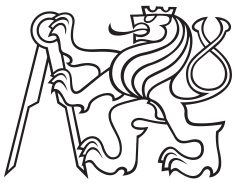
Master Thesis



Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Radioelectronics

Instrument design for punch detection in boxing training

Filip Bobuski

Supervisor: Prof. Ing. Pavel Sovka, CSc.,Ing. Pavel Máša, Ph.D.
May 2020

MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Bobuski Filip** Personal ID number: **459917**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Radioelectronics**

Study program: **Open Electronic Systems**

Branch of study: **Communications and Signal Processing**

II. Master's thesis details

Master's thesis title in English:

Instrument Design for Punch Detection for Boxing Training

Master's thesis title in Czech:

Návrh zařízení pro detekci úderu při boxerském tréningu

Guidelines:

1. Briefly describe the equipment and techniques used for boxing training purposes.
2. Design and implement a system for recording signals during the training. Discuss the possibilities of sensor placement and propose a method of data saving. Analyze recorded data to verify the correctness of the solution concerning the sampling and quantizing.
3. Create a database of the strokes under the different training conditions according to point 1. Perform preliminary analyzes to describe the data, including the design of the stroke model. Consider selecting appropriate signal descriptors.
4. Study the possibilities of methods based on matched filtering or correlation for impact detection.
5. Evaluate the results of the impact detection for differently strong and differently placed punches and summarize the results.

Bibliography / sources:

[1] S. M. Kay: Fundamentals of Statistical Signal Processing. Estimation Theory. Prentice Hall, New Jersey, 1993
[2] S. M. Kay: Fundamentals of Statistical Signal Processing. Detection Theory. Prentice Hall, New Jersey, 1998
[3] H. V. Poor: An Introduction to Signal Detection and Estimation. Springer-Verlag, New York, 1988
[4] J. G. Proakis at all: Algorithms for Statistical Signal Processing. Prentice-Hall, New Jersey, 2002

Name and workplace of master's thesis supervisor:

prof. Ing. Pavel Sovka, CSc., Department of Circuit Theory, FEE

Name and workplace of second master's thesis supervisor or consultant:

Ing. Pavel Máša, Ph.D., Department of Circuit Theory, FEE

Date of master's thesis assignment: **27.01.2020** Deadline for master's thesis submission: **22.05.2020**

Assignment valid until: **30.09.2021**

prof. Ing. Pavel Sovka, CSc. Supervisor's signature	doc. Ing. Josef Dobeš, CSc. Head of department's signature	prof. Mgr. Petr Páta, Ph.D. Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

Acknowledgements

I would like to thank prof. Ing. Pavel Sovka, CSc. and Ing. Pavel Máša, Ph.D., for exceptional mentoring and guidance he provided me throughout the making of this thesis. I would also like to thank my family and friends who helped me in my studies.

Declaration

I declare, that I have produced the submitted work independently and that I have provided all the information sources used in accordance with the Methodological Guideline on Ethical Principles in the Preparation of Graduate Final Theses.

Prague, May 22, 2020

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolkých závěrečných prací.

V Praze, 22. května 2020

Abstract

In this thesis, we set ourselves to investigate the mode of operation of an acceleration measuring device mounted on a boxing bag whose task is to detect an acceleration impulse caused by a punch. Devices position is on the top part of the bag, in the line of sight of the athlete. That particular device placement can be superior from the ease of use point of view. As immediate feedback regarding the progress of the exercise is possible without any other equipment necessary. Several other commercial devices with similar functionality are mentioned and discussed. Proper technique and mechanics of the boxing punch is stated and its resulting signal impulses described. Signal impulses for different punch placements are shown and described. Design and assembly of hardware based on the parameters of expected signals, together with its programming, are presented. Databases of punch signals for different punch placements are constructed. Median signal model needed for proper detection is proposed and justified. A statistical model of a signal is formulated, and UMP and GLRT detectors based on it derived together with sufficient statistics. Necessary unknown parameters were estimated by the maximum likelihood method. Detection accuracy and overall performance are evaluated, and possible alternative improvements discussed.

Keywords: detekce, estimace, box, úder, pohyb, akcelerometr, GLRT

Supervisor: Prof. Ing. Pavel Sovka, CSc.,Ing. Pavel Máša, Ph.D.
Technická 1902/2
160 00
Praha 6-Dejvice

Abstrakt

Cílem této práce je prozkoumat způsob provozu zařízení připevněného na boxovací pytel, jehož úlohou je detekovat impulzy zrychlení způsobené údery. Zařízení je umístěno ve vrchní části pytle, v zorném poli sportovce, kde mu poskytuje okamžitou zpětnou vazbu bez nutnosti dalšího vybavení. Diskutována jsou také další, komerčně dostupná zařízení sloužící k podobným účelům. Je popsána základní technika boxerského úderu a charakteristika impulzů způsobených údery s různou lokalizací. Práce obsahuje návrh, realizaci a naprogramování hardwaru přizpůsobeného předpokládaným signálům. Jsou vytvořeny databáze signálů úderů pro různá umístění a popsány. Jako model signálu byl zvolen medián. V práci je uveden statistický model signálu a odvození z něho vycházejících UMP a GLRT detektorů. Nezbytné neznámé parametry byly odhadnuty pomocí ML odhadu maximalizujícího jeho pravděpodobnost. Byla vyhodnocena detekční přesnost a celková výkonnost detektoru a navržena možná alternativní řešení.

Klíčová slova: detection, estimation, boxing, punch, MLE, motion, accelerometer, GLRT

Překlad názvu: Návrh zařízení pro detekci úderu při boxérském tréninku

Contents

Figures

Tables

Chapter 1

Introduction

In the sport of boxing, heavy bag boxing drills are an essential part of the practice of any boxer. It trains his power, endurance and technique among many other abilities. Therefore improving upon the experience of a boxing drill could be beneficial. For example, tracking the number of punches in the round or the cadence of the series of punches gives the boxer and exact benchmark for future improvements and quantify his progress thus making his preparation more precise and effective. It can also make it more enjoyable with stimulating challenges. But for all this functionality, reliable punch detection is imperative. In this domain, there are several commercially available devices such as Hykso, Corner, PunchLab, UFC force tracker and others. We will focus on devices mounted directly on a boxing bag, which brings about several advantages for user experience. A device placed on a boxing bag in or slightly above the eye-height of the athlete allows for immediate visual observation of the displayed statistics. Such placement is superior for user experience, but maybe suboptimal for punch detection. The various unpredictable parameters such as boxing bag height, weight, composition or suspension height make it an appealing detection problem. Firstly we will examine the characteristics of prevalent signals sensed by the intended device. Then we will construct a measuring apparatus with three-axis accelerometer for data acquisition. After we create several databases of punches, we will proceed with the signal analysis, that will create a foundational knowledge for signal model creation and detector proposal. The designed detector will be implemented, and its performance tested on the created databases.

Chapter 2

Contemporary state of problems

There are many electrical devices which assist athletes with training and judges with more accurate decisions in competitions even in martial arts. We will focus on commercial alternatives first.

Two most popular wearable punch trackers are Hykso and Corner. They are using two trackers mounted under the wrist wraps of the boxer. Data collected with accelerometer and gyroscope sensors are processed via artificial intelligence trained on previous data (in case of Hykso). Those options offer intensity tracking, punch recognition, detection and other advanced functionality due to their advantageous tracker placement on the wrists. Another wearable solution called exIMU [11] is based on high range accelerometer and gyroscope for motion tracking and Kalman filter for data processing. Devices more similar to ours which are placed on a punching bag are for example PunchLab, UFC Force Tracker or punch sensor [17].

PunchLab is an application on a mobile phone which is fastened to the punching bag in an eye-height of a boxer. In this way, it serves both as a tracking device and as a display.

The UFC Force Tracker is a stand-alone device mounted on the bottom of the bag, and when connected to a mobile phone or tablet, shows information about athletes training in real-time. Those are the most known commercial products available on the market. Other found solutions were usually based on a force sensors [13] implemented by flexiforce sensors, FITRO agility plate [14] or similar type of sensors. However, commercial character limits public information about their internal operation.

For more professional or laboratory use, the following devices were found. Apparatus mentioned in [18] is built around a torsion bar and a target attached on a lever arm attached to the said torsion bar. When a target is hit, statistics about the force of the punch is calculated from the torque gauge and its known properties. In 1996 Klapman [4] patented a device embedded in the boxing glove with a pressure sensor in the punching area of the glove connected to a micro-controller and accelerometer sensor placed on the bottom part of the wrist area. Data were wirelessly transmitted to a computer for processing. A similar device was devised by a team of Punthipayanon, Sirichet, Senakham and Tanormsak [12]. It is using a FlexiForce force sensor with ten-millimetre diameter, which was sandwiched by a pair of flat equal-area circular rigid

pucks and embedded between padding layers of the boxing gloves at the knuckle scoring area. Again micro-controller in the glove processed the data, encrypted it and sent to a computer. The possibility to accurately measure force comes with a hefty price. The need for custom made boxing glove, whose manufacturing is a labour intensive handicraft work increases the cost of the whole apparatus. Also, every glove is different, so every one of them has to be individually calibrated. Commercial alternatives also offer force measurements, but without the knowledge of the weight of the punching bag or the boxer, it is just a rough estimate.

Unfortunately, in the scientific articles, information relevant to our use case was sparse. Still, research in the patent library had given us overall awareness about solutions implemented potentially outside the commercial use. We found solutions predominantly using force and pressure sensors. Solutions proposed in [5], [10], [9], [8] were made to fit in the punching bag and all used some kind of a force measuring sensor. Mentioned were custom built vibration sensors, piezoelectric sensors or special electric switches [7], that can be are bridged when contacted by a moving object.

We also found some unusual approaches. A punch detection and force measurement proposed in [6] is based on an interacting magnetic field, where the moving body includes a material highly permeable to a magnetic field, and the target includes means for generating a magnetic field in its vicinity. The magnetic properties of the target are being detected by using a magnetic field detector.

The wast majority of found solutions were either meant to be built into the target area or designed as a wearable motion tracking devices. Our solution as a universal stand-alone device adaptable to a given target apparatus differs in this aspect. In our endeavours, we attempt to create a signal processing method of reliable punch detection and design a solution which would offer decent detection performance with hardware fixed to the boxing bag without the need to endure the punch directly.

Chapter 3

Hardware

Before we could select any hardware, we needed to investigate its delimiting parameters based on the signal we wanted to collect. If we selected these parameters incorrectly, we would degrade our desired signal. Because our primary goal was to create a database of signals from which we would then extract parameters for our detectors, selection of optimal hardware with sufficient redundancy is crucial. The variables which were necessary to determine are accelerometer sensitivity range, sampling frequency and quantization.

Firstly let's inspect the sensitivity range. Studies concerned with maximal force of a punch diverge greatly in their conclusions. From [21] stating maximal force of $4096N$ we could deduce that on a $50kg$ boxing bag we would get a

$$a = \frac{F}{m} = \frac{4096}{50} \approx 82ms^{-2}$$

, which is approximate $8.2g$ of acceleration. The [22] states maximal punching force of almost $8000N$, that gives a maximal acceleration of around $16g$. The article [19] states 5.3g a and [20] even 3.2g. To get an assurance in the value, we decided to perform our own experiment, which should assure us of the correctness of used figures. From boxing gym SKPŠ Boxpraha we selected a testing boxer. He was 37 years old weighted 110 kg and was 189 cm high. Measuring apparatus was composed of Arduino UNO board with an attached SD card breakout board for storage. As a sensor, we used ADXL377, which is a high-g triple-axis analog accelerometer with a range of $\pm 200g$. Arduino UNO has a 10-bit A/D converter so any received value would be in the range from 0 to 1023. The device was secured with a non-elastic sticky tape on the boxing bag slightly above the eye height of the boxer. The z-axis of the sensor pointing in the direction of the punch. Boxer was instructed to exert as much force as he was able into a single punch directed into the centre of the bag. Punch was directed in the direction of the z-axis of the sensor. Collected data from the z-axis, from one of the experiments, are shown in figure 3.1.

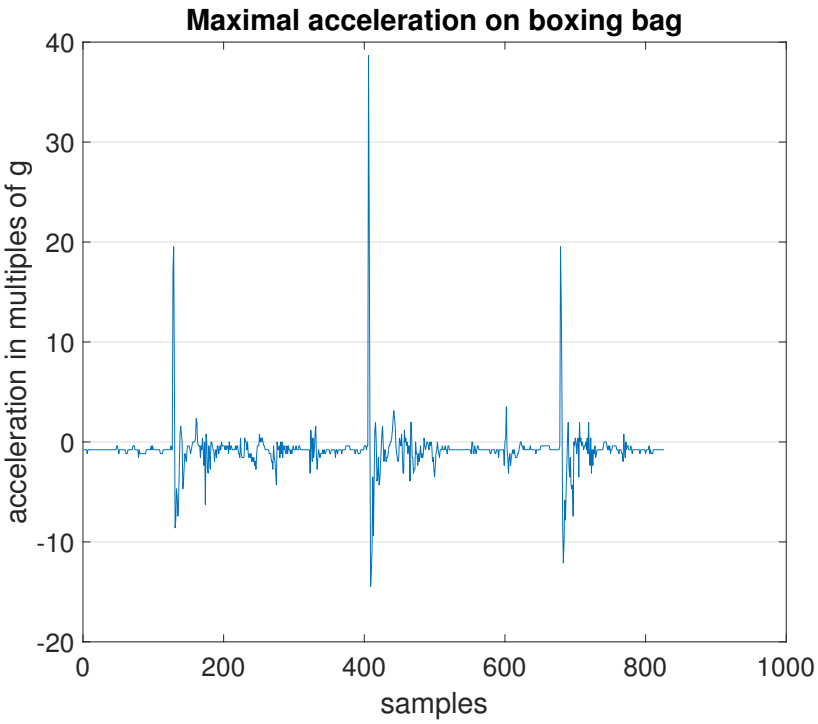


Figure 3.1: Signal of punches with maximal force (z-axis)

We can see that one of the punches reached to almost 40 g. It is necessary to mention that this value was measured on a particular boxing bag which was on particular suspension and changes in the weight, composition of padding or length of the suspension of the bag could influence this result. We repeated this experiment with 46 kg boxing bag with suspension length of ± 2 m. Boxer had 90 kg and was 178 cm tall. Figure 3.2 illustrates the results attained from the experiments.

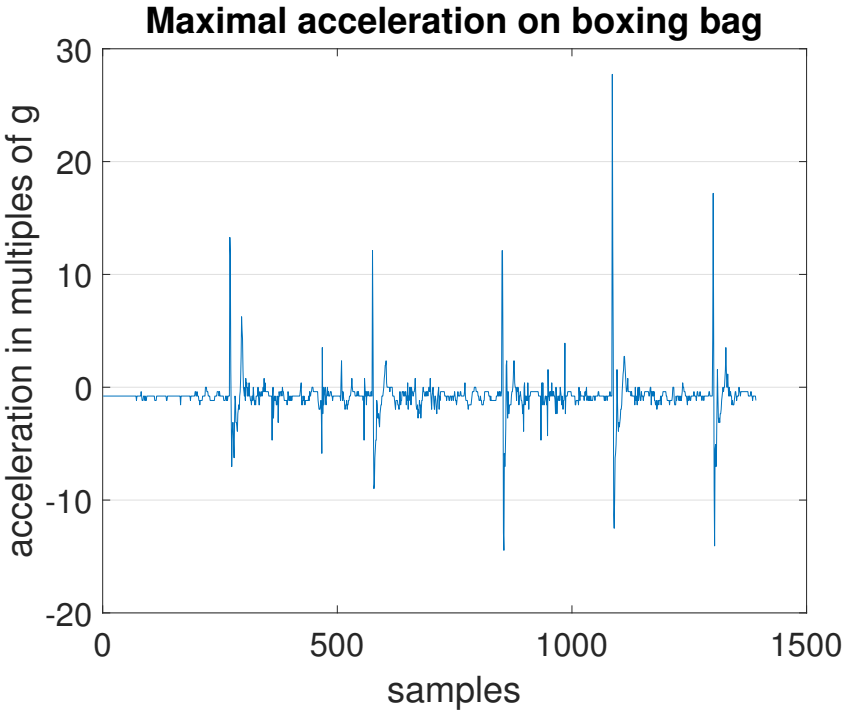


Figure 3.2: Signal of punches with maximal force

From experiments done, we could assume that the maximal value of 39g was a singular extreme case. Most of the acceleration caused by the punches was securely below 20g. From these findings, we concluded that the accelerometer range of ±16g would be sufficient. The possible saturation of the sensor with big punches was an acceptable sacrifice for finer quantization which would be beneficial for the detection of smaller punches which was expected to be much more difficult task than detection of the big ones. Next, we focused on the sampling frequency. To be sure that we had redundancy in the sampling frequency, we used very efficient code for Arduino UNO and SD cards called *AnalogBinLogger*. Bill Greiman created it for his library urlSdFat which provides read/write access to FAT16/FAT32 file systems on SD/SDHC flash cards much faster than regular libraries. Thanks to this code, we reached the sampling frequency of 10000 samples per second. As a sensor, we used again ADXL377 with its z-axis pointing into the direction of a punch. In this experiment, we executed multiple hard punches on a bag, which was continuously prevented from excessive swinging by an assistant. The resulting signal is shown in a figure 3.3.

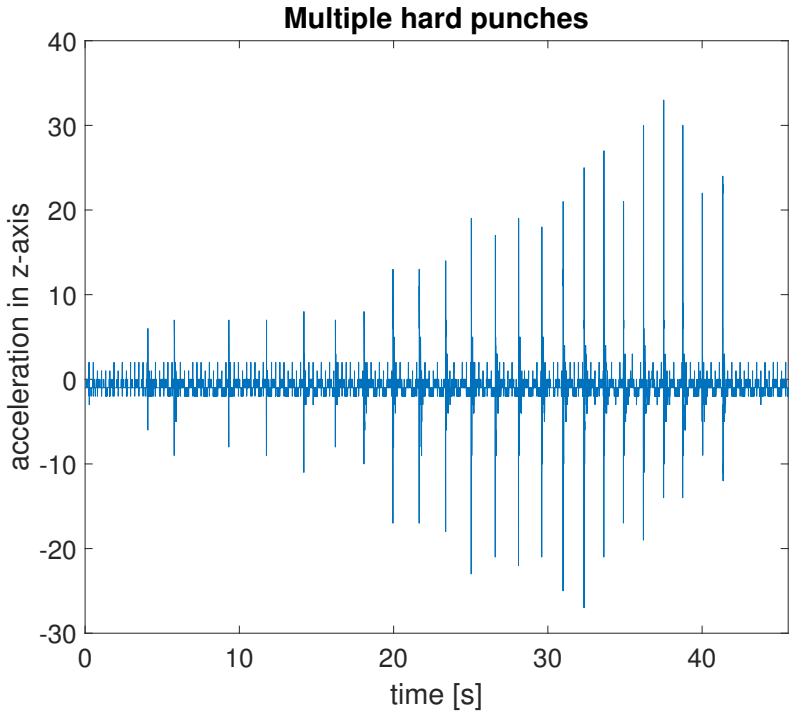
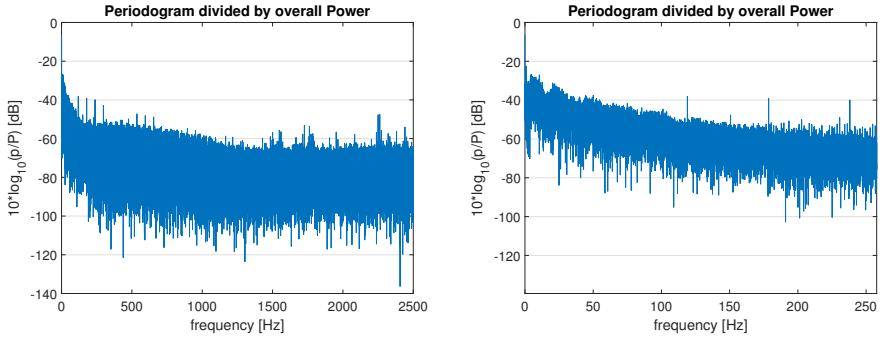


Figure 3.3: Centered signal from ADXL377 (z-axis)

The signal shown is in its raw state only centred on a zero, because ADXL377 has −200g as 0, 0g as 511 and +200g as 1023. You can see many punches (23 overall) and signal oscillation around zero caused mainly by noise and rough quantization of the A/D converter. We estimated the power spectrum of the signal by the periodogram normalized by a total power, so it is clear how big of a portion does each spectral line represent, see figure 3.4.



(a) Periodogram of the signal 3.3

(b) Detail of lower frequencies

Figure 3.4: Periodogram of the signal in figure 3.3 in (a). Detailed look on the lower frequency of same periodogram (b)

We could see that around 150Hz is a place where the spectrum amplitude drops below −50dB which means that all higher frequencies contain less than 10^{-5} of the total power and therefore can be safely discarded without

the loss of quality of our desired signal. To be sure of the correctness of our conclusions, we filtered out a signal with frequencies above 150 Hz and compared the resulting signal with original, see figure 3.5.

