BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INTELLIGENT SYSTEMS
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

COUNTING PEOPLE USING A PIR SENSOR

POČÍTÁNÍ OSOB POMOCÍ PIR SENZORU

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR **MARTIN BENEŠ**
AUTOR PRÁCE

SUPERVISOR **prof. Ing., Dipl.-Ing. MARTIN DRAHANSKÝ, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2018

Abstract

PIR (passive infrared) sensor are mainly used to detect a presence of a person and notifying a system to react appropriately. The aim of this thesis is to try using the PIR sensors to localize the person and present approach to estimate a count of people. To do so, thesis suggests a heat signal processing pipeline including wavelet transformation feature extraction, fuzzy logic system with classifiers based on linear regression. The performed experiment and its results are presented an evaluated.

Abstrakt

PIR (pasivní infračervený) senzor se používá zejména pro detekci přítomnosti osoby a oznámení systému pro příslušnou reakci. Cílem této práce je užití PIR senzorů pro lokalizaci osoby a návrh způsobu pro určení počtu lidí ve snímaném prostoru. Pro tento účelem je navržen způsob zpracování jeho výstupního analogového signálu, počínající extrakcí příznaků pomocí spojité vlnkové transformace, klasifikačního modelu založeném na fuzzy logice a lineární regresi. Na konci jsou představeny a vyhodnoceny experimentálně získané výsledky.

Keywords

PIR sensor, classification, fuzzy model, linear regression

Klíčová slova

PIR sensor, klasifikace, fuzzy model, lineární regrese

Reference

BENEŠ, Martin. *Counting People Using a PIR Sensor*. Brno, 2018. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor prof. Ing., Dipl.-Ing. Martin Drahanský, Ph.D.

Counting People Using a PIR Sensor

Declaration

Hereby I declare that this bachelor's thesis was prepared as an original author's work under the supervision of Prof. Ing., Dipl.-Ing. Martin Drahansky, Ph.D. a Prof. (FH) Univ.-Doz. Mag. Dr. habil. Guido Kempter. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

.....
Martin Beneš
May 14, 2019

Acknowledgements

I want to express a great gratitude to my Czech and Austrian supervisors Prof. Drahansky and Prof. Kempter as well as Dipl.-Ing. Ritter from FH Vorarlberg for assigning me to this amazing topic and aiming the focus of the research. My sincere thank you also goes to Ing. Tomas Goldmann and Ing. Daniel Sadjak for providing facilities for the research. Last but not the least, I am greatful to my family and friends for unceasing support and encouragement.

Contents

Chapter 1

Introduction

Our body as same as everything surrounding us emits some radiation. The dominant wavelengths belong to the infrared spectrum and our body senses it as a heat. If we pass its significance for living creatures and the fact that the presence of the right amount of infrared radiation is essencial for all the life as we know it, there is also a lot of usage in the industry or generally – technology.

Infrared waves are used in various devices. From nightvision devices, astronomical telescopes to personal electronics (infraport, TV remote controller). This thesis focuses on the usage in PIR sensors – electronic devices that changes its output based on the amount of received infrared radiation.

The PIR sensors are used around us a lot even though we might not know it. We all know the waving of hands towards the sensor in a hallway so the light would turn on and we could tie our shoes, or the self-opening door in shopping malls or self-rotating door in banks. These mechanisms mostly use PIR sensors.

Figure 1.1: PIR sensor: automatic doorway. [6]

PIR sensors offer even more than stating a presence of person. It is possible to process sensor output signal to get more information about sensed space - a position of person, a number of people. Especially when more sensors are used.

The localization using PIR is still matter of intense research, a number of articles has been written on it. This thesis suggests multisensor attitude and a usage of fuzzy logic to merge sensor's outputs.

Chapter 2

State of the art

2.1 Physics of radiation

In the modelled system equations calculating with infrared radiation are being used – to understand them properly it is necessary to describe what is a radiation, where does it come from and how can we measure it.

As it was already mentioned in the chapter 1, every object whose temperature is higher than absolute zero emits an electromagnetic radiation.

$$T_{obj} > 0\ K \equiv -273.15°C \tag{2.1}$$

It is caused by a charged subatomical particles (electrons, protons) that are undergoing an acceleration, emitting an energy in a form of photon – electromagnetic radiation.

Characteristics

The electromagnetic radiation have a number of measurable characteristics. The most significant ones are *frequency* f and *wavelength* λ. Due to the constant speed of the radiation v aka speed of light $c = v = 3 \cdot 10^8\ m \cdot s^{-1}$ not dependent on the frequency, they are propotional and mutually transferrable.

$$f = \frac{v}{\lambda} = \frac{c}{\lambda} = \frac{3 \cdot 10^8}{\lambda} \tag{2.2}$$

Electromagnetic waves are being divided into categories according to their usage by the wavelength λ. With increasing wavelength λ it is gamma, X-rays, ultraviolet (UV), visible light, infrared (IR) and radio waves. This is called electromagnetic spectrum and it is shown in the image 2.1.

Another measurable characteristic is an energy of the radiation Q,E or W. It is linearly dependend on its frequency f and can be computed using Planck constant $h = 6.63 \cdot 10^{-34}\ J \cdot s$. [20]

$$W = h \cdot f \tag{2.3}$$

The power of the radiation Φ is called *radiant power* or rather *radiant flux*. As a regular power it is energy per time, since the radiation is four-dimentional, partial derivations must be used.

$$\Phi = \frac{\partial W}{\partial t} \tag{2.4}$$

Figure 2.1: Electromagnetic spectrum.

Name	Symbol	Unit	Definition
Radiant flux	Φ	W	Power transfered by a radiation.
Radiant exitance	M	$W \cdot m^{-2}$	Sent W per sender's surface.
Irradiance	E	$W \cdot m^{-2}$	Received W per receiver's surface.
Radiant intensity	I	$W \cdot sr^{-1}$	W per unit solid angle.
Radiance	L	$W \cdot sr^{-1} \cdot m^{-2}$	I per sender's area projected to a direction.

Table 2.1: Radiation characteristics. [17]

The radiant power per unit surface is a flux density. It is called either *radiant exitance* M when emitting or *irradiance* E when receiving.

$$M = \frac{\partial \Phi_{emitted}}{\partial S_{sender}} \tag{2.5a}$$

$$E = \frac{\partial \Phi_{received}}{\partial S_{receiver}} \tag{2.5b}$$

The Stefan-Boltzmann law defines irradiance of electromagnetic radiation as

$$I = \sigma \cdot T^4 \tag{2.6}$$

where $\sigma = 5.6704 \cdot 10^{-8} W m^{-2} K^{-4}$ is the Stefan-Boltzmann constant and T is a thermodynamic temperature.

Power per unit solid angle I is called *radiant intensity*. With dividing by an area of the item projected from certain direction we get an amount of power emitted in that direction called *radiance* L.

$$I = \frac{\partial \Phi}{\partial \Omega} \tag{2.7a}$$

$$L = \frac{\partial I}{\partial S cos(\theta)} \tag{2.7b}$$

Other characteristics of radiation can be seen in the table 2.1. [14] [17]

2.2 Temperature homeostasis

The animal bodies require physical and chemical conditions in order to work properly (or at all). One of the physical aspects is a temperature. There are generally three types of animals – *ectotherms*, *endotherms* and *mesotherms*.

Ectotherms do not regulate its body temperature and rely on an external source, endotherms keeps it constant independently on the environment, so called *homeothermy*[1]. Mesotherm strategy is then something in between.

Endotherm groups are birds and mammals, the most significant ectotherm group are reptiles. They compensate it with basking in the sun. The thermal characteristics of these groups can be seen in the figure 2.2.

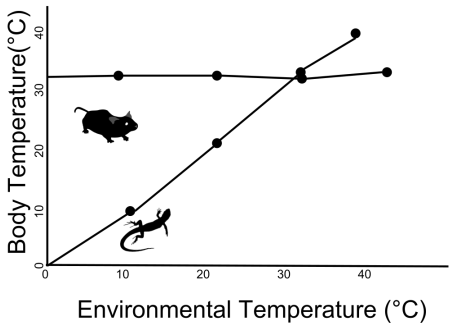

Figure 2.2: Thermal regulation graph.[12]

Human body temperature T_{HB} varies in $\langle 36°C; 38°C \rangle$, in the hyperthermia it can rise up to $40°C$. The figure 2.3 shows the radiation wavelength composition. The peak wavelength (temperature $37°C$ or $310.15\ K$) can be calculated with the Wien's displacement law.

$$\lambda_{max} = \frac{b}{T} = \frac{2.8977729 \cdot 10^{-3}}{310.15} = 9.3431\ \mu m \tag{2.8}$$

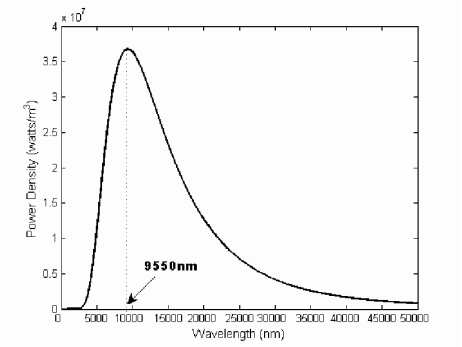

Figure 2.3: Human body radiation wavelength.[16]

[1]Homeothermy is an aspect of homeostasis. It means keeping its inner body temperature within the preset limits.

2.3 Radiation perception

Radiation processed by organisms

This thesis describes a particular way how to use the infrared radiation. Very important beginning of such work is always studying existing applications. Unforgettable one is a nature – how evolution enabled various organisms to use it.

Many animals can process parts of electromagnetic spectrum. Eyes enables mammals, cephalopods and arthropods to sense a visible light, some insects can even see a part of UV. Additionally organisms including human often have thermoreceptors in their skin so they can get information about intensity of infrared radiation around them.

Visible light PIR sensor structure is obviously inspired by a human eye. Human eyes can process radiation $\lambda \in \langle 380\ nm; 760\ nm \rangle$ called visible light, one of the bands of an electromagnetic spectrum. Seeing means receiving a light from a light source reflected by the surface of an observed object to our retina. A biological system composed of light-sensitive cells *rods* and *cones* propagates the information through nerves to brain.

A ray coming to an eye is going through a converging lens, which changes its trajectory aiming to the retina, in the best case to the most sensitive place with a lot of the rods and cons called *Fovea centralis*. [28]

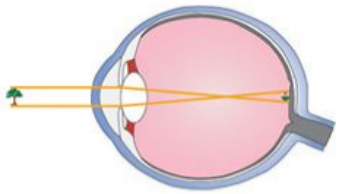

Figure 2.4: Structure of an eye. [4]

Heat The infrared rays surrounds us during our whole life, we sense it as heat. The heat receptors called *thermoreceptors* in our body are located on its surface (in the skin), but also in organs. The structure of a skin is shown in the figure 2.5.

There is a difference between sensing a visible light and an infrared radiation. Visible light comes mostly reflected from the surface, while the IR can originate only from the primary source – warm item.

Our skin contains two types of thermoreceptors: sensing cold, colder than a body temperature and hot, hotter than a body temperature. The skin structure is shown in the figure 2.5. This is already well described, on the other hand the evaluation center of these receptors in the brain and its mechanisms is not fully understanded yet and a matter of current research. [15]

Some animals even use sensing the heat as a primary way how to survive. Several groups of snakes (pythons, rattlesnakes, boas and others) use it when hunting warm-blooded animals (mouses, rats, rabbits etc.). Blood-eating organisms (vampire bats, south-american heteroptera *Triatoma infestans*) have IR receptors to look for a vein under the skin.[7]

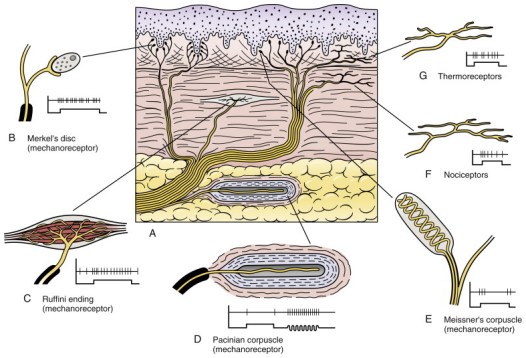

Figure 2.5: Structure of a skin. [10]

Discovery For the first time, the infrared electromagnetic waves were observed and named in 1800 by German-English astronome sir Frederick Harschel. He dispersed light by a prism and found out that the temperature of the light is growing with wavelength, the red light had the highest one.

When he measured the temperature behind the red light, there was no visible light on the table but the thermometer was showing even higher temperature moving beyond the red spectrum. Harshel pronounced hypothesis that except the visible light there must be also invisible one which we can not see. [2]

A great coincidence is that he was an astronome, he even discovered the planet Uran, and it is his discovery, the IR waves, which now enables us to explore and understand the universe. [25]

Infrared radiation processed by machines

PIR (*passive infrared*) sensor is an electronic device that scans electromagnetic radiation at wavelength $\lambda \in \langle 700\ nm; 2.5\ mm \rangle$ aka frequency $f \in \langle 120\ MHz; 430\ THz \rangle$. [8]

Principles of PIR sensor There is a number of approaches how to construct such a sensor. The point is to convert the electromagnetic energy in electric voltage and send it away via wire to be processed by hardware or software.

