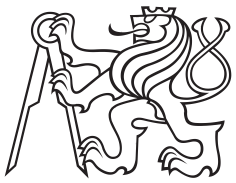
Bachelor's Thesis



Czech Technical University in Prague

F3

Faculty of Electrical Engineering

Department of Cybernetics

Intelligent Building Control by Hand Gestures

Adam Uhlíř

Open Informatics

2014

Supervisor: Ing. Jaromír Doležal Ph.D.

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Adam U h l í ř

Studijní program: Otevřená informatika (bakalářský)

Obor: Informatika a počítačové vědy

Název tématu: Řízení inteligentní instalace budov pomocí gest

Pokyny pro vypracování:

1. Seznamte se se standardem řízení inteligentní instalace KNX.
2. Vyberte vhodné mobilní zařízení pro detekci gest.
3. Navrhněte způsob řízení vybraných prvků a funkcí inteligentní instalace pomocí gest.
4. Implementujte systém a jeho funkci ověřte na instalaci KNX v prostorách Centra Asistivních technologií.

Seznam odborné literatury:

[1] Harper Richard, ed: Inside the smart home. London: Springer, 2003. 264 s. ISBN 1-85233-688-9
[2] The KNX standart of intelligent building conrol [online]. http://www.knx.org
[3] Myo - Gesture control armband by Thalmic Labs [online] https://www.thalmic.com/en/myo/

Vedoucí bakalářské práce: Ing. Jaromír Doležal, Ph.D.

Platnost zadání: do konce zimního semestru 2015/2016

L.S.

doc. Dr. Ing. Jan Kybic
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 8. 7. 2014

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

BACHELOR PROJECT ASSIGNMENT

Student: Adam U h l í ř

Study programme: Open Informatics

Specialisation: Computer and Information Science

Title of Bachelor Project: Intelligent Building Control by Hand Gestures

Guidelines:

1. Get familiar with the KNX standard of intelligent building control.
2. Choose a suitable mobile device for hand gestures detection.
3. Design a system for control of selected intelligent installation elements by using hand gestures.
4. Implement the system and verify its function at the installation at the Center of Assistive Technology.

Bibliography/Sources:

[1] Harper Richard, ed: Inside the smart home. London: Springer, 2003. 264 s. ISBN 1-85233-688-9
[2] The KNX standart of intelligent building conrol [online]. http://www.knx.org
[3] Myo - Gesture control armband by Thalmic Labs [online] https://www.thalmic.com/en/myo/

Bachelor Project Supervisor: Ing. Jaromír Doležal, Ph.D.

Valid until: the end of the winter semester of academic year 2015/2016

L.S.

doc. Dr. Ing. Jan Kybic
Head of Department

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, July 8, 2014

Acknowledgement

I would like to thank my parents for constant support and believe in my dreams. Also I would like to thank my supervisor Ing. Jaromir Doležal Ph.D. for his guidance, advices and patient. Moreover I would like to thank Bc. Martin Cihlář for his help with his RTLS system. And in the end I would like to thank RNDr. Petr Olšák for creating this thesis TeX template.

Declaration

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 vysokoškolských závěrečných prací.

V Praze dne 6.1.2015

.....

Abstrakt / Abstract

V blízké budoucnosti můžeme očekávat prostoupení inteligentních technologií do způsobu bydlení, takzvané Smart Homes. Tato bakalářská práce se zaměřuje na zvýšení komfortu ovládání Smart Homes pomocí ovládání gesty. Navrhl jsem a implementoval systém na platformě Android, který využívá lokalizaci uživatele v bytě a náramek Myo pro detekci gest.

Klíčová slova: Inteligentní dům, KNX, náramek Myo, gesta.

Překlad titulu: Řízení inteligentní instalace budov pomocí gest

In near future we can expect usage of more advanced technologies in our houses. My thesis focus on increasing comfort of controlling KNX powered Smart Homes with hand gesture control. I designed and implemented system on Android platform, which use Myo armband for hand gesture recognition, in conjunction with integrated indoor localization.

Keywords: Smart Home, KNX, Myo armband, gestures.

Contents /

/ Figures

Chapter 1
Introduction

In our lives we are becoming more and more dependent on the everyday use of technology. Many people use computers for work, entertainment, learning and much more. Smartphones are an essential part of our lives and the use of intelligent systems is growing in every field of human work.

In the near future, a new era is predicted, many devices will be smart (eq. small computers) and they will be connected into networks or to the Internet. They will communicate and share data with each other. This will provide us more comfort and support in our work. The era is commonly called „Internet of Things“.