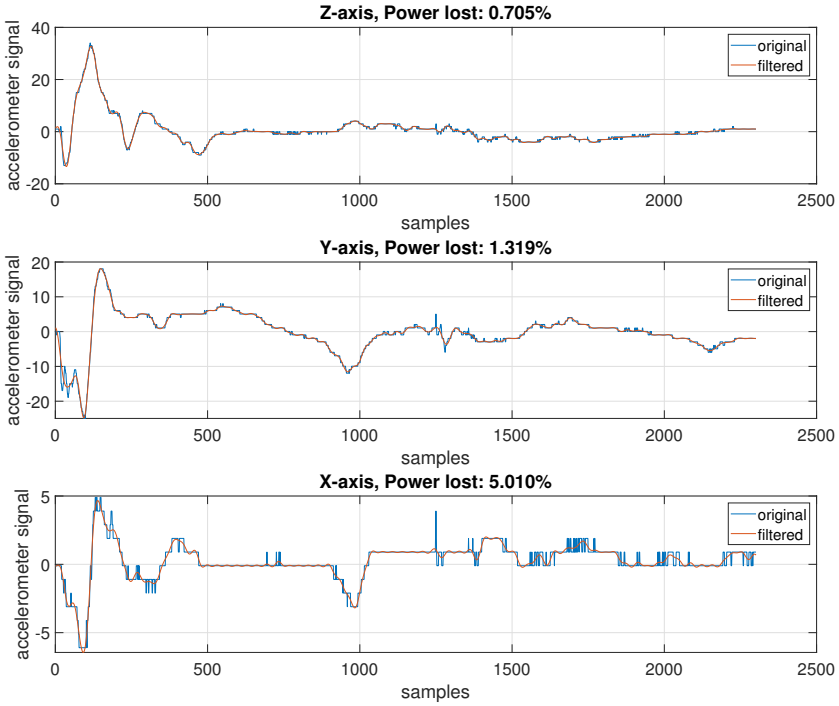


Figure 3.5: Comparison of original signals and signals with frequencies above 150Hz filtered out. Signals and power lost plotted for each axis.

As we could see no significant loss on the quality or power of the punch signal. Therefore our theoretical minimal sampling frequency had to be, according to the Nyquist–Shannon sampling theorem, twice of the highest frequency we wanted to preserve. Our sampling frequency had to be a least 300Hz. According to an industry-standard, we should take at least six times higher sampling rate. I this case that meant at least 1500Hz.
Such a sampling frequency narrowed down our choices of sensors. All considered sensor are shown in 3.1.

	max f_s [Hz]	precision [bits]	type	range [g]
ADXL377	10000	10	analog	± 200
ADXL326	10000	10	analog	± 16
ADXL345	3200	13	digital	$\pm 2, 4, 8, 16$
MPU9250	4000	16	digital	$\pm 2, 4, 8, 16$

Table 3.1: Considered accelerometer sensors for measuring apparatus.

All our experiments so far were realized with ADXL377. But for further

use, its vast range of ±200g was excessive and led to visible steps in the signal caused by inadequate quantization precision. Unfortunately even ADXL326 with its ±16g range had still too low-value resolution for sufficient capture of weaker punches. Finally, we decided to use MPU9250 3.8 rather than ADXL345 because of its higher resolution and sampling rate. Another advantage was variable range setting which could be adjusted if the ±16g would seem too broad. Other alternatives in consideration for microcontroller were Arduino Due with its 84MHz and 12-bit A/D converter. However, it is physically large and introduces different ARM architecture which could bring incompatibility with older scripts written for UNO. Options outside the Arduino family were Teensy 3.2 or 3.6. Their 72 MHz and 186 MHz clock speed respectively and 13-bit A/D converters were quite tempting. But because of ease of use, we decided to stick with the 16 MHz Arduino Uno.

3.1 Communication between Arduino UNO and MPU9250

PARAMETER	CONDITIONS	MIN	TYP	MAX	Units
SPI Operating Frequency, All Registers Read/Write	Low Speed Characterization		100 ±10%		kHz
	High Speed Characterization		1 ±10%		MHz
SPI Operating Frequency, Sensor and Interrupt Registers Read Only			20 ±10%		MHz
I²C Operating Frequency	All registers, Fast-mode			400	kHz
	All registers, Standard-mode			100	kHz

Figure 3.6: SPI and I2C protocol speed specifications from data sheet [24].

I2C protocol was too slow for us. To transfer 12000 16 bit values would with fastes I2C mode of 400kHz take around 0.48 seconds, which is unacceptable. So we used the SPI 4-wire protocol that uses two control lines and two data lines. The MPU-9250 always operates as a Slave device during standard Master-Slave SPI operation. With respect to the master, the Serial Clock output (SCLK), the Serial Data Output (SDO) and the Serial Data Input (SDI) are shared among the Slave devices. Each SPI slave device requires its own Chip Select (CS) line from the master. CS goes low (active) at the start of transmission and goes back high (inactive) at the end. Only one CS line is active at a time, ensuring that only one slave is selected at any given time. The CS lines of the non-selected slave devices are held high, causing their SDO lines to remain in a high-impedance (high-z) state so that they do not interfere with any active devices.

Important SPI Operational Features.

1. Data is delivered MSB first and LSB last
2. Data is latched on the rising edge of SCLK
3. Data should be transitioned on the falling edge of SCLK

4. The maximum frequency of SCLK is 1MHz.

5. SPI read and write operations are completed in 16 or more clock cycles (two or more bytes). The first byte contains the SPI Address, and the following byte(s) contain(s) the SPI data. The first bit of the first byte contains the Read/Write bit and indicates the Read (1) or Write (0) operation. The next 7 bits contain the Register Address. In cases of multiple-byte Read/Writes, data is two or more bytes [24], see figure 3.7.

6. Supports Single or Burst Read/Writes

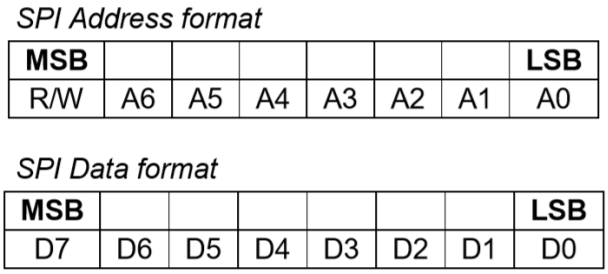


Figure 3.7: Illustration of SPI address and data formatting. Image taken from [24]

To set configure the MPU9250 to an SPI mode, we needed to adjust values in three configuration registers.

Address (Hex)	Name	Value
6B	PWR_MGMT_1	0x80
6A	USER_CTRL	0x10
37	INT_PIN_CFG	0x02

Table 3.2: Registers for SPI configuration with corresponding values.

We needed to reset the sensor control processor because to prevent switching into I2C mode when using SPI; the I2C interface should be disabled by setting the I2C_IF_DIS configuration bit. Setting this bit should be performed immediately after waiting for the time specified by the “Start-Up Time for Register Read/Write" which is typical $11 - 100$ms. Right after that, we need to set in the USER_CTRL register the I2C_IF_DIS bit to 1, which puts the serial interface into SPI mode only[25]. Finally, we needed to write into INT_PIN_CFG value 0x02, which set the BYPASS_EN bit to 1 thus fully disabled the I2C master interface.

Figure 3.8: Selected sensor MPU 9250 on a breakout board.

Now we can communicate with the MPU9250 via SPI interface. The actual wiring of the electronic parts is shown on 3.9.

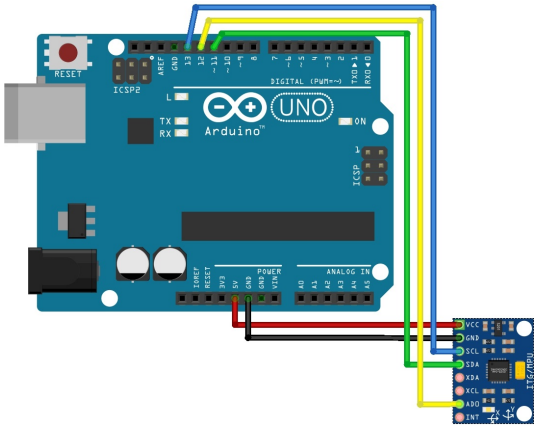


Figure 3.9: Wiring of Arduino UNO and MPU9250 for SPI communication.

To achieve 4000Hz sampling rate, we needed to set sensor into Low-Noise Accelerometer Mode, which disabled gyroscopes, magnetometers and digital motion processor (DMP). So we needed to go to Power Management 2 register PWR_MGMT_2 and disable the gyroscopes by writing value of 0x07. Then we went to Sample Rate Divider SMPLRT_DIV, which divides the internal sample rate to generate the sample rate that controls sensor data output rate, FIFO sample rate. We set its value to 0x00 and disabled it. The last step was in the Accelerometer Configuration 2 register ACCEL_CONFIG_2 in this register we set the value of bit accel_fchoice_b, which is used for bypassing the digital low pass filter DLPF. To disable it, we set it to 1.

Address (Hex)	Name	Value
6C	PWR_MGMT_2	0x07
19	SMPLRT_DIV	0x00
1D	ACCEL_CONFIG_2	0x08

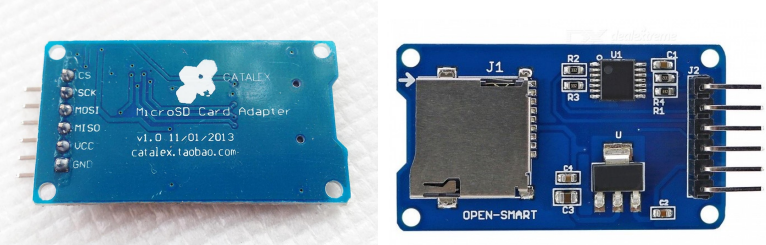
Table 3.3: Registers for setting the sampling rate of MPU9250 to 4000Hz.

With 4000Hz sampling rate, we had 12000 values per second that we had to store. 12000 values that was $12000 \cdot 16 = 192000$bits that was 192 kbit/s.

Arduino UNO has 32 kilobytes of Flash memory, 2 kilobytes of SRAM memory and 1kilobyte EEPROM, so we needed to look for external memory. Our signal had to be at least five seconds long so the capacity should be around $192 \cdot 5 = 960$ kilobits. But it had to be also fast enough. We needed to store each byte of data in at least

$$T_{write} = \frac{1}{4000 \cdot 3 \cdot 2} = 41\mu s$$

EEPROM memory has write speeds in orders of milliseconds, so we looked into SD-cards and SRAM memory. SD cards offer huge capacities very cheaply. It looked like the best way to store data. In our attempts to make the sd-card storage work, we came across several complications. One of the biggest ones, which also took the longest time to uncover had to do with the SD card module which is made for Arduino boards. It is a breakout board with a slot for an SD-card 5V to 3.3V tri-state buffer chip and voltage regulator. Arduino UNO is 5V logic-based board, but SD cards work on 3.3V logic; thus, we need to use a voltage regulator to lower to voltage for the SD-card. The tri-state buffer does the same for the logic values of the inputs. Also, it can switch from output enabled state to high impedance state, which effectively disconnects the data line from the bus. Unfortunately, all of the output enable pins of the SN74LVC125A tri-state buffer are permanently tied to the ground which makes them permanently active at all times. That is not a problem for CS, MOSI and SCK pins because they are inputs, but MISO pin is output to the master. When the CS pin of a slave device is pulled high, it should disconnect from the bus so other slave devices could communicate. In this case, the MISO line is always active and thus disturbs the communications of all other devices on the bus. Used model is shown on 3.10.



(a) Back of the SD card module (b) Front of the SD card module

Figure 3.10: Used SD module

Unfortunately, diagnostics of this problem took longer than expected, and complicated implementation of fast and effective write procedures that would work with digital accelerometer such as MPU9250 was after failed attempts abandoned. An alternative way was suggested, using SRAM memory. It is very fast and some models, compatible with SPI protocol, can be easily connected to Arduino. We selected model 23LC1024, which is one Mbit serial SRAM with SDI and SQI Interface. One Mbit SRAM stores, with data rate 192kbit up to 5.2 seconds of a signal. We decided to use four SRAMs, which

gave us more than 20 seconds of a signal. Connecting four SRAM modules and accelerometer sensor to one SPI bus on the Arduino just with jumper cables becomes quite messy so with just two pairs of SPI pins on Arduino quite impossible without soldering. SPI bus on Arduino UNO rev3 has two sets of connecting pins. One is made from digital pins eleven to thirteen on the right side of the board, and the second one is on the ICSP header. Both sets are shown on 3.11.

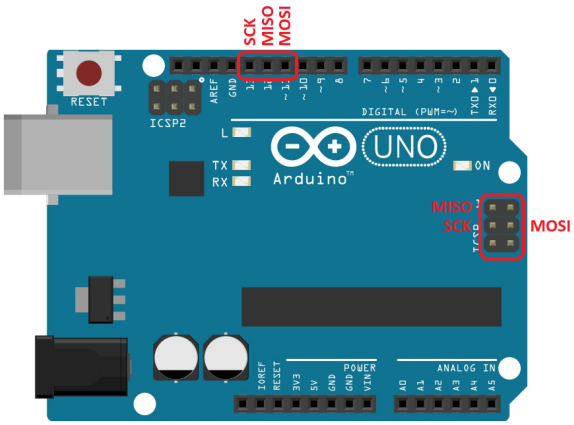
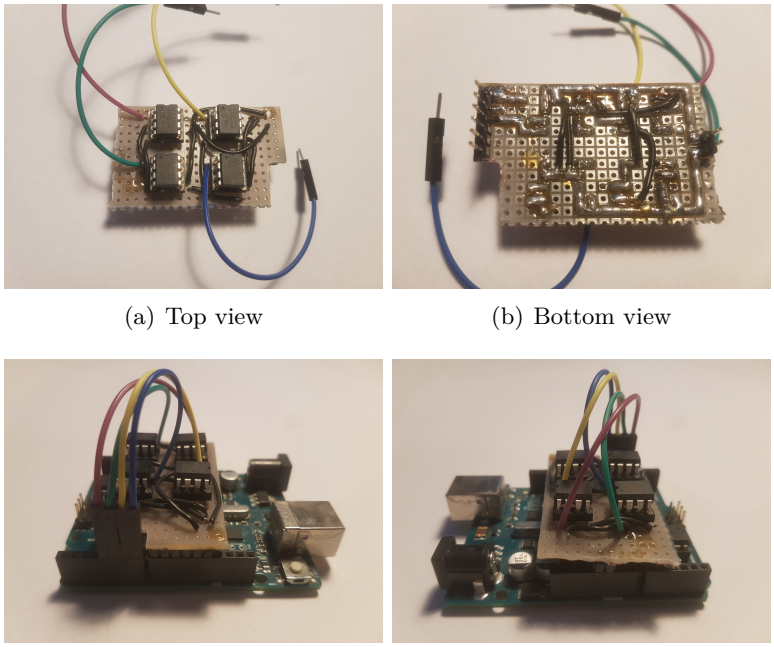


Figure 3.11: SPI pin locations on Arduino UNO

We decided that best way was to construct an SRAM shield which would take up the pins eleven to thirteen and accelerometer would be connected the ICSP header. On a solder finished prototype PCB we created a circuit connecting the SRAMs to the SPI pins and their CS pins were channelled through wires to any remaining digital pins. The finished shield is shown on 3.12.



(a) Top view (b) Bottom view

(c) Right view on Arduino (d) Left view on Arduino

Figure 3.12: SRAM shield

Each 23LC1024 contains an 8-bit instruction register[26]. All SRAMs are accessed via the SI pin (3.13), with data being clocked on a rising edge of SCK, which is mode 0.

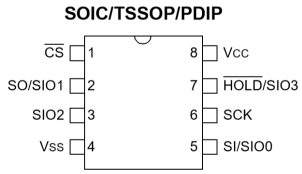


Figure 3.13: SRAM package and pin descriptions. Image taken from [26].

The $\overline{CS}$ pin must be low for the entire operation. They also support multiple modes of operation. Byte operation - write/read just one byte of data in/out the SRAM. Page operation - 23LC1024 has 4096 pages of 32 bytes in this mode the read and write operations are limited to within the addressed page. The address is incremented internally, so just initial address is necessary. Sequential operation - this mode allows to access the entire array. Address counter is automatically incremented and pages boundaries are ignored. When the address counter reaches the end of the array, the address counter will roll over to 0x00000 [26]. We will use the Sequential operation and when we reach the last address of one SRAM, the program will switch to another chip and will continue to write until all SRAMs are full. Mode of operation is selected by setting a specific value to mode register. In our case, value for sequence mode is 0x40.

3.2 Finishing the measuring apparatus

For a container, we picked an ABS plastic enclosure with dimensions 150mm wide, 100mm deep and 60mm high with a screw-on lid. Can be seen on 3.14.

(a) Interior of the box

(b) Whole box with lid

Figure 3.14: Enclosure for the measuring device.

The device is meant to be fastened to a boxing bag. So we drilled multiple holes for fastening screws to keep the Arduino from damage. Also, foam pads were hot glued to the enclosure at the location of Arduino to prevent possible excessive shocks to the board. That could be caused by collisions with the bottom of the enclosure under the Arduino. That could also interfere with the accelerometer measurements. Several holes were cut to the sides of the box throughout the prototyping process mainly for SD card reader and USB-cable connecting the Arduino to the computer. Arrangement of the parts was different for some prototype versions to that is why there are some unused openings. Accelerometer MPU9250 was fixed to the bottom of the box by hot-glue. In the image of 3.15, there is second analog accelerometer ADXL326 hot-glued beside the MPU9250. It was used in certain experiments where a very high refresh rate was necessary.

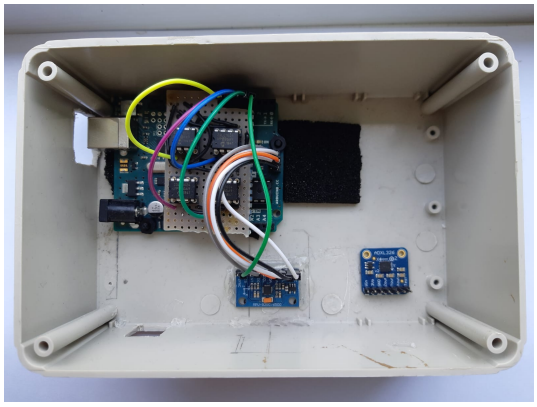


Figure 3.15: View of the measuring apparatus. It consists of Arduino UNO board, SRAM memory shield and MPU9250 accelerometer.

Arduino UNO is screwed down to the bottom of the enclosure with protective foam padding underneath it (the visible black material on the right side of Arduino board). SRAM shield is connected to the power and SPI pins on the Arduino board. The accelerometer is hot-glued to the bottom of the enclosure and connected to the SPI bus of the Arduino via its ICSP header. Sensor hot-glued to the right side of the MPU9250 is analog accelerometer ADXL326, and it was used for experiments where high f_s was necessary, but for database, creation was not used.

3.3 Software

In this section, we would like to outline the main functions and operation principle of our measuring apparatus. It is in no way a thorough description of the matter because, throughout our experiments, the design of the hardware and mainly the code controlling it went through a significant number of adjustments and improvements. Describing all of them here would be impractical. Nevertheless, we attempted to go through the most critical stages of the designing process and mention some of the difficulties we met on the way.

The main goal of the device was to collect and store data. We would focus on the data collection first. For any signal processing, it is crucial to have consistent sampling frequency. Alas, that is not that simple to accomplish in practice. The go-to approach to ensure accurate data collection is the use of interrupts. Usually, a sensor is creating data samples with some given sampling frequency of f_s. The program would have to check the sensor constantly to catch the samples when they arrive to ensure that our program always catches the samples and never misses any. The MPU9250 has its interrupt pin, which can be configured to indicate the arrival of new sensor data.

So, if we decided to make use of this functionality, every time a new data comes in the Arduino main program has to be stopped and interrupt routine executed. In such a routine, an SPI sensor data reads, and SRAM writes could be made. However, several facts had to be taken into account. This routine would have to run 4000 times a second because that is our sampling frequency.

Additionally, Arduino implementation of SPI communication by which Arduino communicates with sensor and SRAM memories is also based on interrupts. That would mean that every second new data interrupt would be called 4000 times and SPI communication 8000 because new data have to be read and saved. That incorporates many interrupt routine overheads. Also, each new SPI communication with sensor contains one byte of administrative information about that data transfer, such as slave address and read/write bits. SPI communication with SRAM memories includes 4 byte instruction overhead. That accounts for an additional 40000 non-data bytes transferred each second.