First way how to do it is *a bolometer*. It uses the fact that resistance of a resistor is different when changing a temperature, as shown in the equation 2.9 for temperature difference ΔT and resistor with original resistance R_0 and new resistance R_t and with temperature coefficient α. So with using the same voltage it measures the electric current and with the Ohm law $R = \frac{U}{I}$ the sensor computes instantaneus resistance.

$$R_t = R_0(1 + \alpha \Delta T) \tag{2.9}$$

Another type is *a thermoelectric sensor* reacting to the different thermoelectric resistance of exposed wire and comparative wire.

The last is *a pyroelectric detector*. The principle is based on electrostatic polarization, changing during the temperature change. [18]

Sensing of the infrared radiation The structure of PIR sensor is inspirated by structure of an eye described in subsection 2.3. An infrared ray incoming to the sensor first goes

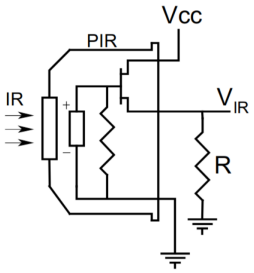

Figure 2.6: Single element pyroelectric detector.[8]

through *Fresnel lens* aiming it onto a pyroelectric sensor as you can see in the figure 2.7. Then the ray is transformed in an electric voltage.

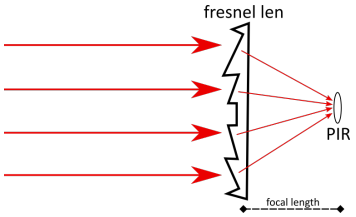

Figure 2.7: Fresnel lens.

This sensor is made for application in sensing of people. People emit radiation with characterics in the figure 2.3. The signal is amplified and filtered by built-in mechanism, described later in 5.1, to well discriminate their presence and absence. An example is a scheme 2.8 of product *PIR STD* made by *B+B Sensors*.

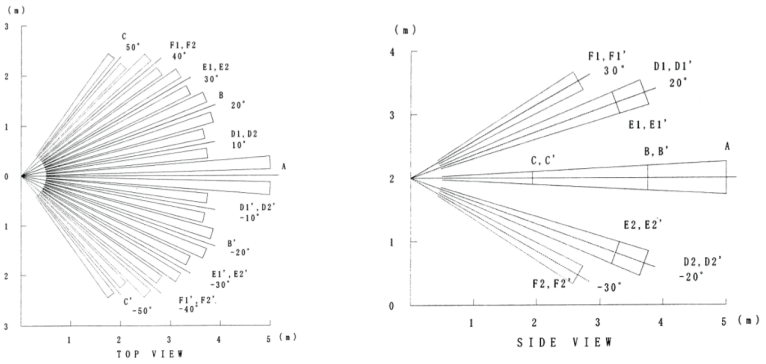

Figure 2.8: Sensing of PIR STD. [9]

2.4 Pattern recognition

The pattern recognition means processing of a signal and localization of predefined objects. The form is dependent on the type of signal and the objects that we search. Generally we can say there are five parts of recognition pipeline.

Sensing

In the world of digital computers sensing means *sampling*, converting a continuous signal into discrete samples. It is present if the signal is being processed online.

Through different signal types various technologies for sensing are used – image, sound, temperature, pressure, weight, smell etc. The sensing procedure in the case of heat signal is described in the section 2.3.

There is a few things that we need to deal during sensing: noise, linearity, callibration, ageing.

Segmentation

The signal is splitted into segments by the time axis that are being processed separately. They can even overlap. Segmentation ensures fast processing saving memory and other resources.

Features extraction

Features are quantitive expression of the input signal, they replace the signal in the following phases. Its purpose is to reduce memory and computational complexity of the processing. Each segment of N samples is transformed to vector of K features, the point is to reduce dimensions, $K << N$, but preserve relevant characteristics. Choosing the right features is therefore key for the following classifier.

To have good results the features should be discriminative (distinguish between classes), invariant to the transformations (translation, rotation, scale, deformation etc.) and decorrelated - mutually independent.

Vast number of described ways how to create features exists – *Principal Component Analysis* (PCA), *Linear Discriminant Analysis* (LDA). They can be used generally, but there is also many special application-dependent features.

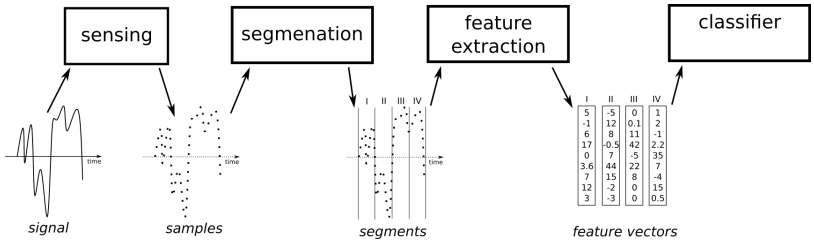

Figure 2.9: Preprocessing pipeline.

Multiple thesis and articles were written on the topic of heat signal processing. For the feature extraction, [27] suggests *Wavelet Transformation*, [26] uses *chirplet*-based features, but also tests other feature-extraction methods: PCA, Expanded-Class LDA or fusion of PCA and LDA feature vectors, [13] calculates with the signal itself.

Classification

Before we will describe the classification phase, several terms must be defined:

- *Detection* is a classifying of presence of observed object or characteristic.
- *Identification* is an assigning the observed object to one of N classes.
- *Detection Error Tradeoff* is a relation of miss to false alarm probability of a classifier. The goal is minimizing both with finding the best settings of classifier parameters.

This thesis performs a detection of a presence of a person. In the case of positive detection identification of the situation is made – what was actually detected and whether it is a person or more people etc.

Finding the most suitable classifier for the task is fundamental, but consequent to the feature extraction method we use. At the end of the day, inputs of all the methods is a vector of features for each segment. There are linear and non-linear classifiers, separating the hyperdimentional space into segments. Then the segment is detected or identified if features vector geometrically lies in the right segment.

It is also possible to perform some kind of transformation before classification if the space is not separable linearly. But other attitudes are also possible like algorithm *K-nearest neighbors*.

An output of a classifier can be either a hard decision or some kind of soft score, which can be later processed by a postprocessor – used to merge data from more classifiers or something else. A classical linear classifier is used in *Linear Regression* or *Support Vector Machine* (SVM). Each neuron of recently very popular *Neural Networks* (NN) can also be represented with linear classifier.

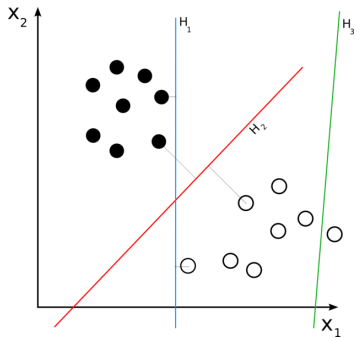

Figure 2.10: Linear classifier.[5]

Postprocessing

During postprocessing different information than pattern are used, procedure is connected to the concrete task. This parts often includes hard decision, if the classifier's output is a soft score – prices are taken into consideration, the simpliest way is using treshold.

Chapter 3

Design Description

The design of the whole project is split in two products - sensing device and the processing and visualising program - a server. Sensor measures signal sensing the observed space with connected PIR sensor and sends the data. The server collects the data from the sensor/sensors and performs the recognition of people and fusion algorithms over the results, if multiple sensors are connected.

Results might be displayed in the visualization program. The data are being sent over network. In the implementation, LAN is used, but it could be possible to use the Internet and send the data to the remote visualizer.

3.1 Sensor device

Heart of the sensing device is a preprogrammed MCU with PIR connected. Such device uses AD convertor to read the signal value from the sensor. The device may also perform fixed-sized segmentation in order to increase the effective data speed with sending more samples within one chunk, but with regard to preserving *real-time* nature.

One of the great issues, that need to be solved, is determining the position of the sensor, or even worse – mutual position of multiple sensors when used. Unless it is known, no fusion can be done. If the intention is to sense the door area and count number of incoming/outgoing people, then a position of the door is required to be known in advance.

Very elegant solution is suggested in [24]. Three PIR sensors are places in fixed mutual position on a board, as shown in the figure 3.1. This brings several advantages, not only it increases the sensing angle, but also solves problems with position of sensors, which is given by the board construction. Such a construction is unsuitable to use in rooms though.

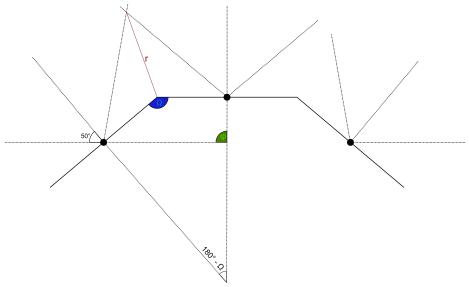

Figure 3.1: Geometry for three PIR sensors.

More flexible solution is probably inserting the dimension of space measured by the sensors and their position directly to the fusion app as parameters. Issue of this is the positioning itself. On the other hand, this solution can be easily extended with more parameters: position of doors that can persons come or leave, windows and heaters that can have unwanted influence and many others.

The device uses a serial communication channel, that can be read by the server. To increase effectivity of the communication, the MCU sends the data in fixed period as the chunk of fixed number of samples. The right size of the chunk must be chosen with regard to the effectivity, but keeping the reaction fast enough. Sampling period and sending period could be changed in the MCU configuration using REST API if the MCU is reachable in a network and keeps a running web server.

3.2 Classification server

On the used communication medium is running application listening to the channel. This server parses the chunks sent by the devices and performs the classification.

Artefact extraction and fuzzification

The signal produced by the PIR has a specific nature, as it is shown in the appendix . There are parts, that are constant, divided by abruptly changing edges, when there is a moving body. The characteristics of the movement change character of the signal.

The signal first is segmented using *continuous wavelet transformation* [1] with Morlet wavelet (3.1) and scale $s_{cwt} = 1.33846$, and the extrems are detected comparing to the $N_n = 8$ neighbors on both sides.

$$\Psi(x) = e^{-\frac{x^2}{2}} cos(5x) \tag{3.1}$$

Detected extrems are used as the borders of the segments, that lay in between them. These numbers were discovered experimentally minimizing the sum of the within-segment variances of the recorded data using formula 3.2

$$s_{cwt}, N_n = \text{argmin}(\sum_{\sigma_i \in \sigma_{s,N}} \sigma_i) \tag{3.2}$$

In the next step segment distances are evaluated. Distance of two adjoining segments aka *edge* is computed as numerical difference of their within-segment mean values.

$$||\mathbf{AB}|| = \mu_{\mathbf{B}} - \mu_{\mathbf{A}} \tag{3.3}$$

Here comes to the game fuzzification. A designed set of fuzzy membership functions ξ_F, ξ_S and ξ_R corresponds with artefacts falling edge (**F**), stagnating signal (**S**) and rising edge (**R**). The E and R are related, they use the same wave form based on transformed $arctan()$ function, parametrizable with tolerance t and center/shift x_0. S is based on gaussian function. The fuzzy set arrange is visualized in the figure 3.2 with $t = 10$.

$$\xi_R(x) = \frac{arctan(4 \cdot (\frac{x - x_0}{t} + 1))}{\pi} + 0.5 \tag{3.4a}$$

[1] Continuous wavelet transformation is a signal processing technique transforming a signal to frequency domain. The process is quite similar to fourier tranformation, instead of sinusoid it uses wavelet, a finite signal with energy $E = 0$, and therefore reacts better to abrupt changes.[19]

$$\xi_F(x) = \xi_R(-x) \tag{3.4b}$$

$$\xi_R(x) = \sqrt{exp(-\frac{(x_0 - x)^2}{2 \cdot (0.6t)^2})} \tag{3.4c}$$

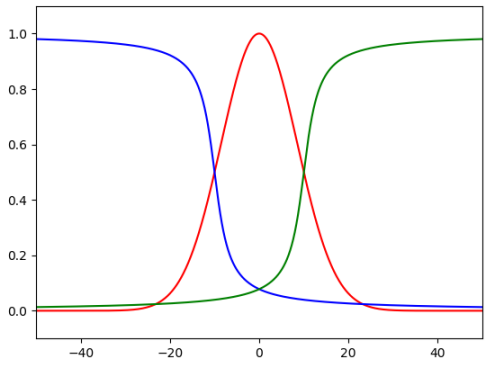

Figure 3.2: Fuzzy sets used for fuzzification.

Every edge gets assigned its fuzzy value for each of the classes. This leads to determining *artefacts*, groups of neighboring segments, which has similar characteristic (falling, rising, stagnating), but it changes in time and CWT separated them as isolated segments.

Score of border $\xi_{ABC}(x_i)$ for the edge x_i is evaluated using formula 3.5. [2] The calculation is done for all the artefact combinations ABC here, where A is characteristic of precedent x_{i-1}, B for the x_i itself and C for the follower x_{i+1}. $A, B, C \in \{F, S, B\}$.

$$\xi_{ABC}(x_i) = (\xi_A(x_{i-1})) \wedge_P (\xi_B(x_i)) \wedge_P (\xi_C(x_{i+1})) \tag{3.5}$$

The combination indicating artefact border are put together as same as the rest. That is reached using formula 3.6.[3]

Border indicating combinations are those, whose precedent and own combination differ, aka. *{SF*, FS*, RF*, FR*, RS*, SR*}* . The rest, *{FF*, SS*, RR*}*, means no edge. The results are tresholded for $T = 0.5$.

$$f_1[i] = \bigvee_{\xi \in \xi_1} {}_{p\Sigma}(\xi) \tag{3.6a}$$

$$f_0[i] = \bigvee_{\xi \in \xi_0} {}_{p\Sigma}(\xi) \tag{3.6b}$$

Newly created artefacts will be used to form feature vectors, filled with each one's characteristics. Suggested metrics are variance and mean value of samples within artefact, linear extrapolation scope of the artefact, linear extrapolation scope of predecessor etc.

