer events. If the WDT is enabled it is capable of resetting the entire micro-controller unless, the WDT is not reset regularly. Normally, WDT is utilized for recovering from deadlocks, unintentional conditions without exit. However, it is possible to utilize the watchdog timer as a wake-up timer in low power modes.
- *Power Management and Sleep Modes* allows the user to tailor the power consumption according the application requirements. Power management is beneficial feature specially with battery-powered solutions when regular re-charging is not effortless to accomplish.

- *SPI and I^2C* serial bus interfaces are used for device-internal communication of the micro-controller and the sensors as well as the SD memory card. The serial interfaces were briefly discussed in section 3.1.1.

Accelerometer ok

As discussed previously, a micro-controller is the core component of the hardware design of the sensor device. Similarly, an accelerometer is the essential sensors providing data for data recording as well as algorithm development. In this study, the utilized accelerometer is the Bosch Sensortech BMA222E [11]. It is a triple-axial accelerometer used for measuring the change in the motion. Additionally, it can be employed in position recognition in constant situations. The axises of the accelerometer are perpendicular. Therefore, it is capable of measuring in all direction. Additionally, it contains several advantageous on-chip functions. It is capable of prior and post filtering acceleration measurements. Additionally, it can detect various conditions and trigger interrupts in result. Moreover, the filters as well as the detectable conditions are highly configurable. These configurations are discussed in the following description. Nevertheless, the accelerometer contains several functionalities without use-cases in this study. Consequently, those features are not covered in the discussions. However, some of the features have no direct use case in this study but are applicable in the future studies that are considered in section 5. Additionally, the sensor configurations varies between two software implementations in this study and they are discussed in the sectin 3.1.2. The kay features of the BMA222E accelerometer are as follows.

- *On-chip interrupt controller* is capable of generating interrupts from various conditions. The conditions are configurable with respect to time or numerous motion statuses. On-chip interrupt controller yields an opportunity to create device applications even without a micro-controller. However, in this study all the interrupts are processed in the micro-controller. Nevertheless, the interrupt controller is utilized to create favorable sampling events instead of continuous sample recording.

- *On-Chip FIFO Register* is capable of storing up to 32 data frames. Depending on the configurations, one frame contains measurements from chosen or all three axis. Additionally, it is configurable whether the data in the register is filtered or unfiltered. Furthermore, the register contains information whether the data is new or old data.

- *Range* of the acceleration measurements is adjustable within four preset ranges, $\pm 2\,\mathrm{g}$, $\pm 4\,\mathrm{g}$, $\pm 8\,\mathrm{g}$ and $\pm 16\,\mathrm{g}$. However, increasing the range decreases the resolution and vice versa. Thus, selecting of appropriate range shall be considered.

- *8-bit Resolution* is applicable for both, acceleration and temperature measurements. As discussed, the range of acceleration measurements is adjustable and affects the resolution. Consequently, the resolution is from $15.63\,\mathrm{mg}$ per *least significant bit (LSB)* to $125\,\mathrm{mg}$ per LSB. The Nevertheless, the temperature resolution is fixed to $0.5\,^{\circ}\mathrm{C}$ per LSB.

- *Low Pass Filter* enables removing of high frequency distortion from the measured signals. Thus, no additional low pass filtering is needed in the micro-controller application. The low pass filter of the accelerometer is configurable with preset frequencies from 7.81 Hz to 1000 Hz. The bandwidth configuration effects the interval, how frequently, new data frame is readable from the register of the sensor. Roughly, the values are update twice often than the filter bandwidth according the Shannon-Nyquist sampling theorem.
- *Offset Compensation* allows removing offsets from the measured signals. At sea level, there is always approximately $1\,\mathrm{g}$ offset present. Specially in integrative calculations the presence of offset accumulates and yield misleading results.
- *SPI and I^2C digital interfaces* are necessary interfaces for configuring the sensor as well as data transmission from the accelerometer from the micro-controller. It is configurable, which interface is utilized.
- *Low Power Consumption* is beneficial in portable and battery-powered solutions such as the sensor device in this study. Additionally, the accelerometer enables improvement of the power consumption with several power modes. However, the power modes are not applicable in this study because, one of the core ideas is to utilize the accelerometer as effectively as possible instead of more power consuming micro-controller.
- *On-chip Temperature sensor* of the controller provides a resolution of $0.5\,^{\circ}\mathrm{C}$. The accuracy of the BMA222E is relatively low in comparison to the TMP112 temperature sensor discussed later in this section. Nevertheless, it is useful as a reference or verification value with the measurements of the TMP112 temperature sensor. [11]

Temperature Sensor ok

Whereas the accelerometer is the main sensor of the sensor device, a temperature sensor is considered as a supportive sensor. Furthermore, the accelerometer contains a built-in temperature sensor. However, its resolution is relatively low. In contrast, Texas Instruments TMP112 is a high-accuracy temperature sensor for various applications from portable devices to industrial controls. It is specially designed for replacing *negative temperature coefficient (NTC)* and *positive temperature coefficient (PTC)* thermistors in high accuracy applications. In this study, the temperature sensor is used for monitoring the skin temperature of the cow. The target of the monitoring is to find correlation between the change of skin temperature and ongoing estrus or proestrus. Therefore, in the case of positive correlation the temperature sensor can confirm the detected estrus. In contrast, a negative correlation can confuse the results and yield a need for further studies. The properties of the TMP112 temperature sensor are described in the following.

- *High accuracy* of
  - $0.5\,^{\circ}\mathrm{C}$ in range from $0\,^{\circ}\mathrm{C}$ to $65\,^{\circ}\mathrm{C}$
  - $1.0\,^{\circ}\mathrm{C}$ in range from $-40\,^{\circ}\mathrm{C}$ to $125\,^{\circ}\mathrm{C}$

without calibration. Furthermore, instructions for the calibration of the sensor are provided in the data sheet.

- *High resolution* of $0.0625\,^{\circ}\mathrm{C}$ in both, 12-bit and 13-bit modes
- *Low power consumption* in two different power modes:
  - $10\,\mu\mathrm{A}$ in active mode
  - $1\,\mu\mathrm{A}$ in shutdown mode
- SMBus™, Tow-Wire and I^2C digital interfaces
- supply voltage range from $1.4\,\mathrm{V}$ to $3.6\,\mathrm{V}$
- *Conversion rate* from $0.5\,\mathrm{Hz}$ to $8\,\mathrm{Hz}$
- *12-bit resolution* from $-55\,^{\circ}\mathrm{C}$ to $128\,^{\circ}\mathrm{C}$. The sensor has an 13-bit mode, when the measu rement range is up to $150\,^{\circ}\mathrm{C}$. [42]

In this study, the temperature sensor is soldered directly on the circuit board. In result, the sensor is not in direct skin contact with the cow. Therefore, the skin temperature is conducted from skin to the sensor via heat conducting aluminum tape.

3.1.2 Software ok

As discussed in previous section, the data recording hardware in this study is a prototype of dairy cow sensor device enhanced with memory card slot. By contrast to the hardware, the data recording software had to be implemented from a scratch for this study. Therefore, this section will discuss of the software design as well as the software implementation. Additionally, we will introduce the software development tools utilized in this study. The design phase of the software development included hardware test for defining the restrictions and capabilities. Consequently, it slowed the design process. Nevertheless, hardware test results that had no effect are not included in the results of this study. However, after proper testing and design, the actual software implementation was rather straight forward process. In this section we will first introduce rduino Integrated Development Environment (Arduino IDE) [6] as our primary software tool in application development. Next, we will present two different software implementations. The first software implementation was utilized as a test and the results were mainly used in for software improvement. Consequently, the second software implementation differs from the first one and is more optimized for long term data recording. Lastly in this software section we will discuss of the data recording procedures. In the following we have short description of Arduino IDE and of its utilized libraries.

- *Arduino IDE* provides a fast and easy software development environment for implementing micro-controller applications without prior expert-level knowledge on micro-controllers. Additionally, Arduino and Arduino community supports with comprehensive set of software libraries for various micro-controller applications.

- *Wire* library provides functionality for Inter-Integrated Circuit (I2C) communication. "This library allows you to communicate with I2C / TWI devices. On the Arduino boards with the R3 layout (1.0 pinout), the SDA (data line) and SCL (clock line) are on the pin headers close to the AREF pin. The Arduino Due has two I2C / TWI interfaces SDA1 and SCL1 are near to the AREF pin and the additional one is on pins 20 and 21." [7]
- *SPI* library provides the interface for uitlizing of SPI bus for inter-integrated chip communication. Furthermore, Arduino's IDE library provides functionalities for both, master and slave devices [8]. Nevertheless, in this study the sensor device shall work as a master.
- *SD* library allows reading from and writing to SD cards. The library supports FAT16 and FAT32 file systems on standard SD cards and SDHC cards. It uses short 8.3 names for files. That is, eight characters for filename and three for extension. [9]
- *EnableInterrupt* library provides interface for configuring interrupt pins. In additoin to enabling and disabling the interrupts, interrupt conditions e.g. rising edge, faling edge, low are configurable. [36]
- *RingBuf* is a library for simple *first in first out (FIFO)* buffers for the Arduino. Buffers created with the library can buffer any fixed size object such as ints, floats and structs. [46]

In addition to the standard libraries, Arduino community provides plenty of open-source software libraries for various applications. In spite of the numerous software libraries, they do not comprehensively include all the functionalities of the supported micro-controllers. Therefore, a profound familiarization with the micro-controller data sheet is mandatory step in order to achieve the optimal performance of the controller. However, the Arduino programming language is merely a set of C/C++ functions. Thus, it enables of implementation of ad-hoc functions such as hardware configurations. In this study, this flexibility was utilized in driver implementation for the accelerometer and temperature sensor since, their original drivers were not correctly supported in Arduini IDE. Furthermore, the data logging interface was created from s scratch. In result, there are two different log types in this study, one for each software implementation. Nevertheless, in both cases, the logs are written on SD-card on common .txt-files. In the following subtopics, we will discuss both software implementations in detail.

First Software Implementation ok

In this study, it was considered that software implementation based on strict initial assumptions could impair the the achieved results. Therefore, the approach for the first software implementation was to avoid utilizing unnecessary or defective functionalities of the accelerometer. In conclusion, it was decided to prioritize high sampling rate over other features. Specially, irreversible pre-processes should be avoided. Thus, e.g. offset compensation was not in use in this software implementation. Furthermore, the offset data can provide vital information of the pose of the sensor. Nevertheless, the data was

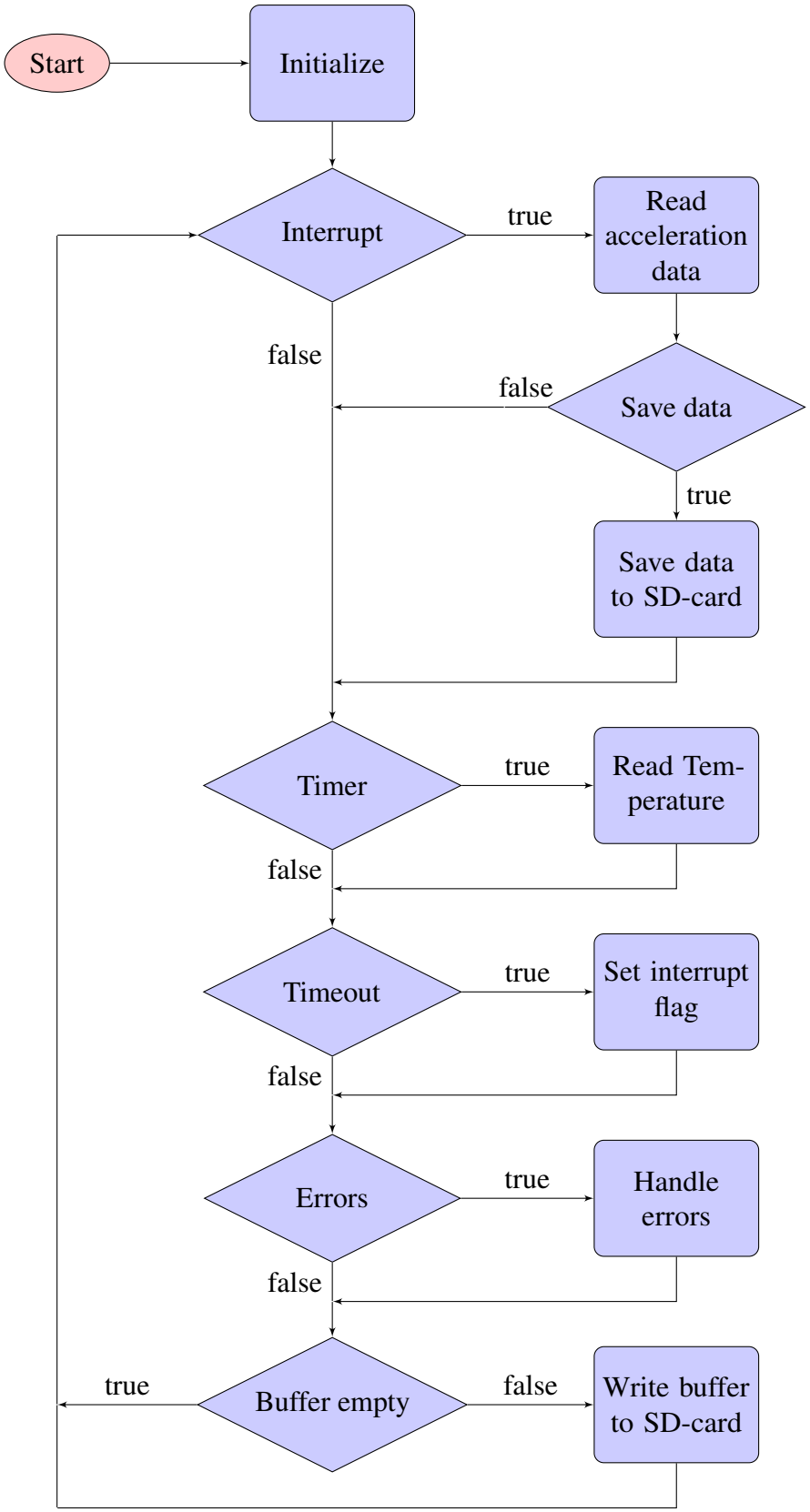

Figure 5: The software flow of the first data recording software. The rounded rectangles represents tasks whereas the diamonds are conditions. Each conditions are followed by branches depending on the current situation. The tasks and conditions forms a repetitive loop of executions excluding the initialize task.

low-pass filtered in order to avoid the folding of frequencies higher than the half of the sampling rate. The maximum sampling rate of the accelerometer is limited in $2000\,\mathrm{Hz}$. Consequently, the minimum update time is $0.5\,\mathrm{ms}$. However, utilizing a filter douples the update time if filtered values are configured as output values. In spite of the high sampling rate of the accelerometer, the usage of secure digital (SD) memory card as a data storage limits the data rate rather effectively. That is, the duration of the SD file operations such as open, write and close exceeds the disposable time at high data rate. In result, it causes the FIFO buffers of the accelerometer to overflow. Consequently, undefined amount of data is lost. In contrast, a low sampling rate could cause a loss of possibly relevant features on higher frequencies. Therefore, the optimization of the data rate is more or less a trade of between data and feature losses.