Having in mind the sole purpose of data collection of our device, we

decided to try a different approach and take advantage of the sequential write capability of our SRAM memory and on-chip FIFO buffer in the sensor. There is a 512 byte FIFO buffer in the MPU9250, and newly arrived data can be stored in the buffer until a sufficient number of bytes are stored. FIFO buffer can then be read all at once, thus lowering the number of SPI transfers. Moreover, SRAM sequential write allows us to send those 4 address and operation bytes at the beginning of the write just once and then after the initial data byte is shifted in. Additional bytes can be clocked into the device. The internal address pointer is automatically incremented. When the address pointer reaches the highest address (1FFFFh), the address counter rolls over to (00000h). So the final decision was just to read the values from the FIFO after a certain number of samples were collected. So we theoretically calculated time duration of each operation. Firstly we needed to read all the collected bytes from the FIFO. It does not matter if it takes longer than 250μs because the sensor can write new data into the buffer while we read it. We set the number of accumulated bytes in the buffer to 510, which corresponded to 85 triplets. It was the highest multiple of 6 below 512 because new sensor data comes in groups of 6 bytes. Waiting for all 512 bytes would force to the sensor to write just the remaining 2 bytes of FIFO. The remaining 4 it would have to either discard because there is no place for them or rewrite the first 4 bytes at the beginning of the FIFO buffer depending on its configuration. Selecting 510 bytes could also act as a safety measure preventing buffer overflow. FIFO read would be necessary every 21.25ms ($T_{fs} \cdot N = 2.5 \cdot 10^{-4} \cdot 85$). Our FIFO read of 510 bytes would take

$$\begin{aligned} \frac{1}{f_{spi}} \cdot N_{buffer} \cdot 8 &= \tfrac{1}{8 \cdot 10^6} \cdot 510 \cdot 8 \\ &= 5.1 \cdot 10^{-4}\text{s} \end{aligned}$$

so around 0.51ms and their storage would take similar time. Finally, to see how fast can we notice that the buffer state changed, we did an experimented. We measured the time between buffer states. It turned out that maximal value was 91μs. So, while calculating how long would to read and write takes, we need to add this number before the read begins. That represents the worst-case scenario. It turned out that we needed $91 + 2 \cdot 510 = 1111\mu$s (1.111ms) to detect read and store 510 bytes of data every 21250μs (21.25ms).

We checked for discontinuities or abnormal values in the stored data, which would be evidence of buffer overflow of other issues. You can verify that on 3.16.

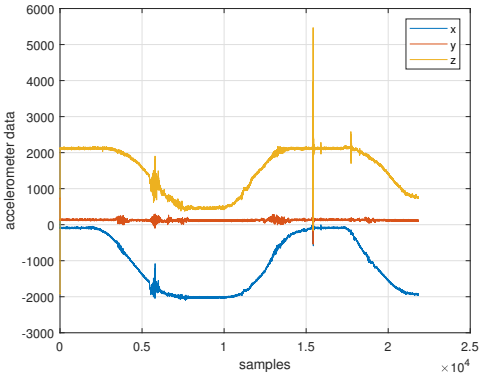


Figure 3.16: Signal from all axes of MPU9250. It is the signal of slow rotation of the box around one edge of the box (y-axis).

Note that any artefacts would occur every 85 sample. It is in no way definite proof that there are indeed no discontinuities, but no were found even from closer inspection and later experiments.

Throughout the process, we would have been storing the buffer state values, which are plotted in 3.17.

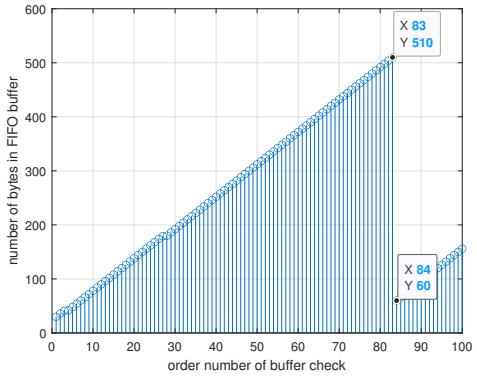


Figure 3.17: Sequence of buffer states observed over time.

From 3.17 we could see that right after the reading and storing procedure, there were already 60 bytes in the buffer. When we timed the process, from the point of detecting 510 bytes in FIFO to the end of data storage and got values around 2536μs. From that it can be verified with calculations

$$\begin{aligned} \frac{t_{measured}}{T_{fs}} \cdot n_{bytes} \cdot n_{triplet} &= \frac{2536}{250} \cdot 2 \cdot 3 \\ &= 60.864 \end{aligned}$$

where $t_{measured}$ is our measured duration, T_{fs} time period of our sampling frequency, n_{bytes} number of bytes in one value and $n_{triplet}$ number of values in triplet that even this number holds true. This had brought us to our final version of the code the was then used to collect data for the database in later

chapters. Some additional clean-up adjustments were made, like formatting for the data into CSV format, for easier processing later in MATLAB. Still, the main mode of operation described here was maintained. In the case of our device used solely for data collection, checking FIFO state and sequential reading approach was a viable option. In future implementations, where additional functionality from the processor would be required, an interrupt routine which allows for intermittent functioning of a processor between data collection would be a better choice. This approach becomes immensely superior with much faster processors.

3.4 Mode of operation

A plastic enclosure containing our device is tightly fastened to the boxing bag with non-elastic adhesive tape in the selected position. Then through the opening cut in the enclosure on the left side of the Arduino board, a USB cable is connected to the Arduino. The cable can be adjusted and fixed with more adhesive tape to prevent its excessive movement. A second end of the cable is connected to a computer on which a serial monitor is opened. On the start of the program in the Arduino, the board runs quick diagnostics of the connected peripherals and initialises them. Also, SRAMs are erased. Immediately after this initialisation takes place, the device starts measuring and saving data from the accelerometer. After all currently used SRAMs are full, a measuring stage is terminated. Data transfer from the SRAMs is started automatically after the termination of the measuring stage. Data is transferred via the cable to the serial monitor on the connected computer until all data are transported. Received data are manually copied a stored and processed in the computer by a researcher.

Chapter 4

Punch signal analysis

This study investigates the signal caused by boxing punch mainly. Boxing punch is from the physical point of view very similar to punches from different martial arts such as kickbox or karate. The main objective with this type of strike is to deliver as much force as fast as possible to the target, by rapidly extending the arm and with the hand closed to a fist hit the target.

4.1 Direct punch in boxing

Straight punch in boxing is one of the most effective punches of all punches in martial arts. It is simple yet, not an easy technique to master. It is one of the essential skills for a boxer. It reaches across a greater range than hooks but can deliver devastating force. Left 'jab' is one of the most frequently used punches in boxing so we will focus primarily on it. The proper boxing technique of the left 'jab' is as follows. It begins by rotation of the left hip forward and sliding of the left leg. The left arm begins to extend forward and is lead via the shortest path to the target. The heel of the right leg lifts slightly as weight of the boxer shifts more forward and pushes the body forward. Elbow of the right arm is resting on the stomach, thus covering the liver, ribs and solar plexus. Hand of the right arm is clenched in a fist and contacts the cheekbone from the side, thus guarding the head and a chin. The punch can be performed on the spot or simultaneously with a forward step.[23] In the following chapters, we work mainly with signals created by this type of punch. We are interested in how the placement of the jab influences the shape of the acceleration impulse signal. So we proceeded with an experiment. Used boxing bag is shown in 4.1. The grey marks show the target positions for the top and bottom punches. Middle punches were led on the white stripes in the middle of the bag. The bag had the following parameters.

height	150 cm
weight	46 kg
radius	17 cm
height of a suspension	150 cm

Table 4.1: Parameters of the used boxing bag. (Suspension height is measured from the fixed point to the top of the bag.)

Figure 4.1: Boxing bag used in experiments. Grey marks show the target positions of top and bottom punches. Middle punches were led on the white stripes in the middle of the bag.

Measuring device was fastened tightly to the bag with non-elastic adhesive tape. Accelerometer was approximately 30 cm down from the top edge of the boxing bag. The orientation of the accelerometer mounted in the device is shown in 4.2

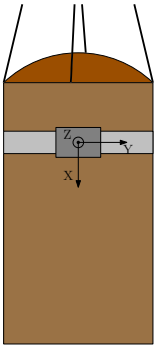


Figure 4.2: Illustration of accelerometer axes alignment on the boxing bag.

The z-axis of a sensor was pointed toward the boxer so sudden movement of the bag in direction of a punch would result in the negative acceleration spike

in the z-axis. MPU9250 accelerometer was used with sampling frequency of 4000Hz and sensitivity range of 16g.

4.2 Top of the bag

We performed one punch targeted on the mark for the top punch which was approximately 7cm under the accelerometer. Detailed view of the punch signal is shown in 4.3.

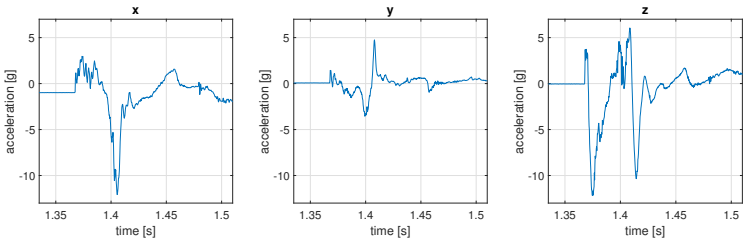


Figure 4.3: Signals of the top punch from x axis (left), y axis (centre), z axis (right).

From the impulse in the z-axis, we can see that before the bag started accelerating backwards, which is represented by the biggest negative spike, it moved a little bit forward. This experiment was repeated twice and also captured in slow motion by a mobile phone camera. From the recording, we assumed that the cause of the small move forward, present in both cases, was a deformation of the boxing bag around the point of impact. The deformation is, in part, dependent on the parameters of the bag. So, the impulse begins with a positive spike caused by the deformation of the bag, followed by the main negative spike caused by the initial rapid movement backwards in the direction of the punch. All the other secondary spikes are influenced heavily by the parameters of the punch and the bag which deformed, moved and bounced on its suspension. From the video analysis, we estimated that glove was contacting the bag for just 22 frames in both experiments. That translates with a frame rate of 240 fps to 0.1 of a second. That was, with a sampling frequency of 4000Hz, around 400 samples. For illustration we can look again on figure 4.3, where there is 0.1s visually noticeable between 1.35s and 1.45s.

4.3 Middle of the bag

For the middle punch we repeated twice the same experiment as mentioned above. This time, the landing position of the punch were the white stripes in the middle of the bag visible on figure 4.1. View of the signals from all axes are shown in 4.4.

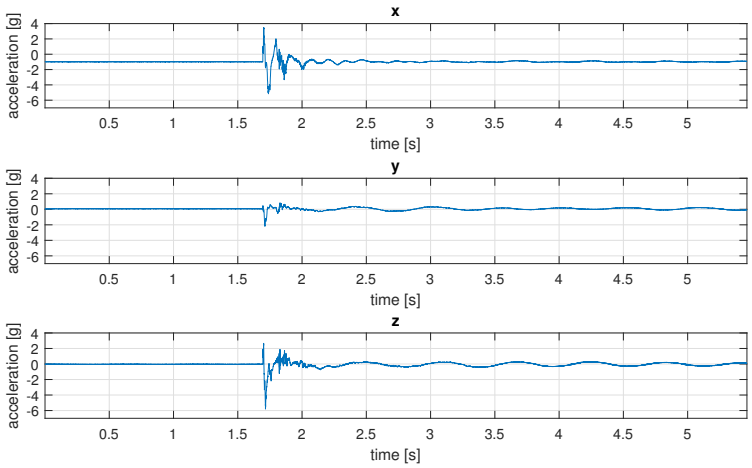


Figure 4.4: Signals of the middle punch from all axes.

More detailed view of the punch impulses are in the 4.5.

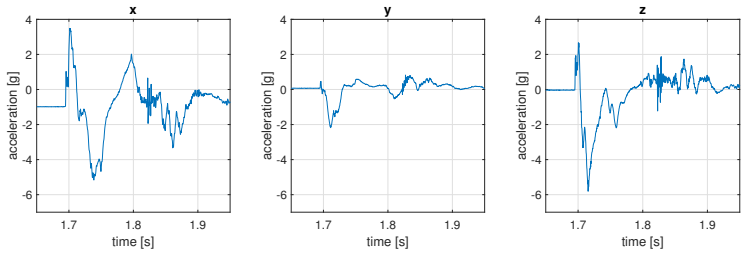


Figure 4.5: Detailed view on impulses from 4.4. X-axis (left), y-axis (centre), z-axis (right).

We noticed that with the middle punches the amplitude of the impulses were smaller than in the previous example with the top punch. Also the artefacts from the bag deformation and erratic movements were much less pronounced.

4.4 Bottom of the bag

For the bottom punch we repeated the same experiment.

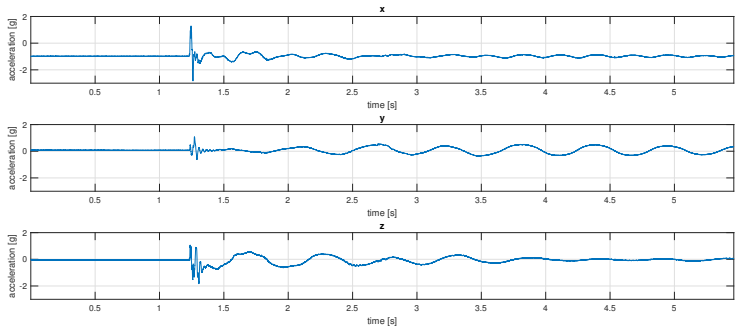


Figure 4.6: Signals of the bottom punch from all axes.

More detailed view of the punch impulses are in the 4.7.

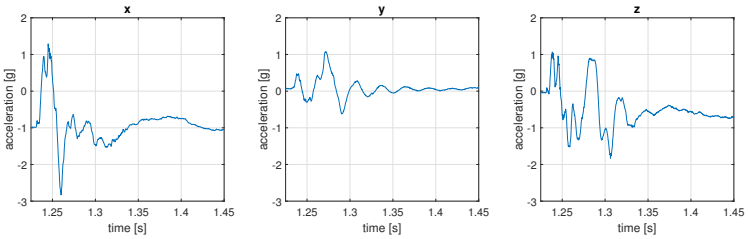


Figure 4.7: Detailed view on impulses from 4.7. X axis (left), y axis (centre), z axis (right).

The initial positive spike was again bigger and amplitude of the impulse smaller. Also following swings of the bag are much more pronounced with bottom punches. Side-by-side comparison of the z-axis punch signals are shown in 4.8.

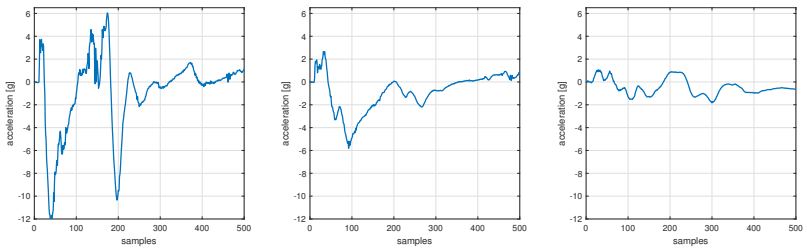


Figure 4.8: Side-by-side comparison of z-axis signals of a top (left), middle (centre) and bottom (right) punches.

I looks like for each punch placement, there are different punch models necessary.

Chapter 5

Theoretical analysis

5.1 What is a detector

Detectors are devices whose main task is to decide when an event of interest occurs and then to determine more information about that event. These devices include radars, sonars, image processing and others. We will attempt to illustrate the primary tasks of detectors on the radar system. The main purpose of radar is determining the presence or absence of an approaching aircraft. To accomplish this task, it sends out an electromagnetic pulse, which, if reflected by a large moving object, will indicate the presence of an aircraft. If an aircraft is present, the received waveform will consist of a reflected pulse and noise due to ambient radiation and the receiver.[1] The received pulse will, however, not precisely match the shape of the transmitted signal. It could be attenuated due to the propagation loss and possibly distorted due to the interaction of multiple reflections. If no aircraft is present only noise will be present. It is the task of a detector to decide whether the echo is or is not present. The second task after successful detection is information extraction, such as speed, direction, size and others. In most of these systems, we are faced with the problem of making a decision based on a continuous-time signal. Still, modern signal processing is done by digital computers which store discrete samples of those signals to process them. So our task is to reach a decision based on an N-point data set or some function of it.[1] The most straightforward detection problem is to decide if a signal is present. Mentioned radar is a great example. Since we want to choose between two possible hypotheses, signal and noise present versus noise only present, we term this the binary hypothesis testing problem. Following chapters produced on the basis of the book [1].

5.2 Neyman-Pearson Theorem

We will address the simple hypothesis testing problem in which the probability density function (PDF) for each assumed hypothesis is completely known. Let us assume that we observe a realisation of a random variable whose PDF is either $\mathcal{N}(0,1)$ or $\mathcal{N}(1,1)$. The notation $\mathcal{N}(\mu,\sigma^2)$ denotes a Gaussian

PDF with mean μ, and variance σ^2. We must, therefore, determine if $\mu = 0$ or $\mu = 1$ based on a single observation $x[0]$. Each possible value of μ can be thought of as a hypothesis so that our problem is to choose among two competing hypotheses. We can modify this problem as a signal detection problem by modifying the interpretation of the second PDF as a DC signal of aplitude $A = 1$. These hypotheses can be summarized as follows[1]

$$\begin{aligned} \mathcal{H}_0 : &\quad x[0] = w[0] \\ \mathcal{H}_1 : &\quad x[0] = A + w[0] \end{aligned}$$

The $\mathcal{H}_0$ is reffered to as the *null hypothesis* and $\mathcal{H}_1$ as an *alternative hypothesis*. Since we need to choose between two hypotheses this problem is known as *binary hypothesis test*. One intuitive approach to solving this problem can be taking the value of incoming sample and compare its probability for each PDF and selecting the one PDF giving the higher value. A threshold for this decision would in our case be $1/2$. Unfortunately is impossible to determine which PDF generated certain sample every time. If, for example sample generated from $\mathcal{H}_0$ had $w[0] > 1/2$ of alternatively sample from $\mathcal{H}_1$ had $w[0] < -1/2$ our detector would make a mistake. Note that with this scheme we can make two types of errors. If we decide $\mathcal{H}_1$ but $\mathcal{H}_0$ is true, we make a *Type I error* otherwise known as *False alarm*. If we decide $\mathcal{H}_0$, but $\mathcal{H}_1$ is true we make a *Type II error*. These two errors are unavoidable, but may be traded off against each other. To do so we only need to change the threshold. The notation $P(\mathcal{H}_i; \mathcal{H}_j)$ is the probability of deciding $\mathcal{H}_i$ when $\mathcal{H}_j$ is true. Illustrations of both types of error shown in 5.1.

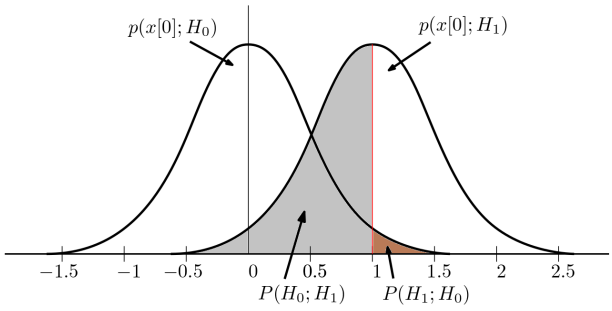


Figure 5.1: Types of errors with threshold on 1

It is impossible to reduce both error probabilities simultaneously. A typical approach then in designing an optimal detector is to hold one error probability fixed while minimizing the other. Usually we constrain the $P(\mathcal{H}_1; \mathcal{H}_0)$ on a very small value, because of potentially disastrous effects of false alarm (initiating attack after false alarm of enemy aircraft). So after we fix the probability of a false alarm P_{FA} we seek to minimize the second error $P(\mathcal{H}_0; \mathcal{H}_1)$ or equivalently to maximize $1 - P(\mathcal{H}_0; \mathcal{H}_1) = P(\mathcal{H}_1; \mathcal{H}_1) = P_D$. Which is the *probability of detection*. This setup is termed the *Neyman-Pearson* (NP) approach to hypothesis testing or to signal detection. In summary we wish to maximize $P_D = P(\mathcal{H}_1; \mathcal{H}_1)$ while having constrained $P_{FA} = P(\mathcal{H}_1; \mathcal{H}_0) = \alpha$.

We can constrain P_{FA} by setting the threshold γ since [1]

$$\begin{aligned} P_{FA} &= P(\mathcal{H}_1; \mathcal{H}_0) \\ &= \Pr\{x[0] > \gamma; \mathcal{H}_0\} \\ &= \int_{\gamma}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}t^2\right) dt \\ &= Q(\gamma) = \alpha \end{aligned}$$

The $Q(\gamma)$ function is the tail distribution function of the standard normal distribution.(citace z wiki která má u sebe zdroje The Q-function, from cnx.org, Basic properties of the Q-function Archived March 25, 2009, at the Wayback Machine) In other words, $Q(\gamma)$ is the probability that a normal (Gaussian) random variable will obtain a value larger than γ standard deviations. From set P_{FA} for example $P_{FA} = 10^{-3}$ we can compute P_D [1]

$$\begin{aligned} P_D &= P(\mathcal{H}_1; \mathcal{H}_1) \\ &= Prx[0] > \gamma; \mathcal{H}_1 \\ &= \int_{\gamma}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}(t-1)^2\right] dt \\ &= Q(\gamma - 1) = 0.023 \end{aligned}$$

Question arises, if this is the maximum value of P_D we can attain? Let us look on the general operation of an detector. It decides either $\mathcal{H}_0$ or $\mathcal{H}_1$ based on a certain dataset. For example a part of an incoming radar signal. That means detection is a mapping from each possible dataset into a decision.[1]

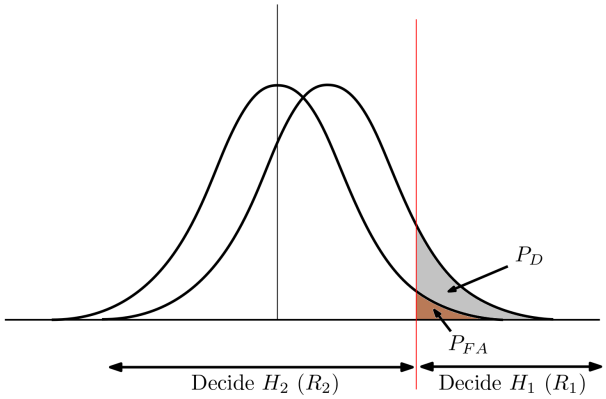


Figure 5.2: Decision regions and probabilities for DC level example.