One of the areas of expected rapid growth is Smart Homes. They are apartments or houses which use intelligent platforms to increase resource efficiency use and to increase the inhabitant's comfort. With more smart electronics connected to this platform, the possibilities will increase enormously. But to expand the common use of Smart Homes, a natural way of controlling the platform, needs to be found.

The common user–controlled interfaces used at present are switches, smartphones, tablets or control panels. These interfaces fulfill their purpose, but with new technologies coming to the market, new options are created and therefore evolution can happen. The most natural way of controlling our surroundings could be through our thoughts, but this control interface is still in early development stage and therefore, so far, not usable. But there is one user interface which is becoming more and more popular, it already has reached a high level of advancement and therefore it is usable in real life. It is hand gesture control.

Nowadays new types of technology are being developed and released; for example, wearable electronics, such as Google Glass or smart watches. These devices are expanding the boundaries of our fantasy / reality. There is one device which is definitely pushing the boundaries of fantasy and which caught my attention — the Myo armband. It is brand new device which is capable of recognizing hand gestures and tracking the movement of your arm. When I discovered this device I immediately knew that I want to work with it. I am a big enthusiast of technology and I want to explore new technologies and contribute to developing them so they could be deployed in real life situations and used by many people.

The goal of this thesis is to research ways that gestures could be implemented for controlling various aspects of a Smart Home. Firstly, general information about gestures and KNX Smart Homes will be presented and then a system for hand gesture control will be designed and implemented using the Myo armband and the KNX platform. Also the mapping of gestures to Smart Home functions will be created.

When I started to work for this thesis Myo armband was still in development. The system which will be implemented will be the first one which will utilize this Myo armband for controlling a KNX platform.

Chapter 2
State of the art

2.1 KNX

2.1.1 Overview

KNX is the world wide standard for building management. It is the successor of BatiBUS, EIB and EHS. It was standardized as an International Standard (ISO/IEC 14543-3), a European Standard (CENELEC EN 50090 and CEN EN 13321-1) and as a Chinese Standard (GB/T 20965) [1]. It is a decentralized network which uses the bus concept for creating networks with advanced elements, which communicate with each other. For communication, data telegrams are used. [2]

KNX uses a layer system similar to the OSI model. From 7 OSI layers, only 5 are used. The comparison can be seen in Figure 2.1.

	OSI Model		KNX/EIB
7	Application Layer	7	Application Layer
6	Presentation Layer	6	-
5	Session Layer	5	-
4	Transport Layer	4	Transport Layer
3	Network Layer	3	Network Layer
2	Data Link Layer	2	Data Link Layer
1	Physical Layer	1	Physical Layer

Figure 2.1. Comparation of OSI and KNX layers, [3]

One of the differences from normal flat installation is flexibility. The KNX network can be more easily reconfigured. For example, when a change of a switch target needs to be made, from hallway lights to room lights, no construction is needed, a software reconfiguration will be enough. Moreover, intelligent learning mechanisms can be applied to the KNX network. These mechanisms will observe user behavior and based on discovered patterns it will modify its own behavior. For example when the user is not at home during the day, the heater will lower the temperature and before the user returns the heater will increase the temperature to a desired level. Also the KNX network requires less cabling then normal flat installations. An example of the difference between the KNX installation and a normal flat installation can be seen in Figure 2.2.

2.1.2 Installation

The KNX installation uses a different media. In KNX technology two types of lines are distinguished — the bus line and the power line. The bus line maintains the data flow and it can be said, that the controlling of the Smart Home is done through this