The software flow of the first data recording software is represented in figure 5. The flowchart consists of tasks (rounded rectangles) and conditions (diamonds). A condition is s state with two or more possible exit branches. In this study, the conditions are of Boolean type with only two following branches. In contrast, tasks are executable functions typically between conditions. Nevertheless, a branch may include a sequence of conditions as well as sequence of beaches. Furthermore, a task might contain smaller tasks and smaller conditions whereas conditions may contain smaller tasks and smaller conditions. Nevertheless, the conditions presented in the flowchart are discussed in the following itemization.

- *Interrupt* is true if an interrupt flag is set by an interrupt from the accelerometer which in this case is a certain FIFO buffer level.
- *Timer* condition is used for reading temperature data from both of the sensors periodically. The temperature variation is not a high frequency process. Therefore, it is rather reasonable to read temperature values more seldom than accelerometer data.
- *Timeout* is a backup condition in the case of missed interrupt from the accelerometer. That is, the accelerometer gives FIFO level interrupt only once, unless the micro-controller reads it empty after receiving the interrupt. Therefore, if micro-controller misses the interrupt it will not receive a new one. In consequence, it can cause a deadlock situation.
- *Errors* condition is for handling of the detecting errors in the software flow. The detectable errors need to be defined. The purpose of error detection is mainly improving the software and evaluation the quality of the recorded data.
- *Buffer empty* condition checks the buffer level of the micro-controller. If the buffer level is more than zero the values in the buffer will be written to the text file on the SD card.
- *Save data* condition detects s safe timing for performing save data task. The save data task includes time consuming SD-card file operations such as close file and open file.

Correspondingly, the software flow contains the following tasks.

- *Initialize* task (which is analogous for *setup* function in Arduino IDE) is executed only once right after the device is powered up. This task initializes the micro-controller itself as well as the desired configurations for the accelerometer and the temperature sensor.
- *Read acceleration data* task reads all the data from the FIFO buffer of the accelerometer and stores the it into the FIFO buffer of the micro-controller. Actually, the serial interface is capable of transmitting less than six data frames at a time. Therefore, the task reads sets of five data frames until there are more than zero but less than five frames available. Lastly, the task reads the remaining frames. In result, the FIFO buffer of the accelerometer should be apty.
- *Save Data to SD-card* task saves the data written on the SD card. According to the Arduino's SD library, the written data is saved only after the file is closed utilizing the close function. Otherwise, the data will be lost and it is not possible to recover it afterwards. Additionally, in order to continue fluent writing process, the save data task re-opens the file. However, if the size of the log file exceeds the set maximum file size, the save data task will open a new log file. Furthermore, the duration of the open and close file operations could exceed the time required to fill up the FIFO buffer of the accelerometer. Thus, the FIFO buffer is being read empty before and after closing the current file and once more after opening a file.
- *Read temperatures* task reads the temperature data of both sensors and stores the value to the memory of the micro-controller. No temperature data are buffered. Thus, only the latest values are written on the log file on the SD-card.
- *Set Interrupt flag* sets the interrupt flag without an interrupt if a timeout condition is met. Once the interrupt flag is set, the micro-controller will read the acceleration data from the FIFO buffer of the accelerometer. Consequently, it is possible for the accelerometer to set new interrupt flag once its FIFO buffer is being filled.
- *Handle errors* task handles predefined errors if one or more of them have occurred. In practice, this task writes the names of the occurred errors into the text file and resets all of the error flags.
- *Write buffer to SD-card* writes a single line of acceleration data from the FIFO buffer of the micro-controller into a text-file on the SD card. However, in normal execution of the software flow, all conditions are most likely false. In result, the software will sequentially write multiple lines of acceleration data before another condition is met.

Second Software Implementation ok

Whereas the first software implementation was created without descent pre-knowledge, the second software implementation benefits from the results of the first data set. As discussed in the results section 4.1 the first data set experienced multiple lost data frames.

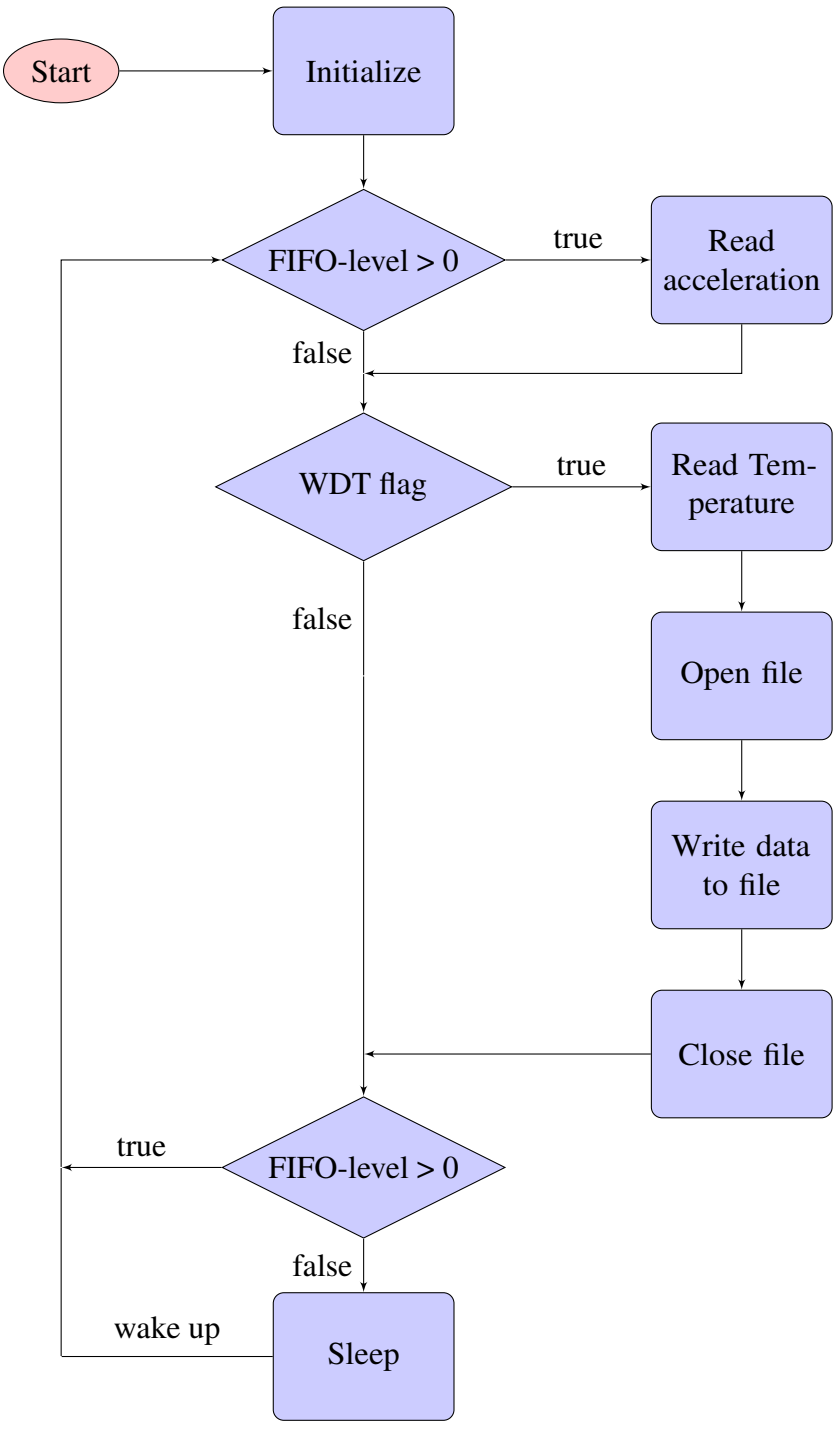

Figure 6: The program flow of the second data logging software

Furthermore, the frequency analysis showed there is no need for high frequency sampling. Actually, most of the frequency spectrum was less than $8\,\mathrm{Hz}$. Therefore, the approach in this second software implementation is in improving the performance. The focus is specially in avoiding lost data frames. Additionally, a secondary target is reducing the device lifetime considering the battery and memory card capacities. However, battery or memory card capacities were not issues with the first data set. Nevertheless, the improvements are beneficial with possible future studies as well. The flowchart of the second software implementation is represented in figure 6. In comparison to the first implementation, the software design is rather simple in the design level. That is, the second implementation consist of only three conditions whereas the first was designed with six. Furthermore, the number of tasks is reduced by one. Additionally, the task are in sequence without intermediate conditions. The conditions of the second software implementation are described in the following itemization.

- *FIFO-level > 0* condition is true if the FIFO buffer level of the accelerometer is non-zero.
- *WDT flag* condition is true if the watchdog timer has set a watchdog timer (WDT) flag.

Correspondingly, the tasks of the second software implementation are as follows.

- *Initialize* task initializes the micro-controller as well as the other components of the sensor device.
- *Read acceleration* task reads all the data from the FIFO buffer of the accelerometer and stores the data into the FIFO buffer of the micro-controller.
- *Read temperature* task reads the temperature of both of the sensors, accelerometer and temperature sensor.
- *Open file* task opens a file for writing the data. The task opens a new file if the size of the current file exceeds a preset limit for maximum file size.
- *Write data to file* writes all the buffered acceleration data into the opened file on the SD card. In addition, the task writes the temperatures of both of the sensors, the number of occurred watchdog timer interrupts and the up-time of the micro-controller into the text file.
- *Close file* task closes the opened file in order to ensure the written data is being saved.
- *Sleep* task sets the micro-controller into a power saving mode. That is, all the other functionalities except the watchdog timer and interrupts are disabled in order to minimize the power consumption. The micro-controller remains in the sleep until the watchdog timer or an interrupt from the accelerometer wakes up the micro-controller. After waking up, the continues the execution of the software flow from the beginning of the repetitive loop.

3.1.3 Procedure ok

Previously we discussed of the software implementations for the data recording. In this section we will discuss of the actual data recording process on real dairy cow farm. The farm utilized in this study is located in coordinates N63°6′3.6″ E23°10′35.5″. The breed of the farm contained of Ayrshire and Holstein cows and they both are used in data recording. The data was recorded in total of three separate occasions. First of the sessions was recorded with the first software implementation whereas the last two with the second software implementation. The recording of the first data set was started on Friday September the 9th at the target farm. The sensor device was attached to the collar of the selected cow. The target of the first data recording was acquiring initial information of the cows behavior as well as performing frequency analysis of the data. Therefore, the cows was equipped also with additional trail camera. A $8\,\mathrm{GB}$ SD-memory card was used as a log file storage during this occasion. Initially it was estimated to sustain approximately 16 days, hence, the recording required approximately $500\,\mathrm{MB}$ per a day. The recording of the first data set was ended on Sunday September the 18th. After removing the devices from from the collar, all the log files as well as trail camera images were stored on computer for further studies. These results are discussed in results section 4.

In contrast to the first data recording event, total of three devices were used on the second and third sessions. In addition to the increased number of devices, the sensor software was improved considering the capacity of data storage and batteries. In result, the memory storage was estimated to sustain roughly for 500 days and batteries at least 60 days. Furthermore, a video camera was mounted on the ceiling of the barn during the second data recording event. Nevertheless, the memory of the camera was fulfilled in 24 hours. The purpose of the video camera was similar to the trail camera in the first session, searching correlation between motion data and the video based behavior monitoring. During the third data recording event, no video devices were used. The second data set was recorded on the test farm from December the 14th to 15th. Meanwhile this data was analyzed and algorithms developed, the third data set was recorded until January the 20th. In addition to the data recording sensors, the cow were equipped with Heatime cow monitoring system. This system was utilized as a reference for the estrus detection algorithms. Figure 7 illustrates the cow with a collar sensor. Additionally, the axises of the accelerometer are shown in the picture. In addition to the picture of cow wearing the sensor, figure 8 shows the heat conducting tape on the bottom side of the sensor device. This aluminum tape is in continuous skin contact with the cow and conducts the heat to the temperature sensor on the circuit board. Furthermore, differences between the software implementations are represented in table 1. Additionally, table contains information of fundamental performance estimation.

3.2 Data Processing ok

Previously, we have been discussing of the data recording hardware and software as well as the data recording process itself. Accordingly, this section focuses in the data processing methods. The data processing methods consists of basic statistics as well as such digital signal processing methods as *fast Fourier transform (FFT)* and digital filters. Thus, the

Table 1: Data recording parameters of different data recording occasions and software implementation

Parameter name	**First software implementation**	**Second Software Implementation**
Low-pass filter bandwidth	62.5 Hz	7.81 Hz
Update interval	8 ms	64 ms
Measurement range	± 4 g	± 8 g
Accelerometer buffer size	20	30
Interrupt latching	12.5 ms	latched
Internal Buffer Size	200 frames	200 frames
SD-memory card size	8 GB	16 GB
estimated SD-capacity	approx. 16 days	approx. 500 days

Figure 7: Cow wearing a sensor device. The axis directions are illustrated with the red arrows in the figure. The sensor is in parallel with a commercial Heatime device also seen in this figure.