Let R^N be a data space containing all possible datasets of the length N. Let us divide this dataset into a two subsets R_1 and R_0, such that R_1 be the set of values in R^N that map into the decision $\mathcal{H}_1$ and R_0 set that map into the decision $\mathcal{H}_0$. For previous DC example the datasets correspond with the value intervals shown in 5.2. R_1 is termed the *critical region* in statistics. The R_0 is complement to the R_1, because $R_1 \cup R_0 = R^N$. The P_{FA} constraint can be then rewritten as [1]

$$P_{FA} = \int_{R_1} p(\mathbf{x}; \mathcal{H}_0) d\mathbf{x} = \alpha \tag{5.1}$$

α is termed *significance level* or *size* of the test. There are many sets R_1 that satisfy equation (5.1). Our goal is then to find such R_1 that maximizes P_D [1].

$$P_D = \int_{R_1} p(\mathbf{x}; \mathcal{H}_1) d\mathbf{x} \tag{5.2}$$

In statisticsd, P_D is called a *power* of the test and *critical region* that attains the maximum *power* is the *best critical region*. The NP theorem tells us how to choose R_1 if we know the $p(\mathbf{x}; \mathcal{H}_0)$ and $p(\mathbf{x}; \mathcal{H}_1)$. To maximize P_D for a given $P_{FA} = \alpha$ decide $\mathcal{H}_1$ if [1]

$$L(x) = \frac{p(\mathbf{x}; \mathcal{H}_1)}{p(\mathbf{x}; \mathcal{H}_0)} > \gamma \tag{5.3}$$

where the threshold γ is set from [1]

$$P_{FA} = \int_{\{\mathbf{x}: L(x) > \gamma\}} p(\mathbf{x}; \mathcal{H}_0) d\mathbf{x} = \alpha \tag{5.4}$$

The equation (5.3) is called *likelihood ratio*. It compares the likelihood the vector $\mathbf{x}$ is from $\mathcal{H}_1$ with the likelihood of $\mathbf{x}$ being from $\mathcal{H}_0$.

5.3 Task assessment

To design a detector with good performance, it is necessary to create an appropriate model of our signal and formulate our detection task accurately. The too complicated model could be challenging to implement effectively and on the other hand inaccuracies in a model that is too simple can impede detectors performance. Our main goal is to detect the presence of a signal impulse indicating that a punch has been thrown on the boxing bag. We denoted our punch signal impulse $s[n]$ representing development of acceleration a during the punch. We took $s[n]$ as a deterministic signal with known character. Unfortunately, we are expecting boxers to throw punches with various force, so the amplitude of s is unknown. We also do not know the specific time of arrival for any given punch, but that is covered in later chapters. Based on our analysis, we could characterize our task as detection of deterministic signals with unknown parameters. Unknown parameters are in our case, the amplitude A of $s[n]$.

5.4 Deterministic signal with unknown parameters

The task of detection of signals which are not entirely known are very important. Arriving signals can be unpredictably deformed due to the uncertain propagation effects, for example. The general model we shall employ in our case is, as we mentioned earlier, a completely known signal s except for a few unknown parameters. We can choose between two main approaches in designing a suitable detector for this composite hypothesis testing task. We can employ generalised likelihood ratio test (GLRT) where we model the

unknown parameters as deterministic and substituted by their maximum likelihood estimates or the Bayesian approach which model the parameters as realisations of a random variable with prior known PDF. In the classical case of unknown deterministic signal parameters, an optimal detector, i.e., a uniformly most powerful (UMP) test (one that produces the highest P_D for all values of the unknown parameters and for a given P_{FA}) will usually not exist. The GLRT, a suboptimal detector, will however, usually produce good detection performance. The reason for GLRT is twofold. First, it is not easy to gain knowledge about the prior distribution of the amplitude from our database, and second, it is easier to implement maximal likelihood estimate (MLE) rather than integration. Let us now consider the problem of detecting a deterministic signal s known except for amplitude in WGN. Specifically

$$\begin{aligned} \mathcal{H}_0 : x[n] &= w[n] & n = 0, 1, ..., N-1 \\ \mathcal{H}_1 : x[n] &= As[n] + w[n] & n = 0, 1, ..., N-1 \end{aligned}$$

where $s[n]$ is known, amplitude A is unknown and $w[n]$ is WGN with variance σ^2. To determine if UMP test exists, for our task, we will provisionally assume A to be known and construct the NP test. If the test statistic and its threshold can be found without knowledge of A, then the test is UMP. The likelihood ratio test decides $\mathcal{H}_1$ if

$$L(x) = \frac{p(\mathbf{x}; \mathcal{H}_1)}{p(\mathbf{x}; \mathcal{H}_0)} \tag{5.5}$$

because of the WGN we can rewrite the equation (5.5) as [1]

$$\frac{\frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp\left[-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(x[n] - As[n])^2\right]}{\frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp\left[-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(x[n])^2\right]} > \gamma \tag{5.6}$$

. Taking logarithm gives

$$-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(-2As[n]x[n] + A^2s^2[n]) > \ln\gamma \tag{5.7}$$

, and

$$A\sum_{n=0}^{N-1} x[n]s[n] > \sigma^2 \ln\gamma + \frac{A^2}{2}\sum_{n=0}^{N-1} s^2[n] = \gamma'. \tag{5.8}$$

Now if $A > 0$ then the NP test decides $\mathcal{H}_1$ if

$$\sum_{n=0}^{N-1} x[n]s[n] > \frac{\gamma'}{A} = \gamma'' \tag{5.9}$$

while if $A < 0$ then it decides $\mathcal{H}_1$ if

$$\sum_{n=0}^{N-1} x[n]s[n] < \frac{\gamma'}{A} = \gamma'' \tag{5.10}$$

Now, we must determine if we know the sign of our amplitude A, because it could potentially cost us utilizing of a UMP test. For that we need to specify the model of signal $s[n]$.

One of the possibilities is to create a physical model of a boxing bag and from that derive analytical waveform of acceleration on a certain point on the bag and that use as a model signal. This approach is very methodical but, in this case possibly not that effective. To successfully model a punching bag accounting for its elastic and plastic deformation can be very difficult and without much benefit. The device is intended to be placed on any boxing bag and work sufficiently. Unfortunately physical model can not account for such variation of unknown parameters such as height and weight of the bag, its toughness and height of its suspension. Choice of boxers gloves also can play a role by the amount of padding in them.

We chose as model of a signal a median model created from signals in Euclidean norms calculated from all three axes on raw accelerometer data. It has many advantages over analytical model that are discussed further in later chapters, it helps us to determine a typical waveform of a signal, but mainly it brings bag-specificity of the signal model which can not be achieved or predicted. Also use of the Euclidean norm prevent the detector from being direction specific. With Euclidean norm punches from all directions will be detected without difference. Its main disadvantage is necessity of calibration phase in which sufficient punch database is created from which median can be calculated. Also, from the nature of the median it follows that our model signal will not ever quite match any given punch which will impede the detectors performance. However, use of Euclidean norm brings an advantage. It is calculated by the following non-linear formula

$$||\mathbf{a}|| = \sqrt{a_x^2 + a_y^2 + a_z^2} \tag{5.11}$$

From the nature of this norm it follows that any increase in the absolute value of acceleration in any axis will result into positive increase of respective norm component. Negative values are impossible from physical point of view. This gives us the affirmation that our amplitude a can acquire only positive values. This fact allows us to take advantage of a UMP test given by equation (5.9). Yet, the real value of the amplitude A is still unknown. The parameter A will be substituted with its MLE $\hat{A}$. The MLE for a scalar parameter is defined to be the value of θ that maximizes $p(\mathbf{x};\theta)$ for $\mathbf{x}$ fixed,i.e, the value that maximizes the likelihood function.[2] Our task has a following form

$$\hat{A} = \underset{\check{A}>0}{\arg\max} \left(p(\mathbf{x};\check{A}) \right) \tag{5.12}$$

with

$$p(\mathbf{x};A) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{n=0}^{N-1} (x[n] - As[n])^2 \right)$$

. We will find the maximum by differentiating the log-likelihood function, which will not change the result because logarithm is monotonically increasing

function, but it will transform the likelihood function into more convenient form

$$\begin{aligned}\frac{\partial \ln(p(\mathbf{x};\hat{A}))}{\partial \hat{A}} &= \ln\left(\frac{1}{(2\pi\sigma^2)^{N/2}}\right) - \frac{1}{2\sigma^2}\sum_{n=0}^{N-1}\left(x[n]-\hat{A}s[n]\right)^2 \\ &= \frac{\partial}{\partial \hat{A}}\left[\ln\left(\frac{1}{(2\pi\sigma^2)^{N/2}}\right)\right] + -\frac{1}{2\sigma^2}\frac{\partial}{\partial \hat{A}}\left[\sum_{n=0}^{N-1}\left(x[n]-\hat{A}s[n]\right)^2\right] \\ &= -\frac{1}{2\sigma^2}\frac{\partial}{\partial \hat{A}}\left[\sum_{n=0}^{N-1}\left(x^2[n]+\hat{A}^2s^2[n]-2\hat{A}s[n]s[n]\right)^2\right] \qquad (5.13)\end{aligned}$$

. Setting (5.13) equal to zero produces [1]

$$2\hat{A}\sum_{n=0}^{N-1}s^2[n] - 2\sum_{n=0}^{N-1}x[n]s[n] = 0$$

$$\hat{A} = \frac{\sum_{n=0}^{N-1}x[n]s[n]}{\sum_{n=0}^{N-1}s^2[n]}. \qquad (5.14)$$

Now, we can substitute $\hat{A}$ in (5.14) for A in (5.7) as a MLE of A. So we take

$$-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(-2\hat{A}s[n]x[n]+\hat{A}^2s^2[n]) > \ln\gamma \qquad (5.15)$$

and substituting $\sum_{n=0}^{N-1}x[n]s[n] = \hat{A}\sum_{n=0}^{N-1}s^2[n]$ into the (5.15) results in

$$-\frac{1}{2\sigma^2}\left(-2\hat{A}\hat{A}\sum_{n=0}^{N-1}s^2[n]+\hat{A}^2\sum_{n=0}^{N-1}s^2[n]\right) > \ln\gamma \qquad (5.16)$$

. So finally we decide $\mathcal{H}_1$ if

$$\hat{A}^2 > \frac{2\sigma^2\ln\gamma}{\sum_{n=0}^{N-1}s^2[n]} \qquad (5.17)$$

or

$$\left|\hat{A}\right| > \sqrt{\frac{2\sigma^2\ln\gamma}{\sum_{n=0}^{N-1}s^2[n]}} \qquad (5.18)$$

[1]. For noise only we expect $\hat{A} \approx 0$ (since $E(\hat{A}) = 0$), and when a signal is present $|\hat{A}|$ should depart from zero. So, principle method of our detector is to use (5.14) to calculate the MLE of amplitude A and then compare it with the threshold given by (5.18). Alternatively, we have from (5.17)

$$T(x) = \left(\sum_{n=0}^{N-1}x[n]s[n]\right)^2 > 2\sigma^2\ln\gamma\sum_{n=0}^{N-1}s^2[n] \qquad (5.19)$$

or

$$\left|\sum_{n=0}^{N-1} x[n]s[n]\right| > \sqrt{2\sigma^2 \ln\gamma \sum_{n=0}^{N-1} s^2[n]} \tag{5.20}$$

[1]. Detector at (5.20) is just a correlator that accounts for the unknown sign of A by taking the absolute value. On the figure 5.3 there is shown detector given by (5.19).

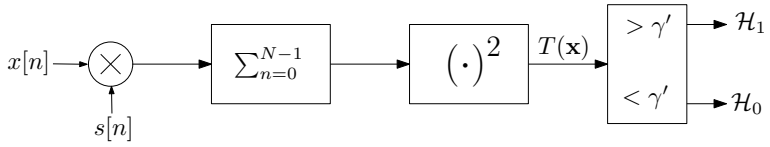


Figure 5.3: Diagram of GLRT for unknown amplitude signal.

From this model we know that our sufficient statistics will be mutual energy of a incoming data and a model of the signal $s[n]$. Now we get into the unknown time shift. If we consider that the signal we want to detect has a length M and is a part of a signal of a length N, where $M < N$, then its PDF could look something like

$$\begin{aligned} p(\mathbf{x}; n_0) = & \prod_{n=0}^{n_0-1} \tfrac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\tfrac{1}{2\sigma^2}x^2[n]\right] \\ & \cdot \prod_{n=0}^{n_0-1} \tfrac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\tfrac{1}{2\sigma^2}(x[n] - As[n-n_0])^2\right] \\ & \cdot \prod_{n=n_0+M}^{N-1} \tfrac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\tfrac{1}{2\sigma^2}x^2[n]\right] \end{aligned}$$

. Where the first and last component represents the probability of a part of a signal before and after the wanted signal, the central component represents the probability of the wanted signal somewhere inside the more extended signal. That simplifies to

$$p(\mathbf{x}; n_0) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \cdot \exp\left[-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1} x^2[n]\right]$$
$$\cdot \exp\left[-\frac{1}{2\sigma^2}\sum_{n=n_0}^{n_0+M-1}(-2Ax[n]s[n-n_0] + A^2s^2[n-n_0])\right]$$

A MLE is found by minimizing

$$\sum_{n=n_0}^{n_0+M-1}(-2Ax[n]s[n-n_0] + A^2s^2[n-n_0])$$

But $A^2\sum_{n=n_0}^{n_0+M-1} s^2[n-n_0] = A^2\sum_{n=n_0}^{n_0+M-1} s^2[n]$ and it is not a function of n_0. So MLE would be found by maximizing

$$\sum_{n=n_0}^{n_0+M-1} x[n]s[n-n_0] \tag{5.21}$$

That this would transform statistics used for our detector from mutual energy to cross-correlation. We would slide the signal model $s[n]$ along the incoming signal $x[n]$ and work with (5.21) resulting function.

5.5 Verification of WGN characteristics

Legitimate use of detectors devised in previous chapter is possible only if noise present in our signals is in fact WGN. We made a static experiment in which we collected around 87000 samples in about 20 seconds, while the sensor was in complete rest. Then we calculated Euclidean norms from these signals and applied Chi-square goodness-of-fit test via its MATLAB implementation of the mentioned test by `chi2gof` function on the resulting vector. The function `chi2gof` returns a test decision for the null hypothesis that the data in vector x comes from a normal distribution with a mean and variance estimated from x, using the chi-square goodness-of-fit test. The alternative hypothesis is that the data does not come from such a distribution. The result h is 1 if the test rejects the null hypothesis at the 5% significance level, and 0 otherwise. This test confirmed normal distribution of the data. Histogram of the data with fitted normal distribution curve is shown in figure 5.4. Variance of collected noise was $\sigma^2 = 193.87$.

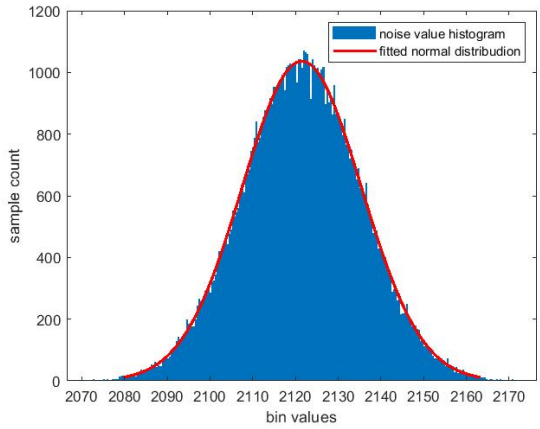


Figure 5.4: Histogram of the collected noise from the static test in Euclidean noise.

Noise has in fact a normal distribution, so use of devised detector is justified.

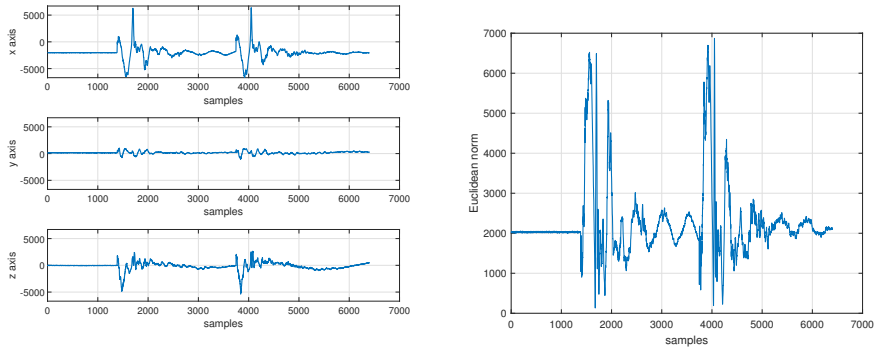
Chapter 6

Experimental verification of proposed designs

In this section, we attempted to implement and test a detector based on principles found in the chapter theoretical (analysis chap. 5). Let us start with the data we were working with. As an output from accelerometer, we were receiving three sets of data, one for each axis. As previously mentioned, we strived to devise a reliable way of signal processing which would detect punch at any time and from any direction. Athlete, usually in the course of a workout shifts his position relative to the punching bag. Also, in the aftermath of a strike, the bag itself can start rotating, thus again shifting the relative position of the sensor to the athlete. We would, therefore, have to apply detection algorithms on all axes, which would not be computationally economical, or somehow combine the outputs of all axes in the detection process. Two potential approaches were considered. Firstly there was principal component analysis. But that is a data-dependent method and is very computationally demanding. Moreover, it produces unnecessary information about the direction of a punch which we did not have a use for. Then there is Euclidean norm (6.1), which is very computationally inexpensive and preserves full energy of a strike no matter its placement. Formula for the Euclidean norm from a three-axis sensor see (6.1).

$$s[n] = \sqrt{x[n]^2 + y[n]^2 + z[n]^2} \tag{6.1}$$

For illustration in figure 6.1, you can see how Euclidean norm changes the shape of a signal.



(a) raw acceleration data in all axes (b) Euclidean norm from signals on the left

Figure 6.1: Comparison of raw acceleration from separate axes (a), with Euclidean norm created from them (b).

For our detector, we chose GLRT. From (5.20) it follows that our data statistics would be mutual energy of incoming data and model signal. We decided that we would use the segmenting window of a width equal to the model signal and shift of one sample, which would effectively change the mutual energy into the cross-correlation. Its values would be compared with the threshold also given in (5.20). We decided for this suboptimal detector because it should be a secure method with the least number of assumptions, also throughout our work not just Euclidean norm was used so double sided test was selected for assurance. Our objective was to detect punches in, more less, real-time. We would create the cross-correlation always after a given period, analyse it and move on to the next segment. Punch would be then detected if the value of cross-correlation function breaches a certain threshold. The assumption was, that all punches were when taken as an acceleration impulse measured on the bag, very similar and that even punches of different forces and placements would yield high values of cross-correlation. On the other side, the similarity of punching bag swing and a punch would be low. In the ideal case, a physical model of punch signal would be created and that used for the signal model creation. Unfortunately, the punch signal is dependent on too many variables such as abilities of the athlete, his physical attributes, a force of the strike, weight, height, composition and suspension height of the punching bag and most notably the placement of the punch. Because every athlete is different and so are punching bags, one analytical model created on the given bag could not be sufficient on a different one, or for that matter on the same one, but with a differently delivered punch. To minimise these issues, we decided to use the median model. It has the advantage of being "bag specific" and created on the spot. The downside is the necessity of a calibration phase in which a controlled series of punches is collected and the median calculated.

6.1 Databases

In this thesis, three databases were constructed. One was created for the preliminary study, analysis and first testing of the designed detector. The second was created to test the detector in real use case scenarios. The third was for further testing of the detector capabilities.

6.1.1 Database I

Our first database has in total of 44 signals. Those are divided into three sections, each dedicated to a given punch placement: the top punches, the middle punches and the bottom punches. The whole database I is completely labelled, so it is known which impulses represent punches and which do not. This database was created for our pilot study and development of our detector together with its implementation. In the top section, there are 15 signals with a total of 32 punches recorded in them. In the middle section, there are also 15 signals with a total of 75 punches, and in the bottom section, there are just 14 signals 70 punches. See table 6.1.

database	number of signals	total number of punches
top	15	32
middle	15	75
bottom	14	70

Table 6.1: Table showing a number of signals and punches for each database.