The linear extrapolation scope can be found by formula 3.7, where l_0, l_1 are y coordinates of first and last point of the line respectively, $\mathbf{A}$ is a artefact samples vector. k is the searched

[2] Operator $\wedge_P$ here means *product fuzzy T-norm*: $A \wedge_P B = f_A(x) \cdot f_B(x)$.

[3] Operator $\vee_{p\Sigma}$ here means *probability sum fuzzy S-norm/T-conorm*: $A \vee_{p\Sigma} B = f_A(x) + f_B(x) - f_A(x) f_B(x)$. Other variants could be considered too - maximum, Lukasiewicz, etc.

line scope, $\Delta_{\mathbf{A},k}$ is the difference score of artefact $\mathbf{A}$ and line with scope k. It can also be used as a feature. Graphical representation is to be seen in the figure 3.3.

$$l[i] = k\mathbf{A}[i] + l_0 \tag{3.7a}$$

$$k = \frac{l_1 - l_0}{|\mathbf{A}|} \tag{3.7b}$$

$$\Delta_{\mathbf{A},k} = \sum_{i \in \{1,\dots,|\mathbf{A}|\}} (\mathbf{A}[i] - l[i])^2 \tag{3.7c}$$

$$l_0, l_1 = \text{argmin}(\Delta_{\mathbf{A},k}) \tag{3.7d}$$

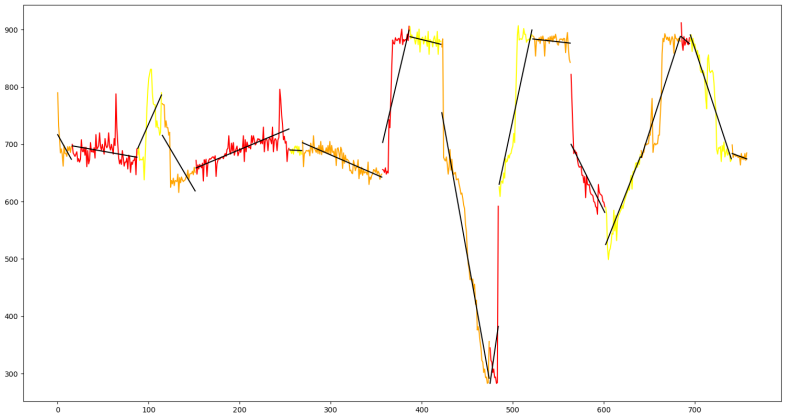

Figure 3.3: Representation of artefacts using line scope.

Classification

Extracted feature vectors for every detected artefact will undergo classification procedure consisting of multiple dichotomic classifiers, each evaluating single decision - whether the movement is present or not, if the movement is in the center or on the aside, if the movement is closer to sensor or far, and if possible, if the person is on the left side or right.

The suggested solution of classification is based on linear regression, searching for the best linear separation of the training data using linear classifier 3.8, where $\overline{x}$ is an input, $\sigma(a)$ is activation function mapping $\mathbb{R} \to \langle 0; 1 \rangle$ and $\overline{w}$,w_0 are searched parameters of the classifier.

$$y(x) = \sigma(\overline{w}^T \overline{x} + w_0) \tag{3.8}$$

The estimation of classifier's parameters $\overline{w}$,w_0 means maximizing the posterior probabilities of each training data as same as solving the equation 3.9a, in the logarithm domain 3.9b.

$$p(t|\mathbf{X}) = argmax \prod_{n \in C_1} y(\overline{x}_n) \prod_{n \in C_2} \{1 - y(\overline{x}_n)\} \tag{3.9a}$$

$$-\ln(p(t|\mathbf{X})) = -\sum_{n=1}^{N}\{(t_n)\ln(y_n) + (1-t_n)\ln(1-y_n)\} = E(\overline{w}) \tag{3.9b}$$

The searched extrem of 3.9b means derivation 3.10a equal to 0, solved numerically using 3.10b. $\eta \in \langle 0;1\rangle$ is a converging parameter, the computation can be stopped either after fixed step count or using minimal solution improvement $\varepsilon < |w^{(\tau+1)} - w^{(\tau)}|$.

$$\nabla E(\overline{w}) = \sum_{n=1}^{N}(y_n - t_n)x_n = 0 \tag{3.10a}$$

$$w^{(\tau+1)} = w^{(\tau)} - \eta\nabla E(\overline{w}) \tag{3.10b}$$

The classifier's output is a 2D fuzzy value matrix, representing the area in front of the sensor. It is inspired by the space representation by cellular automata, which carves it into small homogenous segments. The fuzzy value at certain index expresses a presence of a person at the position. Even though the space because of the construction of sensor has a shape of circular sector, it can be easily represented by classical 2D matrix, as it is shown in the figure 3.4.

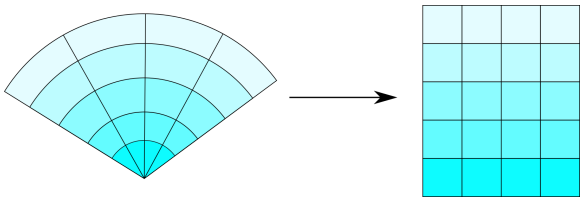

Figure 3.4: Representation of circular sector.

Each of the scores is merged together, creating the resulting matrix, as shown in the figure 3.5. Formula 3.11 illustrates the base of merging process. The trait „object is closer than T“ is scored as direct output of the classifier for the distance trait, for „object is further than T“, the score of the assertion is a fuzzy negation of the classifier output. Classifier for $\kappa = distance$ must be trained for classes with distances $< T, > T$.

In this case, the area has only two distance states – closer or more distant than T. These traits are directly dependent on the labeling. With a sufficient number of training data, a lot of traits can be created, enlarging the matrix and making the results more accurate.

For the assertions „object is on the left“ and „object is on the right“, a classifier trained with appropriate labels is used, and the logic of evaluation is the same. Merging statements is reaching the index score, and is designed using fuzzy T-norm of traits.

$$M_{<T} = A^{\kappa=distance} \tag{3.11a}$$

$$M_{>T} = N(A^{\kappa=distance}) \tag{3.11b}$$

$$M_{<0^\circ} = A^{\kappa=left} \tag{3.11c}$$

$$M_{>0^\circ} = N(A^{\kappa=left}) \tag{3.11d}$$

$$M_{<0^\circ,<T\ M} = N(A^{\kappa=left}) \wedge_P A^{\kappa=distance} \tag{3.11e}$$

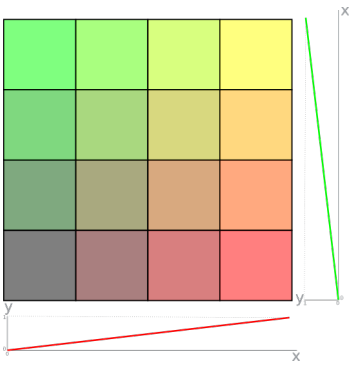

Figure 3.5: Weighting of regression score to area matrix.

Defuzzification

To get the results in a form of coordinate(s) of classified objects a cluster analysis is done. It calculates a clusters of high membership values, for optimalization α-cut on a certain level or tresholded matrix can be used. Algorithm PAM (Partitioning Around Medoids is used), similar to k-means, where item called medoid is used to represent the cluster instead of mean.

$$PAM(k, data) = argmin\left(\sum_{i=1}^{k}\sum_{j=1}^{k}||data_i data_j||\right) \tag{3.12}$$

To estimate the k, elbow method can be used: calculating k-means for different k values and taking the one with minimal within-cluster sum of square (WCSS).[11] [23]

$$WCSS = \sum_{i=1}^{k}\sum_{x\in S_i}||x-\mu_i||^2 \tag{3.13}$$

Very important is computing a distance of two segments. Since the original circular sector segmentation is not homogenous, the distances varies with each distance. It can be calculated though using cosine law. $P = (d, \alpha)$ is a segment given by polar coordinates, d is distance and α is azimuth.

$$|P_1P_2| = \sqrt{(d_1)^2 + (d_2)^2 - 2d_1d_2(\alpha_1 - \alpha_2)} \tag{3.14}$$

Fusion

It could be possible to use multiple sensors. Using more sensors brings advantages: the measuring might be more precise since the person presence classified by two independent sensors is not only more probable but the computed position can get more accurate than when using only one sensor.

The disadvantage is higher price and need for mechanism to process the configuration: mutual orientation and position of the sensors, as same as position of the objects in the room like doors or windows as „human sources“, or borders of the room.

Fusion precedes defuzzification. All the computed sensor areas represented by fuzzy matrices are merged into emptily-preinitialized whole area representation. Since these relations

between the indices are stacionary, it can be precounted after the setting the configuration for the room.

As shown in the figure 3.6, the resulting matrix with fuzzy values creates fragments with shape of either a circular sector of a circle, or a circular sector of a annulus. To avoid overcomplicated computations, these shapes can be simplified by representing them with triangles.

We have a circular sector with center S, radius r, arc edge points L, R and their polar angular deviations from the center are being φ_L, φ_R, global azimuth of the sensor is ω. Such a circular sector can be represented with a triangle SLR, as shown in the formula 3.15.

Such object is in the figure 3.6 displayed split by its height. Height can be easily found, it is a vector SH, where $H = L + \frac{1}{2}\overline{LR}$.

$$S = S \tag{3.15a}$$

$$L = S + r * (cos(\omega + \varphi_L), sin(\omega + \varphi_L)) \tag{3.15b}$$

$$R = S + r * (cos(\omega + \varphi_R), sin(\omega + \varphi_R)) \tag{3.15c}$$

In the case of the circular sector of annulus, we have center S, polar angular deviations φ_L,φ_R and radiuses of the bordering arcs, minimal radius r_1 and maximal radius r_2. This can be overlaid by trapezoid L_1, R_1, R_2, L_2, the $\Delta\varphi_L\varphi_R$ should not be to big, otherwise a significant discrepancies can occur. The formulas for stating the trapezoid points are shown in the formula 3.16.

$$L_1 = S + r_1 * (cos(\omega + \varphi_L), sin(\omega + \varphi_L)) \tag{3.16a}$$

$$R_1 = S + r_1 * (cos(\omega + \varphi_R), sin(\omega + \varphi_R)) \tag{3.16b}$$

$$L_2 = S + r_2 * (cos(\omega + \varphi_L), sin(\omega + \varphi_L)) \tag{3.16c}$$

$$R_2 = S + r_2 * (cos(\omega + \varphi_R), sin(\omega + \varphi_R)) \tag{3.16d}$$

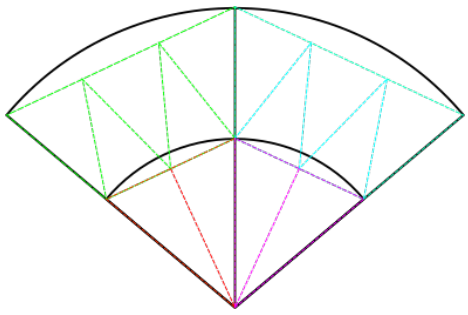

Figure 3.6: Sensed area converted to triangles.

All the objects were transformed into triangles. Now we get back to the global area matrix representation. Below is described how to compute size of area intersection of two triangles. If the global area is split into homogenous mosaic of triangle sectors, the ones intersecting with sensor area ones can be found in advance, just once and saved as a part of configuration.

The ratio of the intersecting sectors of sensor output matrixes over the global area sector gives the coefficient of their influence in the fusion. The not-covered parts of sector are considered as absence.

Two triangles intersection In order to count an intersection of two triangles, we will need formulas 3.17.

Let's consider two triangles ΔABC and ΔDEF, whose intersection is needed to be found. Firstly for each of the edges of ΔABC, AB, BC and CA, collisions with points D, E, F using $lp_{collide}$ and edges DE, EF, FD with $ll_{collide}$ are evaluated and separately sorted for each „hit" by ascending parameter p.

After that each of the vertices D, E, F are tested for laying inside the triangle (not on the edge though). If so, they are marked. Finally all these three lists are merged together with vertices, if they are marked, in clockwise order, aka DE, D^*, EF, F^*, FD, D^*.

$$L_{A,B,p\in\langle 0;1)}(X) = A + p \cdot \overline{AB} - X \tag{3.17a}$$

$$lp_{collide}(A,B,C) = \begin{cases} L_{A,B,p}(X) & L_{A,B,p} = 0 \\ \text{not on line} & \text{otherwise} \end{cases} \tag{3.17b}$$

$$ll_{collide}(A,B,C,D) = \begin{cases} X & lp_{A,B,X} \wedge L_{C,D,X} \\ \text{no intersection} & \text{otherwise} \end{cases} \tag{3.17c}$$

What needs to be point out, the object whose vertices were just counted is always convex. Provided this is acknowledged the area of the object can be counted using altered Pined filling algorithm or another filling algorithm as same as for a triangle.