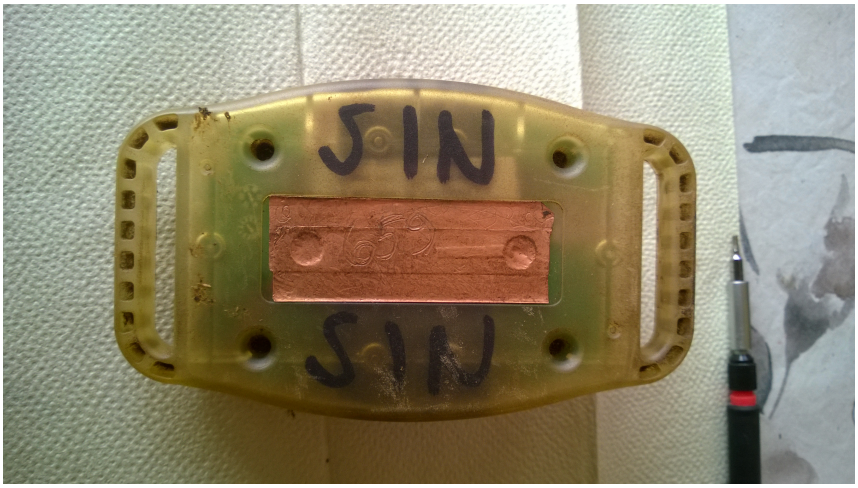

Figure 8: The heat conducting tape on the bottom-side of the sensor device. The purpose of the tape is conducting the skin temperature of the cow to the temperature sensor on the circuit board.

discussion in this section will start from the most essential statistic methods. Second, we will study the Fourier transform as a tool fro the frequency analysis. Third, we discuss of digital filters. The discussion rest of the fact the accelerometer includes filter for distortion attenuation as well as offset compensation. Last, in this section we introduce a sliding window as an essential principle while processing large amounts of data.

However, prior processing of the data, it is essential to parse the data from log files into applicable form. In this study, we use *MathWorks MATLAB* (later Matlab) [28] software for both, log file parsing and data processing. Furthermore, the format of log files

varied depending on the software implementation. Therefore, corresponding data parsing scripts are required. In the log files, the data is written as certain rows containing the desired information. The purpose of the data parsing script is to convert the data from text characters into numeric values. Furthermore, the secondary purpose of the data parser is in reorganizing the data into practical structures for the future purposes. Additionally, the amount of data in log files was as minimal as reasonable. However, the paring sequence is capable to compute additional values such as time and include that into the structs for further purposes. In spite of these computational data processing methods, plotting and visual observation is the essential method in algorithms development as well as in feature identification.

3.2.1 Statistics ok

As already discussed, the data processing methods in this study consists of various tools. The statistics is an effective method of describing large amounts of data in few characteristics. Furthermore, they are relatively easy to compute. Therefore, statistics are one of the fundamental data processing tools in this study. Additionally, the statistics are deployed in several sequences from prior analysis to the various phases of estrus detection algorithms. In spite of the benefits of advanced statistics, in this study we use only the most essential statistical characters. The utilized statistical numbers are discussed in the following itemization [38].

- *Mean* is an average value that attempts to describe the most typical value of the data set. However, mean is sensitive to unevenly and asymmetrically distributed values. Therefore, in such cases median is more valuable depicting the most typical value. Nevertheless, mean is defined as the sum of all values divided the number of values as in the following equation.

$$\bar{x} = \frac{\sum_{i=1}^{n} x(i)}{n} \tag{1}$$

- *Median* also describes the most common value of the data. In contrast to the mean, it is not that sensitive to unevenly and asymmetrically distributed values. Therfore, it is more applicable when data contains such features as large peaks Median is defined as the middlemost value of sorted data. If the number of values in the data is even, the median is the mean of the two middlemost values.

- *Variance* describes the expectation of the squared deviation of a random variable from its mean, and it informally measures how far a set of (random) numbers are spread out from the mean:

$$s^2 = \frac{\sum_{i=1}^{n} (x(i) - \bar{x})^2}{n} \tag{2}$$

- *Standard deviation* describes the amount of variation in the data set. A low standard deviation means the values tend to be close to the mean value, whereas a high value means the values tend to be far from the mean. In contrast to the variance, the

standard deviation describes the “typical” distance between the values and the mean. The standard deviation is actually the square root of variance:

$$s = \sqrt{\frac{\sum_{i=1}^{n}(x(i) - \bar{x})^2}{n}} \tag{3}$$

- *Minimum* is the smallest value in the data set.
- *Maximum* is the largest value in the data set.
- *Range* is the absolute difference between the minimum and the maximum values.

3.2.2 Fourier Transform ok

Frequency analysis of a signal is fundamental phase in filter design as well as in adjustment of the sampling rate. According to the Shannon-Nyquist theorem, the minimum sampling rate shall be at least twice the signal frequency. Otherwise, high frequency components may fold over the lower frequencies and distort the signal. In this study, we did not make initial assumptions of the signal frequency of the acceleration signal. Thus, the sampling frequency was set as high as possible. The purpose is to use the frequency analysis to choose more suitable low-pass filter bandwidth for the second data recording application. Nevertheless, understanding of frequency analysis requires understanding of periodic signals. The basis of fourier transform is in Fourier analysis. That is, in the seventeenth century mathematician Jean Baptiste Fourier discovered that periodic signals could be approximated as series of simple sine and cosine functions. In fact, the Fourer transform is linear mapping of a signal from time domain into frequency domain. COrrespondingly, inverse Fourier transform is inverse linear mapping from frequency to time domain. Continuous time Fourier transform is representend in equation 4 [21].

$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(x)^{-i\omega x} \mathrm{d}x \tag{4}$$

Similarly as normal Fourier transform, textitDiscrete Fourier transform (DFT) is also linear mapping of a signal from time domain to frequency domain. However, continuous time equations apply on continuous time signals whereas discrete time equations on discrete signals. Corresponding discrete time Fourier transform is represented in equation 5.

$$F_n = \sum_{k=0}^{N-1} f_k e^{-i(2\pi k/N)n} \quad , \mathrm{n} = 0, \ldots, \mathrm{N} - 1 \tag{5}$$

Normally, DFT would require approximately n^2 iterations for mapping the time domain to the frequency domain. With large samples this becomes mathematically demanding and time consuming process. Therefore, there are alternative approaches called Fast Fourier Transforms (FFT). They require less computing and time, approximately only $n \log n$ iterations. In practice, the fast Fourier Transform is solved numerically instead of solving it mathematically. However, the results may not be as exact as it is with DFT [21] [35]. Nevertheless, in this study we will use the FFT functions included in the standard

libraries of Matlab software [29]. The theoretical background of FFT is rather profound and therefore, it is not in the scope of this study. The fft function of Matlab does not actually consider frequency at all. In contrast, it provides only a numeric solution. Therefore, corresponding frequency vector needs to be created and mapped manually to the solution. Otherwise, the fft function of Matlab is fast and rather straightforward function.

3.2.3 Digital Filters ok

Filters are one of the fundamental tools in signal processing. Properly designed filter removes such undesired components from the signal as distortion and offset. Typically signal is smoother after applying a filter. Typically, filters are applied in the beginning of the signal processing sequence. However, there are no restrictions of applying filters in multiple phases of the sequence. In general, filters can be divided in four different types, low-pass, high-pass, band-pass and band-stop filters. Low-pass filters are desirable in removing high-frequency components. Normally, the purpose is removing of distortion. Usually, in these cases the source of distortion is external and not a part of the original phenomenon e.g. mechanical vibrations, acoustic and electrical noises. Similarly, high-pass filters are applied when low frequency components e.g. offset and DC voltage are not in the focus of the current application. Furthermore, band-pass and band-stop filters are special combinations of low-pass and high-pass filters. Band-pass filters are designed to pass only specific frequency band and removing components below and above it. Conversely, band-stop filter is design for filtering only a specific frequency band and allowing other frequencies to pass. [14] [21]

Filters can also be separated as analog and digital filters. Analog filters are based on a physical combination of analog electrical components such as resistors, capacitors and inductances. In contrast, digital filters are a sequence of arithmetic operations. Consequently, digital filters can be implemented on specific digital circuits or in software applications. Nevertheless, analog filters are usually present also in digital applications as preliminary and protective filters prior digital sampling. Specially, it is crucial to filter out frequency components above the Nyquist frequency in order to void folding. In this study, we will concentrate only on digital filters. Furthermore, the approach is to imitate the filters inside the Bosch Sensortec BMA222E accelerometer. The Accelerometer contains two configurable on chip filters, 2nd order low-pass filter and 1st order high-pass filter [11]. The bandwidth of the low-pass filter is configured directly via the control register of the sensor. However, the high-pass offset compensating filer depends on the selected bandwidth of the low-pass filter as well as specific preset. That is, the cutoff frequency of the high-pass filter is either one-hundredth or one-thousandth of the low-pass cutoff frequency. Unfortunately, the data sheet of the sensor did not provide further information of the filters except the filter order. Therefore, these filters are replicated in this study based on the available information.

Digital filters can be divided into two categories, *finite impulse response (FIR)* filters and *infinite impulse response (IIR)* filters. The topology of FIR filter is represented in figure 9 whereas the topology of IIR filter is shown in figure 10. The names of the filters follow their impulse response. The FIR filter has a finite impulse response whereas the impulse response of IIR filter is infinite. As seen in the FIR filter equation 6, the output

of the filter is dependent on only current and previous input signal levels. Therefore, the output signal cannot be longer than the degree of the filter. By contrast, the IIR filter is also dependent on its own current and previous outputs as seen in the equation 7. Thus, if the filter is stable it converges but never reaches absolute zero after an impulse input.

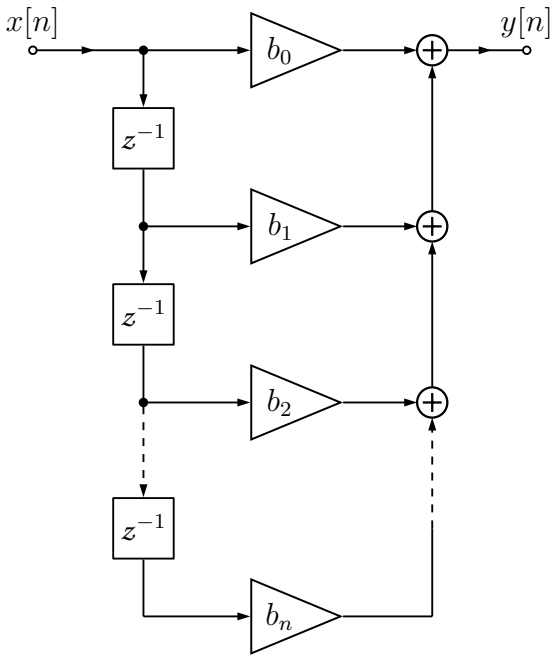

Figure 9: An example of an n^{th} order FIR filter

$$y[k] = b_0x[k] + b_1x[k-1] + b_2x[k-2] + \cdots + b_{n-1}x[k+1-n] + b_nx[k-n] \quad (6)$$

$$\begin{aligned} y[k] = b_0x[k] + b_1x[k-1] + b_2x[k-2] + \cdots + b_nx[k-n] \\ -a_1y[n-1] - a_2y[n-2] - \cdots - a_my[k-m] \end{aligned} \quad (7)$$

With this ground knowledge of digital filters we are able to create adequate replicates of the filters of the accelerometer. In addition, to the knowledge, we need an effective filter designing tool. The Matlab softeare provides a satisfactory filter designing tool [30] for our purposes. The tool allows of defining the type on order of the filter. Additionally, it provides offers the frequency response of the filter for validation. In this study, both of the filters were assumed as IIR filters. Specially the 1^{st} order high-pass filter would not provide a satisfactory frequency response for offset compensation. Therefore, IIR filter in this application is more reasonable. Nevertheless, same kind of logic could not be used defining the type of the low-pass filter. However, the IIR filter has steeper cutoff frequency than same order FIR filter. Therefore, it also was designed as IIR filter. Nevertheless, designing of an effective 1^{st} order high-pass filter is challenging. Therefore, it was eventually decided to design also the high-pass filter as a 2^{nd} order IIR filter.

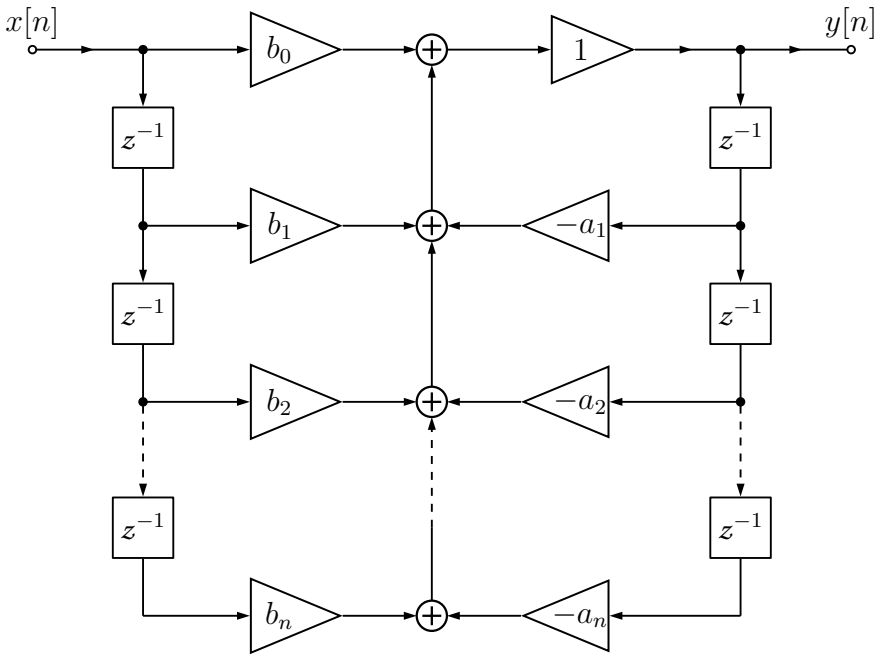

Figure 10: An example of an n-th order IIR filter in direct form I

3.2.4 Sliding Window ok

In addition to the methods discussed previously, we deploy one more data processing method called sliding window. In this study, the sliding windowing means applying certain calculus on sliced data. Furthermore, in this study, the result of s single window operation is always a single value instead of array of values. An example of sliding window function is represented in equation 8. In this example, the window operator calculates the mean value of the data inside the window instead of entire data set. The n is the size of the window in data samples whereas k is the index of the entire data set. In this method of the increment of the m can vary according to the requirements. That is, increment of 1 requires a lot of calculations and results significant overlapping. By contrast, increment of $m > n$ requires less computation and results no overlapping in results. However, this approach causes certain concerns in the data processing sequence. That is, if the $m > 1$ it reduces the number of resulting data points. Therefore, it is necessary to re-scale the corresponding time array. Nevertheless, this kind of windowed operation is analogous to windowing in digital signal processing [40] [31]. The equation 8 represents an example of windowed operation. This example calculates the mean value inside the window. As it seen in the equation, the resulting array will decrease in size if the $m > 1$. Furthermore, this method creates special cases on the edges of the data array. That is, the indexes of the values might be either negative or exceed the size of input array. Therefore, all values with negative or exceeding indexes shall be assumed as zero in the calculations. However, this distorts the resulting edge values and any data points near edges should be considered with care.

$$\bar{x}(k) = \frac{\sum_{i=mk-n}^{mk} x(i)}{n} \tag{8}$$

3.3 Estrus Detection Algorithms ok

Previously in this study we discussed of the data recording hardware and software. Additionally, we surveyed through the fundamental methods of data processing in this study. Consequently, this subsection discusses of the estrus detection algorithms. As already declared, the algorithms are developed on the recorded data. Therefore, data has been recorded and analyzed prior to develop these algorithms. In this study, we created three different estrus detection algorithms. These algorithms are discussed in detail in the following subsections. In spite of their differences, all the algorithms are based on the measurements of the accelerometer and they consists of similar preliminary and closing sequences. Furthermore, these algorithms detect rather proestrus than the actual estrus. That is, the estrus behavior does not differ significantly from normal behavior from the sensor perspective. By contrast, during the proestrus the cow behavior is highly active and therefore, more detectable phase. Furthermore, detection of proestrus instead of estrus provides more time for the cattle tender to prepare the insemination. Therefore, this kind of approach is reasonable in algorithm development. Additionally, the algorithms are developed, tested and evaluated with Matlab. Accordingly, implementing the algorithms on the sensor device itself is out of the scope of this study. In spite of the differences between the estrus detection algorithms, they all obey the following principal algorithm pattern.