The top database has a lower punch count because those punches were landed very close to the measuring device, and big acceleration spikes were expected so greater time interval between punches was made. Also, punches of this placement should seldom occur and will not be recommended for the optimal function of the device because of the potential damage to the device. User will be advised to place the bag and the device to a position that minimises the probability of the top punches. In the middle section, there are only 14 signals because it was later discovered that a mistake was made while collecting the data, and one signal was in the database twice. Overview plots of individual database signals are shown in figures 6.5, 6.6, 6.8. For the sake of clarity, only z-axis signals are shown, because z-axis was pointed parallel to the direction of the punches thrown, so the impulses there are the clearest as we can see from figures 6.2,6.3,6.4.

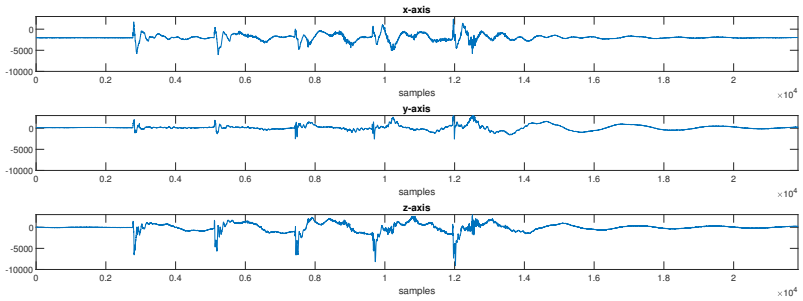


Figure 6.2: Signal overview of all axes from top database signal.

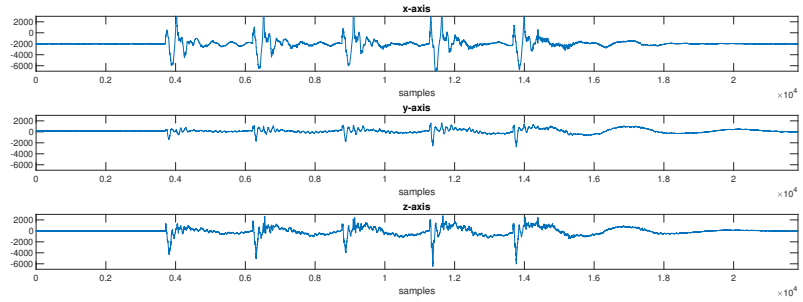


Figure 6.3: Signal overview of all axes from middle database signal.

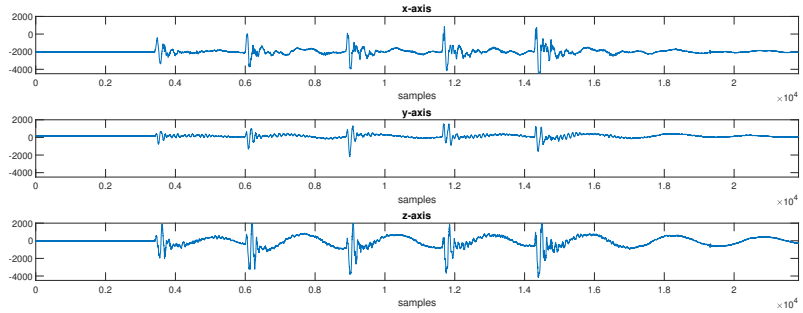


Figure 6.4: Signal overview of all axes from bottom database signal.

Also, as we can see, there is a difference in signal shape and amplitude of punches from different placements.

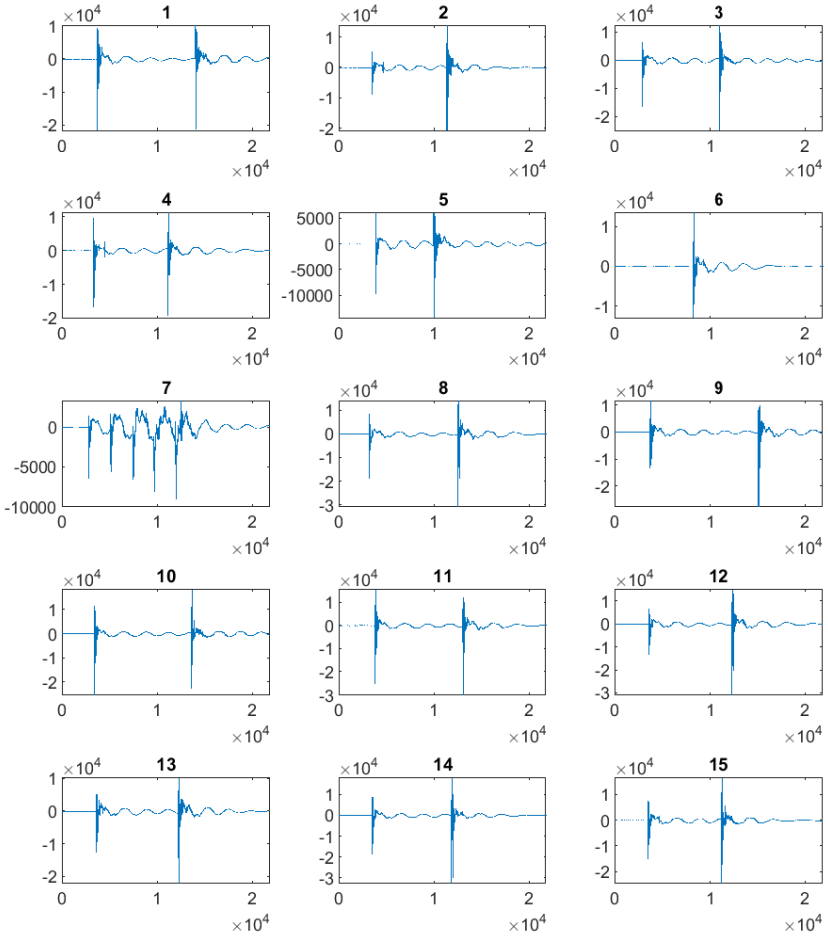


Figure 6.5: Overview of top database signals from the z-axis.

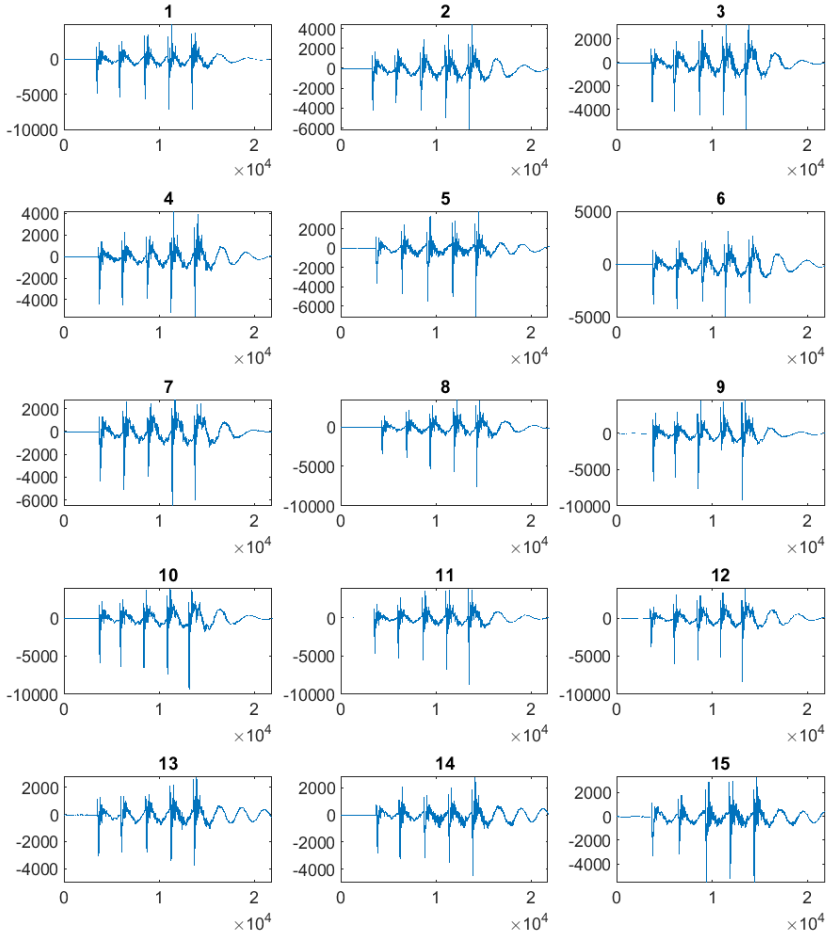


Figure 6.6: Overview of middle database signals from the z-axis.

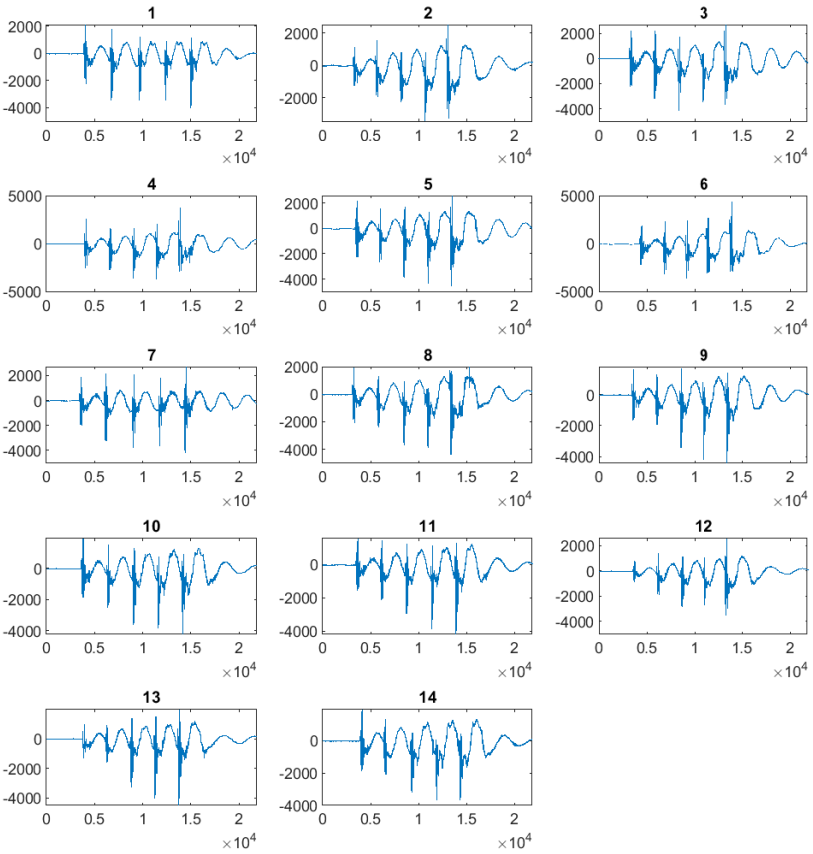


Figure 6.7: Overview of a bottom database signals from the z-axis.

6.1.2 Database II

The second database contains 8 signals about 24 seconds long, which contains all together 337 punches. This database was created to test designed real-time detectors in real use case scenarios. However, that was not the goal of this thesis. Signals were collected from two different boxers and include general punch combinations varied in force, speed and position. Boxers were advised to simulate their typical boxing bag training drills, for closer approximate real case training situation. Example of a z-axis signal from one of the signals in the second database is shown in figure 6.8.

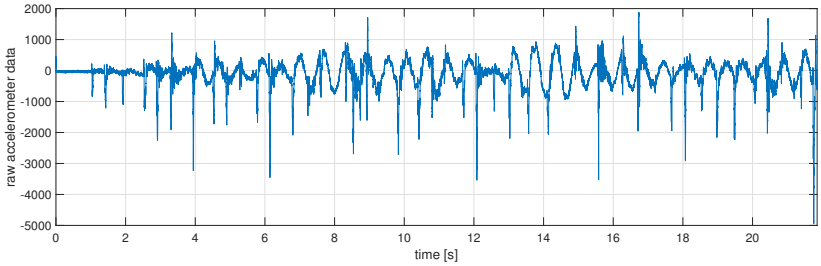


Figure 6.8: Overview of a z-axis signal from a signal in the second database. Notice the greater amount of differently spaced and high impulses caused by randomly selected punches by the boxer.

6.1.3 Database III

The third database was created for multiple different purposes. It is also, as database I, entirely labelled. It contains two sections, each focused on different competence area of the detector. The first section consists of multiple 14 signals, each containing just a fast series of 10 punches in the z-axis direction with punch frequency around 10Hz. This section should test the cadence limit of the detection. For an example signal from this section see figure 6.9.

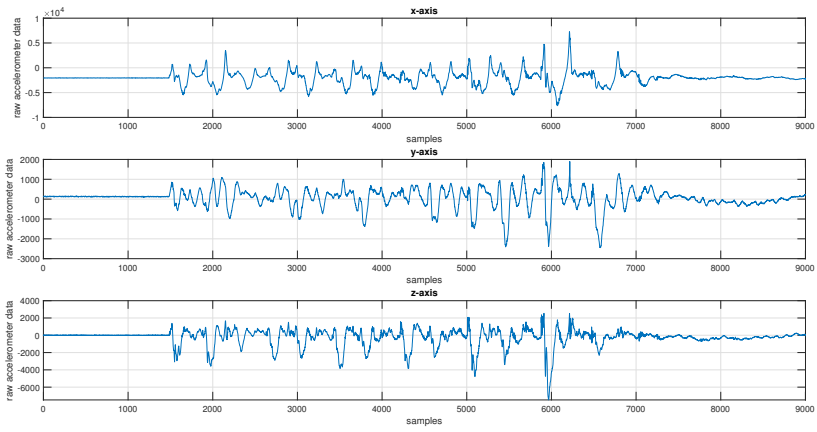


Figure 6.9: Example signal from database III containing 10 fast punches to the middle of a boxing bag.

Those 14 signals are divided into 4 signals for the top and 5 signals for middle and bottom punch placements each. Together this section contains 140 punches. The second section is dedicated to punching placement invariance. There are two different signals for each vertical punch placement (a top, middle, bottom). Both contain 6 punches. The first one there are 6 hard punches dealt from different positions around the boxing bag, see figure 6.10.

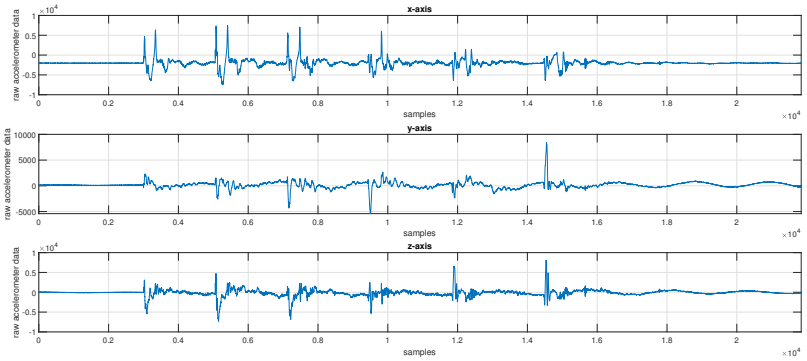


Figure 6.10: Overview of an example signal from database III containing 6 strong punches with changing position around the bag. Notice the signals in y and z-axis and their changing amplitude. That is caused by different punch placement.

In the second signal, the same format is maintained, but punches are dealt with increasing force from weak to strong, see figure 6.11.

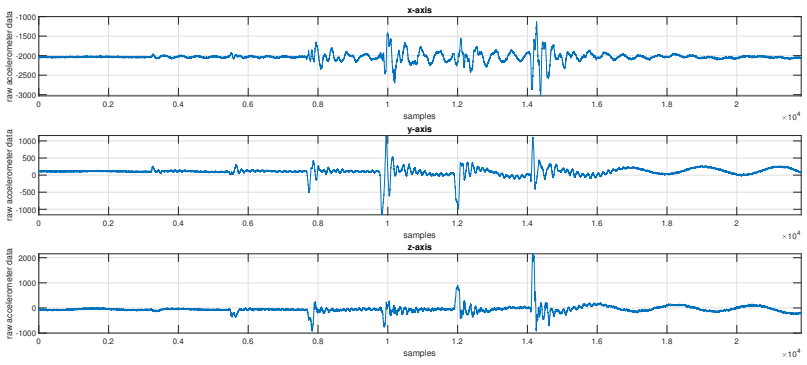
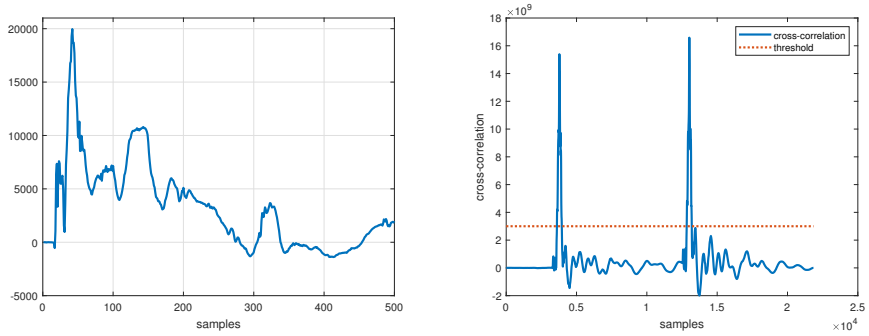


Figure 6.11: Overview of an example signal from database III containing 6 punches with increasing force and changing position around the bag. Notice the signals in y and z-axis and their changing amplitude. That is caused by different punch force. The direction change of the last three impulses clearly represents the change in the punch position.

Unfortunately, database III was not used in this thesis to keep the scope of this thesis reasonable and due to the time considerations of the thesis. They will be used for the following research for a deeper understanding of the capabilities of our detector.

6.2 Making median

From this point forward, we will use the database I for our analysis and detector design, if not stated differently. In our database I, we had long signals most of them containing multiple punches. To create a median, we had to extract signals of the punches from the signals, align them properly and create a median from them. Doing this by hand was not feasible, because of the inaccuracies in the potential misalignment of the signals. Furthermore, in the case of a real-world working device, this process had to be automatic because it is a vital part of the calibration phase mentioned earlier. We would progress through the database I from the top down. We began with the punches to the top of the bag. There we had five teen signals shown in figure 6.5. Signal 7 has an example of five punches, the rest has only two punches, except for 6 which has just one. To extract and align signals of punches from the database I, we would use the cross-correlation function. For that, we needed to pick one of the signals as an initial template in place of our modelled signal. In the finished device, this step should also be automated, but for now, we would manually pick one of the punches. In this case, it was the first punch from signal number 1. It is shown on 6.12(a).



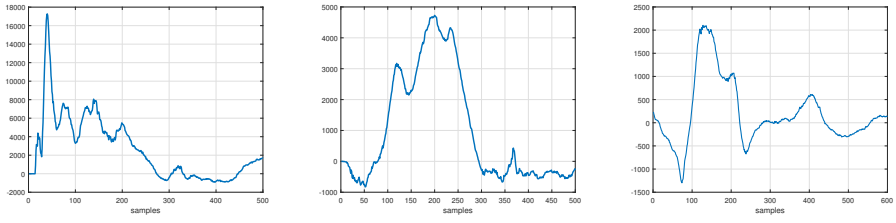
(a) Signal of a punch picked as a template for cross-correlation.

(b) Example of centred cross-correlation function of our template signal and database signal number 11

Figure 6.12: Punch impulse used to initiate the detection algorithm (left) and Cross-correlation of that impulse with database signal. (right)

From high-speed camera recordings of a punch and visual inspection of the signal, a length of the model signal of 500 samples. Cross-correlation then was made between our selected model signal and all five teen database signals. To calculate the threshold given in equation (5.20), we need to know the variance of the noise present in our signal. From our static test done in section 5.5 we know that $\sigma^2 = 193.87$. Then the significance level of 0.95 was selected that gave us $\gamma = 3$. The last thing is the calculate the energy of a median signal which we do not yet have. So, a subjective threshold value was manually selected after visual examination of the cross-correlation function. On figure 6.12(b) you can see an example of cross-correlation function of template signal

with database signal number 11. Notice the two high extremes. If we look at the figure 6.5 we can see that they are, in fact, in a position occupied by the two punch impulses in the signal 11. The threshold, in this case, could seem unnecessarily low, but its placement took into consideration cross-correlations from the whole section. Its value was around $3 \cdot 10^9$. Detection algorithm checked for the highest value above a given threshold. If there were such value, it would pinpoint its position and extracted a corresponding punch impulse from the original signal accordingly. The extracted signal had the same length as the template signal. After successful extraction, it removed detected extreme and its surrounding values in the correlation and substituted them with zeros. The width of the cleared area was dependent on a character of the correlation function. It was adjusted accordingly, to prevent high values surrounding the extreme from triggering the detection algorithm. Median was then calculated from all extracted signals. To increase that possibility of detecting all of the punches correctly without and false detections, we decided to reiterate this process of median making and each time substitute newly created median signal in place of the previous one. As any particular punch signal is not model signal but the particular realisation of that model, it could differ from the other impulses and therefore could produce deformed correlation which could lead to misalignment of the impulses or to false detections. For example, it could detect swings of the boxing bag as punches and by that contaminate the extracted punch database, which could produce lower quality median. To avoid this, we decided to substitute our present template signal with, hopefully, better and better median with each iteration. That means that this process was initialised with one of the punch signals as a model. Then after the first iteration, it was replaced by a median. We would let this run for a 10 iterations and thus taught the detector how to recognise the punches. Hopefully, with every iteration, it would be creating a better 'cleaner' database and median. After the 10 iterations, the detector was able to detect all 32 punches, with 0 false detections. Final median can be seen in figure 6.13(a).