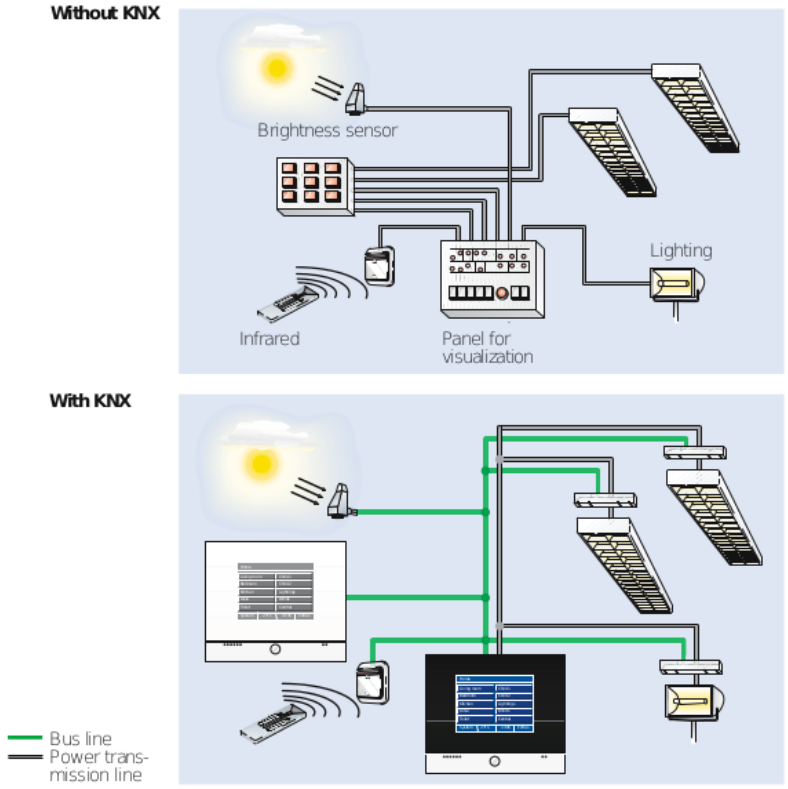


Figure 2.2. Differences between normal and KNX installation, [2]

line. The power line is, on the other hand, supplying energy for element's functions (eq. lights, heating and so on). Some elements do not have high power requirements and therefore they can be powered directly from the bus line.

As mentioned in [3], there are 4 types of media for a bus line — twisted pair, power line, radio and Ethernet.

2.1.2.1 Twisted pair

- TP–0 (type 0) — It posses data throughput 4,8 bit/s and originally is taken from BatiBUS.
- TP–1 (type 1) — It is originally taken from EIB and it posses data throughput 9,6 bit/s.

The most common media for installation is TP–1. It carries 29V DC voltage and allows up to 1000m per physical segment. It also allows usage of line repeaters, which can be used maximally 4 times forming 4 segments in a 4 000m long line.

2.1.2.2 Power line

- PL–110 (110 kHz) — As TP–1, this medium was taken from EIB, the data throughput is 1,2 kbit/s.
- PL–132 (132 kHz) — It is originally taken from EHS and it posses data throughput of 2,4 kbit/s.

2.1.2.3 Radio

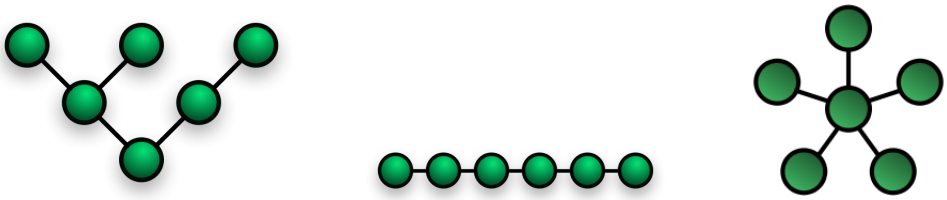
Another medium is radio on frequency 868 MHz which was developed as part of the KNX wireless standard. It can transmit up to 38,4 kbit/s.

2.1.2.4 Ethernet

The KNX standard supports Ethernet media as well. This media can be used together with *KNX over IP* specification. Usually an extra element which acts as a router between the Ethernet and other network media is needed.

2.1.3 Topology

The KNX network is very flexible regarding network topology as stated in [2]. The only constraint is is that this network can not contain loops. Besides that we can use topology models such as tree topology, star topology or line topology — examples of these topologies can be seen on Figure 2.3.



a) b) c)

Figure 2.3. Examples of usable topologies.

The basic unit of KNX network topology is a *line*. Every line needs to have its own power supply. If we consider the most common TP–1 media, 64 elements can be placed on a line. According to [2] there are some limitations for placement of elements on a line.

- The maximum total length of one line without using repeaters can be 1 000 meters.
- The maximum distance between the power supply and the last element must be 350 meters.
- The maximum distance between two elements must be 700 meters.
- The minimum distance between two power supplies must be 200 meters.

If there is a demand for a higher number of elements connected to the network, lines can be connected with a *area line*, which can group together up to 15 lines forming an *area line*. In a network there can be maximally 15 area lines.

Therefore the maximum number of elements connected to the network can be up to 14 400 elements. If further measures would be applied, the maximum number of elements on each line can be extended to up to 255 elements, which would extend the maximum number of elements to 57 375 elements in total (according to [2]).

2.1.4 Addresses

In the KNX technology two types of addresses are used — physical addresses and group addresses.

2.1.4.1 Physical addresses

Every element connected to the KNX network has its own unique physical address. This address is meant for unambiguous addressing of elements inside the network. The physical address is related to the topology of this KNX network, it is 16 bits long and consists of 3 parts — the area address (4 bits), the line address (4 bits) and the position address (8 bits). An example of a physical address can be seen in Figure 2.4.