1. **Process data** — Data processing consists of algorithm specific data functions. These functions may include data filtering and other computations. These functions are discussed in detail in the following subsections. All the remaining phases of the algorithms follows the same pattern discussed below.

2. **Sum results** — The results of data processing are summed within time windows. The length of the time window should approximate the duration of the proestrus. Additionally, these time windows shall overlap in order to provide more continuous impression of ongoing state of estrus cycle. Furthermore, excessively long time windows without overlapping could delay the detection of the estrus. Thus, cause the failure of insemination as a consequence.

$$u(k) = \sum_{i=nk-m}^{nk} x(i) \quad \text{, where} \tag{9}$$

   k is the index of summed results, n is an incremental step size and m is the size of the time window.

3. **Remove offset** — The resulting data after summing may differ significantly within algorithm depending on the parameters as well as between the algorithms. Therefore, any existing offset in the data should be removed. This means adjusting the data so that the normal behavior appears around the zero. The median of the data describes

the amount of the offset more reliable than mean value. Hence, median is less sensitive to proestrus peaks in the data.

$$u(k) = x(k) - \tilde{x} \quad \text{, where} \tag{10}$$

$\tilde{x}$ is the median value of x.

4. **Normalize** — After removing the offset, the data shall be normalized. In this case, the normalizing means scaling the data so that no extreme value shall exceed the range of from -1 to 1.

$$u(k) = \frac{x(k)}{\max |x|} \tag{11}$$

5. **Threshold** — Finally, thresholds are used for indicating the beginning and the end of the proestrus. That is, exceeding the first threshold indicates the beginning of the proestrus and next, the going below indicates the end of the proestrus. The thresholds for deciding whether the cow is in estrus or not should be separate. The end of proestrus signals to the cattle tender to prepare for insemination. The principle of the proestrus detection is presented in the following Matlab code.

```
% Initialize the estrus status
proestrus = false;
estrus = false;
% Process through the entire data set
for i = 1: length(x)
    % Check if the value exceeds the threshold
    if (x(i) > threshold1 && proestrus == false)
        proestrus = true;
    % Check if the value falls below the second threshold
    else if (x(i) < threshold2 && proestrus == true)
        proestrus = false;
        estrus = true;
end
```

Additionally, each change of the estrus state should trigger an alert.

6. **Plot data** — Plotting of the resulting data visualizes the results. Nevertheless, it does not effect to the results directly. However, plotting makes the algorithm evaluation quicker and easier by brief visual inspections. Furthermore, plotting can reveal some features that might not be detected in pure numeral form.

As these were the principal steps of all the algorithms, the following subsections provide more detailed sequence of each algorithm in this study.

3.3.1 Activity Monitoring ok

The concept of this estrus detection algorithm is analogous to an accelerometer based energy consumption estimation [20]. Moreover, the energy consumption correlates with physical activity. Thus, such a similar method should be applicable also in activity measurement and, consequently in estrus detection. This algorithm employs the entire data set. Howeer, the working principle is rather simple. In general, it only sums the absolute value of the motion vector and observes the results within a certain time frame. The more detailed algorithm operations are discussed in the following sequence.

1. **Filter** — The data consists of an offset of approximately $1\,\mathrm{g}$ which effects to results. Actually, cows are rather passive animals and existing offset overrules the actual motion data. Therefore, removing the offset improves the performance of the algorithm and it shall be the first step in the algorithm sequence.

$$y(k) = a_0x(k) + a_1x(k-1) + a_2x(k-2) - b_1y(k-1) - b_2y(k-2) \tag{12}$$

2. **Compute** — The actual core of this algorithm is the length of the acceleration vector and it is defined as
$$u(i) = \sqrt{x^2(i) + y^2(i) + z^2(i)} \tag{13}$$

3. **Sum results** — The results are integrated within certain time window using the sliding window principle.

4. **Remove offset** — The median value of the resulting array is considered as the mean offset. The offset is deducted from all the data points.

5. **Normalize** — After removing offset, remaining data is divided by the maximum of the absolute value of the data. In result, all the data points are between ± 1.

6. **Threshold** — Next, exceeding of preset threshold is considered as the beginning of the proestrus. Conversely, falling below second threshold indicates the end of proestrus and the beginning of estrus.

7. **Plot data** — Finally, the data is presented in plot. The plot shows the algorithm results against time. In addition to the data, also the beginning and the end of proestrus are shown in the plot.

The benefit of this algorithm is its simplicity. The offset compensation filtering can be performed in the accelerometer with few easy configurations. Furthermore, the absolute value of the motion vector is replaceable with the sum of absolute values of the motion vector if necessary. Also summing of the results within a certain time frame is s straight forward process. Furthermore, the steps from four to seven are not required in real algorithm implementation. Conversely, the purpose of these final steps is to modify the information into more understandable form. However, continuous sampling deteriorates the possibility of deployment of power saving features of the micro-controller. This might be a critical issue with battery-powered solution. Nevertheless, the results of this algorithm are discussed in section 4.2.1.

3.3.2 Variance Detection ok

In contrast to the previous algorithm, this algorithm attempts to avoid the requirement of continuous data streams. That is, using of small standard-sized samples in certain intervals. The achieved benefit is the improved possibility of utilizing the power saving modes of the micro-controller. In result it increases the battery life and thus, cuts the maintenance costs of the device. In addition to the standardized samples, this algorithm consists of different calculus.

Originally, the concept of a variance based algorithm arose among the analysis of the first data set. The data consisted of two distinct period. The periods differed in both, variance and offset. In theory, both of these features could have been used as a sound for the algorithm. However, an offset based algorithm seemed unreliable, hence, the collar was not in a fixed pose on the cow collar. Thus, the pose of the sensor can change accidentally and corrupt the results. Furthermore, the offset was present and varied only among the x-axis. By contrast, the variance in motion were present evenly on each axis. Additionally, variance in more tolerant against accidental change of the pose of the sensor. Thus, the variance together with the sampling principle discussed were selected as the key methods for the second estrus detection algorithm. The work flow of the algorithm is explained in the following description.

1. **Get a sample** — In this algorithm, the data is processed in samples instead of data stream. The length of the samples is configurable. Naturally, there are no restrictions for overlapping the samples. However, the purpose of the algorithm is to reduce the amount of required data. Thus, overlapping is not reasonable approach from the perspective.

2. **Compute Variance** — A data sample consists of predefined number of data frames. Here, the computing of the variance means calculating the variance of a single sample. However, in this study we use the entire data set as an input data. Thus, we need an sliding window method 3.2.4 for sampling as well as calculating the variance. Furthermore, in this point we calculate the variance for each axis separately. Therefore, the outcome consists of three output arrays. The following equation represents the working principle of sliding variance window.

$$u_x(k) = \frac{\sum_{i=mk-n}^{mk} (x(i) - \bar{x})^2}{n} \quad , \tag{14}$$

   where n is the size of the data sample, m is the interval between the samples and k is the index of the output data. Furthermore, the algorithm implementation shall consider avoiding of exceeding the range of the input data indexes.

3. **Sum results** — The results are integrated within certain time window using the sliding window principle.

$$u_{tot} = u_x + u_y + u_z \tag{15}$$

4. **Remove offset** — Similarly as before, the median value of the resulting data is considered as the offset and it will be deducted from all of the data points.

5. **Normalize** — After removing the offset, the resulting data set is divided by the maximum of the absolute extreme value. In result, all the data points are between values ± 1.

6. **Threshold** — Next, exceeding of preset threshold is considered as the beginning of the proestrus. Conversely, falling below second threshold indicates the end of proestrus and the beginning of estrus.

7. **Plot data** — Finally, the data is presented in plot. The plot shows the algorithm results against time. In addition to the data, also the beginning and the end of proestrus are shown in the plot.

By contrast to the previous algorithm, the benefit of variance detection is in the potential power saving. That is, continuous sampling requires continuous readiness of the micro-controller, which is the most power consuming component the the sensor device in this study. Additionally, this algorithm is tolerant to offsets. Thus, it is not required to apply any offset filtering prior calculating the variance. However, this algorithm is more complex, hence it has more tunable parameters than the first one. Therefore, it is more complex to tune and might result non-satisfactory results. The results of the variance detection algorithm are discussed in section 4.2.2.

3.3.3 Inactivity Detection ok

The first two estrus detection algorithms were based on continuous sampling or calculating of variance of samples. Considering the micro-controller environments, these are relatively demanding operations. Furthermore, micro-controllers work in single thread. Thus, it is not possible to execute other tasks in parallel. In results, it degrade the performance capabilities of the device. Additionally, both of the first algorithms allocates significant part of the available memory for storing the intermediate and final results. By contrast, the third algorithm attempts to reduce the amount of required computation in the micro-controller. Basically, there are to options to ease to workload of the micro-controller. First, simplifying the arithmetic or reducing the amount of data frames reduces the workload. Second, utilizing the on-chip functionalities of the accelerometer also reduces the workload of the micro-controller. Actually, the first option is sort of a combination of the first two algorithms and possibly could provide satisfactory results. However, in this study we want to have three different approaches. Thus, the second option of utilizing the features of the accelerometer becomes more reasonable.

Once we have chosen to utilize one of the on-chip functionalities, we still need to decide which of the functionalities we shall utilize. As we already discussed in section 3.3.2, the pose of the device is not satisfactory reliable option for being the observed measure. Therefore, angle detection features of the accelerometer are excluded. However, the accelerometer contains a functionality which provides a new perspective to the activity monitoring. The functionality is *called no-motion / slow motion detection*. Thus, it detects

the absence of motions instead of detecting motion directly. In fact, the accelerometer triggers an interrupt when detects the state of no-motion. Normally, the cow lies down half of the day and changes its place occasionally. These resting resting and rumination activities should appear as detected inactivity. Correspondingly, the highly active period of proestrus should appear as reduced inactivity. Thus, proestrus period triggers less interrupts than the normal state of the cow.

As already discussed, the implementation the algorithms for the micro-controller is not in the scope of this study. Therefore, the no-motion detection is imitated in a Matlab script as a part of the algorithm. This methods allows us to test and re-configure the no-motion detection parameters freely. The no-motion detection has two configurable parameters, motion threshold ans delay. The motion threshold means the value that the slope of the motion shall not exceed. Correspondingly, the delay is the period when the slope shall not exceed the threshold in order to generate an interrupt condition. The complete algorithm description is discussed in the following sequence.

1. **Get Slope** — The slope is the difference between sequential acceleration values. That is, the previous value subtracted from the current value as shown in the following equation.

$$u[i] = x[i] - x[i-1] \tag{16}$$

   Additionally, the slope is offset independent hence, it actually is a first order FIR filter. Furthermore, it has infinite attenuation on zero frequency, sampling frequency and all of its harmonics.

2. **Detect inactivity** — According to the specifications of the accelerometer, a no-motion interrupt is triggered if the absolute value of the slope remains below of a preset threshold for a delay. The following Matlab code represents a software implementation for the no-motion detection.

```
% Initialize the output data array as zeros
data_out = zeros(length(data_in), 1);
% Initialize the index of previous exceeding of the threshold
prev_i = 0;
% Process through the entire data set
for i = 1: length(data_in)
    % Comapre slope value to the threshold
    if abs(data_in(i)) > threshold
        prev_i = i;
    end
    % Compare the index reminder to duration threshold
    if i - prev_i ≥ duration
        data_out(i) = 1;
        prev_i = i;
    end
end
```

In this algorithm, the output array has the same length as the input data array. Furthermore, it is initialized as zeros. During the execution, the algorithm processes through the entire data set. Whenever a no-motion condition is met, the algorithm sets the value of one into the output array. In consequence, the resulting passivity array consists mostly of zeros and few ones. Thus, the zeros in the output array represent activity or undefined activity whereas ones represent detection of inactivity. Additionally, because input and output array are equally long, there is no need to scale the time array of the measurement. Thus, it enables the discovering of the level of inactivity in desired time frame.

3. **Sum results** — The results are integrated within certain time window using the sliding window principle.

4. **Remove offset** — The median value of the resulting array is considered as the mean offset. The offset is deducted from all the data points.

5. **Normalize** — After removing offset, remaining data is divided by the maximum of the absolute value of the data. In result, all the data points are between ± 1.

6. **Threshold** — Next, exceeding of preset threshold is considered as the beginning of the proestrus. Conversely, falling below second threshold indicates the end of proestrus and the beginning of estrus.

7. **Plot data** — Finally, the data is presented in plot. The plot shows the algorithm results against time. In addition to the data, also the beginning and the end of proestrus are shown in the plot.

The obvious benefit of this algorithms is performing the possibility of performing the core functions on the accelerometer instead. Thus, it releases the resources of the micro-controller for other tasks or for power saving modes. Furthermore, the configuration of the delay sets the minimum interval between the interrupts. Therefore, it is easily possible to calculate the theoretical maximum amount on interrupts within a certain time frame. In consequence, it helps to estimate the power consumption and the battery-life of the device. Nevertheless, the results of the inactivity detection algorithms are discussed and evaluated in section 4.2.3.