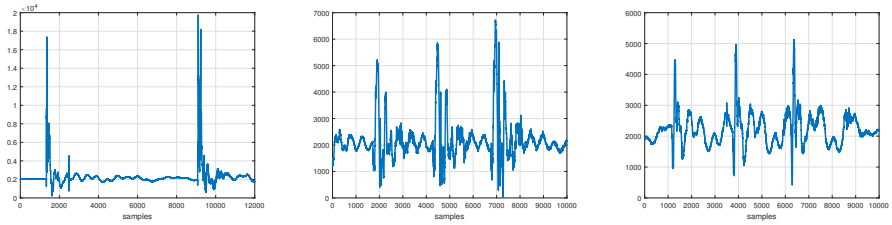


(a) Top median (b) Middle median (c) Bottom median

Figure 6.13: Medians created for each punch location after 10 iterations of detection algorithm. All signals are in euclidean norm.

This process was repeated with the section of the middle punches. Again with 100% of the punches detected, with 0 cases of false detection after 10 iterations. Resulting median is shown in figure 6.13(b). Finally, we

ran the same algorithm on the bottom section. Here we encountered some complications. Punch to the bottom of the bag is furthest from the sensor. Therefore it was harder to recognise. Also, some of the energy of the strike was converted into swinging of the bag, which was much more pronounced in the signal than with the other two punch placements. For illustration in figure 6.14 are shown three signals, each from different punch placement section of database I. We could notice that the lower the punch, the more noticeable is the swing component.



(a) Top signal (b) Middle signal (c) Bottom signal

Figure 6.14: Comparison of swing signal manifestation with different punch placement in Euclidean norms.

With the bottom punches, the swing was noticeable even in the cross-correlation function as shown in figure 6.15. We managed to detect only 67/70 punches, with 11 false detections in the section after 10 iterations. Moving the threshold lower to increase true detections did not have a positive effect and only increased the number of false detections substantially. From the figure 6.15 we could see that just by threshold detection, we could not extract the first punch, without also extracting part of the swing at the end of the signal.

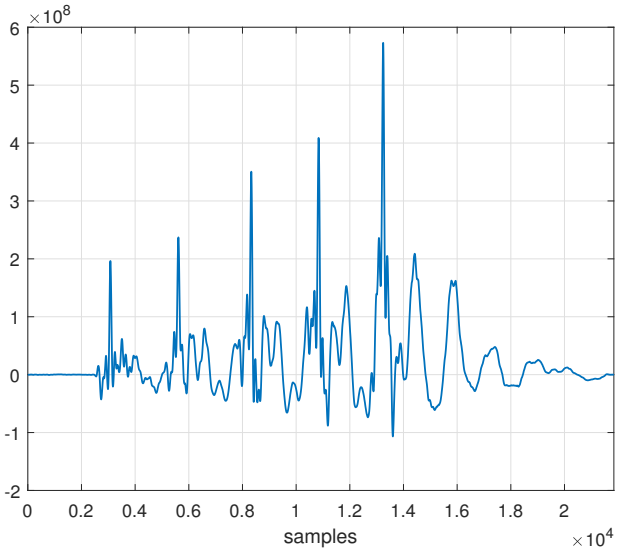
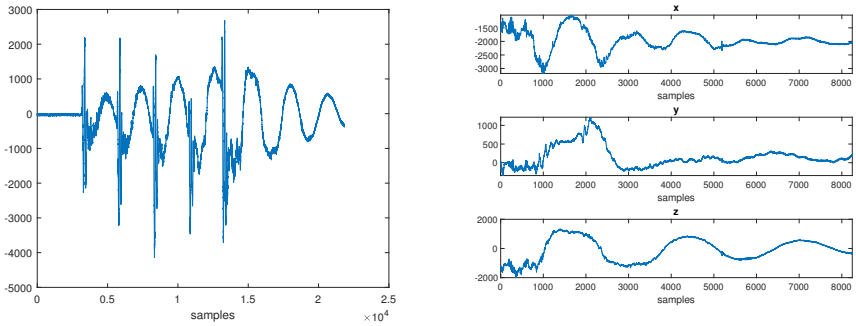


Figure 6.15: Centred cross-correlation function example from the bottom section.

It would have been great, if we could remove the swing component from our data. We decided to investigate the frequency characteristics of the swings. We looked through the database for a clear segment of swing signal. We decided to use the ending part of one of the bottom signals. In the figure 6.16(a) we can see the z-axis signal of the selected database signal.



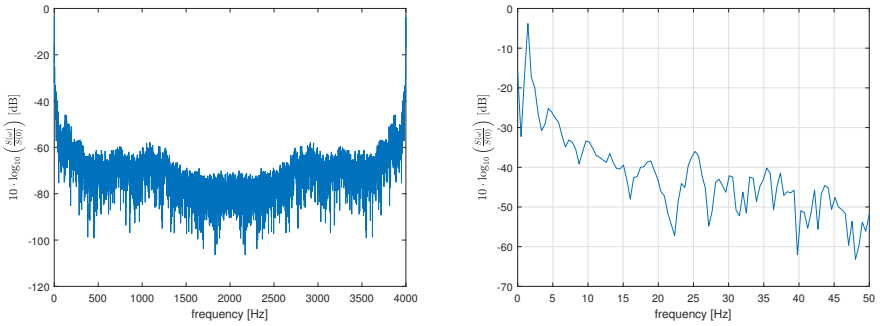
(a) Z-axis part of the database signal (b) Swing signal in all axes

Figure 6.16: On the left, there is z-axis database signal from which the ending part was used. On the right is the selected swing signal shown in all axes.

Our biggest concern is the signal in the z-axis, where the swing signal is most pronounced. We looked at the frequency spectrum created from the z-axis swing signal by discrete Fourier transform [3] via `fft()` (MATLAB implementation of Fast Fourier transform). In figure 6.17 we can see logarithm of a power spectrum calculated by

$$S_{\log} = 10 \cdot \log_{10}\left(\frac{S(\omega)}{S(0)}\right)$$

.

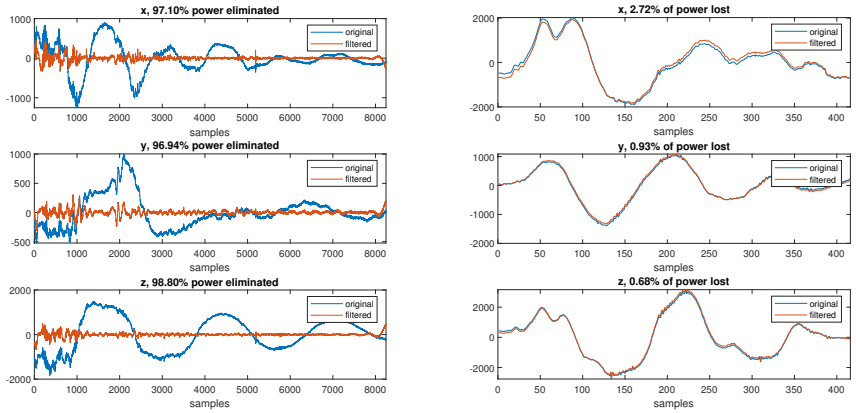


(a) Whole log spectrum of a swing signal in z-axis (b) Closer look on the lower frequencies

Figure 6.17: On the left, there is whole spectrum of the swing signal. On the there is a detail on the lower part of the spectrum.

We could see from figure 6.17(b), that around 10Hz the spectrum drops under −30dB which means, that any frequency higher than 10Hz constitute

less than 10^{-3} of signal power. To make sure that filtering out frequencies up to 10Hz would be enough, we placed zero values in place of all spectral lines up to 10.18Hz and performed inverse Fourier transform. Comparison between original and filtered swing signal can be seen in figure 6.18(a). The power loss was computed between filtered signal and a centred swing signal. Se we can see we were able to eliminate around 97.1% from x-axis signal, 96.94% from y-axis and more than 98.8% from z-axis.

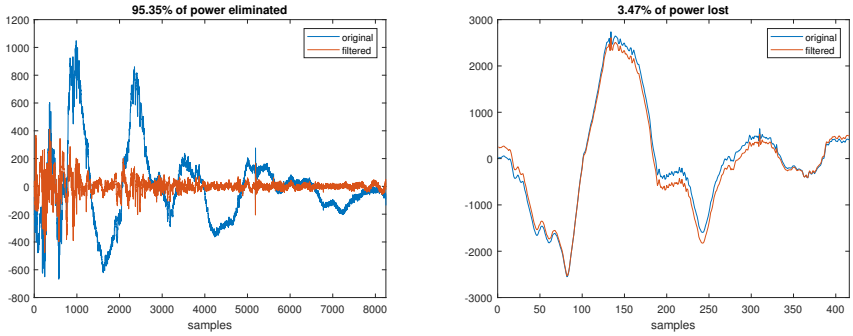


(a) Comparison of swing signals before and after filtering.

(b) Comparison of punch signals before and after filtering.

Figure 6.18: On the left, there is comparison between original swing signal and filtered swing signal. On the right, there is similar comparison for the punch signal.

Next we wanted to check how the filtering affects the punch signal. We took punch signal from the same database signal and filtered spectral lines under the 10Hz (It was not exactly 10Hz because the punch signal is short and frequency spectrum was low resolution even after zero padding.) Resulting signal and comparison to original is shown in 6.18(b). Here we lost 2.72% in x-axis, 0.93% in y-axis and 0.68% in z-axis which are very good numbers. As we were using Euclidean norm for all signal processing, we could filter the signal after the norm had been created. The results were not significantly different. Results can be seen in figure 6.19.



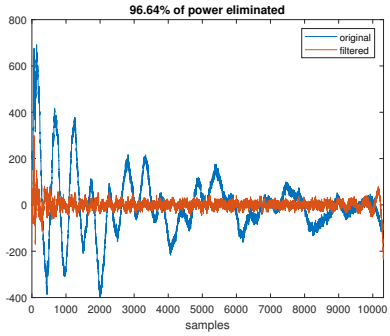
(a) Comparison of swing signals in Euclidean norm before and after filtering.

(b) Comparison of punch signals in Euclidean norm before and after filtering.

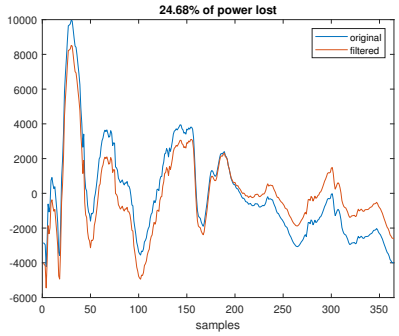
Figure 6.19: On the left, there is comparison between original swing signal and filtered swing signal both in Euclidean norms. On the right, there is similar comparison for the punch signal.

Finally, we investigated how did the filtering affect the top and middle punches. We selected signals from both remaining sections of database I and performed the same filtering of frequencies up to 10Hz. Resulting signals are shown in 6.20. Both top and middle punches were affected much more than the bottom punch, with their 24.68% and 16.38% power loss respectively.

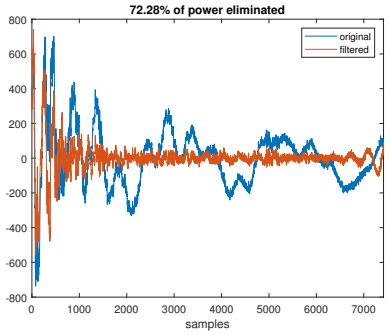
Filtering of the swing signals was much more successful with the swings caused by the top punches (96.64% power elimination). Least efficient was the filtering in elimination of the middle swing signals with just 72.28% of power eliminated. But these numbers are just for overview of filtering effects on our signals. More experiments would have to be made to investigate possible influences of signal contamination to draw clear conclusions. Observed effects were not that severe for us to defer from advancing in following research.



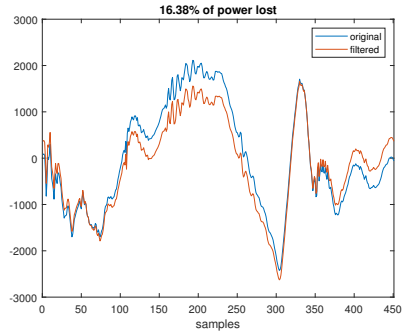
(a) Comparison of swing signals in Euclidean norm before and after filtering.



(b) Comparison of punch signals in Euclidean norm before and after filtering.



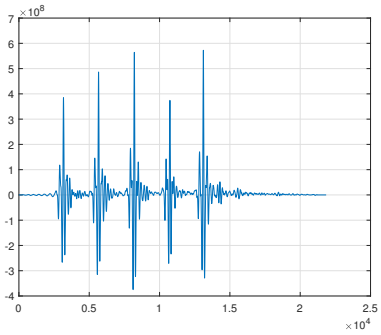
(c) Comparison of swing signals in Euclidean norm before and after filtering.



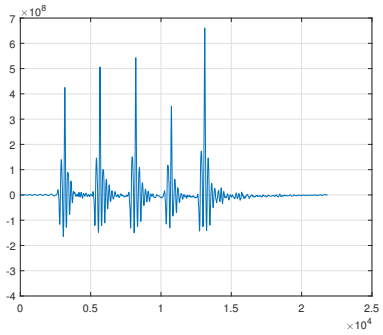
(d) Comparison of punch signals in Euclidean norm before and after filtering.

Figure 6.20: In the 6.20(a) and 6.20(b) we can see filtering results for top swing and punch signals and in 6.20(d) and 6.20(d) there is the same comparison for the punch signal. All the signals are in the Euclidean norm.

There is a possibility of signal filtering before or after the creation of Euclidean norm. When comparing effect of both filtering approaches on the cross-correlation functions, we obtained signals in figure 6.21.



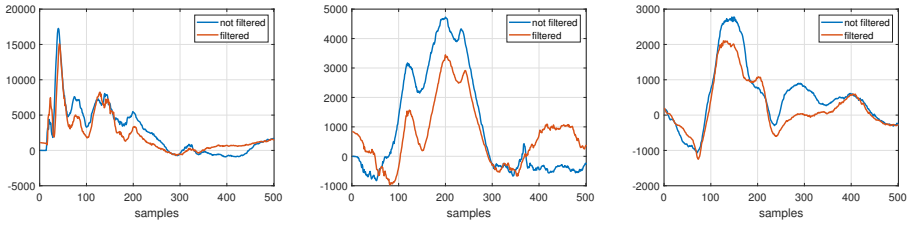
(a) Filtering after Euclidean norm



(b) Filtering before Euclidean norm

Figure 6.21: Comparison of cross-correlation function created with filtering after Euclidean norm 6.21(a) and before the Euclidean norm 6.21(b).

Both correlations we created from middle section. Cross-correlation on the left figure 6.21(a), was created from signals that was made into a Euclidean norms first and then filtered. On the right figure 6.21(b), filtering took place before the Euclidean norm. As we can see from the figure 6.21(b), this correlation has much smaller variance at some extremes even attains higher values. But there was still an argument of computational difficulty. Filtering three signals instead of just one is substantially less efficient and it should be taken into consideration when designing a working device. For all future experiments we would use filtering before the Euclidean norm. When we performed the extraction algorithm on the filtered data in the bottom section, we managed to detect 70/70 punches with 0 false detections. Attained median from the bottom section is shown in figure 6.13(c). To have uniform detection method for all punches, we needed to go back to the top and middle punches and test their sections and medians on filtered data. In top and middle filtered sections we still managed to reach 100% punches detected with no false detections. In the figure 6.22 we can see the comparison of median that have been created from original data (blue signals) and from filtered data (orange signals). As we can see the medians created from the filtered data are smaller in amplitude which is very likely caused by loss of signal power due to the filtering.



(a) Top median (b) Middle median (c) Bottom median

Figure 6.22: Comparison of medians created from original and filtered sections for each punch placement.

Next we had to test the medians on the testing signals. Now when we have the medians, we could finally calculate the thresholds that our model dictates. To calculate the threshold given in equation (5.20), we need to know the variance of the noise present in our signal. From our static test done in section 5.5 we know that $\sigma^2 = 193.87$. Then significance level of 0.95 was selected that gave us $\gamma = 3$. Last thing is the calculate an energy of a filtered median signal. For the top median the $E_{top} = 9.4 \cdot 10^9$, for middle $E_{middle} = 2.1 \cdot 10^9$ and for bottom $E_{bottom} = 5.4 \cdot 10^8$ So from equation (5.20)

we have

$$\begin{aligned}
\gamma''_{top} = \sqrt{\gamma'_{top}} &= \sqrt{2 \cdot \sigma^2 \ln \gamma \sum_{n=0}^{N-1} s^2[n]} \\
&= \sqrt{2 \cdot 193.87 \cdot \ln(3) \cdot 9.4 \cdot 10^9} \approx 2 \cdot 10^6 \\
\gamma''_{middle} = \sqrt{\gamma'_{middle}} &= \sqrt{2 \cdot 193.87 \cdot \ln(3) \cdot 2.1 \cdot 10^9} \approx 9.5 \cdot 10^5 \\
\gamma''_{bottom} = \sqrt{\gamma'_{bottom}} &= \sqrt{2 \cdot 193.87 \cdot \ln(3) \cdot 5.4 \cdot 10^8} \approx 4.8 \cdot 10^5
\end{aligned}$$

So far we used following subjectively selected thresholds (6.2).

section	subjective threshold
top	$20 \cdot 10^8$
middle	$4 \cdot 10^8$
bottom	$0.5 \cdot 10^8$

Table 6.2: Subjectively set thresholds for all sections.

We could see that thresholds given by the model and actual working thresholds differ greatly. This was due to additional noise in the signals that our model was not aware of. Except WGN noise and a punch signals in our real signals there were remains of the boxing bag swings from imperfect filtering and noise caused by erratic bag movement immediately after the punch, which were causing extremes in cross-correlation functions right on the sides of extremes caused by a punch and thus causing false detection if the extreme was high enough. Example of such extremes is in figure 6.23. Mentioned noise is non-gaussian.

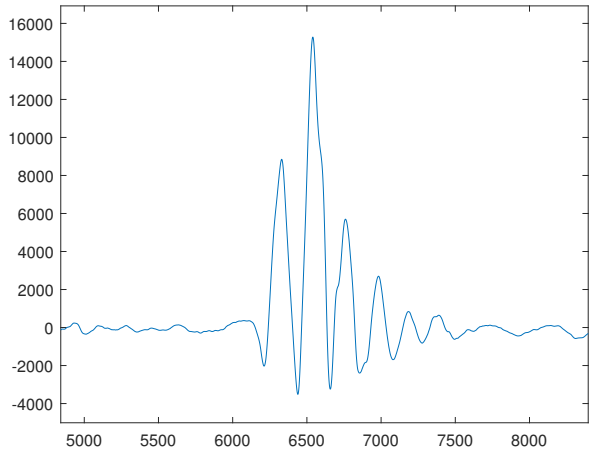


Figure 6.23: Example of side extreme caused by the noise in cross-correlation function.

Around 25000 samples of mentioned noise were selected and its variance for top, middle and bottom was found and input into the formulas for thresholds.

$$\begin{aligned}
\gamma''_{top} = \sqrt{\gamma'_{top}} &= \sqrt{2 \cdot \sigma^2 \ln \gamma \sum_{n=0}^{N-1} s^2[n]} \\
&= \sqrt{2 \cdot 8.2 \cdot 10^5 \cdot \ln(3) \cdot 9.4 \cdot 10^9} \approx 1.3 \cdot 10^8 \\
\gamma''_{middle} = \sqrt{\gamma'_{middle}} &= \sqrt{2 \cdot 4.1 \cdot 10^5 \cdot \ln(3) \cdot 2.1 \cdot 10^9} \approx 9.2 \cdot 10^7 \\
\gamma''_{bottom} = \sqrt{\gamma'_{bottom}} &= \sqrt{2 \cdot 5.1 \cdot 10^4 \cdot \ln(3) \cdot 5.4 \cdot 10^8} \approx 7.7 \cdot 10^6
\end{aligned}$$

With this change, the difference between modelled and theory thresholds decreased significantly. The most significant difference is between top thresholds, where the subjective threshold is 15 times greater than modelled. Better results were achieved with middle (4.3 times greater) and bottom thresholds (6.5 times greater).

For the following steps, we decided to use the subjectively set thresholds. We had taken the first 10 punches from each section and created medians for each section from them. We selected 10 punches because demanding 75 punches as a calibration procedure is hardly practical. So the motivation behind this step was to find out if we could get comparable detection results with fewer punches. The process was again initiated with taking first punch as a model signal. Again we set 10 iterations for redundancy, but most of the times, all punches are detected and aligned correctly after two or three iterations. Then we took the created medians and ran the testing on the remaining signals in the section. For the top punches there were 22 punches left, for the middle section there were 65 punches remaining and for the bottom section 60. In the top section, only 17 remaining punches were detected. However, with some manual adjustment of the threshold value, all 22 could be detected without any false detections. Middle and bottom medians created from just 10 punches successfully detected all 65 and 60 punches in their respective sections with no false detections. Results are summarised in the table 6.3.

section	threshold	punches present	punches detected	false detections
top	$20 \cdot 10^8$	22	17	0
middle	$4 \cdot 10^8$	65	65	0
bottom	$0.5 \cdot 10^8$	60	60	0

Table 6.3: Detection results attained with medians created with just first 10 punches from each section and with thresholds subjectively set from previous experiments with the database.