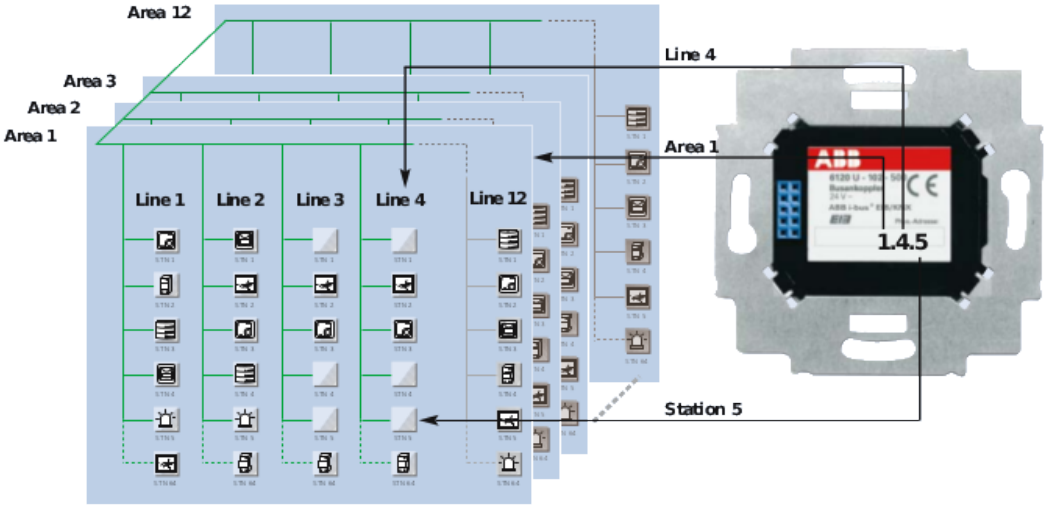


Figure 2.4. Example of physical address, [2]

Physical addresses are not used in normal traffic, they are used for configuration of the KNX network.

2.1.4.2 Group addresses

Group address numbering is based on elements function differentiation. The same as Physical addresses, group addresses are 16 bits long and are divided into 3 groups — main group (4 bits), middle group (3 bits) and lower group (8 bits). The 16th bit is saved as a broadcasting address. When a telegram is sent to a broadcast address, all elements in the network will receive it. The division of elements into groups is not standardized and usually their are logically divided. One example of Group address interpretation could be: main group / middle group / lower group = room / group function (lighting, heating etc.) / position of element ceiling, wall etc.).

2.1.5 Elements

According to [2], elements which are connected to the KNX network can be divided into 4 general groups based on their function.

- System devices — power supplies, communication interfaces (serial interface RS-232, USB, IP), connectors, choke, line and area couplers.
- Sensors — push buttons, transducers (wind, rain, light, heat, etc.), thermostats, analogue inputs, analogue and binary inputs.
- Actuators — switching actuators, dimming actuators, actuators for blinds, heating actuators.
- Controllers — logic units, logic modules.

System devices are elements which build the infrastructure of the network. They can provide bridges to other technologies, for example older standards such as EHS. Moreover, they can extend the network with elements which are able to provide new types of connectivity such as WiFi, Bluetooth and so on.

Sensors report the state of the Smart Home. They are the ones that start actions, because of the outside inputs. They share this data across the network with other elements.

Actuators execute tasks in the Smart Home. They receive data from sensors, from external networks and based on this data, they evaluate their own further actions.

Controllers are elements which make the network intelligent. They process data from the network and make more complex analysis for more complex functions and decisions.

2.1.6 User control interfaces

The most common control interface is taken from a normal home installation. It consists of switches and regulators. These control elements are placed around whole Smart Home. Also these elements can be accompanied with more advanced elements, such as noise listener (for clapping) or motion detectors.

Another quite common control interface of KNX standard is control panels. These panels can consist of touch screens, but there are types of panels which do not have touch screens and have physic control elements. Usually panels are static control devices placed in one place. This interface can be accompanied with remote controls.

Figure 2.5. RTI remote control which can be used for Smart Home control

Another control interface takes advantage of external connectivity of the KNX standard such as *KNX over IP*. These are smartphone and tablets applications which let you have an overview of your home and also control it. This is possible even when the user is not home (in case that KNX is connected to Internet).

2.2 Hand gestures

Defining a hand gesture is not an easy job. It can be perceived on a physical level as the motion of muscles forming the trajectory of a hand. But this perception is lacking context. Usually a gesture is a supporting communication process adding more stress or importance to currently stated information.