4 Results ok

In previous sections, we have discussed of the background of this study as well as surveyed through the research tools and methods. In the background section we considered the history of milk production and discussed of modern farming technologies. Additionally, we viewed recent research and studies of dairy cow monitoring solutions. Correspondingly, the research section introduces the sensor device for data recording in this study. Second, we discussed of the software implementations for the sensor device. Next, we surveyed through data processing methods utilized during this study. Last in the research section we introduced three different estrus detection algorithms suitable for micro-controller devices. Consequently, this section discusses of the results of this study. The discussion will start from viewing general results. The general results include intermediate results utilized in software implementation as well as in algorithm development. Furthermore, we discuss briefly of the secondary results occurred during this study. Next after general results, we represent the results of the estrus detection algorithms and evaluate the algorithm performance.

4.1 General Results ok

In this subsection we discuss of general results of this study. The general results consists of intermediate and secondary results of this study. The intermediate results are essential information in algorithm development and improvement of the devices software. Mostly, these results are based on the first recorded data set discussed in section 3.1.3. The cow was wearing the sensor for 16 days and the data recording period included an proestrus according to the cattle tender. However, the data record consisted of only six days of valid motion data and the data from the remaining ten days was corrupted for unknown reason. Unfortunately, the proestrus occurred during the remaining ten days. Therefore, estrus or proestrus detection were not possible with the first data set. Nevertheless, the first data set provided us the frequency spectrum of dairy cow motion. Furthermore, we were able to analyze the performance of the SD-memory card as a data storage. Additionally, the first and second data sets were utilized in behavior monitoring. Nevertheless, the results were invalid in lack of proper time synchronization. However, the general results are discussed in detail in the following subsection.

4.1.1 Frequency Spectrum ok

As discussed, the first recorded data set did not include estrus or proestrus. Thus, it could not have been used directly in algorithm development. However, it provided us beneficial secondary information such as frequency spectrum of the motion data. The frequency spectrum of the first data set was analyzed with a Fast Fourier Transform (FFT). The principles of FFT were discussed in section 3.2.2. Actually, in this study we utilized the fft function of the Matlab software, which is actually a discrete time Fourier transform (DFT) applicable for discrete signals as our data records. The fast Fourier Transform was applied directly on raw data. That is, preliminary data processes such as filters were not used in order to avoid corrupting the frequency spectrum. However, the on-chip low-pass filter of

the accelerometer was enabled with the cut-off requency of $62.5\,\mathrm{Hz}$. The corresponding update time was $8\,\mathrm{ms}$. Because other filters were no used, the data consisted of an offset on each axis. In consequence, the frequency spectrum of each axis consists of significant amplitude on zero frequency and nearby. Optionally, the presence of the offset components could have been reduces with an high-pass filter. Nevertheless, the presence of the peak values nearby zero in the frequency domain are no disturbing the analysis. The results of the frequency analysis are represented in figure 11.

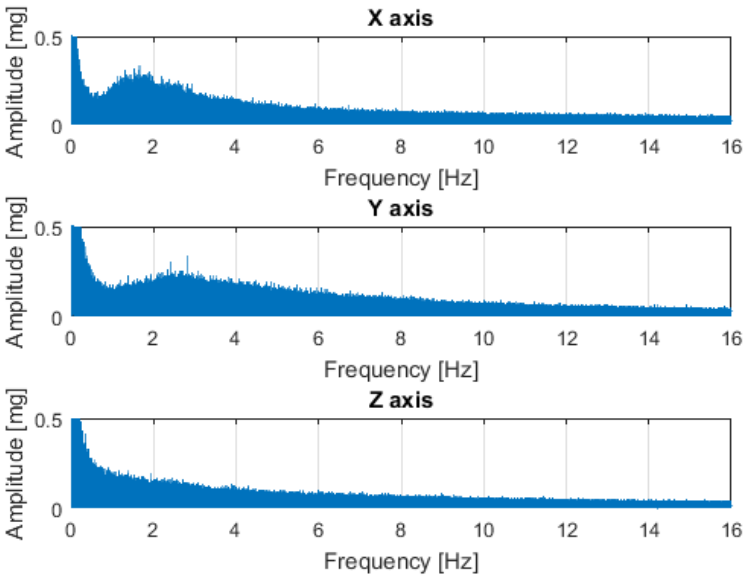

Figure 11: The results of the frequency analysis of the first data set. The frequency spectrum of each axis consists of significant components at frequencies close to zero-frequency and minor components at non-zero frequencies.

As seen from the figure, most of the cow activity has the frequency of $8\,\mathrm{Hz}$ or less. Additionally, there is a significant peak value nearby zero. The peak is a result of passivity of the cow as well as result of the data offset. Actually, the peak was so high that it was necessary to crop it out from the figure in order to emphasize the non-zero frequencies. Nevertheless, the frequency analysis shows us that $62.5\,\mathrm{Hz}$ cut-off frequency is unnecessary high. Actually, the cut-off frequency could be reduced down to $7.81\,\mathrm{Hz}$. The corresponding update time is 16 seconds. This frequency information was applied in the second data recording software implementation. The benefits were in reduced amount of required memory capacity of SD-memory card, less buffer overflows and data retention and possibility of utilizing the low power modes of the micro-controller. Nevertheless, the frequency analysis was not performed on the second and third data sets.

4.1.2 SD File Operations ok

The approach of the first data recording software was to maximize the cutoff frequency of the accelerometer. Thus, acquire data in maximum frequency and avoid the loss of

potential features in the signal. The maxmimum cutoff frequency of the ccelerometer is $1000\,\mathrm{Hz}$ and the corresponding update time is $0.5\,\mathrm{ms}$. However, the file operations of the SD-memory card were relatively slow and formed bottleneck for the maximum cutoff frequency. The preliminary testes with the sensor device established the $62.5\,\mathrm{Hz}$ as the absolute maximum cutoff frequency. Nevertheless, it caused occasional buffer overflow and data retention. In spite of the preliminary testes, the frequency $62.5\,\mathrm{Hz}$ was configured as the cutoff frequency during the first data record. However, the software was prepare for logging possible buffer overflows in order to validate the recorded data. Furthermore, the software logged also the duration of SD file operation times. The results of file operation logs are presented in figures 12 and 13. The figure 12 compares the time consumption of open file and close file operations. The close file operations is fundamental to execute regularly, hence it secures and saves the written data. Correspondingly, a file must be open before it could be written on. Therefore, closing a file and opening a file are sequential tasks for enabling data writing and securing the written data. The figure 13 represents the write times of each write data execution.

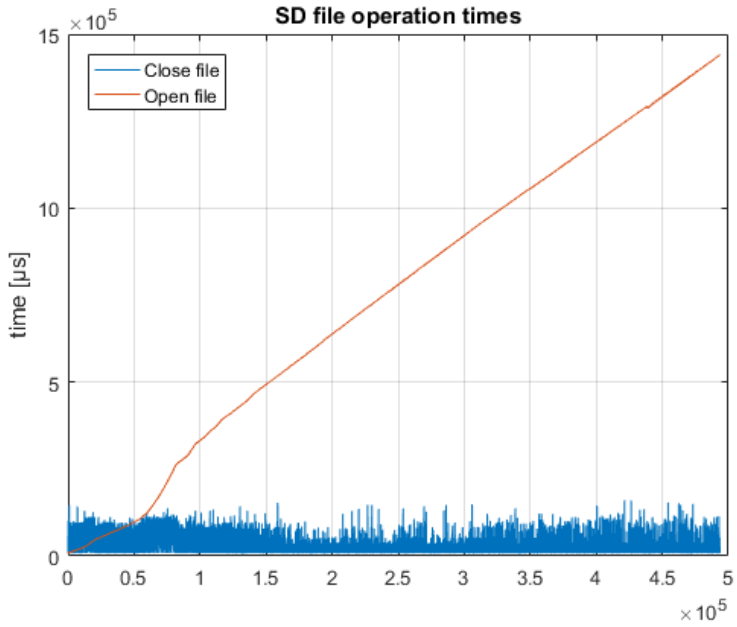

Figure 12:

It should be noticed that, the x-axis of these figures does not represent the real time or the amount of data written into the SD card. The x-axis simply represents the number of executions of the corresponding file operations. Thus, these results are not applicable in evaluation of the operation time versus the file size. Unfortunately, it would have been more relevant information, hence, the open file operation is related to the amount of data on the SD-card [39]. In summary, the close file and write file operations were not related to the file size. Conversely, the open file time increased all the time. The minimum, maximum and average file operation times are represented in table 2. If we compare the average file operation time to the filling time of the FIFO buffer of the accelerometer $8 \times 32 = 256\mathrm{ms}$, it is obvious the buffer will overflow. Nevertheless, in the beginning of the recording

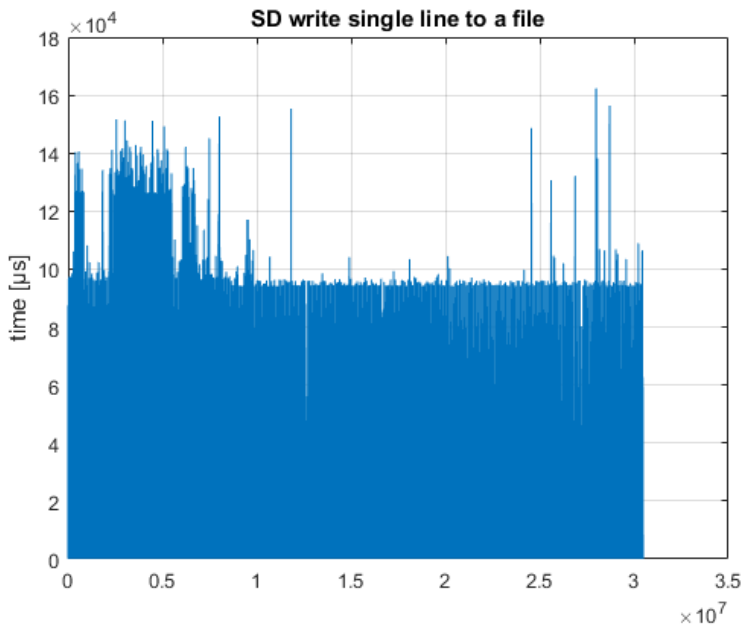

Figure 13:

Table 2: The durations of the SD file operations of the first data record.

Measure	**Open File** [ms]	**Write Data** [ms]	**Close File** [ms]
Minimum	7.26	0.056	8.04
Average	747.14	9.98	19.75
Maximum	1440	16.24	160.03

process the duration of the file operations were more tolerable.

4.1.3 Behavior monitoring ok

In the beginning of this study, the behavior monitoring was in our scope as a secondary target. In order to reach this target we used trail and action cameras in parallel with the sensor device. The purpose was to find correlation between visually detected behavior and the characteristics in the data. While recording the first data set, the cow was equipped with a trail camera. Both, trail camera and the sensor device were attached to its collar with a suitable collar band. The trail camera was configured to take a picture periodically after each 30 s. The battery and memory card of the trail camera were estimated to last maximum of 24 h. The pictures from the trail camera were downloaded on computer in parallel with the data from the sensor device. Unfortunately, the pictures were quite garbled and it was difficult to infer the possible behavior. Nevertheless, the picture series contained obvious eating period. However, the synchronization of the pictures and the recorded data was not effortless to do. Both of the devices lacked descent and reliable clock. Furthermore, the initial starting time differed too much, which caused significant initial time shift between the data and pictures. Therefore, it was concluded that behavior monitoring is possible if data and visual information are more properly synchronized.

In the beginning of the recording of the second data set, a video camera was mounted on the ceiling of the cowshed. However, the cowshed was too large with plenty of other cows. Therefore, tracking of a cow was a challenging task. The cows with sensors were marked with colors in order to ease the visual tracking. In spite of these actions, the cows were most of the time invisible. Furthermore, the second data set suffered the same time synchronization issues as the first one. That is, when a cow appeared in the video it was usually traceable at least for several minutes. Nevertheless, there were no kind of holding point to find reliable correlation between the visual observations and the data. Thus, the secondary objective of behavior monitoring was rejected.

4.1.4 Skin Temperature ok

In this study, we attempted to utilize the temperature sensor to either confirm or reject the results of estrus detection. In order to achieve this target, the sensor was equipped with a heat conducting tape. The tape was supposed to conduct the heat from the skin of the cow to the temperature sensor on the circuit board as discussed in section 3.1.3. In addition to this temperature sensor, we are able to utilize the temperature sensor of the accelerometer. Nevertheless, the resolution of the accelerometer was less satisfactory than the resolution of the temperature sensor. Unfortunately, the results of the skin temperature measurements were not promising. The readings seemed to follow some external source of heat instead of the skin temperature of the cow. In result, the measured temperatures varied approximately from $5\,^{\circ}\mathrm{C}$ to $25\,^{\circ}\mathrm{C}$ along the day as seen in the figure 14. Additionally, it was attempted to the differential of the sensor as a measure. Nevertheless, it did not seem beneficial approach. Thus, also the skin temperature measurements were omitted.

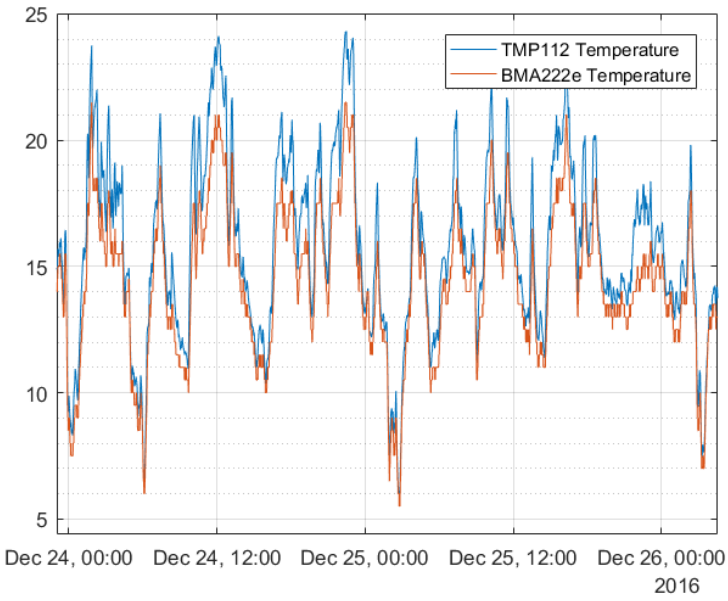

Figure 14: The skin temperature measurements of a cow. The measurements seems to follow some external source of heat rather than the actual skin temperature.

4.2 Algorithm Evaluation ok

In previous subsection, we discussed of the results in general. The results of the first recorded data set were used for improving the sensor device software as discussed in section 3.1.2. Conversely, the results of the second data set were used for developing and testing of the estrus detection algorithms as discussed in section 3.3. An additional third set of data was utilized in testing and tuning of the developed algorithms. Accordingly, this section discusses of the results of the estrus detection algorithms. First, we overview the results of estrus detection in general level.