That meant that calibration phase of just 10 punches is a viable option. Up until this point we used for detection experiments thresholds carefully set for each section and taking into account every punch in the section so we could see how effective detection we could attain. But with the prospect of

automated calibrated procedure in mind, using just first 10 punches for the calibration means that threshold value, should also be set from those punches only. So to test this procedure we created a new set of thresholds subjectively set according to the first 10 punches in the section. We set the highest possible threshold which detected all of the 10 punches successfully, from a visual inspection of the cross-correlations. Some reserve was added between the top of the lowest extreme of the cross-correlation and the threshold, making a possibility of even weaker punches being detected. With these thresholds in place we reached results in table 6.4.

section	threshold	punches present	punches detected	false detections
top	$23 \cdot 10^8$	22	17	0
middle	$7 \cdot 10^8$	65	57	0
bottom	$1.2 \cdot 10^8$	60	57	0

Table 6.4: Detection results attained with new 'automated' thresholds.

Notice the increase in threshold values. Because they are higher than in previous test 0 values in false detection column are self-evident. However in middle and bottom sections there are lower true detection values. These are caused by weaker punches further into the section which had cross-correlation values lower than threshold. This brings about considerations about a structure of calibration phase. It should contain not just normal and harder punches, but also some of the weaker punches, even though it could be difficult to detect them even in calibration phase. Further contemplations are necessary.

Also, we must not forget that user would have to go through a calibration phase for each punch placement separately and thus do 30 careful punches just to calibrate the device and then there is the necessity of testing each incoming punch against three different punch models. It is feasible, but impractical. We decided to try to use one median for all punches. We selected middle median because it is most frequent punch and also it is more similar to both the top and bottom punches that they are to each other. Used middle median was created from first 10 punches and then we ran the detection algorithm as before but with top and bottom section using middle median as a model signal and checked how it performed with both original subjective and new automated thresholds, see table 6.5.

section	punches present	punches detected with 'automated' threshold	punches detected with original threshold
top	22	10	12
bottom	60	59	60

Table 6.5: Results from detector using just middle median for detection in top and middle sections.

This test was run for both original and newly created 'automated' threshold. In top section there was some anticipated drop in true detection compared

to previous experiments with both thresholds, but in bottom section the true detection with original threshold detected all 60 punches and even with automated threshold the detection increased from 57 to 59. This better performance in bottom section was likely due to comparative difference in middle median energy. Effects of differences in median energy will be investigated in detail later. So we can see that using just middle median will degrade detection performance in some cases, but advantages of using just one median makes it an attractive choice. But even if we could use just one median, we have still three thresholds that are dependent on the positioning of the punch which is always unknown. Before we try to investigate reduction in number of thresholds we should look at number of zero values around detected extremes, which are used to eliminate multiple detections of one extreme.

6.3 Zero width area

As previously mentioned, after detector detects an extreme it reduces certain number of cross-correlation samples around maximal value of that extreme to zero. That is effectively limiting the maximal punch frequency which is detectable by our device, because any other extreme in this area caused by an actual punch will be cleared away. Currently we are replacing with zeros 1000 samples before and after the detected extreme. That means that extremes had to be at least 1000 samples apart which with $f_s = 4000$ samples/sec gives us maximal punch frequency less than 4Hz. From our own experiments with high speed cameras we reached around 10 punches per second. Boxer was instructed to throw as many punches in the air as possible in 2 seconds. Punches should be at least half of arms length long for a strike to be counted as a punch. In this experiment we reaches around 10 punches per second. Guinness world records website states that world record for full contact punch strikes in one minute hold Robert Ardito with his 428 punches. That averages into 7.13 punches per seconds. That reasserted our believe in legitimacy of maximal burst speed of 10 punches per second as a legitimate limit for our device. This limit would translate into maximal possible width of the area around cross-correlation extreme to be reduced to zeros of 400 samples on each side. We tested the newly set area width of 400 samples with original threshold values and for each section its matching median was used. Results from this experiment are shown in table 6.6.

section	punches present	detected punches	false detections
top	22	22	5
middle	65	65	3
bottom	60	60	5

Table 6.6: Detection results from experiment with elimination area width of 400 samples and original threshold. For each section corresponding median was used.

Then we repeated the experiment with 'automated' threshold and matching medians and got results in table 6.7.

section	punches present	punches detected	false detections
top	22	17	0
middle	65	57	0
bottom	60	57	0

Table 6.7: Detection results from experiment with elimination area width of 400 samples and 'automated' threshold. For each section corresponding median was used.

As expected number of successfully detected punches stayed the same with original threshold, but decreasing the number of zero samples around the extreme increases the chance of false detection. With automated threshold which was made to be the highest one while detecting all punches in calibration phase, we were able to also hold the number of detected punches constant even without any false detections. We repeated detection tests for top and bottom sections just with middle median and got results in table 6.8 for automated threshold and table 6.9 for subjective threshold.

section	punches present	detected punches	false detections
top	32	14	0
bottom	70	70	0

Table 6.8: Detection results from experiment with elimination area width of 400 samples and automated threshold. Using for all sections middle median. Whole section was used as with previous experiment.

section	punches present	detected punches	false detections
top	32	17	0
bottom	70	70	51

Table 6.9: Detection results from experiment with elimination area width of 400 samples and subjective threshold. Using for all sections middle median. Whole section was used as with previous experiment.

It was possible to reach 100% punches detected by adjustment of the threshold to following results in table 6.10.

section	punches present	detected punches	false detections
top	32	32	21
bottom	70	70	51

Table 6.10: Detection results from experiment with elimination area width of 400 samples and adjusted threshold for 100% detection. Using for all sections middle median. In this experiment we used the whole sections because none of it was used for creating of the median. That is why the numbers of punches present is greater.

Detector still managed to detect 100% of present punch impulses even with just middle median. Unfortunately the less precise match of median to the data caused the secondary extremes on the sides of the main ones increase in height and thus caused more false alarm detections.

6.3.1 Thresholds

As we mentioned earlier we were always using three median models and three thresholds. One for each punch placement. Then we tried to use just middle model for detection with original thresholds which was not that efficient, especially in the top section. As we mentioned earlier that could be partially caused by differences in median energies. Because cross-correlation functions are created by multiplying two shifted signal, their own energy is positively correlated to the energies of the multiplied signals. Comparison of median signals is in figure 6.24.

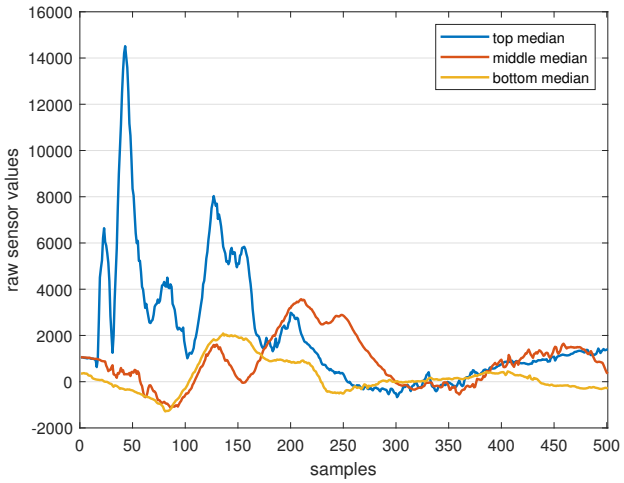


Figure 6.24: Comparison of scale differences of median signals.

This difference is naturally arises from the difference in distance of the corresponding punch placement. Top punches are much closer to the sensor and their median will naturally have greater energy than bottom punches which are much further away. The energies of the individual signals are shown in figure 6.11.

section	signal energy
top	$5.34 \cdot 10^9$
middle	$9.69 \cdot 10^8$
bottom	$2.71 \cdot 10^8$

Table 6.11: Signal energy of each median.

Top median had around twenty times the energy of the bottom one. Normalising of the medians energies could potentially greatly influence the current

substantial differences in threshold values for different punch placements. Each median signal was divided by its energy.

$$\begin{aligned} A\sum_{n=0}^{N-1} s^2[n] &= 1 \\ A &= \frac{1}{\sum_{n=0}^{N-1} s^2[n]} \end{aligned}$$

From that follows

$$\begin{aligned} A\sum_{n=0}^{N-1} s^2[n] &= \sum_{n=0}^{N-1} s_n^2[n] \\ As^2[n] &= s_n^2[n] \\ \sqrt{A}s[n] &= s_n[n] \end{aligned}$$

Each sample of the median signal have to be multiplied by square root of an inverse of the signal energy. Comparison of normalized medians is shown in figure 6.25.

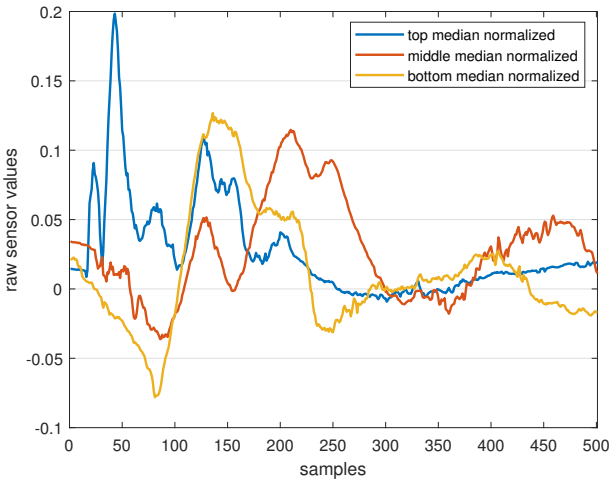


Figure 6.25: Comparison of normalized median signals.

To compare how this change affects the cross-correlation functions, we took all of the cross-correlation functions from each section and created a histogram for each section. Then we calculated cumulative sum on the histogram data. The resulting data vector was then divided by total sum of the histogram data, thus normalized, so it spanned interval from 0 to 1. In this form these vectors approximated cumulative distribution function. In figure 6.26 we can see the huge differences in threshold values for different punch placements.

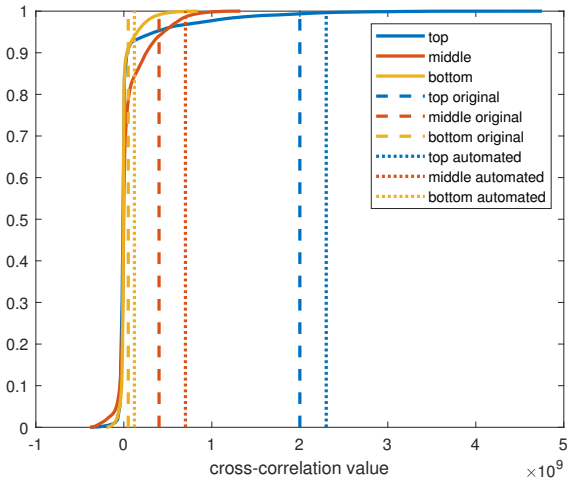


Figure 6.26: Normalized cumulative sums of cross-correlation values, with used thresholds.

Finding one common threshold for all punches looks very difficult. No the figure 6.27 there is the same type of graph only created from cross-correlations with normalized median signals corresponding to its section.

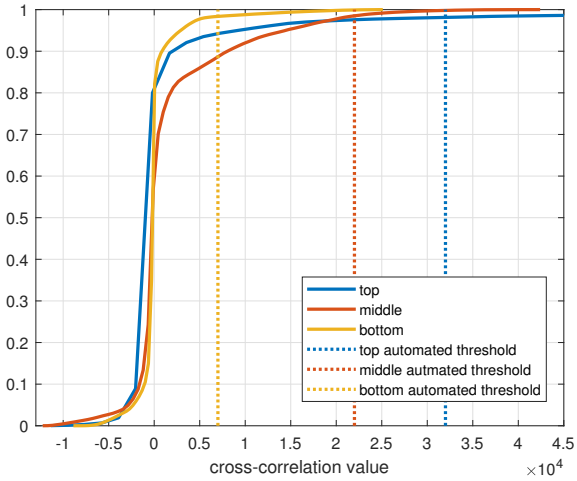


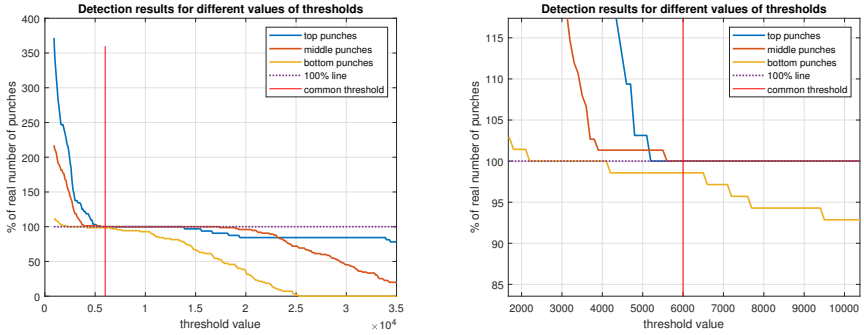
Figure 6.27: Normalized cumulative sums of cross-correlation values created with normalized medians on corresponding sections. Thresholds shown are automated the same way as previous automated thresholds from new cross-correlation values.

Thresholds for this graph were created the same way as the previous 'automated' thresholds. They are slightly below the level of the lowest cross-correlation extreme among the first 10 extremes. Their values are shown in table 6.12.

section	threshold for normalized median
top	$3.2 \cdot 10^4$
middle	$2.2 \cdot 10^4$
bottom	$0.7 \cdot 10^4$

Table 6.12: New 'automated' threshold values for normalized medians.

Differences between our new set of thresholds were much smaller. We did an experiment where, we iterated through increasing values of threshold and tracked the number of detected punches in all sections simultaneously. For each value of the threshold, we ran our detection algorithm in all three sections and recorded its performance. After we iterated through the whole range of threshold values, we normalised the collected detection performance vectors for each section by the actual number of punches present in each respective section. That means that if the detector detected 35 punches in the bottom section that has in total 70 punches, it would be 50% after the normalisation. Values above 100% represent false detections. Throughout the detection 1000 zero width area was used. Results from this experiment are shown in figure 6.28.

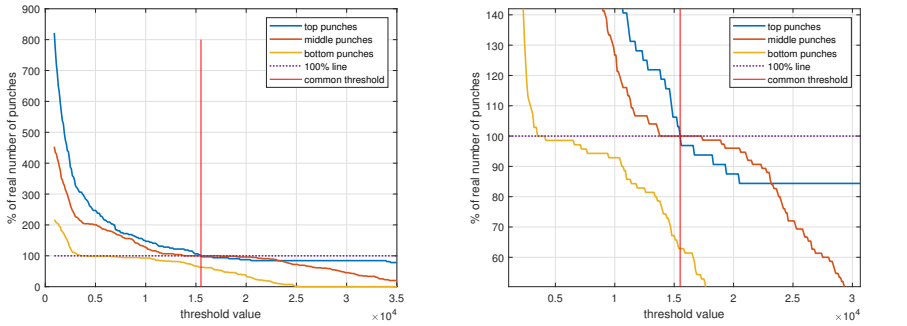


(a) Full graph (b) Detail on the area of common threshold

Figure 6.28: Downward curves represent the percentile part of detected punches for particular section. Horizontal line shows reference for the 100% of punches in each section.

Downward curves represent normalised detection performance of respective section for given value of threshold. in the figure 6.28, there is a 100% line which represents a level on which there is 100% of given section detected. We would like to find a interval of threshold values on x-axis in which all three curves lay on the 100% line. That would represent a common threshold value with which we could reach a 100% detection in all three sections of database I with 0 false detections. Unfortunately from figure 6.28(b), which shows a detailed look on the common interval, we can see that it is not possible to find a threshold value for which would all sections be on 100%. We decided to set a threshold of 6000 marked with the red vertical line in 6.28(b) as a best possible

threshold. It gives 100% detection for top and middle sections and 98.5% for bottom section. We could possibly choose even value around 5500, but that is on a limit of false detection in middle section and thus increasing its risk in future detections. This choice takes into account expected probability of punch placements which favours middle punches substantially. This is merely an assumption, but supported by experience. The same experiment was done with just 400 samples on each side of a detected extreme corresponding to maximal detectable punch frequency of 10Hz. Results are shown in figure 6.29.



(a) Full graph (b) Detail on the area of common threshold

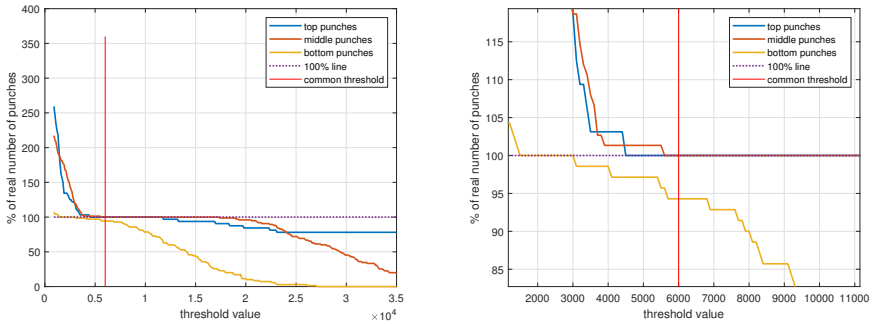
Figure 6.29: Downward curves represent the percentile part of detected punches for particular section with 400 samples reduced to zero around each extreme. Horizontal line shows reference for the 100% of punches in each section.

Common threshold increased to around 15500. This threshold would give us 100% from both top and middle punches but only 62.86% of bottom punches. For 500 zero samples, which corresponds to maximal punch frequency of 8Hz, we got threshold around 12000. Of course to get more accurate data about detection of high frequency punching series we would have to run this experiment on specialized database.

Unfortunately this 'optimal' threshold could be difficult to implement into calibration phase. Also, it is very difficult to do this in general because top, middle and bottom punches are expressions without fixed definition. In our case they were defined based on our particular boxing bag and its suspension parameters. With different bag hanged in different height these definitions and thus expected distribution of punches would change. This could be solved to some extent by requirements or recommended setting of a boxing bag for optimal device function. Nevertheless to summarize our findings. If we set our maximum detectable punch frequency to 4Hz we got 100% detection of top and middle punches and 98.5% of bottom punches. With maximal punch frequency of 10Hz we get, with threshold on 15500, 100% for both top and middle punches, but only 62.86% of bottom punches.

Main disadvantage of this detector is that it still presumes the knowledge of the punch placement and uses matching median model for each punch placement. We tested the same detector just using the middle median model.

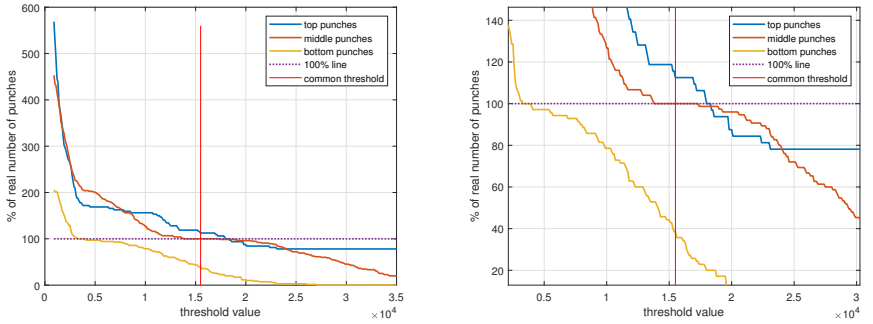
Results from the threshold iteration experiment with 1000 zero sample area width are shown in figure 6.30.



(a) Full graph (b) Detail on the area of common threshold

Figure 6.30: Downward curves represent the percentile part of detected punches for particular section with middle median and with 1000 samples reduced to zero around each extreme. Horizontal line shows reference for the 100% of punches in each section.

The best common threshold from the previous experiment with matched medians also seems to be the best fit in this case. Again we reached 100% detection in top and middle sections and 94.29% in bottom section. Which are surprisingly good results. For the 400 zero sample width area we got results shown in figure 6.31.



(a) Full graph (b) Detail on the area of common threshold

Figure 6.31: Downward curves represent the percentile part of detected punches for particular section with middle median and with 400 samples reduced to zero around each extreme. Horizontal line shows reference for the 100% of punches in each section.

Here we got 112.5% of the real number of punches detected in the top section, which corresponds to $32 \cdot 1.125 = 36$ detections so $36 - 32 = 4$ false detections. In the middle, we have 100% detection and in the bottom 38.57%, which is an inferior result. If we considered using a detector with just the middle median, we should go with 1000 zero sample width area (4 Hz maximal

punch frequency).

To properly test the newly found thresholds, we tested them on our database II created for real world use case scenarios. We joined all 8 signals from the database together. Slow-motion video recording of the data collection was inspected to verify the exact number of punches in each signal visually. We used our threshold of 6000 on cross-correlation created with our normalised middle median with 1000 zero sample width area and applied our detector on the database signal. There were 337 real punches present in the signal. Our detector was able to detect 223, which accounts for around 66%.