Global area representation It is pretty simple to convert the problem of merging all segments of sensor output matrices into transforming all the segment shapes to triangles. Then they can be all put together into global area dependently on their mutual positions.

The global area must be segmented as well with a segment being representant of a part of the area. The fuzzy value for each such segment is derived from the sensor area segments, that correspond by position. Since the overlap does not have to be, and probably never will be perfect, there must be an algorithm, that would generate a polynome of areas influencing the value of the segment as well as coefficient for influencer.

Intuitively, the global area could be segmented homogenously into squares. This layout has a lot of advantages. Problem is, that the algorithm generating the coefficients will count with a ratio of surfaces. The intersection of areas of triangle and square is not as easy as intersection of areas of two triangles. To use this, the area could be rather than into squares split into triangles, example shown in the figure 3.7.

Formally, formulas 3.18 specify area as 3D function with third dimension being a bool, one of the values $0, 1$ and conversion these 3D coordintes into 2D coordinates of vertices of the triangle.

$$A(x,y) = [x * a, y * a] \tag{3.18a}$$

$$T(x,y,b) = \begin{cases} A(x,y), A(x+1,y), A(x+1,y+1) & b = 0 \\ A(x,y), A(x+1,y+1), A(x,y+1) & b = 1 \end{cases} \tag{3.18b}$$

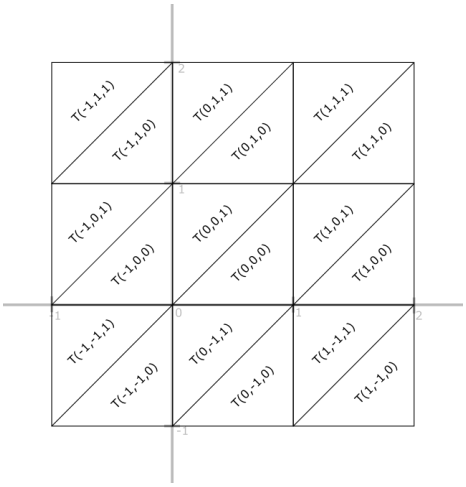

Figure 3.7: Representation of global area with triangles.

Fusion The algorithm for getting intersecting area of two triangles can be now iterated for all the pairs of triangles of global area and triangles from all sensor areas. The result will be fraction of overlap of each pair. For each area segment is held a vector of triangles from sensor areas, which overlaps it - the area is not zero. The coefficient of influence of these triangles and their fusion is shown in the formula 3.19.

$$c_{T,a} = \frac{S(T \cap a)}{S(T)} \tag{3.19a}$$

$$T_{val} = \bigvee_{\forall a : S(T \cup a) > 0} (c_{T,a} \wedge a_{val}) \tag{3.19b}$$

Chapter 4

Data

A output of a sensor as you can see in the figures 4.1 and 4.2 is very discriminative – with a bare eye a movement from no movement is distinguishable. A detection of presence used in light sensors can be implemented with tresholding, this attitude is not very suitable for classification of anything else except the presence itself.

In a calm state the sensor sends a constant signal[1]. Movement in the sensed area causes abrupt changes of output. When the object either leaves the area, or stays completely calm, the signal changes are getting slower and after a little while the output voltage gets in the calm state again.

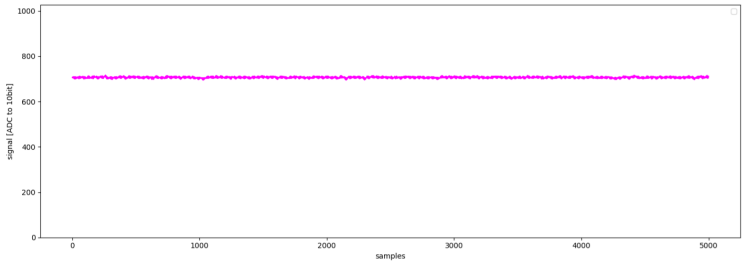

Figure 4.1: Signal of zero movement.

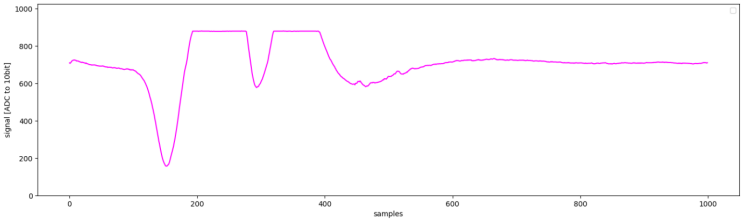

Figure 4.2: Signal of person walking around.

Therefore, the nature of the signal does not seem to need a complicated method to perform a classification on it. *Fourier transformation* and *wavelet transformation* were considered for feature extraction. The abrupt changes in the signal could be problem for

[1]Constant signal in the terms of electricity, slightly polluted with a background noise etc.

FT, because it can not represent it efficiently[27], but unlike sinusoids, the wavelets exist for a finite duration and they are suitable for representing abrupt changes.

The wavelet transformation is defined as a function $F_{(s,k)}(x)$. Parameters s,k are scale and shift, changing them in predefined unit and interval creates a matrix, as you can see in the figure 4.4.

$$F_{(s,k)}(x) = \sum_{n=1}^{N} x[n]\Psi^*_{(s,k)}[n] \tag{4.1}$$

Firstly The data were offline analysed using *Matlab*, which has implemented wavelet transformation. The Matlab is not used because it is proprietary software and the program would be dependent on it. During the analysis it was found, that the data are very well separable using continuous wavelet tranformation.

```
1 data = csvread('walk02.csv');
2 wscalogram('image', cwt(data));
```

Figure 4.3: Matlab code performing cwt.

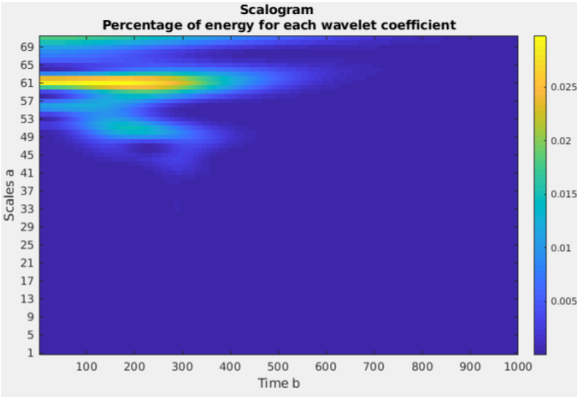

Figure 4.4: Matlab *cwt()* output of *walk03.csv*.

The training and testing data were recorded as a part of the work, results are shown and described by the appendix A.

Labeling the data set

Classification is implemented as supervised learning. The supervisory is implemented as labels for recorded files. For achieving such labels a special script *classifiers/labeller.py* was made. This script uses *segment.py* of Monitor app to make the labeling possible.

The labeling itself is made by human. We have a recorded file with signal in the format .csv and video file of the area as well in the format *.mp4. During the labeling, firstly the signal is segmented ino artefacts using classes Segment, Edge and Artefact. After that artefacts are shown to the person performing the labeling.

When the program is started, it requires entering a session name to label. After that measurements of the session are being processed. In order to work properly, if the session is named X, the session measurements must be called X_1.csv, X_2.csv, ...

Each artefact is shown as a signal plot, center frame of the video and the questions, that the person ought to answer. The first question is „Is the person present?“. Clicking 0 or 1 on the keyboard with active window with a frame the program shows another frame or question dependent on the answer. The consequent questions in the case of presence of the person are „Is the person in the center?“, „Is the parson on the left?“ and „Is the person close?“. After answering the next artefact is processed.

The labeling process is shown in the figure 4.5.

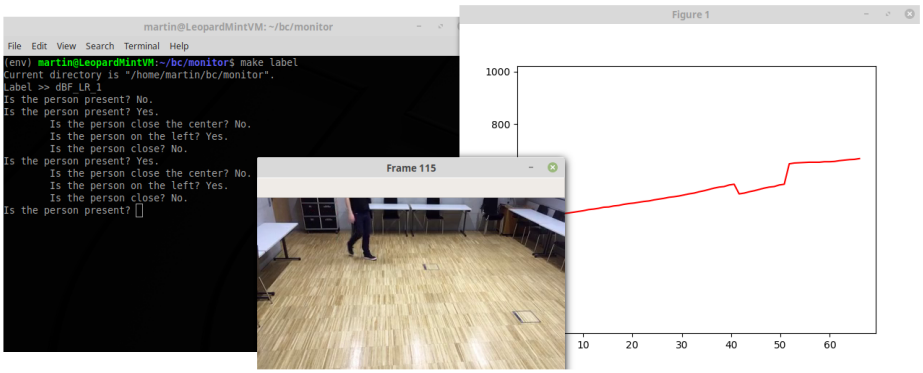

Figure 4.5: Labeling method of the data set.

Chapter 5

Implementation

The implementation was done in two parts – implementation of sensor device, that senses the signal and sends it in chunks, described in 5.1, and the server, that reads the data sent by sensor device and performs the classification, described in 5.2.

5.1 Sensor device

The sensor device consists of PIR sensor and programmed MCU that samples analog signal from sensor and sends it via WiFi to a local multicast group.

Nowadays most of the PIR sensors available on market have only a binary *switching* output. When signal reaches a set treshold, output is set to logic „1“ for a unit of time. This mechanism is suitable for a light sensor or door sonsor, completely useless for the needs of this project though.

The only found sensor that offers an analog output was *PIR STD* by *B+B Sensors*.

PIR STD

PIR STD is a product of *B+B Sensors*. It is the only found passive infrared sensor, that has except of the switching binary ouput also analog output. Except of operating voltage V_{cc} and ground GND pins, it also has reference voltage input, which needs to be connected to $\frac{V_{cc}}{2}V$. The optical resistance pins can be also used for additional classification, fusion with the infrared signal classification results and making the final result more accurate.

Pin layout table from the sensor operational manual is shown in the table 5.1.

Pin	Code	Type	Description
1	ANA	O	Analog output
2	REF	I	Reference voltage
3	GND	O	Ground
4	OUT	O	Binary (switching) output
5	GND	O	Ground
6	VCC	I	Operating voltage
7	LDR	O	Optical resistance
8	LDR	I	Optical resistance

Table 5.1: Pin layout of PIR STD.[9]

The *PIR STD* scheme shown in the figure 5.1 processes the signal in three stages going from left to right. The first two stages filter and amplify the signal, output of this is the sensor analog output. The third phase generates binary output from the analog.

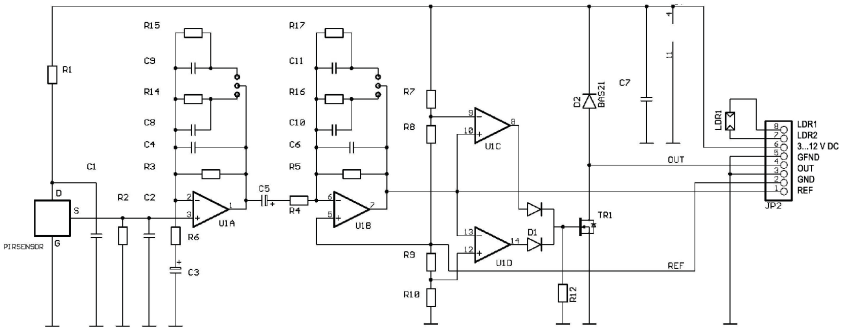

Figure 5.1: Scheme of PIR STD.[9]

I. stage The first stage starts at S of the PIR sensor, following with noise/lowpass filter consisting of the amplifier $U1A$ and the feedback components $R3$, $C4$, $C8$, $R14$, $C9$, $R14$. There is also highpass filter done by $R6$ and $C3$. The output of this stage is a signal with frequency between f_{L1} and f_{H1} amplified A_{U1A} times.

$$f_{H1} = \frac{1}{2\pi R_{3,14} C_{4,8}} \tag{5.1a}$$

$$f'_{H1} = \frac{1}{2\pi R_{3,15} C_{4,9}} \tag{5.1b}$$

The choice of resistor $R_{3,14}/R_{3,15}$ and capacitator $C_{4,8}/C_{4,9}$ is done by connected switch. The value of resistance and capacitance of the components can be counted with formula for parallel connection.

$$R^p_{A,B} = \frac{1}{R_A} + \frac{1}{R_B} \tag{5.2}$$

$$C^p_{A,B} = C_A + C_B \tag{5.3}$$

The same for the resistors $R_{5,16}$ and $R_{5,17}$ and capacitators $C_{6,10}$ and $C_{6,11}$.

$$f_{L1} = \frac{1}{2\pi R_6 C_3} \tag{5.4}$$

$$A_{U1A} = 1 + \frac{R_{3,14}}{R_2} \tag{5.5a}$$

$$A'_{U1A} = 1 + \frac{R_{3,15}}{R_2} \tag{5.5b}$$

II. stage The second processing stage focuses on amplification. It also includes lowpass filtering done by $C5$ and $R4$ and highpass filter performed by the feedback of amplifier $U1B$. The greater amplification is also made by the divider bridge ($R8$, $R9$, $R10$, $R11$) connected to the positive input. The output of frequency between $max(f_{L1}, f_{L2})$ and $min(f_{H1}, f_{H2})$ amplified $A_{U1A} \cdot A_{U1B}$ times is an analog output connected to the pin 1.

$$f_{H2} = \frac{1}{2\pi R_{5,16} C_{6,10}} \tag{5.6a}$$

$$f'_{H2} = \frac{1}{2\pi R_{5,17} C_{6,11}} \tag{5.6b}$$

$$f_{L2} = \frac{1}{2\pi R_4 C_5} \tag{5.7}$$

$$A_{U1B} = 1 + \frac{R_{5,16}}{R_4} \tag{5.8a}$$

$$A'_{U1B} = 1 + \frac{R_{5,17}}{R_4} \tag{5.8b}$$

III. stage The third phase performs top-bottom tresholding generating binary output used in simple industrial application. It is not used in the project.[3]

Connecting the sensor

PIR sensor is connected to the MCU. Except for source, ground and output which are connected directly, *PIR STD* has also reference voltage input, that should be approximately $\frac{V_{cc}}{2}$ V. To ensure that a voltage divider is used with two resistors of the same resistance R_X. During the testing 100 $k\Omega$ resistors were used. The circuit is shown in the figure 5.2.