2.2.1 Structure of gesture

Every gesture is a dynamic process which has its own structure and therefore it can be divided into several phases. As mentioned in [4] for Human–Computer Interaction (HCI) purposes, we can assume 3 phases of gesture:

- preparation
- stroke
- retraction

In the the preparation phase, the hand is changing position from its resting position to a position where the stroke will start. The stroke is the main phase, where the gesture is actually performed. It is also part of the movement, that expresses the information of the gesture. Then the retraction follows, which either moves the hand into the resting position, or to a starting position for the next gesture.

2.2.2 Type of gestures

To have an overview of what hand gestures are most frequently used for, lets have a look into the taxonomy presented in [4]. This taxonomy was developed for the HCI point of view and you can see its overview in Figure 2.6.

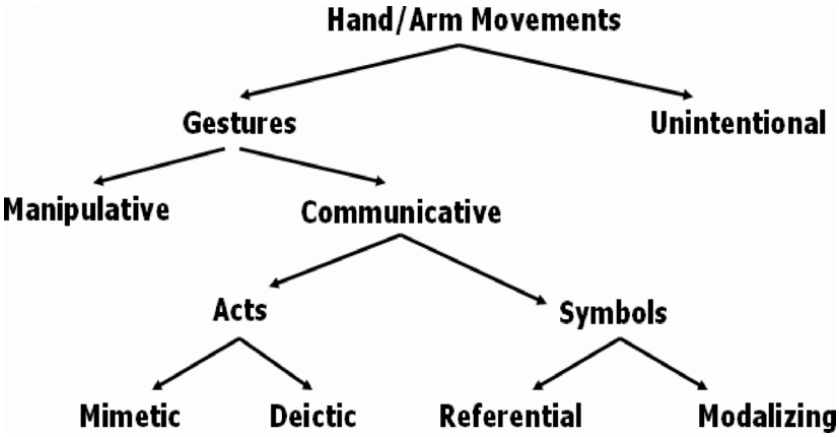


Figure 2.6. Taxonomy of hand gesture, [4]

Generally, hand movements are divided into two categories — unintentional movements and gestures.

Unintentional movements are the ones, which do not contain any information. Gestures are then divided into manipulative and communication gestures. Manipulative gestures are used for manipulating objects in the real word, for example taking a cup of tea and bringing it to the lips. Communication gestures usually accompany speech and have the purpose of communication. These are divided into the last two categories — acts and symbols. Acts are gestures within which information is related directly to the movement. They can be mimetic (imitating some action) or dietic (pointing acts). Symbols, on other hand, are connected to linguistic roles.

2.2.3 Classification of hand gestures

There are several ways to approach the classification of hand gestures. They can be generally divided into two categories — visual and non–visual based on data from which gestures are classified.

2.2.3.1 Visual based classifications

As the name suggests, visual based classification uses image analyses to determine hand position or a hand gesture from images (or footages). This approach evolved tremendously during the past decades, because it utilized a reduction of the size of electronic parts (especially cameras), reducing the price of these parts and at the same time increasing the computational power.

One of the products which uses this approach is the XBox Kinect, developed by Microsoft. According to [5] it uses an infrared camera alongside the usual RGB camera. The infrared camera emits a 640x480 grid of infrared beams from which a 3D vector is constructed. These data are then used for recognizing the human skeleton in pictures.

Another product which uses this approach and also uses technology similar to Kinect is Leap Motion. Unlike Kinect, Leap Motion focuses only on hand recognition, finger tracking, not the whole human body detection. This minimizing of detection radius positively affected the accuracy of classification. The manufacturer states an accuracy of 0.1 cm, but by experiments with real conditions conducted in [6], an accuracy of 0.7 cm was measured. Leap Motion uses several infrared beamers and detectors for recognizing hand and fingers above the device.

Figure 2.7. Image of Kinect camera.

Figure 2.8. Image of Leap Motion.

Technologies which utilize Visual-based classification have already reached a high level of accuracy, which makes them usable in real life. They have, definitely, many benefits. They do not bother the user with peripherals, therefore increasing user comfort. As it was already pointed out, they have a high success rate. But on the other hand they restrict movement of the user, since the place of detection is usually based on the view of a camera(s).

2.2.3.2 Non visual based classifications

The second approach was developed mainly in the previous century. It uses mechanical or other contact sensors which measure hand position and from data of these sensors it classifies hand gestures.

One of the old concepts using this approach is called Data Glove. This concept is presented in [7] and consists of a cotton glove on which flex sensors were attached.

Figure 2.9. Image of Data Glove concept.

Those sensors measured finger angles and hand motion, from the data of these sensors hand gestures and position was calculated. The concept can be seen in Figure 2.9.