In this study, the Heatime activity and rumination monitoring system was utilized as a reference system for our estrus detection. Thus, each cow wearing our sensor device was also equipped with Heatime sensor. In this study, tThe Heatime is considered a reliable activity measurement system. Additionally, significantly increased level of activity is considered as a proestrus. Furthermore, the Heatime allow us to see the activity history of the cow. Thus, the former activity periods can confirm the the recent activity period as a proestrus if they are within the duration of a normal estrus cycle. Nevertheless, none of the detected estruses were not verified by a veterinarian. Thus, the results of this study as well as the results of the Heatime should be considered with certain suspicion. That, is there might be false detections or other unexplained occasions appearing in the data output.

As discussed earlier, a total of six cows were used for the actual estrus data recording. The first three cows were labled with numbers, 319, 659 and 9885. Of the first data set only data of two cows, 319 and 9885, were valid to use in the algorithm development. Both of these two cows had two or more activity periods in the log of the Heatime. The Heatmime log of a cow 319 is represented in figure 15. In this log, the cow 319 had two highly active periods that are considered as a proestrus. The next three cows utilized in data recording were 767, 787 and 812. Of those only the data of the cows 767 and 787 were valid and the data of 812 was corrupted. Thus, total of four cows were used in algorithm development and evaluation in this study. All these data recordings lasted approximately 30 days and each consisted a period of increased activity. The figure 16 represents what the increased activity looks like in raw data prior data processing. The increase activity is seen as an increase of motion amplitude during a half-day period. The full list of increased activity levels are represented in table 3. This table also consists description whether the period of increased activity is within our data recording period or not.

Previously, we discussed of the activity periods detected by the Heatime system. However, it is only the reference for our own estrus detection algorithms. Therefore, the following subsections will evaluate the algorithms separately. However, in order to validate the results of the algorithms, it is mandatory to define the necessary measures of quality. Considering the estrus, the cow either is in proestrus or is not in proestrus. Thus, four simple but exact fundamental measures, *true positive*, *true negative*,*false positive* and *false negative* are suitable this evaluation. Furthermore, the definitions of the fundamental measurements are as follows.

- *True positive* means a condition when the algorithm detects an ongoing estrus and the cow is in estrus.
- *True Negative* is a condition when the estrus detection algorithm yields negative for

Table 3: Increased activity periods detected with Heatime cow monitoring system. Increased activity is considered as proestrus. These detections are used as a reference in this study.

Cow ID	**Occurrence of increased activity**	**Notifications**
319	30-11-2016	High activity period considered as proestrus. However, this period is outside of our sampling period.
319	19-12-2016	Second period of high activity considered as proestrus. This activity period is within our data recording period.
9885	24-11-2016	First high activity period of this cow and it is considered as a proestrus. However, the occurrence is outside of our data recording period.
9885	22-12-2016	The second activity period of this cow is considered as proestrus. However, this activity period is delayed possibly because of stress cause by the new sensor. Nevertheless, the proestrus is within our data recording period.
9885	04-01-2017	Third high activity period of the cow 9885. In contrast to the previous occurrence, this is slightly early. However, also this occurrence is within our data recording period.
767	21-01-2017	The first high activity period of this cow. Unfortunately, the occurrence is outside of our data recording period.
767	13-02-2017	The second activity period of this cow is within of our data recording period.
787	08-01-2017	The first activity period of this cow. However, the amplitude of the activity is not very high. Therefore, it is considered as proestrus with restrictions. Additionally, the occurrence is outside of our data recording period.
787	19-01-2017	The second increased activity period of the cow. This time the activity is high enough and the event should be considered as proestrus. However, also this occurrence is outside of our data recording period.
787	10-02-2017	The third occurrence of increased activity of this cow. The occurrence is within our data recording period. Thus it can be used as a reference in algorithm development.

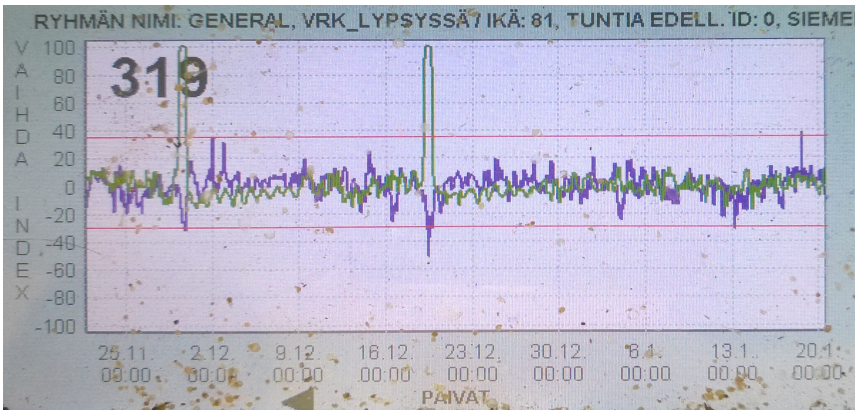

Figure 15: The activity data of Heatime system of the cow 319. The cow has had estruses approximately on October the 30th and December the 20th. The latter of the estruses should be detectable in this study, hence it is within our data recording period.

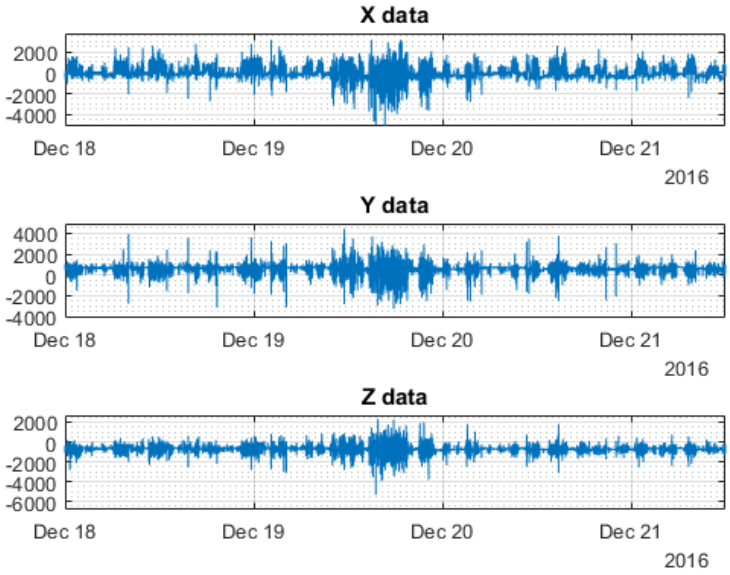

Figure 16: The raw acceleration data of each axis of the cow 319. The time frame of this picture is restricted around the estrus.

estrus and the cow is not in estrus.

- *False positive* states that the estrus detection algorithm yields positive for estrus but the cow is not in estrus.
- *False negative* mean a condition when the estrus detection algorithm yields not in estrus but the cow is in estrus.

In addition to these fundamental measures, it is beneficial to have some additional descriptive evaluations for the algorithms. Thus, amplitude difference between normal and

increased activity periods can describe the risk of false detections.

The following subsections discusses of each estrus detection algorithm separately. The discussion includes the description of the performance and reliability of the algorithm. Additionally, we argue over the algorithm specific parameter configurations and their effect on the results. In this study the parameters are tuned with iterative methods. That is, testing various values and evaluating the results before applying new values. This iteration is continued until, the results are satisfactory and further tuning does not improve the results. Similarly, we attempt to reach some boundary vales for the parameters that increase the possibility of false results. The evaluation starts with the activity measurement algorithm. The algorithm rather straight forward and does not require advantageous computation. Next, we evaluate the variance detection algorithm. The algorithm is based on the variance of periodic and short data samples. Thus, it is more complex than the first algorithm. Furthermore, it consists more tunable parameters than the fist one. Lastly, we evaluate the inactivity detection algorithm. In spite of the name of the algorithm, it attempts to detect lack of in activity, which corresponds the highly active proestrus period. Similarly to the variance detection algorithm, the inactivity detection algorithm contains more tunable parameters than the first one. These parameters are discussed in the evaluation.

4.2.1 Activity Monitoring ok

In this section, we evaluate the activity measurement based estrus detection algorithm. As discussed earlier, total of four cows were used in algorithm development, hence data of two cows were corrupted. Luckily, the data of four cows consisted of five periods of increased activity that could be considered as proestrus. Thus, we have five detectable estruses. This algorithm is very straight forwarding and does not consist any complex phases except the high-pass filtering as an offset compensation. However, the high-pass filter consist of simple multiplications and summing the results. Thus, it is not that complex machine-wise. Furthermore, in real application the filtering is executed in the accelerometer with simple configurations. Therefore, the filtering should not be considered as s formal part of the algorithm. Nevertheless, the offset compensation is a necessary step, hence the offset would distort the results as discussed in the section 3.3.1.

In general, the activity monitoring algorithm performed the estrus detection satisfactory. That is, it did not yield any false positive or false negative results according the logs of the Heatime system. An example of a detected estrus is represented in figure 17. In this figure, the highly active proestrus period appears as a peak in the middle of the monitoring period. Additionally, the amplitude difference between the peak value and other periods are significant. Thus, in this case the risk of false positives and false negatives is minor. Furthermore, the timing of the detection correlates with the Heatime. However, the results were not this clear with all the cows. In the worst case the normal activity reached nearly the value of 0.6 whereas the maximum is 1. Thus, in that case there is obvious risk of false detection.

Because this algorithm was highly straight forward it lacked of real tunability of its parameters. However, the size of the sliding integration window was variable 3.3.1. Thus, the effect of different sizes of the integration window was tested with this algorithm. The results of different sized integration windows are represented in figure 18. As seen in

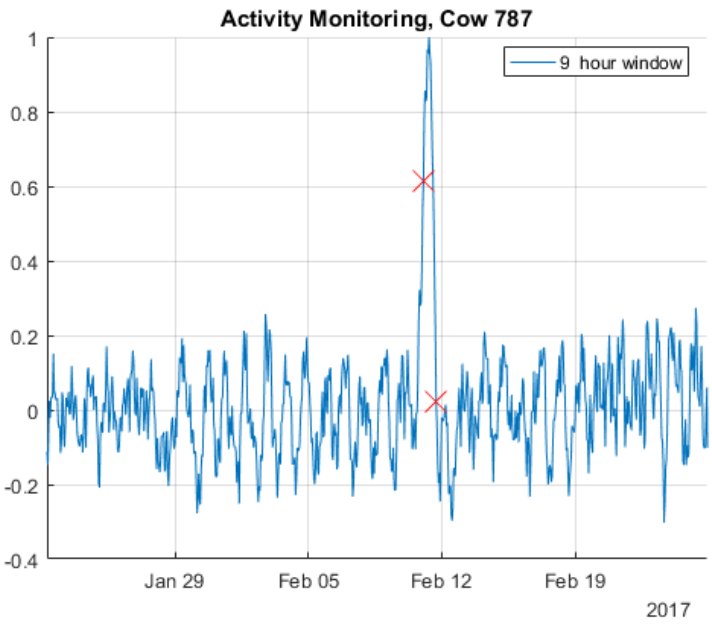

Figure 17: The results of activity measurement of the cow 787. The difference between the estrus and rest of the period is most distinct within this algorithm. Thus, the risk of false positive and false negative results is least significant.

the figure, a long integration time delays the detection of the end of proestrus period and, consequently delays the detection of the estrus. In this case, the $36\,\mathrm{h}$ window delays the estrus detection past the actual estrus. Thus, the integration time is too long. Conversely, a short integration time yields the risk of triggering estrus too early. In this figure, there is a hollow section during the activity period. There is an obvious risk for this intermediate decremented to trigger the end of proestrus too soon. In conclusion, integartion window from $9\,\mathrm{h}$ to $12\,\mathrm{h}$ is an optimal integration window. This conclusion correlates with the duration of the typical dairy cow proestrus.

4.2.2 Variance Detection ok

Above, we evaluated the results of the activity monitoring algorithm. The algorithm, is the most simple algorithm in this study and it is based on continuous data sampling. In general, the algorithm performed the estrus detection satisfactory. However, it utilizes the micro-controller most of the time and therefore, it is not the preferred solution considering the power consumption. Conversely, the variance detection algorithm attempts to minimize the utilization of the micro-controller by using short and seldom data samples instead of continuous data streams. Consequently, this algorithm contains two more configurable variables than the activity monitoring algorithm.

The first configurable parameter is the size of the sample. The size of the sample has a major effect on the workload of the micro-controller. That is, the variance requires of calculating the mean value and as many square roots as the sample size. Furthermore, it requires working memory to store all the intermediate values prior yielding the final

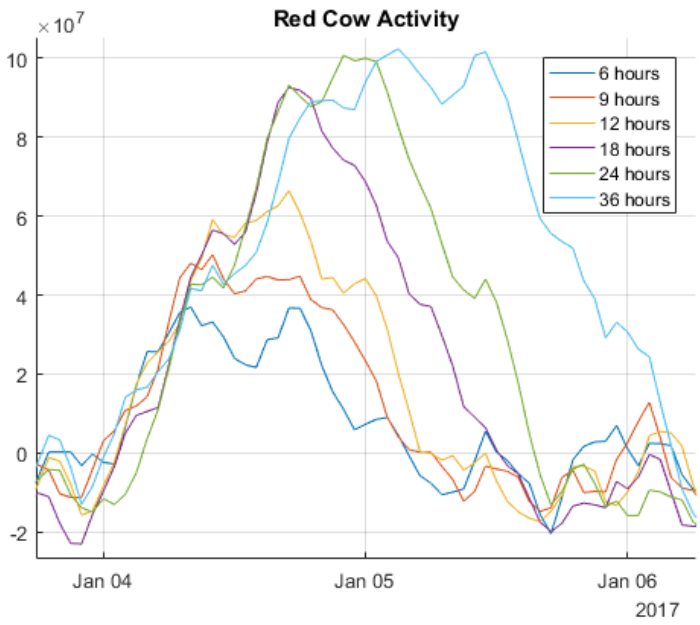

Figure 18: The activity plots of the second proestrus of the cow 9885. The figure illustrates the affect of varying the size of the integration window. Wider window increases the amplitude of the estrus and eases the detectability. However, simultaneously it delays the moment of detection.

output value. Thus, the size of the sample shall remain as small as possible. However, it is possible that too small sample does not include enough of variation. Consequently, it might degrade the results. The second configurable parameter is the sampling interval. As already discussed, it is also possible to overlap the samples 3.3.2. However, such an approach is against the attempt to reduce the utilization of the micro-controller. Otherwise, frequent sampling ensures all the behavior of the cow has being sampled and processed according the specifications of the algorithm. Nevertheless, seldom sampling improves the power saving capabilities of the micro-controller. However, the activities of the cow are likely missed with too seldom sampling hence, the cow moves infrequently and the moving time is relatively short. In conclusion, the purpose of tuning of the algorithm is to find the balance between sample size and sampling interval. An example of detected estrus with balanced variables in represented in figure 19. The sample size of 32 data points corresponds of two second sampling period whereas the sampling interval of 72 seconds is more than one minute. Consequently, the algorithm has reduced the amount of required samples by $97.2\,\%$ in comparison to the activity monitoring algorithm.