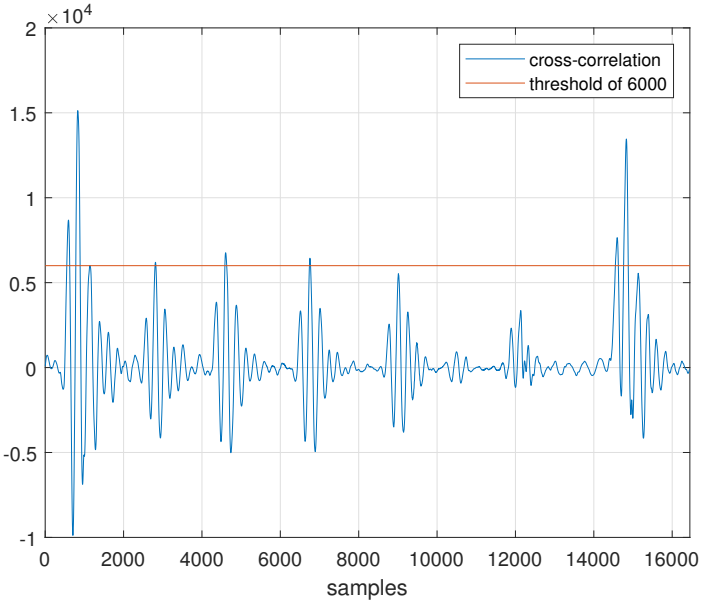


Figure 6.32: Part of the cross-correlation function of the database II signal and middle median signal with 1000 zero sample area width.

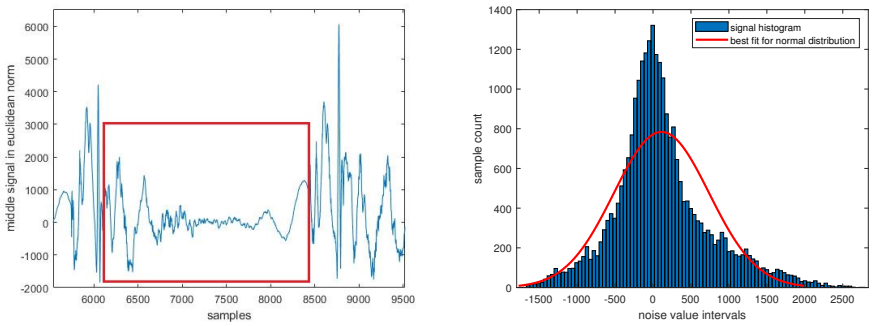
Figure 6.32 could serve as a illustration for how shifting the threshold to the left or right in figures 6.28 to 6.31 would affect the detection process. In figure 6.32, we can see that using the middle median on all punches no matter the placement combined with filtering artefacts creates high side-extremes that prevent us from lowering the threshold and detecting more punches. If we lowered the threshold to detect even the fifth punch impulse from the left we would got several false detections caused by the side-extremes from the first and last impulses. A lot of these side extremes would not be eliminated even by 1000 zero sample area width, which would cause, in case of lowering the threshold, a lot of false detections. This creates an effective limit to our detector. Test with database II resulted into inferior 66% detection despite having good results in database I and calls for alternative approaches.

Chapter 7

Alternative approaches

7.1 Refined model

As mentioned in the conclusion of the previous chapter, our biggest obstacle was false detections caused by side extremes in cross-correlations, which are more pronounced with smaller zero sample area width around main extremes. These are caused partially because our signal models are just medians from few punches and thus do not fit exactly on our signals, but mainly because in between punch signals there are areas of random signals caused by a general movement of the bag in between of the punches. Our detectors do not expect it because it is not in our model. We extracted from the filtered middle database in euclidean norm several parts containing disordered signals in between the punches and created a histogram which is shown in figure 7.1(b). Example of a signal segment that we analyse is shown in the red rectangle in figure 7.1(a).



(a) Marked example segment of signal from the middle database. The part of a signal in red rectangle in between two punches is the signal we want to analyse.

(b) Histogram of the collected signal segments, with best fit curve of normal distribution.

Figure 7.1: Analysis of unknown signals in between punches.

We could see that this signal could not be considered a WGN. It seemed that for the current signal model, we would not get better results, and a different approach is necessary. With the previous detector, we tried to use

simple signal model containing just the signal model in noise, without any other interfering signals under the assumption that only one main interfering signal in our data will be boxing bag swing, which will be eliminated by filtering. Then a long segmenting window of around 5 seconds was applied on the data and the data processed. Unfortunately, by not effective filtering or not sophisticated enough model, unsatisfactory attenuation of interference was reached. It is for several reasons, but mainly due to the unpredictability of the boxing bag parameters with which the device would be used. We must not confuse these random signals with the main frequency of boxing bag swing. Which is accounted for in by swing calibration phase, in which the bag is swung without any interference, its spectrum analysed and filter calibrated. These movement artefacts remain even after the filtering has taken place. For possible improved detection, we proposed another model which deals with the unpredictable movements of the bag the following way. Our new model could have the following form

$$x[n] = A \cdot s[n] + m + w[n]$$

. The only difference from previous model is presence of m which represent DC component of the signal. This model is expected to be used with a relatively short segment window. It we select small enough window, the low frequency signal of boxing bag swings would be noticeable in that short signal segment only as a relatively constant DC component on which the punch signal is added. It is admittedly a simplification, but it forbids the detector from being just a energy detector which expects in the absence o a punch value around zero. We would still use GLRT as a detector and use a MLE estimate of the amplitude A with consideration of the MLE estimate of the DC component and then input into GLRT detector. First let us derive the GLRT

$$\begin{aligned}
L(x) = \frac{p(\mathbf{x}|\mathcal{H}_1)}{p(\mathbf{x}|\mathcal{H}_0)} &> \gamma \\
\frac{\frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp\left(-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(x[n] - As[n] - m)^2\right)}{\frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp\left(-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(x[n] - m)^2\right)} &> \gamma \\
-\frac{1}{2\sigma^2}\left(\sum_{n=0}^{N-1}(x[n] - As[n] - m)^2 - \sum_{n=0}^{N-1}(x[n] - m)^2\right) &> \ln\gamma \\
-\sum_{n=0}^{N-1}\left(A^2 s^2[n] + 2Ams[n] - 2As[n]x[n]\right) &> 2\sigma^2 \ln\gamma
\end{aligned}$$

From that it follows

$$2A\sum_{n=0}^{N-1} s[n]x[n] > 2\sigma^2 \ln\gamma + A^2\sum_{n=0}^{N-1} s^2[n] + 2Am\sum_{n=0}^{N-1} s[n] \tag{7.1}$$

$$\tag{7.2}$$

Dividing (7.2) by A gives

$$\sum_{n=0}^{N-1} s[n]x[n] \quad > \quad \frac{\sigma^2 \ln \gamma}{A} + \frac{A}{2} \sum_{n=0}^{N-1} s^2[n] + m \sum_{n=0}^{N-1} s[n] \tag{7.3}$$

The (7.3) holds true only if $A > 0$. Now we have to calculate the MLE of unknown amplitude A.

$$\hat{A} = \arg\max_{\check{A}>0} \Big(p(\mathbf{x}; \check{A}) \Big) \tag{7.4}$$

From its likelihood function

$$p(\mathbf{x}; A, m) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp\left(-\frac{1}{2\sigma^2} \sum_{n=0}^{N-1} (x[n] - As[n] - m)^2 \right) \tag{7.5}$$

We will find maximum by differentiating the log-likelihood function (taking it as a function of A).

$$\frac{\partial \Big(\ln(p(\mathbf{x}; \hat{A}, m)) \Big)}{\partial \hat{A}} = -\frac{1}{2\sigma^2} \left(2\hat{A} \sum_{n=0}^{N-1} s^2[n] + 2 \sum_{n=0}^{N-1} ms[n] - 2 \sum_{n=0}^{N-1} s[n]x[n] \right) \tag{7.6}$$

Lastly, by setting it equal to zero we got

$$-\frac{1}{2\sigma^2} \left(2\hat{A} \sum_{n=0}^{N-1} s^2[n] + 2 \sum_{n=0}^{N-1} ms[n] - 2 \sum_{n=0}^{N-1} s[n]x[n] \right) = 0$$

and from that finally

$$\hat{A} = \frac{\sum\limits_{n=0}^{N-1} x[n]s[n] - m \sum\limits_{n=0}^{N-1} s[n]}{\sum\limits_{n=0}^{N-1} s^2[n]} \tag{7.7}$$

As we can see the MLE of A depends still on m so taking a logarithm and then derivative of (7.5), where A will be taken as known estimated $\hat{A}$ we have

$$\frac{\partial \left(\ln(p(\mathbf{x}; \hat{m})) \right)}{\partial \hat{m}} = -\frac{1}{2\sigma^2} \left(2\hat{A} \sum_{n=0}^{N-1} s[n] + 2N\hat{m} - 2 \sum_{n=0}^{N-1} x[n] \right) \tag{7.8}$$

and setting (7.8) to zero gives

$$-\frac{1}{2\sigma^2} \left(2\hat{A} \sum_{n=0}^{N-1} s[n] + 2N\hat{m} - 2 \sum_{n=0}^{N-1} x[n] \right) = 0$$

$$\hat{m} = \frac{1}{N} \sum_{n=0}^{N-1} x[n] - \frac{\hat{A}}{N} \sum_{n=0}^{N-1} s[n] \tag{7.9}$$

By substituting for m in (7.7) we have

$$\begin{aligned}\hat{A}\sum_{n=0}^{N-1}s^2[n] &= \sum_{n=0}^{N-1}x[n]s[n] - m\sum_{n=0}^{N-1}s[n] \\ \hat{A}\sum_{n=0}^{N-1}s^2[n] &= \sum_{n=0}^{N-1}x[n]s[n] - \left(\frac{1}{N}\sum_{n=0}^{N-1}x[n] - \frac{\hat{A}}{N}\sum_{n=0}^{N-1}s[n]\right)\sum_{n=0}^{N-1}s[n]\end{aligned}$$

then by collecting $\hat{A}$

$$\hat{A}\left(\sum_{n=0}^{N-1}s^2[n] + \frac{1}{N}\sum_{n=0}^{N-1}s[n]\sum_{n=0}^{N-1}s[n]\right) = \sum_{n=0}^{N-1}x[n]s[n] - \frac{1}{N}\sum_{n=0}^{N-1}x[n]\sum_{n=0}^{N-1}s[n]$$

and finally we get

$$\hat{A} = \frac{\sum_{n=0}^{N-1}x[n]s[n] - \frac{1}{N}\sum_{n=0}^{N-1}x[n]s[n]}{\sum_{n=0}^{N-1}s^2[n] + \frac{1}{N}\sum_{n=0}^{N-1}s[n]\sum_{n=0}^{N-1}s[n]} \tag{7.10}$$

Equality (7.10) no longer depends on $\hat{m}$. By substituting in (7.3) for $\hat{m}$ we get

$$\sum_{n=0}^{N-1}s[n]x[n] > \frac{\sigma^2\ln\gamma}{\hat{A}} + \frac{\hat{A}}{2}\sum_{n=0}^{N-1}s^2[n] + \left(\frac{1}{N}\sum_{n=0}^{N-1}x[n] - \frac{\hat{A}}{N}\sum_{n=0}^{N-1}s[n]\right)\sum_{n=0}^{N-1}s[n] \tag{7.11}$$

Then we compute $\hat{A}$ for each incoming segment and input that into (7.11), which gives us threshold for that particular segment. We should mention that $\hat{A}$ in the denominator of the first component of (7.11) could be a potential complication for small values of $\hat{A}$. More thorough study of this model have to be made before put into practice. Such a change in signal model could reduce our difficulties with many false detections due to low threshold.

7.2 Exponential forgetting

Previous detectors were not made for real-time processing. In attempt to implement some real-time detection we designed computationally undemanding algorithm based on the same statistics as previous models. As a statistics we used

$$T[n] = \left|\sum_{m=0}^{N-1}x[m]s[m]\right| \tag{7.12}$$

constantly updated with each incoming sample. Threshold value is initialised with first calculated value of statistic and then updated with following formula

$$\gamma[n] = \lambda\cdot(\gamma[n-1]) + (1-\lambda)\cdot T[n] \tag{7.13}$$

Detection was then validated if the T[n] exceeded the adaptively set threshold and at the same time was greater than some fixed second threshold derived

according to the statistics properties. Fixed threshold was necessary for elimination of possible detections caused by noise. After each detection a detection was halted for next 500 to avoid repeating detections. That effectively gave us maximal detectable cadence of punches of 8Hz (with $f_s = 4000$ Hz). With suitably set value of λ this detector had remarkable detection performance. In our database II it detected 297 punches from 337 with 40 false detections. That equates to 88.13% true detection rate. Database II is not labelled so this false detection count was arrived at by visual inspection of the threshold and the statistic and thus is not a precise evaluation. This approach could offer potentially great performance and will be object of our further study.

Chapter 8

Conclusion

The assignment of this thesis was to investigate the possibilities of an algorithm designed for punch detection. To reach this goal, we divided this work into several steps. First, we got acquainted with the mechanics of a boxing punch and acceleration signals that it can generate on the surface of the boxing bag. Then we, based on the characteristics of that signals, suggested, constructed and programmed hardware capable of capturing desired signals without loss of quality. Different punch signals then were analysed. Neyman Pearson theorem was used to derive the greatest likelihood ratio test and sufficient statistics for our task. Maximum likelihood estimates of amplitude and time-shift were derived, and sufficient statistics of cross-correlation was set. Then white Gaussian noise was verified by an experiment and its parameters necessary for the detector thresholds determined. In the practical part of the thesis, punch signals were measured, organised into databases I, II and III, labelled and reviewed. Euclidean norm was selected as a solution to the random placement of the punches. Median model from Euclidean norm was suggested and justified. From this finding, it follows that the calibration phase is necessary before the practical use of the device. Algorithm for automatic punch signal detection and alignment was devised and used for iterative median calculation. Contamination by swing signals was dealt with by the analysis of its spectrum and application of a high-pass filter. We found that filter effectiveness was best in signals from the bottom section and decreased in middle and top sections of database I. Also, lowest energy loss, due to the filtering, of punch signals was recorded for bottom punches and increased with middle and top punches.

From medians, we calculated thresholds given by the GLRT detector for each database and compared them with subjectively chosen thresholds based on the visual inspection of the respective databases. Difference between them was substantial by the orders of 1000. We normalised their energy to the same level created both subjective and calculated thresholds for the normalised medians and difference between them significantly decreased to orders of single multiples. That gave us the possibility of a potential common threshold, which we successfully found with 100% correct detection rate in the top and middle sections and 98.5% with 0 false detections in the bottom section of database I for a 1000 zero-width area. Also, we found

that detector performance is positively correlated to zero area width. Lastly, we tested a detector based on just middle punch median which got 100% detection with 0 false detections in the top and middle sections and almost 95% detection in the bottom section of database I for 1000 zero-width area. Additionally, we tested the detector algorithm on a database II and 66% of 337 punches were detected. Finally, alternative approaches were suggested. A more refined signal model was proposed, and GLRT and its necessary ML estimates derived. This approach is yet to be tested. Lastly, an attempt for a real-time detector was made and successfully implemented with remarkable detection performance of 306 from 337 punches were detected with 0 false detections in the database II.

To conclude, our primary detector had an outstanding performance in database I, where punches were carefully placed, spaced, and all dealt with similar reasonable power. However, in a real training scenario introduced by database II, its performance lowered significantly. This detector could be potentially used as an instrument by which an extensive labelled database of punches could be created, by drawing on its capacity to reliable detect adequately strong punches, with very low false detection rate. Database created in this way can be then used for training of a neural network, which could have greater potential for accurate detection of even weak and particular punches. An alternative approach based on an exponential forgetting showed promise of high detection performance with its 88.13% detection rate in its pilot test on database II. Further research into both approaches will be carried out in future studies.

Bibliography

[1] Kay, Steven M. *Fundamentals of Statistical Signal Processing. Vol. 2*, Prentice-Hall, 1993.

[2] Kay, Steven M. *Fundamentals of statistical signal processing: estimation theory. Vol. 1*, Prentice-Hall, (1993)

[3] Monson H. Hayes. *Statistical Digital Signal Processing and Modeling* (1st. ed.). John Wiley & Sons, Inc., USA. (1996)

[4] Klapman, M., *Boxing Glove Accelerometer.* United States Patent 5723786. (1998)

[5] Wiber, L., *Impact detection device* United States Patent US8011222B2. (2011)

[6] Desroses, M., Venkiah, Ch., *Device for detecting the nature of a blow and measuring its force, method for using same and use thereof for refereeing a sport* United States Patent US 20060161356A1,(2006)

[7] Helmer, Richard James N., Hahn, Allan G., *Impact detection method & apparatus* United States Patent US 20120279311A1, (2011)

[8] Sheedy, R., *Boxing device and related methods* United States Patent US007909749B2, (2011)

[9] Helmer, Richard James N., Hahn, Allan G., *Heavy bag workout monitor systems* United States Patent US 20120053016A1, (2011)

[10] Hartman,Kenneth D., Overmyer,Steven A., *Electronic martial arts training device* United States Patent US4974833A, (2011)

[11] Minakov, Ivan & Passerone, Roberto. *exIMUs: An Experimental Inertial Measurement Unit for Shock and Impact Detection in Sport Applications.* 170. 235 – 249. 10.1007/978-3-319-47075-7_28. (2016)

[12] Punthipayanon, Sirichet and Senakham, Tanormsak, *Wireless real-time punching force measurement and landing location capture boxing scoring system.* Department of Sport Science, Faculty of Physical education,

Srinakharinwirot University, Thailand, Department of Sport Science, University of Taipei, Taiwan, Daculty of sport Science, Kasem Bundit University, Bankok, Thailand, (2015)

[13] Venkatraman, Jay; Manwar, Rayyan; Avanaki, Kamran M. *Development of a Punch-O-Meter for Sport Karate Training.* Electronics 8, no. 7: 782. (2019)

[14] Šiška, ľuboslav & Brodáni, Jaroslav & Štefanovský, Miloš & Todorov, Svätoslav. *Basic reliability parameters of a boxing punch.* 16. 241-244. (2016) 10.7752/jpes.2016.01038.

[15] Nikovski, D., Jain, A. *Fast adaptive algorithms for abrupt change detection.* Mach Learn 79, (2010), 283–306.

[16] Dennis Kimm, David V. Thiel, *Hand speed Measurements in boxing.* (2015)

[17] *Punch sensor*
`http://usbloadcell.com/index.php#store`

[18] Chadli, Samir and Ababou, N. and Ababou, Amina, *A New Instrument for Punch Analysis in Boxing*, Procedia Engineering, (2014), doi:10.1016/j.proeng.2014.06.073

[19] Dyson, R., Smith, M., Fenn, L., Martin, C. *Differences in lead and rear hand punching forces, delivered at maximal speed relative to maximal force, by amateur boxers*, Beijing The China Institute of Sport Science (2005)

[20] Tong-Iam, Rat, Pornthep Rachanavy and Chaipat Lawsirirat. *Kinematic and Kinetic Analysis of Throwing a Straight Punch: The Role of Trunk Rotation in Delivering a Powerful Straight Punch.* Journal of physical education and sport 17, 2017:2538.

[21] Atha J, Yeadon MR, Sandover J, Parsons KC. The damaging punch. Br Med J (Clin Res Ed). 1985;291(6511):1756–1757. doi:10.1136/BMW.291.6511.1756

[22] Jacob Mack, Sarah Stojsih, Don Sherman, Nathan Dau, Cynthia Bir, *Amateur boxer biomechanics and punch force.* Biomedical Engineering Department, Wayne State University, Detroit, Michigan, USA (2010)

[23] Bobuski, J., *Osnova pro trenéry boxu*, národní boxérská federace Kuby, Czechia, (1996)

[24] InvenSense Inc., *MPU-9250 Product Specification Revision 1.1*,Document Number: PS-MPU-9250A-01, (2016)

[25] InvenSense Inc., *MPU-9250 Register Map and Descriptions Revision 1.6*, RM-MPU-9250A-00, (2015)

[26] Microchip, *23A1024/23LC1024*,DS20005142C,(2014)

Appendix A

List of attached files

A.1 Database I

Top section

- top_data_1
- top_data_2
- top_data_3
- top_data_4
- top_data_5
- top_data_6
- top_data_7
- top_data_8
- top_data_9
- top_data_10
- top_data_11
- top_data_12
- top_data_13
- top_data_14
- top_data_15

Middle section

- middle_data_1
- middle_data_2
- middle_data_3
- middle_data_4
- middle_data_5
- middle_data_6
- middle_data_7
- middle_data_8
- middle_data_9
- middle_data_10
- middle_data_11
- middle_data_12
- middle_data_13
- middle_data_14
- middle_data_15

Bottom section

- bottom_data_1
- bottom_data_2
- bottom_data_3
- bottom_data_4
- bottom_data_5
- bottom_data_6
- bottom_data_7
- bottom_data_8
- bottom_data_9
- bottom_data_10
- bottom_data_11

- bottom_data_12
- bottom_data_13
- bottom_data_14