A more modern approach was created by the Canadian company called Thalmic Labs. They use EMG sensors for detection of muscle activity of our hand and based on this data determine the hand gesture. More detail can be found in Section 2.3.

The advantage of non-visual based approach is high accuracy, since data usually express the actual state of the hand (for example, the angles between fingers). In general, this method does not limit users in movement around a large area, because the device can be carried with the person. This is also the main disadvantage, because it can decrease the comfort level for users.

2.3 Myo

Myo is an armband device developed by Thalmic Labs in Waterloo, Canada. It is a device for hand gesture recognition and arm movement tracking. The goal of Myo is an easy and comfortable control of our PC, smartphones and other products by using hand gestures.

2.3.1 Muscle sensors

For hand gesture detection Myo uses technology based on electromyography (EMG). Finger muscles are attached to the elbow and this fact is used by Myo for its hand gesture detection. Myo is supposed to be placed on the forearm where special EMG sensors monitor the electrical activity of these muscles and from data collected, the type of hand gesture is determined.

Myo contains 8 of these sensors [8], which were specially developed by Thalmic Labs. Because the usual electromyograph is big and expensive, this device is not suitable for such a use case. Moreover, Thalmic Labs was greatly challenged because human anatomy structure differs from one individual to another. Therefore, they developed sensors which are able to overcome issues such as indirect contact with human skin, for example because of hair cover or skin defects. To overcome this issue of human physiology uniqueness, the developers in Thalmic Lab use machine learning algorithms.

Figure 2.10. Myo armband device.

2.3.2 Gestures and locking mechanism

Because of machine learning algorithms, the set of gestures is limited and from the developer's point of view, not changeable. At the time of writing this thesis (December 2014) Myo possessed five gestures, which can be found in Figure 2.11. Also since Myo is a new device which was officially released in summer of 2014, there is still a lot of development from Thalmic Labs in process. Mostly they are trying to improve the accuracy of their recognition algorithms by releasing new firmware versions, but it is also probable that new gestures will be introduced after current gestures demonstrate suitable reliability.

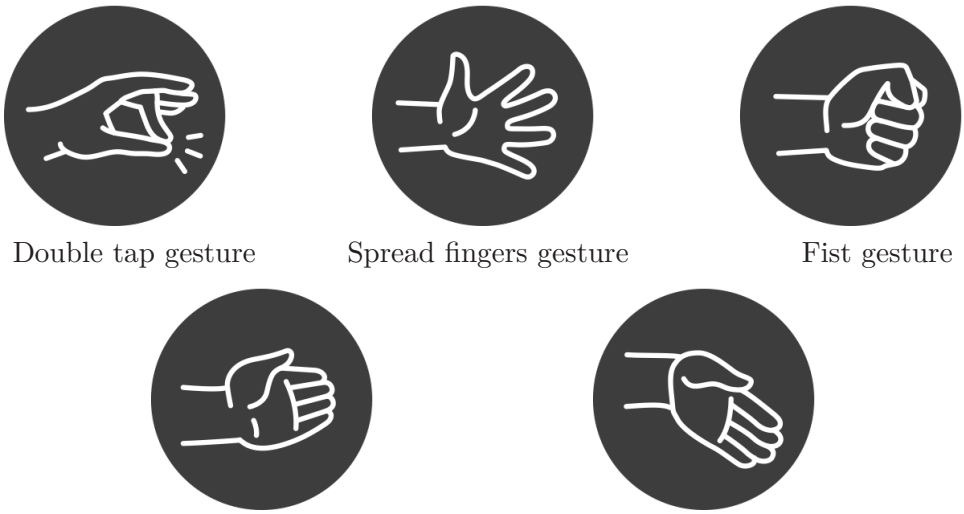


Figure 2.11. Set of Myo gestures.

To prevent the misclassification of hand movements as gesture, Myo has a locking mechanism. The mechanism keeps Myo in stand–by mode waiting for only one gesture — Double tap gesture — all other gestures are ignored. When the unlocking gesture is made, Myo starts to look for other hand gestures for few seconds. If, in this period, no

gesture is made, Myo will lock itself again. This locking mechanism can be modified, disabled or extended from developer's point of view.

2.3.3 Motion sensor and haptic feedback

Myo is also equipped with a nine-axis inertial measurement unit (IMU), which detects arm movement. IMU contains a three-axis gyroscope, a three-axis accelerometer and a three-axis magnetometer.

Moreover Myo is able to give user haptic feedback with three types of intervals — short, medium and long vibrations. This enables communication with the user. For example, for acknowledging a recognized hand gesture, etc.