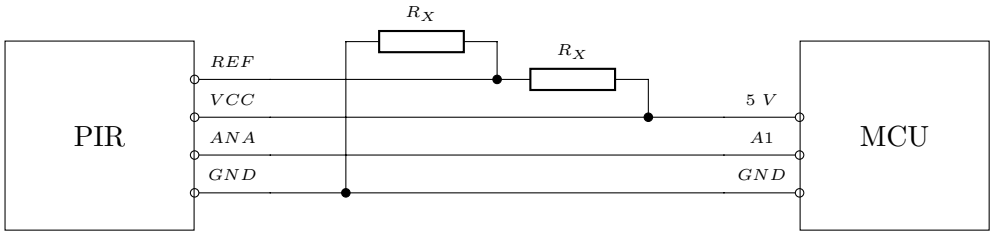

Figure 5.2: Connection of PIR and MCU.

Product design

A printed circuit board was designed (figure 5.3) and a prototype was printed with a PCB drill in electronic laboratory at *FH Vorarlberg* to illustrate the possible size and design of the final product.

The dimensions of final product were minimized to $50 \times 50 \times 25$ mm. It was possible to prototype a board connecting the NodeMCU microcontroller, sensor PIR STD, 2AA battery case and control elements (reset button, status LED) within these dimensions. The sensor case design was briefly sketched too.

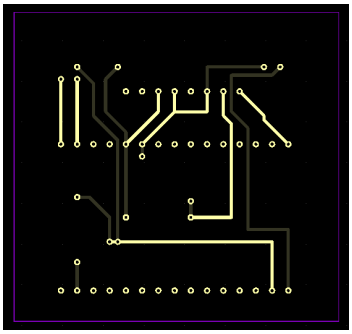

Figure 5.3: Circuit board.

Sampling

The module is programmed to read signal in sampling frequency and send the data to server.

AD conversion The projection of voltage to a value is done by the built-in functionality, accessible by standard library function `analogRead()`. The sets of analog values A and digital values D according to documentation of the function[1] and the sensor[9] and the morphism c are shown in the formula 5.9.

$$A = \langle 0; V_{cc} \rangle \tag{5.9a}$$

$$D = \{0; 1; ..., 1023\} \tag{5.9b}$$

$$c : A \to D = x_A \to \frac{1023 x_A}{V_{cc}}, x_A \in D \tag{5.9c}$$

Sampling frequency The usable sampling frequency can be estimated: the fresnel lens of *PIR STD* splits the area into $10° = \frac{\pi}{18}$ circular sectors. Object moving around the sensor in the distance 0.5 m with speed 15 $km.h^{-1} = 4.1667\ m.s^{-1}$ (very fast run) passes the central circular sector in

$$t = \frac{s}{v} = \frac{0.5 * tg(10°)}{4.1667} = 0.02116\ s \tag{5.10}$$

This means the frequency of the movement through the circular sectors is

$$f = \frac{1}{t} = \frac{1}{0.02116} = 47.259\ Hz \tag{5.11}$$

According to Shannon theorem[22], the sampling frequency must be at least twice as big as the maximal frequency in the signal, which leads to

$$Fs \geq 2 * 47.259 = 94.518 \tag{5.12}$$

Rounding up gives us minimal sampling frequency 100 Hz, or sampling period 10 ms. resulting with $2B$ sample in throughput μ

$$\mu = F_s * |\text{sample}| * 8\frac{bit}{byte} = 100 * 2 * 8 = 1.6\ kbps \tag{5.13}$$

MCU Program

The NodeMCU is programmed to sample data with fixed sampling frequency and form the sequential segments out of it. A N-sized segment is then sent away using ESP8266 present at the module. A multicast technology is used, enabling multiple servers to work over the data concurrently and also ease of initialization of communication, where the channel is predefined, so no mutual IP address is needed. The sending frequency can be derived from sampling frequency F_s with formula 5.14.

$$F_{send} = \frac{F_s}{N} \tag{5.14}$$

The program is written in C++ and programming the module is done with Arduino IDE. Several libraries are used that need to be downloaded to the IDE from its repository. ESP8266 libraries for networking (multicast, HTTP) and EEPROM library to operate EEPROM persistent memory.

MCU runs HTTP server, enabling remote configuration of the MCU at the runtime without reprogramming and reset. It is possible to set sampling frequency or period with the altering other correspondingly as same as sending period, frequency, or segment size with a preset implementation limit. The REST API also enables to set the multicast channel, both address and port, turning debugging informations on/off to serial line.

REST API of MCU HTTP server

The MCU runs a REST API, that can configure it. The API includes following options:

Resource	Description
`/`	Welcome page.
`/config`	Configuration of module.
`/config/mcast`	Configuration of multicast channel.
`/config/sample`	Configuration of sampling.
`/config/send`	Configuration of sending.
`/log`	Logs.

Table 5.2: REST API resources overview.

`/config/mcast` Configures the multicast channel.

`enabled` Enables/disables sending. Defaultly enabled.

`address` Sets multicast channel address. Defaultly `224.0.0.1`.

`port` Sets multicast channel port. Defaultly `1234`.

`/config/sample` Configures the sampling.

Exactly one of the parameters must be present (ignored otherwise):

`frequency` Sets sampling frequency, subsequently changing sampling period (5.15) and segment size (5.16). Defaultly 100.

`period` Sets sampling period, subsequently changing sampling frequency (5.15) and segment size (5.16). Defaultly 0.01.

`/config/send` Configures the sending.

Exactly one of the parameters must be present (ignored otherwise):

`frequency` Sets sampling frequency, subsequently changing sampling period (5.15) and segment size (5.16). Defaultly 0.5.

`period` Sets sampling period, subsequently changing sampling frequency (5.15) and segment size (5.16). Defaultly 2.

`N` Sets segment size, subsequently changing sampling frequency and period (5.15). Defaultly 200.

The segment size must lay between 10 and 1024. If set value causes invalid value of segment size, request will be ignored. The conversion to segment size N is shown in the formula 5.16.

$$f = T^{-1} \tag{5.15a}$$

$$T = f^{-1} \tag{5.15b}$$

$$N = T_{send} T_s^{-1} \tag{5.16a}$$

$$N = f_s f_{send}^{-1} \tag{5.16b}$$

$$N = f_s T_{send} \tag{5.16c}$$

$$N = (T_s f_{send})^{-1} \tag{5.16d}$$

`/log` Shows last events of the sensor in a form of logs.

5.2 Classification server

Monitor is the classification server, that collects data from the sensors and performs classification algorithms described in 3.2 with it.

The main software design is based on MVC (model-view-controller). The model reads the data from file (offline), or right from serial line or multicast channel (online, realtime) and performs a classification over them. The view is presenting the output of the model calculation to the user.

Data sources

Monitor can read data from serial port, multicast channel or replay saved data from a file. This feature is done using class `Reader` from the module `comm` and inherited classes in modules `comm_serial`, `comm_mcast`, `comm_replay`.

Class Reader corresponds with the design pattern Singleton, holding one single instance for each serial port, multicast channel and file to avoid opening multiple handles and potential data race.

Instantiation of the object is done in the first demand for it in the call of static method `getReader()`. Each object then possesses separate reading thread, that ensures updating of the data. Getting the last received segment is through the method `getSegment()`, a thread lock is used. The whole design is shown in the figure 5.4.

The data source objects, inherited from `Reader` are assigned to the objects in view part. These graphical objects hold reference to the `getSegment()` method of linked object in model part for access to the updated data and to present them to the user.

The example of reading data using the `Reader` interface and its inherited classes is shown at the listings 5.5 (offline) and 5.6 and 5.7 (online). When the data are read offline from file, the data are read as one chunk per file and processed together, the online reading is performed in loop waiting for chunk to come from the sensor module.

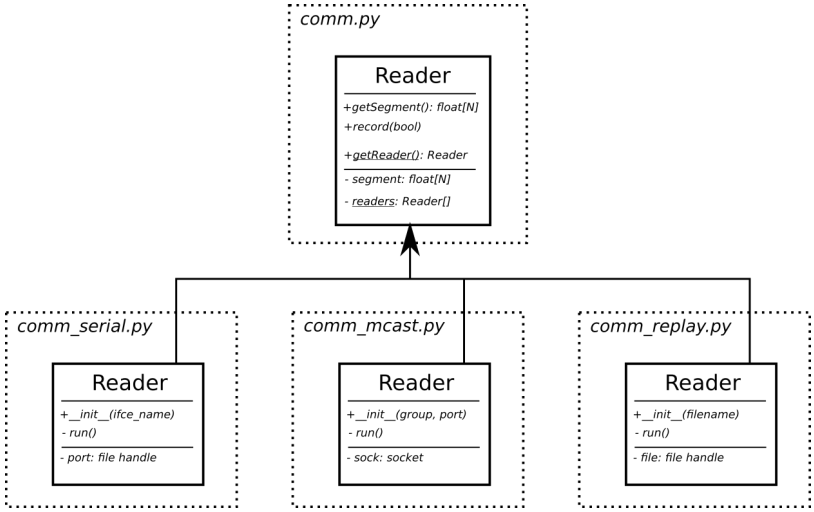

Figure 5.4: UML diagram of Reader classes.

```
1 import comm_replay
2 # ...
3 x = comm_replay.Reader.readFile(path+'/'+filename+'.csv')
4 # ...
```

Figure 5.5: Reading file using `comm_replay.py` module.

```
1 import comm_serial
2 # ...
3 reader = comm_serial.Reader.getReader('/dev/ttyS0')
4 while True:
5   x = reader.getSegment()
6   # ...
```

Figure 5.6: Reading serial file using `comm_serial.py` module.

```
import comm_mcast
# ...
reader = comm_mcast.Reader.getReader('224.0.0.1',1234)
while True:
  x = reader.getSegment()
  # ...
```

Figure 5.7: Reading multicast channel using `comm_mcast.py` module.

Classification model

The segmentation and parsing into artefacts, which cover the feature extraction, is held by the module `segment.py` with class diagram 5.8. The `parseArtefacts()` in this case is a factory for artefact vector. In implementation it uses both `Segment` and `Edge`, these classes can be even used separately, but the recommended way how to perform the feature extraction using this module is presented at the listing 5.9.

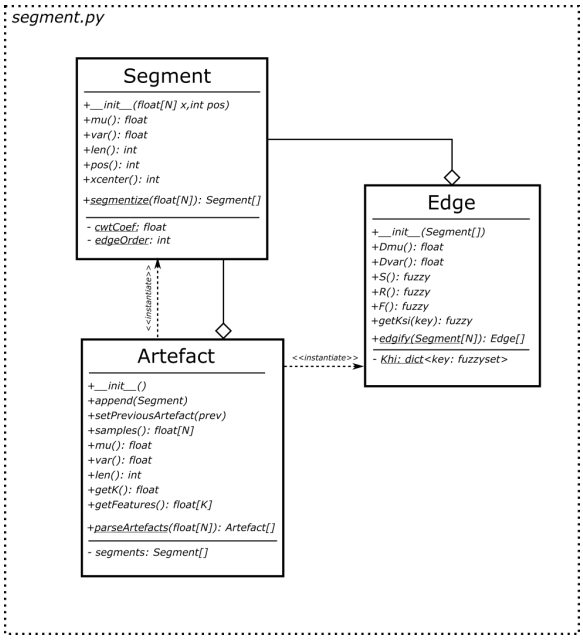

Figure 5.8: UML diagram of Segment classes.

```
import segment
# ...
artefactsVector = segment.Artefact.parseArtefacts(x)
featuresVectorVector = [a.getFeatures() for a in artefactsVector]
# ...
```

Figure 5.9: Feature extraction using `segment.py` module.

The each of the feature vectors is then separate input to the classifier, represented by the module `model.py`. The recommended usage is shown at the listing 5.10

```
1 import model
2 # ...
3 classifiers = model.Classification.getTrained()
4 for featuresVector in featuresVectorVector:
5   areaMatrix = classifiers.evaluate()
6   # ...
```

Figure 5.10: Classification using `model.py` module.