In addition to searching the optimal values for the estrus detection, one objective is finding the boundary values for the algorithm. The boundary values are such that the algorithm becomes unstable and yields false results if the parameters are not within the boundaries. Figure 20 represents one of the boundary solutions. That is, the algorithm has triggered a false estrus on February the 18^{th} because of too seldom sampling interval. However, increasing the sample size to 64 data points stabilizes the algorithm but is is more demanding for the micro-controller to calculate. Correspondingly, less than 16 data

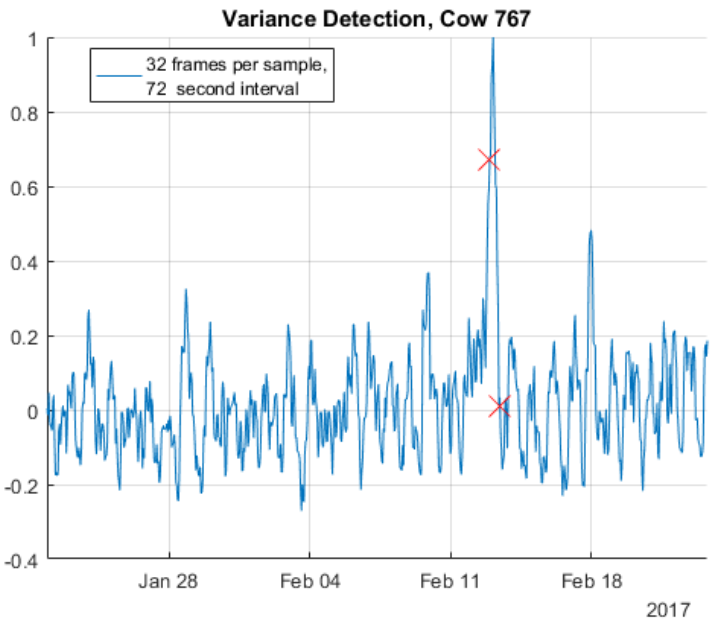

Figure 19: The variance detection results of the cow 767. There are multiple false positive detections in the data period. In general, there is no obvious difference between the estrus and non-estrus periods. Nevertheless, a true positive estrus is detected on February the 13^{th}.

points yields false results with sampling interval of 32 seconds. However, decreasing the sampling interval to 24 seconds stabilized the algorithm. However, the sampling interval is three times more frequent in comparison to the optimal values.

In general, the variance detection algorithm performed the estrus detection satisfactory. However, in comparison to the activity monitoring algorithm, this algorithm showed marks of instability. Thus, the sample size and sampling interval shall be configured with care. Nevertheless, the algorithm capable of reducing the up-time of the micro-controller significantly. Therefore, it might be rather preferred option than the activity monitoring. Additionally, the amplitude difference between the peak value and the other periods remained within same values as it was with the fist algorithm. That is, the maximum value of non-proestrus period with the optimized parameters were 0.5 whereas it was 0.6 with the first algorithm.

4.2.3 Inactivity Detection ok

In contrast to the two previous algorithms, the inactivity detection has an alternative approach to the estrus detection. The first two algorithms are measuring the level of activity whereas this algorithm detect absence of activity. However, eventually the estrus detection is based in absence of inactivity. Finally, in could be considered as activity detection. Nevertheless, the another difference between inactivity detection and the other two algorithms is execution of the core functions of the algorithm in the accelerometer instead of utilizing the micro-controller. Therefore, the power saving capabilities if this

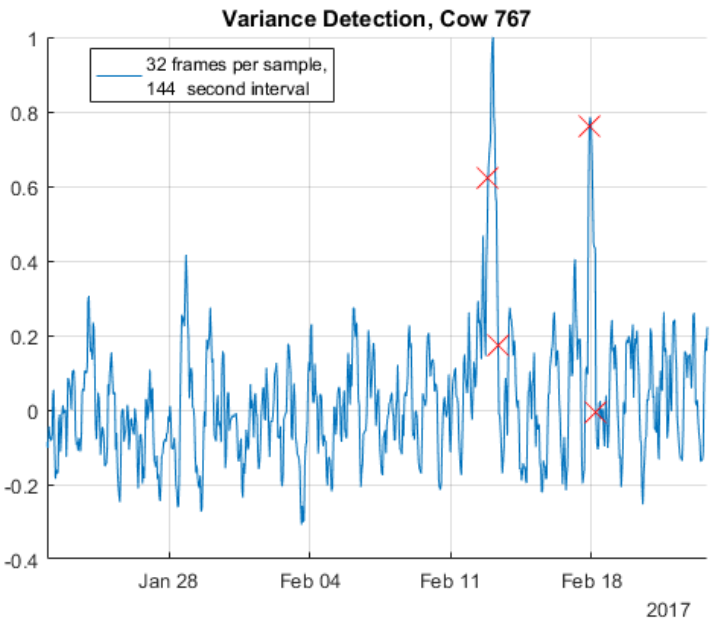

Figure 20: More seldom sampling yield false positive results as seen in this picture.

algorithm is the highest in this study. Furthermore, the minimum interval between generated interrupts are configurable as discussed in section 3.3.3. In result, the micro-controller books seldom interrupts and integrates them within the integration window of 9 h to 12 h. Additionally, the absence of interrupts is related to highly active proestrus behavior. Furthermore, in contrast to the first two algorithms, the peak value points downwards as seen in the figure 21. The inactivity detection algorithm contains two configurable parameters, motion threshold and delay. In this figure the motion threshold is set to 2 g and the delay is the maximum of 336 s. As seen in the figure, the amplitude of non-proestrus periods is low in comparison to the two previous algorithm.

Furthermore, the inactivity detection algorithm seemed the most stable and reliable algorithm in this study. It was highly tolerable to the parameter configurations. Eventually, only most irrational parameters yielded unstable behaviour of the algorithm as seen in the figure 22. In this case, the motion threshold is as low as any motion of the cow. Thus, the no-motion conditions are not met that often. Similarly, if the motion threshold exceeds the normal motion values of the cow, 4 g, the no-motion interrupts are always generated and therefore, there are no difference between proestrus and other periods. Additionally, decreasing of the delay had no significant impact on the results. Decreasing the delay only generates more interrupts and consequently, increases the up-time of the micro-controller and degrades the power saving capabilities.

In summary, the inactivity detection algorithm is highly stable against parameter configurations. Furthermore, it was able to detect all of the estruses. Moreover, the amplitude difference between proestrus and other periods were the most significant with this algorithm. The maximum non-proestrus amplitude with the optimized parameters was 0.2 whereas with other algorithms the amplitude were 0.6 and 0.5. Furthermore, this algorithm provides the highest capability for power saving. Firstly, the delay configuration

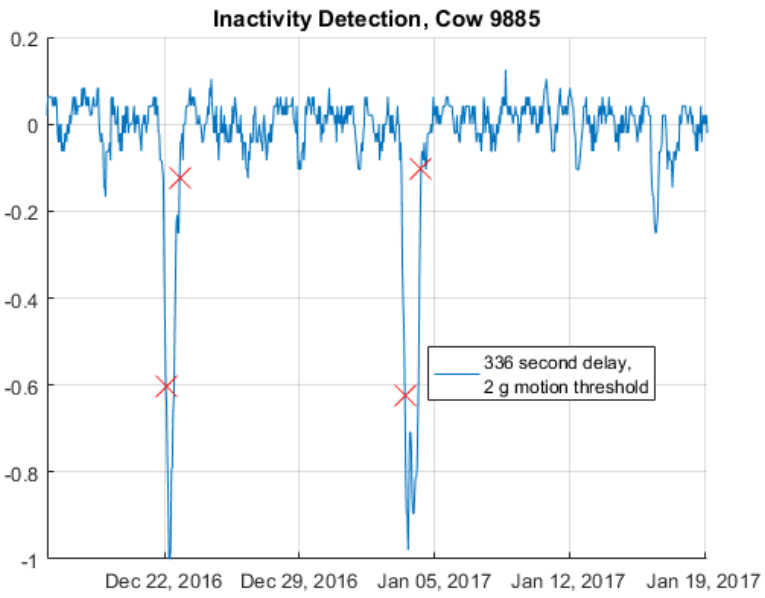

Figure 21: The results of inactivity detection algorithm of the cow 9885. The parameters are 336 second delay and $2\,\mathrm{g}$ motion threshold. Both of the estruses are detected as true positive on December the 22$^{\mathrm{nd}}$ and January the 4$^{\mathrm{th}}$. No false positive or false negative detection occurred. The amplitude difference between proestrus and non-estrus is obvious.

defines the minimum interval between the interrupts, which is $336\,\mathrm{s}$. That is, a maximum of 10.7 interrupts per hour. Secondly, the micro-controller is not required to perform any demanding calculations. It is only responsible of integrating the number of interrupt within the integration window of $9\,\mathrm{h}$ to $12\,\mathrm{h}$. Thus, within all of the measures of the quality an performance, the inactivity detection is the preferred algorithm for dairy cow estrus detection.

5 Conclusions ok

Previously, we discussed of the results of this study. The results included general and intermediate results that were used in algorithm development as well as in improvement of the sensor device software. Additionally, we evaluated the results of the estrus detection algorithms. This subsection summarizes the achieved results and draws comprehensive conclusions. First, we will discuss of sensor based estrus detection. The discussion will first evaluate the consent of sensor based estrus detection. Additionally, we will focus on the deficiencies of this study and propose corrective actions. Next, we will cover the occurred failures of this study. Moreover, we will discuss of some consideration related to data recording and wearable sensors in general. Lastly, in this section we propose suggestions for following studies.

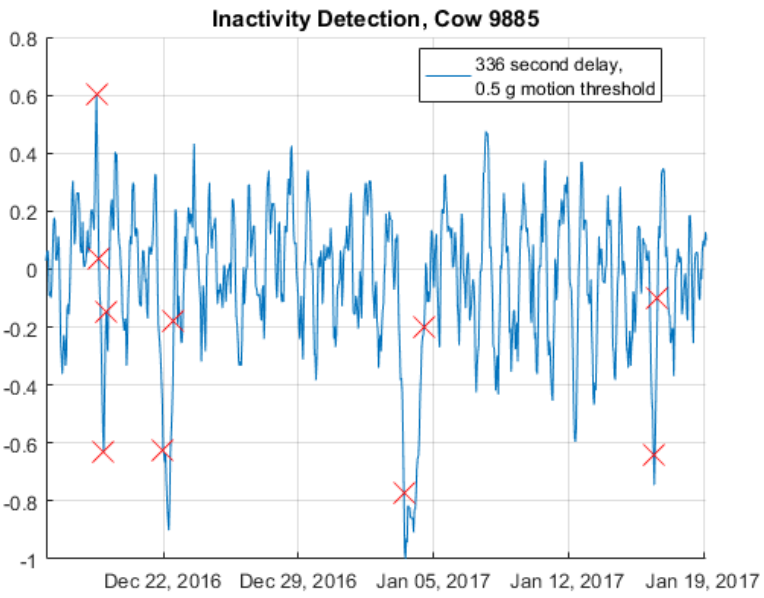

Figure 22:

5.1 Sensor Based Estrus Detection

This study introduced a sensor device for dairy cow estrus detection. The introduction included as well as the hardware configuration as well as the software implementation. Furthermore, we discussed of three different estrus detection algorithms suitable for the device. Practically, the algorithms were developed and tested with Matlab software on computer after recording of real cow data. The algorithms were different in their approach to the topic. However, all of them were based on the accelerometer data only. In summary, all of the algorithms performed the estrus detection satisfactory after tuning of the algorithm specific parameters. Nevertheless, there were some differences in the reliability as well as in the clearance of the results. Furthermore, the most significant differences between the algorithms were in the power saving capabilities, which is essential with battery-powered devices.

The activity monitoring based estrus detection algorithm was a straightforward integration of the accelerometer data. The only configurable parameter of the algorithm was the length of the integration time. In result, The most suitable integration time corresponds approximately the duration of the proestrus phase of the estrus cycle, that is roughly from 9 to 12 h. Extending of the integration time only delayed the triggering of the begin of estrus. Thus, it was not beneficial. Conversely, integration time less than 6 h yielded a risk of triggering estrus too early in the middle of the proestrus phase. These results were so obvious, hence, it was decided to use 9 h window with the rest of the algorithms without further tuning of it. Furthermore, the algorithm was quite robust. That is, the results were clear and there were no high risk of yielding false results within the suggested integration time. However, the algorithm is sensitive to data offset and it shall be removed before any other computation. Nevertheless, offset did not prevent from detecting the estrus in this

study. Actually, it only reduced the amplitude difference between the proestrus and other periods.

In contrast to the activity monitoring algorithm, the variance detection algorithm was sensitive to the parameters. That is, too short sample or too long sampling interval effected to the results significantly. However, ...

Similarly to the variance detection, the inactivity detection algorithm had two additional tunable parameters. However, inactivity detection did not seem to be sensitive to the tuning of the parameters. In this study, any reasonable parameter values yielded true positive and true negative results. Furthermore, no false results occurred. Nevertheless, the most radical parameter values begun to affect to the amplitude difference between proestrus and other phases of the estrus cycle. In conclusion, the performance and reliability of this algorithm seemed most promising in this study. Furthermore, this algorithm is the most suitable for the micro-controller based solutions. That is, most of the computations are performed in the accelerometer and the micro-controller only summarizes seldom interrupt the accelerometer generates. Additionally, it enables efficient use of power saving modes of the micro-controller which is critical with battery powered solutions.

The dairy cow estrus is detectable with triaxial accelerometer only and no other sensors are required. The results were verified with one single commercial product. Thus, the results may not be completely reliable and valid. Furthermore, none of the detected estruses were verified by a vet.

The estrus detection algorithms developed in this study are suitable for micro-controller environments. Therefore, algorithm computation with computers is not required. Consequently, there is no need to large data storage or transmitting raw sensor data wirelessly.