2.3.4 Bluetooth Low Energy

For communication with the external world, Myo uses new Bluetooth standard - Bluetooth Low Energy (or also sometimes referred as Bluetooth Smart).

Bluetooth Low Energy (BLE) was designed to overcome several Bluetooth disadvantages. The goal of Bluetooth is data transfers — images, footages or any other data. It has a quite high data throughput, the high speed version of the Bluetooth 4 standard has, according to [9], a data throughput of 24 Mb/s. But the down side of this throughput is high power consumption.

There are some cases which do not require such a high throughput, for example it needs to communicate only the current status of the device (for example — door opened/closed). Some cases require the lowest possible power consumption, since it is highly probable that these devices will be battery powered. Specifically for these types of cases, Bluetooth Low Energy was developed. It has throughput 0,27 Mb/s, but it has from 2 to 10 times lower power consumption than the normal Bluetooth standard. This new standard is targeted for the upcoming era of *The Internet of things*.

2.3.5 Hardware a API

Because EMG sensors generate a large amount of data, an ARM Cortex M4 processor is used as part of Myo [8]. This processor is used for applying machine learning algorithms to classify executed hand gestures. After a hand gesture is classified, Myo notifies a pared device through BLE which hand gesture was performed.

Thalmic Labs provides SDK to access Myo functions. This SDK provides a high level API, that means that developers do not have direct access to raw sensors data, but only to classified hand gestures and motion data from IMU. Currently, Myo SDK supports Windows 7 and 8, Mac OS, Android and iOS platforms.

2.4 Indoor localization

For outdoor localization there is a widely used Global Position System (GPS), but for indoor localization there is no widely used system and there are many approaches to this problem. I will mention several systems and approaches which can be used.

Figure 2.12. Example of LANDMARC set up presented in [10].

2.4.1 LANDMARC system

The system described in [10] is LANDMARC. This system uses RFID tags and readers for estimation of location. In a place where localization is needed several RFID readers are placed with a stationary network of Reference RFID tags. Then, to the object which needs to be localized, a Target RFID tag is attached. An example of a set up can be seen in Figure 2.12

Localization is then done through comparing the signal strengths of the Reference tags and the Target tag, assessing the nearest neighbor Reference tag, thereby localizing the Target tag.

2.4.2 Localization through Inertial sensors

One different approach to indoor localization uses data from inertial sensors, for example, from a smartphone. This approach models our movement through the usage of sensors such as accelerometer, magnetometer and gyroscope. According to [11] the accuracy of this approach can be 1,5m if the smartphone is carried in-hand and 2m if the smartphone is carried in a pocket.

2.4.3 WiFi RTLS using RSSI

Another approach to indoor localization is Received Signal Strength Indication (RSSI). There are several ways to use this value for estimating the location. One system which uses this approach was developed by Martin Cihlář at the Czech Technical University as a bachelor thesis [12] and it is called Real–Time Location System, using the WiFi standard.

This system uses the RSSI fingerprinting method for a network of WiFi Access Points (APs). Across the rooms several WiFi APs are placed. Before actual localization, a collection phase needs to be done, where data points are collected around whole place. These data points are made of RSSI values of all WiFi APs given to the marked location by user. Afterward, the location is estimated by comparing the database of measured data points and therefore estimating the location. As mentioned in the Bachelor thesis [12], the average accuracy of their implementation is around 2 meters.

2.4.4 Quuppa

In Finland there is company called Quuppa, which developed a highly accurate indoor position technology called *Intelligent Locating Technology™*. This technology is able to localize a user with 0.1 — 1 meter accuracy with high refresh rate up to 100Hz.[13] This technology is based on Nokia's research called High Accuracy Indoor Positioning (HAIP).

For localization they use Bluetooth Low Energy standard. They achieved this accuracy by measuring not the RSSI value, but the angle of the signal arrival.[13] For this they use special hardware with specially designed antennas for estimating this angle of arrival of the signal. Then most probably by using triangulation, they are able to localize user.

Chapter 3
Aim of the thesis

Smart Homes are an advanced platform, which is now moving living standards to a new level. This platform enables a person to control main elements inside the home using digital technologies. Control of such a home can be generally divided to two categories:

- user control - for example toggling lights in rooms, regulating heating and so on.
- automatic control - for example, the heat is turned off during the day when nobody is home, but before users come back home, the heat will turn on.

There is nothing to be improved on automatic control from the point of view of the user experience, user control, on the other hand, has potentional to be improved.

As stated in chapter 2.1.6, the most common control interfaces for Smart Homes are switches, remote controllers or smartphone/tablet applications. All these interfaces require either direct manipulation (switches), or carrying a control device with the person (remote controllers, smart devices).