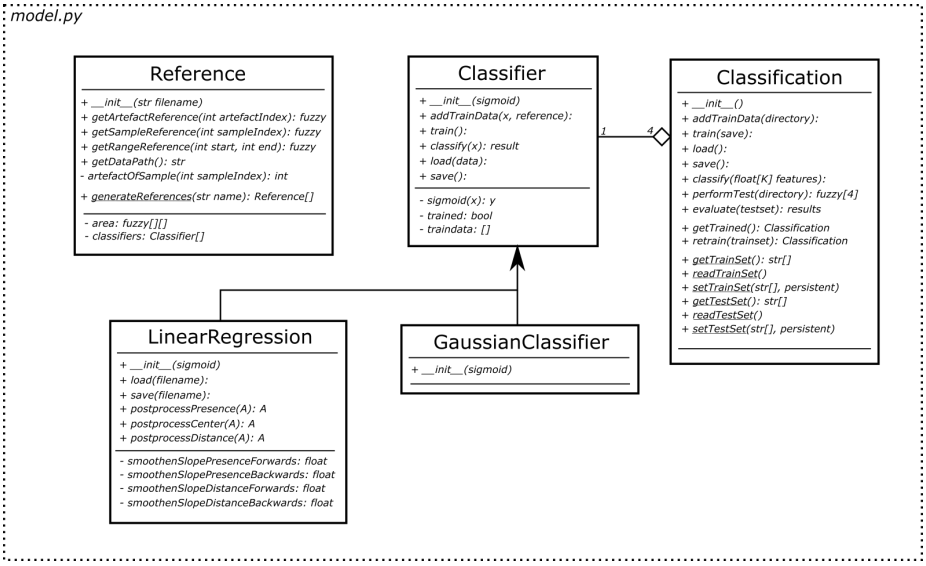

Figure 5.11: UML diagram of Model classes.

The output of each classifier was postprocessed in order to maximize the possible score. The evaluation metrics is described in the section 6.1. The postprocessing is doing over each received chunk separately.

Operation, that is often used is using single smoothening filter with a memory, defined formally by formula 5.17, filtering input artefact vector a to output $f(a)$, using inner memory vector m and saturation function $S(x)$

p is smoothening parameter with optimal value approximately $p \in (0, -1\rangle$, where 0 means no influence and -1 means absolute influence of previous artefact.

$$S(x) = \begin{cases} 1 & x > 1 \\ 0 & x < 0 \\ x & \text{otherwise} \end{cases} \tag{5.17a}$$

$$m[0] = 0 \tag{5.17b}$$

$$f_p(A) = \forall i \in \langle 0; |A|) : f(a[i]) = S(A[i]+m[i]), m[i+1] = S(S(A[i]+m[i])+|A[i]|*p) \tag{5.17c}$$

To find the best fitting parameter p value, equation 5.17 can be altered in order to run the optimalization by maximizing conjunction with vector of reference keys κ_X for each trait over the whole train set $\forall A$, as shown in the formula 5.18. This operation is introduced later in the table 6.1.

$$argmax_p(\sum_{\forall A} f_p(A) \cdot \kappa_X(A)) * 100\% \tag{5.18}$$

Chapter 6

Experiments

6.1 Evaluation

A several metrics can be used to evaluate the success rate of decision. In following equations, A stands for tuple of $|A|$ artefacts being evaluated, $A[i]$ for an artefact from this set. For each artefact from testing set, a label $\kappa_X(A[i])$ for all traits is known.

For labeling purposes, a variation of Lukasiewicz logic was used to cover the states, which Bool logic is not able to handle, as the position of person in case area is empty. The operation in the table 6.1 is represents comparing the computed score and the real label returning a score of success of the decision.

Input conditions for fuzzy number $a \in \langle 0; 1\rangle$ and $\kappa \in \{1, 0, N\}$, which is in *json* file format represented by *True*, *False* and *None* literals respectively. The output is a success rate, multiplied by 100 can be converted to percentage.

To emulate the decision by tresholding a fuzzy sigmoid can be applied to a, mapping fuzzy values to fuzzy values, which are more decisive. An example is shown at the equation 6.1. The usage and choice of such function distorts results significantly.

$$\sigma(a) = \begin{cases} 1 & a \geq 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

The metrics used and described below use also notation described by the equations 6.2. These tuples contains only artefacts, that are positive for the trait X (A^1_X), or negative (A^0_X).

$$A^1_X = (\forall a \in A : \kappa_X(a) = 1) \tag{6.2a}$$

$$A^0_X = (\forall a \in A \kappa_X(a) = 0) \tag{6.2b}$$

a	κ	$a \cdot \kappa$
a	1	a
a	0	$N(a)$
a	N	*excluded*

Table 6.1: Classifier output comparing to reference.

First possible metric is a simple average of artefacts with its scores by the formula 6.3 for trait X. Similarly can be counted separately success rate for A_X^1 and A_X^0.

$$\overline{A_X} = \frac{1}{|A|} \sum_{a \in A} a_X \cdot \kappa_X(a) \tag{6.3}$$

The metric 6.3 does not take account of artefact length as a weight. To do so, alternation 6.4 with the weight added is suggested.

$$\overline{A_X}' = \frac{1}{\sum_A |a|} \sum_{a \in A} (a_X \cdot \kappa_X(a))|a| \tag{6.4}$$

6.2 Smoothing parameters optimalization

Optimalization of parameters in postprocessing smoothing function was performed according to the 5.18. The parameters for each trait were computed, as shown by the table 6.2.

Direction	**Presence**	**Distance***	**Center**	**Left**
Forwards	−0.085	−0.001	−0.005	N
Backwards	−0.0025	−0.1	−0.01	N

Table 6.2: Experimentally computed smoothening parameters values.

6.3 Results

The table 6.3 includes certainty of decision to presence or absence of the trait, equal to $100\overline{A_X}'$. In the case of presence, positive means presence of body in area, negative means absence, for distance it is body being further than 4.5 m when positive, closer when negative. Center is positive for body being in front of the sensor, negative for body on the side.

Since the training is done by numerical solving of gradient descent, every training results in different result. To have more reflective results, accuracy was measured as mean of results of 10 successive trainings and evaluating of the test data set for each trait.

The parts of data set (4m_* and 5m_*) were not used due to the lack of other more distinct data especially in the distance trait (for example 8m_*) which significantly distorts the negative rate.

Aspect	**Presence**	**Distance**	**Center**	**Left**
Positive rate	75.491	74.266	61.866	52.540
Negative rate	87.083	70.681	52.823	53.451
Total rate	83.743			

Table 6.3: Resulting accurracy on testing data.

Reading the table 6.3 as probability would be miscomprehension, to achieve the posterior probability for these traits, the outputs must be converted from fuzzy to bool logic (e.g.

tresholding) and the formula 6.5 to compute the resulting posterior probability is used over them. $T(x) \to 0, 1$ is tresholding function mapping fuzzy values to bool values with treshold 0.5.

$$p(\kappa \in 0, 1|A)_X = \frac{1}{\sum_{A^\kappa} |a|} \sum_{a \in A^\kappa} (T(a_X))|a| \tag{6.5}$$

Aspect	Presence	Distance	Center	Left
Positive rate	75.972	75.785	63.725	
Negative rate	86.542	69.793	53.436	

Table 6.4: Posterior probability on testing data.

The results of posterior probability, shown in the table 6.4, show, that it is definitely possible to use PIR sensors for the purpose of localization of people and possibly even counting them. The results itself in the tables 6.3 and 6.4 are probably not sufficient for a deployment in the real situations. The relatively low results can be caused by multiple aspects.

The most obvious aspect that could influence the accuracy of the classification of the testing data is the method of labeling. The reference medium for labeling were video recordings, synchronized with the recording session in the monitor by a Monitor generating a sound at the beginning of each recording and then synchronisation using the sound track of the video to mark and crop the recording.

Another aspect causing unfavourable result values could be labeling itself. It was done manually, so the human aspect is one of the reasons as same as the fact, that labels were created for each splitted artefact itself. Potential mistakes in segmenting and merging the segments into artefacts could be propagated into the results, even though there was an effort to reduce this threat in class Reference, which performs operations across the artefacts.

Not mentioned, though quite important is a size of dataset. During the whole development, about 200 15 s recordings were done, each then splitted into 10 - 30 artefacts, which each possess a separate label. For the training and testing purposes only 81 recordings were later included, 45 with no movement (empty area) and 36 included movement.

Suggestions for improvement

As mentioned before, the results are not very positive for the deployment in the real situation, e.g. controlling of objects etc. With further improvements and reducing of the mentioned risks it could be a very good and promising principle for certain fields, as physical security, internet of things and others.

The improvement is necessary to reach it. Firstly, more data and different movement types must be recorded in order to improve both the accuracy and possibility to use the sensor in general. The possible way could be changing not only the orientations and trajectories of the movement, but also the speed, different surroundings, sizes of the area, temperature differences in the background (usage in sauna or a room freezers).

The multiple persons present, even though mechanism considered possibility of counting the people, were not tested or measured at all during the measuring.

The speed of the processing program is not breakneck. Possible way could be profiling the program and optimizing the critical parts by rewriting them in compiled language

(C++). Since the program includes a lot of vector algebra, optimalization with moving the computation to GPU / SSE / AVX could be also a solution. During the offline processing, parallelism could be a good approach to increase a speed of processing.

Multiple sensors do not only have to be used in one single room being merged by fusion, but another improvement could be designing a server, monitoring and identification of the people by a network of PIR sensors and cameras. Such a mechanism was already matter of research in several theses, such as [21].

Chapter 7

User Interface

Monitor as a classification server is endowed with simple GUI for the needs of monitoring the signal real-time and recording of the data sets used for training, testing and evaluating the results. The interface is shown in the figure 7.1.

Since the programming language used for the Monitor was Python3 due to many amazing libraries focused on the machine learning and classification (numpy, scipy, matplotlib), for GUI was chosen a library *TKinter* – as a part of standard Python3 library is always present whenever Python3 is, the usage is pretty simple and it is easy to connect with matplotlib, graph plotting library. The whole GUI is still designed as a prototype, for the system deployed to the real situations it would be probably too slow and not flexible enough. As an alternative, Qt might be considered.

Structure of the view

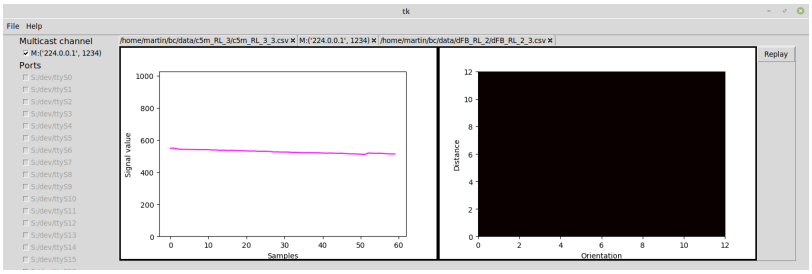

Figure 7.1: Monitor GUI prototype with sample data.

The classical menu bar with dropdowns appears in the top of the window. Dropdown *File* contains option to switch to the record mode over the source in the active tab and opening the tab with a source from file. The side menu contains possible sources to read: multicast channels or serial ports. Checking opens a new tab with a real-time view of the signal of the source as well as the area view, classified by the system.

In order to record a signal into a file, a recording session needs to be created and arranged. The session is made from recordings. The lengths of recordings and time delays before each recording are adjustable. This enables recording session to fulfil the needs for any kind of movement to be made. To work properly with the program, the names of the exercises must match the name of the session and they must have appendixes _1, _2, ... An example is shown in the figure 7.2.

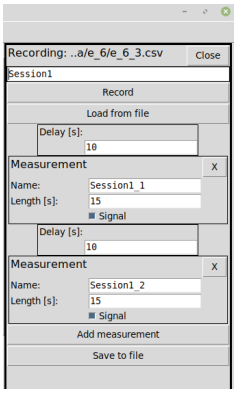

Figure 7.2: Monitor GUI recording session dialog.

Structure

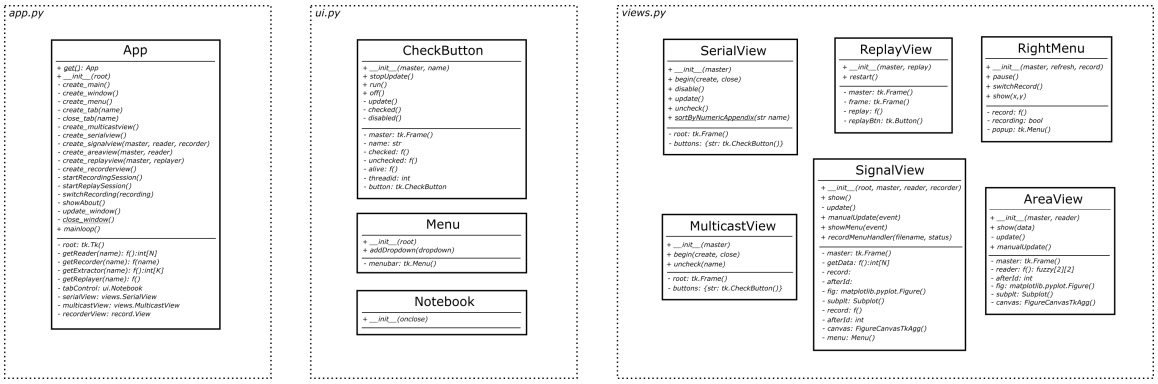

Figure 7.3: Class diagram of view part.

The whole Monitor is made as model-view-controller pattern. The view consists of module *app.py* with the class App controlling the processes inside of the view. Two more modules are present, *ui.py* and *views.py*.