5.2 Failures and considerations

Previously, we discussed of sensor based dairy cow estrus detection based on the results of this study. The conclusion was that the estrus of a dairy cow is detectable with an accelerometer base wearable sensor. Furthermore, it seemed the estrus is rather detectable with various algorithms. However, there were differences in the performance and reliability of different algorithms. Whereas the previous subsection focused on the successful part of this study, in this subsection the scope is in failures and suggestions for improvements.

Firstly, the body temperature measurement was a total failure. That is, the heat conducting system did not conduct the heat enough. Conversely, the results of the temperature measurements seemed to follow the temperature of the surrounding air. It was also seen from the temperature curve of the accelerometer **??**, hence, both of the temperatures seemed synchronous. Additionally, it was tested to filter the results and calculate the differential of the two temperatures sensors. Nevertheless, no correlation with a ongoing estrus or proestrus were found in this study. In contrast to the sensor setup of this study, the temperature sensor should be placed into more direct skin contact. Thereby, the sensor would more likely measure the body temperature rather than the surrounding air. Nevertheless, this discussion is only hypothetical. Furthermore, the body temperature was assumed to rise only up to two degrees during the estrus. Therefore, detecting of such minor alterations with micro-controller suitable algorithms may not be reliable at all.

Secondly, total of three attempts to record cow data were ruined because of an error

state of the sensor device. Unfortunately, there was no method to define the cause of the error directly. However, it seemed that the device continued working even in the error state but the data was invalid. However, further analysis of the error data provided an assumption that the accelerometer had reset meanwhile the micro-controller had not. Consequently, the micro-controller stopped receiving any interrupts from the accelerometer. Furthermore, the range of the accelerometer was set to default of $2\,\mathrm{g}$. Meanwhile, the micro-controller applications continued of executing of its software loop. Nevertheless, without fetching data normally. Therefore in both software implementations, the watchdog timer triggered an event for reading single data frame from the FIFO memory of the accelerometer. Considering the number of the invalid samples and the period of watchdog timer matched with the time cow was wearing the sensor. In order to avoid such failures in the future, the software should include a method for detecting erroneous state of the accelerometer. Additionally, the hardware might be causing blackouts to the accelerometer. This possibility should be inspected and the hardware design improved according the discoveries.

Lastly, the hardware design did not provide optimal performance for high sampling rate as seen in section 4.1. Therefore, the first recorded set of data was partially corrupted because of frequent buffer overflows. Furthermore, high sampling rate fulfilled the SD memory card relatively fast. Thus high sampling rate together with SD memory card would not be preferred solution in long term data recording.

5.3 Suggestions for Future Studies

This study has introduced three different sensor based algorithms for dairy cow estrus detection. Furthermore, we have evaluated the algorithms and concluded that the estrus id detectable with accelerometer based wearable sensor. Additionally, we have discussed of secondary products and other observations during this study. Lastly in this study, we will discuss of suggestions for potential future studies based on the introduced results and conclusions.

Firstly, the detected estruses in this study were not confirmed at any point. The results were verified only with another sensor device as a reference. Thus, the results of this study are not completely reliable. Therefore, the algorithms developed in this study shall be implemented directly for micro-controller. Furthermore, they shall be tested with dairy cattle in real-time instead of posterior data processing sessions. In order to fully validate the algorithms, the results shall be verified by a veterinary. That is, each cow shall be inseminated once an estrus has been detected. Consequently, the detection is considered as true positive if the insemination is successful and the cow becomes pregnant. Nevertheless, the normal success rate of insemination as well as the estrus cycle shall be taken into account.

Secondly, the amount of data used for algorithm development was quite small. That is, only four months of cow data was valid to use in algorithm development. Luckily, the data consisted total of five estruses. Nevertheless, statistically sampling is minor considering the head count world wide. Furthermore, it is assumed there are significant differences in cow activity levels, hence, it varied already in this study. Additionally, a complementary study should cover various breeds in various environments e.g., tie-stall

and pasture. Furthermore, we did no discuss of algorithm tuning thoughtfully. Actually, the methods of this study tuned the algorithms for each cow individually after analyzing all of the data. Thus, the parameters suitable for one cow might not be suitable for another. However, more of recorded data would enable the developing of algorithms with self-adjustable parameters. Consequently, it would improve the real-time performance of the estrus detection algorithms discussed in this study.

Thirdly, behavior monitoring was considered as on optional topic in this study. A successful behavior monitoring would provide prominent information for the farmer about the state of the cattle. Furthermore, exceptions in behavior could indicate concerns in the cows health and treatment could be started before issues escalate. In this study, we recorded in-cowshed video for studying of behavior recognition. However, the the video quality was insufficient and there were no valid time synchronization between the video and motion data. Thus, this option was discarded in early phase of this study. However, based on the brief preliminary study, this option seemed promising. Therefore, study of valid motion and video data with proper time synchronization would provide algorithms for such topics as rumination and lameness detection.

References

[1] June 24, 2013. URL: https://www.proagria.fi/ajankohtaista/lehmien-tuotos-kasvaa-jokaisella-poikimakerralla-804 (visited on 01/19/2017).

[2] United States Department of Agriculture. *Agricultural Statistics 2015*. URL: https://www.nass.usda.gov/Publications/Ag_Statistics/2015/Ag_Stats_2015_complete%20publication.pdf (visited on 01/18/2017).

[3] United States Department of Agriculture. *Livestock and Poultry: World Markets and Trade*. Foreign Agricultural Service. Oct. 2016. URL: http://usda.mannlib.cornell.edu/usda/fas/livestock-poultry-ma//2010s/2016/livestock-poultry-ma-10-12-2016.pdf (visited on 01/18/2017).

[4] L. Mattias Andersson et al. "Wearable wireless estrus detection sensor for cows". In: *Computers and Electronics in Agriculture* 127 (2016), pp. 101–108. ISSN: 0168-1699. DOI: http://dx.doi.org/10.1016/j.compag.2016.06.007. URL: http://www.sciencedirect.com/science/article/pii/S0168169916303581.

[5] L. M. Andersson et al. "Wearable wireless sensor for estrus detection in cows by conductivity and temperature measurements". In: *SENSORS, 2015 IEEE*. Nov. 2015, pp. 1–4. DOI: 10.1109/ICSENS.2015.7370219.

[6] Arduino. URL: https://www.arduino.cc/en/Main/Software (visited on 09/16/2017).

[7] Arduino. URL: https://www.arduino.cc/en/Reference/Wire.

[8] Arduino. URL: https://www.arduino.cc/en/Reference/SPI.

[9] Arduino. URL: https://www.arduino.cc/en/Reference/SD (visited on 09/23/2017).

[10] *ATmega16U4/ATmega32U4 8-bit Microcontroller with 16/32K bytes of ISP Flash and USB Controller*. Rev. 7766J – 04/2016. Atmel Corporation. Aug. 2007.

[11] *BMA222E Digital, triaxial acceleration sensor*. BST-BMA222E-DS004-6. Rev. 1.3. Bosch. Apr. 2005.

[12] Leandro dos Anjos Brunassi et al. "Improving detection of dairy cow estrus using fuzzy logic". en. In: *Scientia Agricola* 67 (Oct. 2010), pp. 503–509. ISSN: 0103-9016. URL: http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-90162010000500002&nrm=iso.

[13] Heli Castrén. *Kotieläinten käyttäytyminen ja hyvinvointi*. Mikkeli: Helsingin yliopiston Maaseudun tutkimus- ja koulutuskeskus, 1997.

[14] *Digital Filters*. 2nd ed. Springer-Verlag Berlin Heidelberg, 2011. ISBN: 978-3-642-14325-0. DOI: 10.1007/978-3-642-14325-0.

[15] Eric Dufresne. Mar. 20, 2005. URL: https://upload.wikimedia.org/wikipedia/commons/d/db/Cows_on_a_farm_-_by_Eric_Dufresne.jpg (visited on 08/26/2017).

[16] GEA Farm Technologies GmbH. URL: http://www.gea.com/en/binaries/DairyFarming_CowScout_Brochure_EN_0315_tcm11-21065.pdf (visited on 10/21/2017).

[17] *Heatime HR User Guide*. Rev. 2.13.X.X. SCR Engineers LTD. July 2014. URL: http://www.heatime.dk/wordpress/wp-content/uploads/2012/07/HeatimeHR-User-Manual-2.13.X.X.pdf.

[18] Jan Hulsen. Ed. by Juho Kyntäjä. 1079. Vantaa: ProAgria keskusten liitto, 2014. ISBN: 978-951-808-192-3.

[19] R. Jónsson et al. "Oestrus detection in dairy cows from activity and lying data using on-line individual models". In: *Computers and Electronics in Agriculture* 76.1 (2011), pp. 6–15. ISSN: 0168-1699. DOI: http://dx.doi.org/10.1016/j.compag.2010.12.014. URL: http://www.sciencedirect.com/science/article/pii/S0168169910002656.

[20] Dong-Won Kang et al. "Prediction of energy consumption according to physical activity intensity in daily life using accelerometer". In: *International Journal of Precision Engineering and Manufacturing* 13.4 (2012), pp. 617–621. ISSN: 2005-4602. DOI: 10.1007/s12541-012-0079-2. URL: http://dx.doi.org/10.1007/s12541-012-0079-2.

[21] A.A. Khan. *Digital Signal Processing Fundamentals*. Charles River Media Computer Engineering. Da Vinci Engineering Press, 2005. ISBN: 9781584502814. URL: https://books.google.fi/books?id=%5C_xeQNRlxzG4C.

[22] Tarja Kortesmaa and Aimo Jokela, eds. *Ruoka- ja luonnonvaratilastojen e-vuosikirja 2016*. Luonnonvara, ja biotalouden tutkimus 64/2016. Luonnonvarakeskus, 2016. ISBN: 978-952-326-322-2.

[23] S. López et al. "On the analysis of Canadian Holstein dairy cow lactation curves using standard growth functions". In: *Journal of Dairy Science* 98.4 (2015), pp. 2701–2712. ISSN: 0022-0302. DOI: http://dx.doi.org/10.3168/jds.2014-8132. URL: http://www.sciencedirect.com/science/article/pii/S0022030215000466.

[24] IceRobotics Ltd. URL: http://www.icerobotics.com/wp-content/uploads/2017/05/IceRobotics_Product_Brochure_2017.pdf (visited on 10/21/2017).

[25] Moocall Ltd. URL: https://moocall.com/products/moocall-sensors (visited on 10/21/2017).

[26] Well Cow Ltd. URL: http://wellcow.co.uk/bolus/ (visited on 10/21/2017).

[27] Paula Martiskainen et al. "Cow behaviour pattern recognition using a three-dimensional accelerometer and support vector machines". In: *Applied Animal Behaviour Science* 119.1–2 (2009), pp. 32–38. ISSN: 0168-1591. DOI: http://dx.doi.org/10.1016/j.applanim.2009.03.005. URL: http://www.sciencedirect.com/science/article/pii/S0168159109000951.

[28] MathWorks. URL: https://se.mathworks.com/products/matlab.html#overview (visited on 09/17/2017).

[29] MathWorks. *Fast Fourier Transform*. URL: https://se.mathworks.com/help/matlab/ref/fft.html (visited on 11/07/2017).

[30] MathWorks. *Introduction to Filter Designer*. URL: https://se.mathworks.com/help/signal/examples/introduction-to-filter-designer.html (visited on 11/08/2017).

[31] G.J. Miao. *Signal Processing in Digital Communications*. Norwood, 2007.

[32] M. Pastell et al. "A wireless accelerometer system with wavelet analysis for assessing lameness in cattle". In: *Biosystems Engineering* 104.4 (2009), pp. 545–551. ISSN: 1537-5110. DOI: http://dx.doi.org/10.1016/j.biosystemseng.2009.09.007. URL: http://www.sciencedirect.com/science/article/pii/S1537511009002748.

[33] Anne Penninkangas. "Nautojen hoitoympäristö". In: *Kotieläintilan luomuopas*. Tieto tuottamaan 78. Maaseutukeskusten Liitto, 1998. ISBN: 951-808-064-X.

[34] Pixabay. Mar. 31, 2016. URL: https://pixabay.com/fi/lehm%C3%A4-farm-pilttuu-el%C3%A4imet-1318283/ (visited on 08/26/2017).

[35] K.R. Rao, D.N. Kim, and J.J. Hwang. *Fast Fourier Transform - Algorithms and Applications*. Signals and Communication Technology. Springer Netherlands, 2012. ISBN: 9789048118892. URL: https://books.google.fi/books?id=i9P3sgEACAAJ.

[36] Mike Schwager. URL: https://github.com/GreyGnome/EnableInterrupt.

[37] *SD Specifications Part 1 Physical Layer Simplified Specification*. Version 6.00. SD Association. Apr. 2017.

[38] Raimo Seppänen and Seppo Tiihonen. "Matematiikka". In: *MAOL-taulukot*. 9th ed. Helsinki: Kustannusosakeyhtiö Otava, 1991.

[39] *SPI Block Guide*. Rev. 03.06. Motorola, Inc. Feb. 2003. URL: https://web.archive.org/web/20150413003534/http://www.ee.nmt.edu/~teare/ee308l/datasheets/S12SPIV3.pdf.

[40] Li Tan. *Digital Signal Processing: Fundamentals and Application*. Burlington: Elsevier Science, 2007. ISBN: 9.

[41] ThoughtCo. *Dairy Farming - The Ancient History of Producing Milk*. Feb. 9, 2017. URL: https://www.thoughtco.com/dairy-farming-ancient-history-171199 (visited on 08/19/2017).

[42] *TMP112 High-Accuracy, Low-Power, Digital Temperature Sensor With SMBus and Two-Wire Serial Interface in SOT563*. SBOS473E. Rev. E. Texas Instruments. Mar. 2009.

[43] *UM10204 I^2C-bus specification and user manual*. Rev. 6. NXP Semiconductors. Apr. 2014. URL: https://www.nxp.com/docs/en/user-guide/UM10204.pdf.

[44] Jorge A. Vázquez Diosdado et al. "Classification of behaviour in housed dairy cows using an accelerometer-based activity monitoring system". In: *Animal Biotelemetry* 3.1 (2015), p. 15.

[45] Eeva Vornanen. "Naudan hoitoympäristö". In: *Luomunaudan ja -sian ruokinta ja hoito*. Tieto tuottamaan 94. Maaseutukeskusten Liitto, 2001. ISBN: 951-808-097-6.

[46] Aaron Wisner. URL: https://github.com/wizard97/ArduinoRingBuffer (visited on 09/23/2017).