The *ui.py* is a adapter over several TKinter classes with slight extension of its functionality according to the needs of App class. It contains CheckButton connected with a data source, which repeats testing the connected source availability and enables or disables itself by that. Notebook extends *TKinter.ttk.Notebook* with a simple graphical widget, Menu represents the menubar.

The whole graphical design consists of multiple parts with various roles, assembled together in one window. For scalability reasons each of this „views“ is implemented as a separate class inside the module *views.py*. Classes MulticastView and SerialView are used in the side menus with checkbuttons for each data source, multicast channel or serial port. SignalView and AreaView are the main part of the tab system (notebook), visualizing the received segment and the counted area fuzzy matrix. The ReplayView can be seen when a replay is shown as a data source, it contains the button controlling the replaying of the file.

Chapter 8

Conclusion

Nowadays PIR sensors are mostly used to detect a presence of a person. The main aim of the research was to try to use PIR sensors for monitoring the situation in the sensed area and state the count of people. For this matter a system, that would use the data from the PIR sensors and scan the monitor situation, should have been designed. To do so, various classification and recognition algorithms were studied and considered. Suggested solutions by the task were using either predefined fuzzy logic system or artificial learning system. One of the tasks was also to implement suggested algorithm and verify its functionality on real situations.

The issues related to the topic, PIR sensor and MCU as for hardware and classification algorithms and fuzzy logic in particular, were studied in detail. To understand the area even more thoroughly, a number of theses and articles were read. The approach of evaluating the room occupancy using infrared radiation is still new and seems very promising.

Acquired data enabled designing a theoretical system based on fuzzy logic and linear regression for classification itself. The system could be used not only to state either presence or absence of a person in front of the sensor, but also his position and possibly even the count of people, if multiple are present. The design also suggests a way how to use multiple sensors.

The implementation of the design includes the main processing pipeline for one sensor, where the signal can be either pre-recorded and read from a file, or sensed in real time and transferred by serial port, or by multicast group. For monitoring and debugging purposes, a prototype of GUI was made, showing the real time signal and the processing output.

The PCB of a possible product, a sensing device, and the look of it, was suggested as well. A prototype of PCB was drilled.

The classifiers were trained with supervisor. To do so, a train and test data set were recorded and labelled with simple labeling method, and experiments were done over it. The specified formal points of the task were all fulfilled.

The designed system considers much wider usage, than was implemented, as multiple-sensor usage and the fusion algorithm. Multiple-person situations were not tested. Still the usage of PIR sensors in this application has bright prospects and despite the challenges, that are still in the way, it is definitely worth being focus for onward research.

Bibliography

[1] Arduino: *analogRead()*. Arduino Reference. February 2019. [Online; visited 29.3.2019].
Retrieved from: https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/

[2] Caltech: *Herschel Discovers Infrared Light*. IPAC Caltech. [Online; visited 14.12.2018].
Retrieved from: http://coolcosmos.ipac.caltech.edu/cosmic_classroom/classroom_activities/herschel_bio.html

[3] Colliard-Piraud, S.: *Signal conditioning for pyroelectric passive infrared (PIR) sensors*. AllAboutCircuits. July 2016. [Online; visited 18.2.2019].
Retrieved from: https://www.allaboutcircuits.com/industry-articles/signal-conditioning-for-pyroelectric-passive-infrared-sensors/

[4] Cox, E.: *Optics of the Human Eye*. carolina.com. December 2016. [Online; visited 23.12.2018].
Retrieved from: https://www.carolina.com/teacher-resources/Interactive/optics-of-the-human-eye/tr39501.tr

[5] Cyc: *SVM Separating Hyperplanes*. Wikipedia. February 2008. [Online; visited 23.12.2018].
Retrieved from: https://commons.wikimedia.org/wiki/File:Svm_separating_hyperplanes.png

[6] Duncan, R.: *Automatic doorway*. PublicDomainFiles. December 2018. [Online; visited 14.12.2018].
Retrieved from: http://www.publicdomainfiles.com/show_file.php?id=13392932418863

[7] E.O. Gracheva, N. I.: *Molecular basis of infrared detection by snakes*. *Nature*. 2010. doi:10.3410/f.2579956.2234055.

[8] Ess, D. V.: *Pyroelectric Infrared Motion Detector*. [Online; visited 15.12.2018].
Retrieved from: https://cdn-learn.adafruit.com/assets/assets/000/010/138/original/an2105.pdf

[9] GmbH, B. T.-T.: *Operational manual: Passive infrared motion sensor PIR STD*. bb-sensors.com. August 2015. [Online; visited 25.3.2019].
Retrieved from: https://shop.bb-sensors.com/out/media/Operation_manual_PIR-STD_passive_infrared_motion_detector_new_1.pdf

[10] Gregory D. Cramer, S. A. D.: *Clinical Anatomy of the Spine, Spinal Cord, and Ans.* Elsevier. 2014. ISBN 978-0-323-07954-9.

[11] Hana Řezanková, V. S., Dušan Húsek: *Shluková analýza dat.* Professional Publishing. 2009. ISBN 978-80-86946-87-8.

[12] Hawer, J.: *Thermal regulation graph.* Wikipedia. May 2018. [Online; visited 15.12.2018].
Retrieved from:
https://commons.wikimedia.org/wiki/File:Thermal_Regulation_Graph.svg

[13] Hyun Hee Kim, S. L., Kyoung Nam Ha: *Resident Location-Recognition Algorithm Using a Bayesan Classifier in the PIR Sensor-Based Indoor Location-Aware System.* 2009. doi:10.1109/tsmcc.2008.2008099. [Online; visited 22.12.2018].

[14] Quantities and units – Part 7: Light. Standard. International Organization for Standardization. Geneva, CH. November 2008.

[15] J. Brendan Ritchie, P. C.: *The bodily senses.* Oxford University Press. 2015. ISBN 0-19-960047-3.

[16] Jian-Shuen Fang, D. J. B., Qi Hao: *Path-dependent human identification using a pyroelectric infrared sensor and Fresnel lens arrays.* Optics express. February 2006. doi:10.1364/OPEX.14.000609. [Online; visited 23.12.2018].
Retrieved from: https://www.researchgate.net/figure/Black-body-radiation-curve-of-human-body-at-37-o-C_fig1_26273296

[17] Kreidle, M.: *Měření teploty - senzory a měřící obvody.* BEN - technická literatura. 2005. ISBN 80-7300-145-4.

[18] Lysenko, V.: *Detektory pro bezdotykové měření teplot.* BEN - technická literatura. 2005. ISBN 80-7300-180-2.

[19] Mallat, S.: *A Wavelet Tour of Signal Processing.* Academic Press. 2009. ISBN 978-0-12-374370-1.

[20] NASA: *The Electromagnetic Spectrum.* NASA webpage. March 2013. [Online; visited 14.12.2018].
Retrieved from:
https://imagine.gsfc.nasa.gov/science/toolbox/emspectrum2.html

[21] Ogollah, O. F.: *Room occupancy monitoring and counting device.* April 2015. final year project.
Retrieved from:
http://eie.uonbi.ac.ke/sites/default/files/cae/engineering/eie/ROOM%20OCCUPANCY%20MONITORING%20AND%20COUNTING%20DEVICE.pdf

[22] Parker, M.: *Digital Signal Processing 101.* Newnes. 2010. ISBN 978-1-85617-921-8.

[23] Půlpán, Z.: *Odhad informace z dat vágní povahy.* Academia. 2012. ISBN 978-80-200-2076-5.

[24] Raes, G.-W.: *Infrared Body Sensing*. Logos Foundation. October 2007. [Online; visited 5.1.2018].
Retrieved from: https://www.logosfoundation.org/ii/infrared_sensing.html

[25] Smith, S.: *What is Infrared Radiation?* YouTube. [Online; visited 14.12.2018].
Retrieved from: https://www.youtube.com/watch?v=KwvXcUe10OQ

[26] Tarun Choubisa, S. B. M., Mohan Kashyap: *Comparing chirplet based classification with alternate feature-extraction approaches for outdoor intrusion detection using a pir sensor platform.* 2011. doi:10.1109/CISP.2017.8125869. [Online; visited 22.12.2018].

[27] Wang, L.: *Human infrared signal recognition using single PIR detector*. 2017. doi:10.1109/ICACCI.2017.8125869. [Online; visited 22.12.2018].

[28] WikiLectures: *Light, eye and vision.* WikiLectures. [Online; visited 14.12.2018].
Retrieved from: https://www.wikilectures.eu/w/LIGHT,_EYE_AND_VISION

Appendix A

PIR signal recording

The movements that data shows were prepared in several directions and the type of movement can be read from the figure captions.

The trajectories are concentred circles with sensor in center and radius values 3 m, 6 m, 9 m and 12 m. The direction is either left-to-right (LR) or right-to-left (RL). The workplace is shown in the figure A.1. [1] The movement was recorded multiple times.

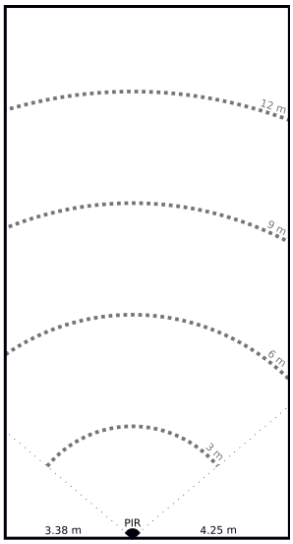

Figure A.1: Measurement workplace.

Other cases are recorded as well: person walking towards (*C_BF*) or walking away (*C_FB*) from the sensor and no movement (*E*).

[1] For example *6m_RL* is movement 6 m from sensor right-to-left.

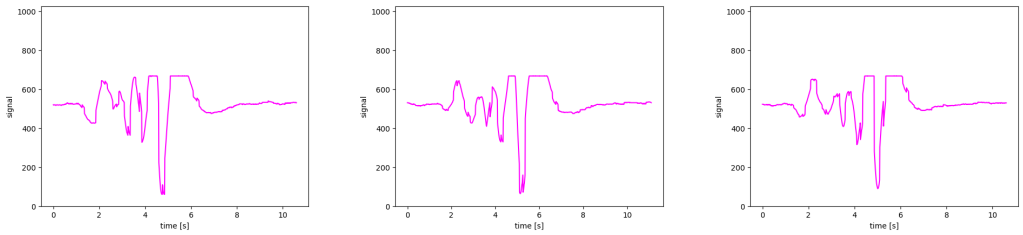

Figure A.2: c4m_LR_1.

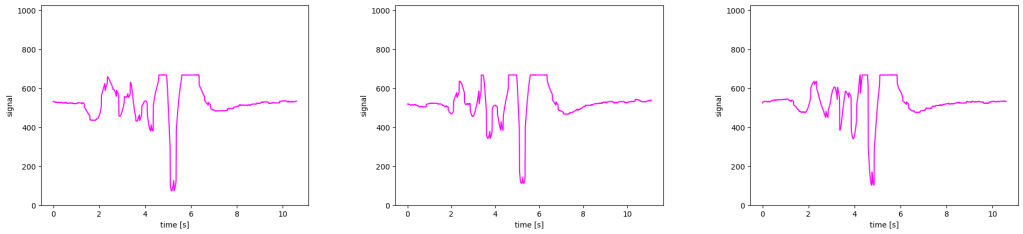

Figure A.3: c4m_LR_2.

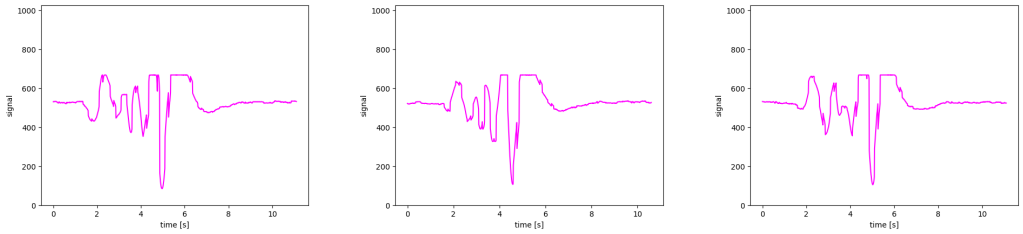

Figure A.4: c4m_LR_3.

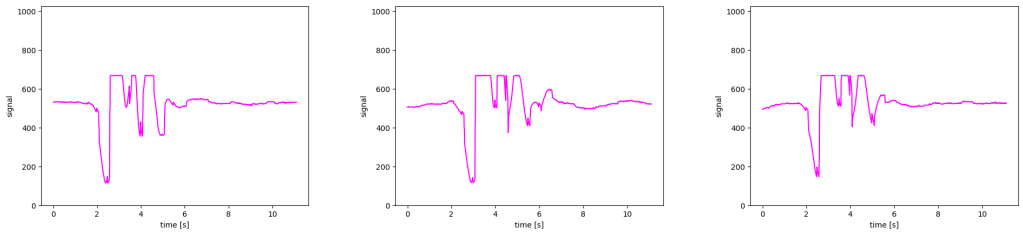

Figure A.5: c4m_RL_1.

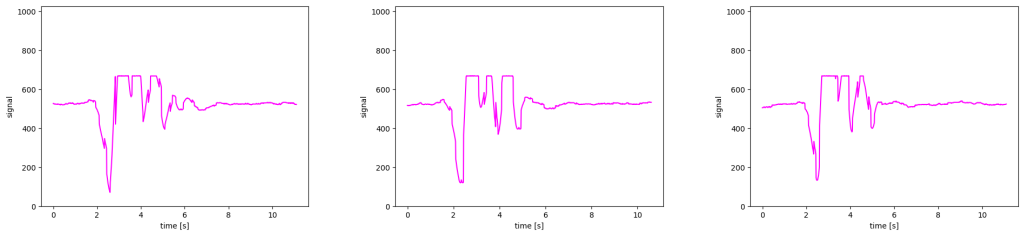

Figure A.6: c4m_RL_2.

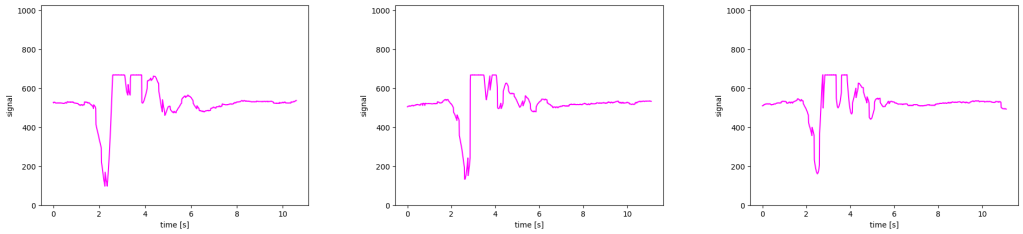

Figure A.7: c4m_RL_3.

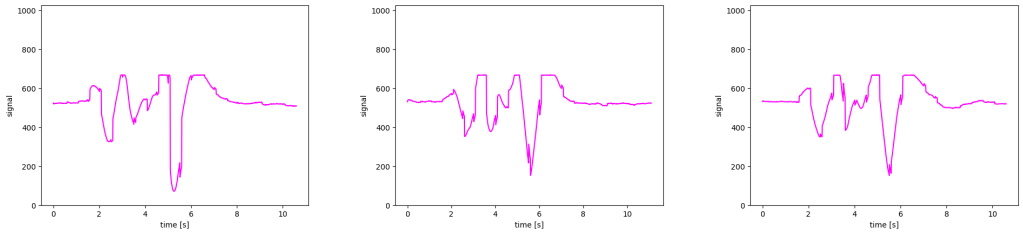

Figure A.8: c5m_LR_1.

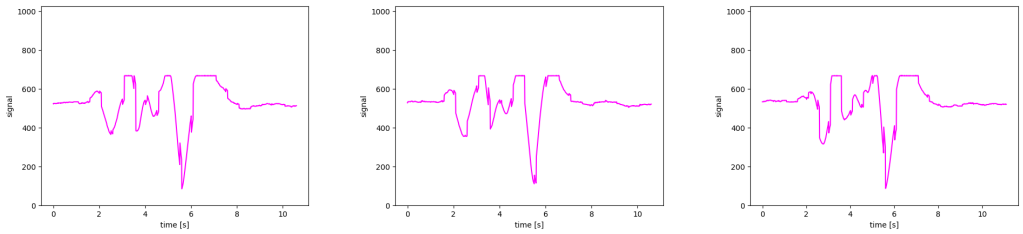

Figure A.9: c5m_LR_2.

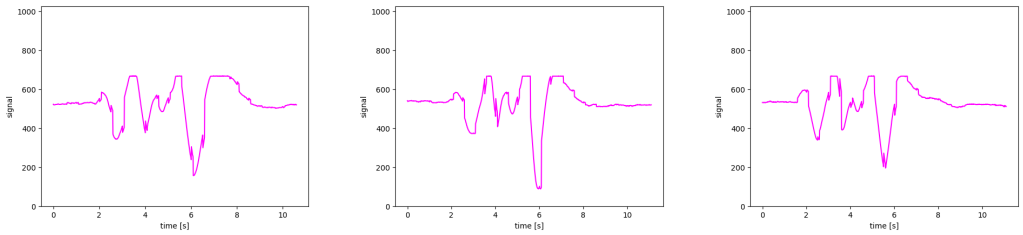

Figure A.10: c5m_LR_3.

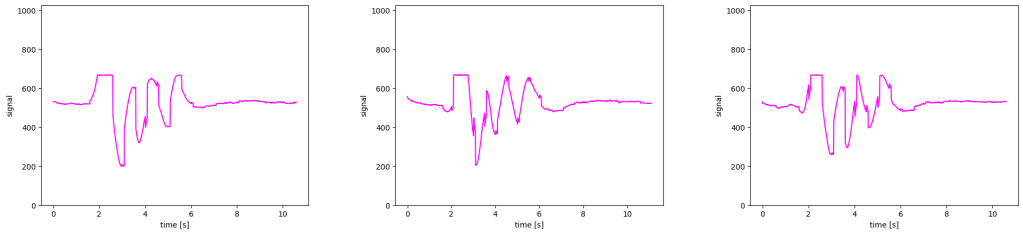

Figure A.11: c5m_RL_1.

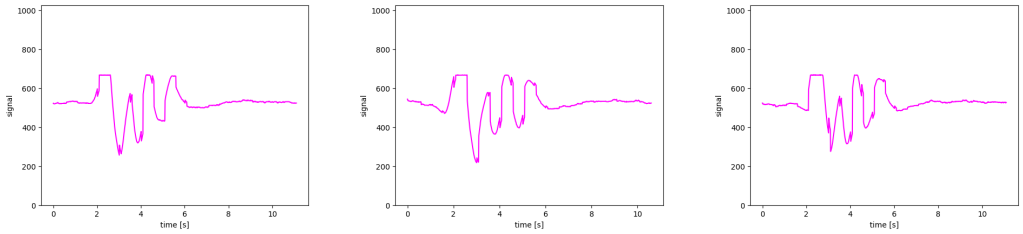

Figure A.12: c5m_RL_2.

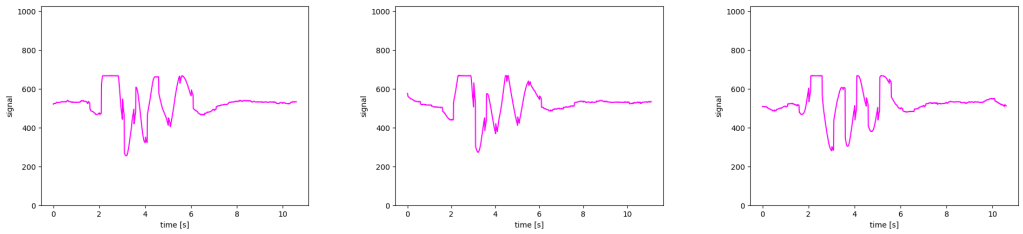

Figure A.13: c5m_RL_3.

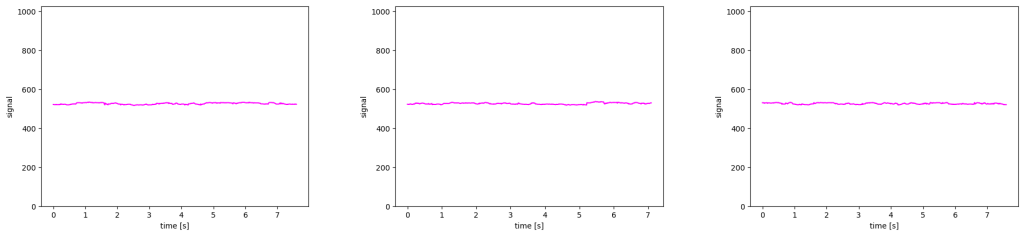

Figure A.14: e_1.

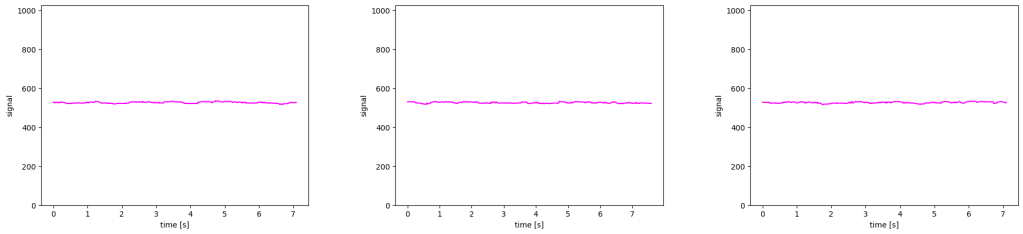

Figure A.15: e_2.

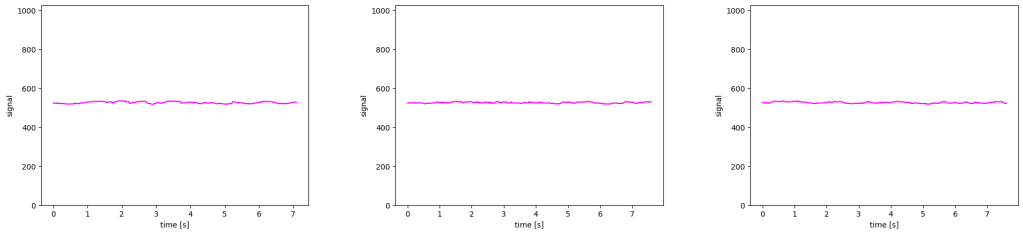

Figure A.16: e_3.

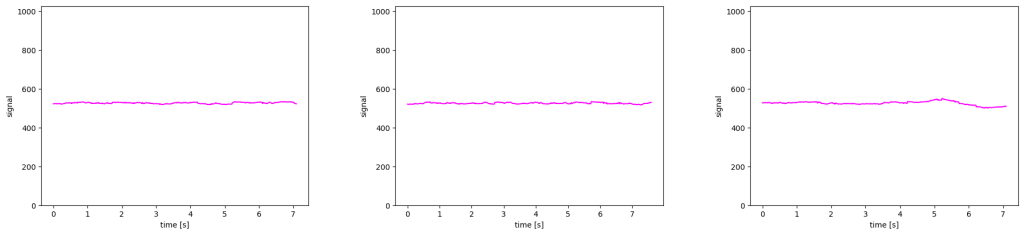

Figure A.17: e_4.

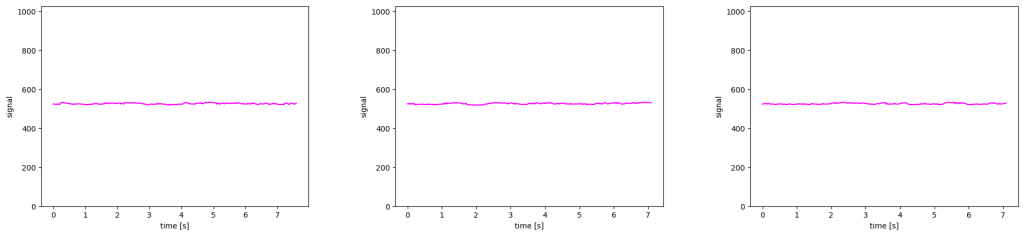

Figure A.18: e_5.

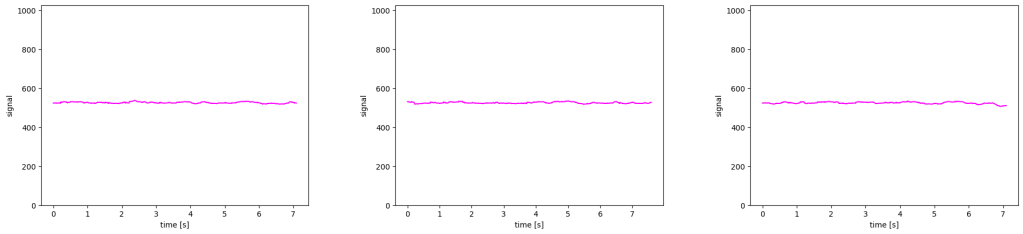

Figure A.19: e_6.

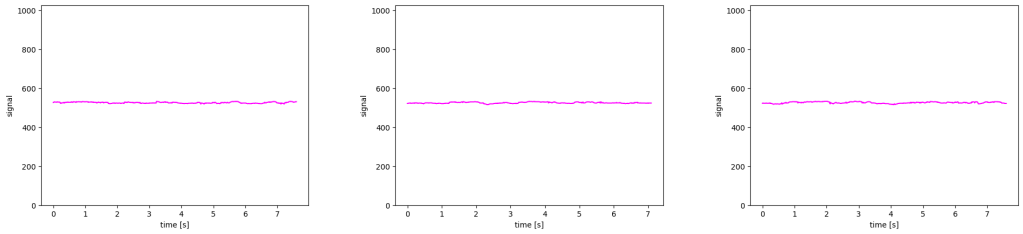

Figure A.20: e_7.

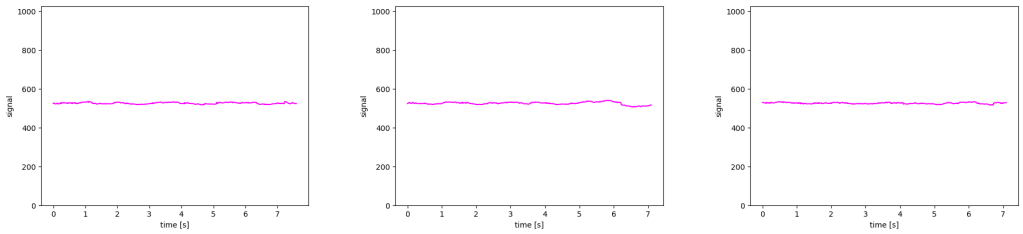

Figure A.21: e_8.