When interaction is desired, the user needs to manually manipulate the device. That include steps like finding the device which usually interrupts current activity. A diagram of such action can be seen in Figure 3.1. This approach could be improved by using a different type of control interface. As the topic of this thesis suggests, the focus will be on hand gesture control.

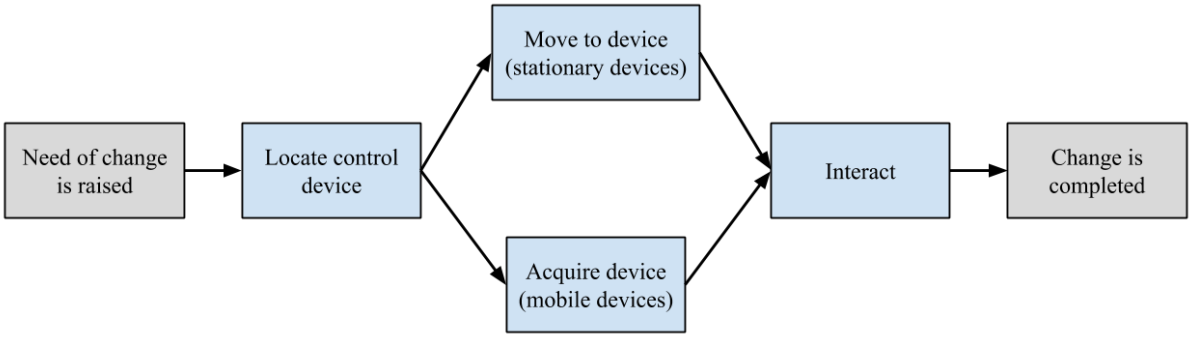


Figure 3.1. Diagram of usual interaction with control elements.

Hand gesture control can reduce the number of steps in diagram presented in Figure 3.1 from 3 steps down to just one step — interaction. Either no control device is needed for interaction, or a control device is implemented in ways that does not limit the user in movement nor limits his usual activities (for example as Myo armband).

The vision behind this thesis is a system, which would completely eliminate the need of direct interaction with control devices of Smart Homes. The system should fulfill following points:

- The system should not limit the user in movement nor his work.
- The system should be based on hand gesture recognition.
- The system should be able locate the user inside the Smart Home and distinguish location at least on a room level.
- The system should take into consideration the location context.
- The system should be easily configurable.
- The system should be usable throughout the Smart Home.

Chapter 4
Analysis

In the previous chapter an ideal system was proposed. This chapter will focus on an analysis of the proposed system and the formation of real system which will be in the end implemented.

4.1 Hand gesture recognition

As mentioned in Chapter 2.2.3 there are two general approaches to hand gestures recognition — visual based and non visual based.

If we would like to use a visual based approach for an ideal system described in the previous chapter, cameras would have to cover every part of the Smart Home. Moreover several angles would have to be covered in every part of Smart Home, so situations where user is standing with his backs toward camera would be eliminated. That is not impossible, but building such a infrastructure could be quite expensive, especially in bigger buildings. Therefore use of non–visual based approach is more suitable for such a system.

When we search for current non–visual based hand gesture detection solutions, not many solutions are found. But one technology appears to be quite interesting and usable for the purpose of the proposed system. It is an armband called Myo detailed in Chapter 2.3.

As Myo is placed on forearm, it does not disturb the user in any way. It does not limit the user in any movement. The only limitation is the range of the BLE , but that is an easily solvable problem using a grid of BLE Access points. One disadvantage of Myo is that it has limited range of detectable gestures (currently just 5 gestures, more details in Chapter 2.3.2). This fact emphasizes the importance of locating user with Myo inside a Smart Home so that location context could be assigned to performed gestures.

Since Myo matches its features to my vision, it will be used in this thesis for hand gestures recognition.

4.2 Localization

As already mentioned several times, the location of the user inside the Smart Home is crucial. It gives an opportunity to the user to interact with the closest surroundings and to specifically target his commands.

Ideally, user location should have the highest accuracy possible, but at least it has to be able to distinguish between rooms in the Smart Home. For now we will assume that the Smart Home will have only one floor, which means that localization will contain only in 2 dimensions. Moving to 3 dimensions should be a goal for further improvements.

4.2.1 BLE research

My first idea was to approach this localization problem using the signal of BLE coming from Myo armband. This approach had one major benefit. No extra device would be needed to be carried along with user, which would be very user friendly.

My idea was to use a network of BLE clients which would periodically measure RSSI from Myo and then using the principle of trilateration (Figure 4.1) to estimate approximate user location.

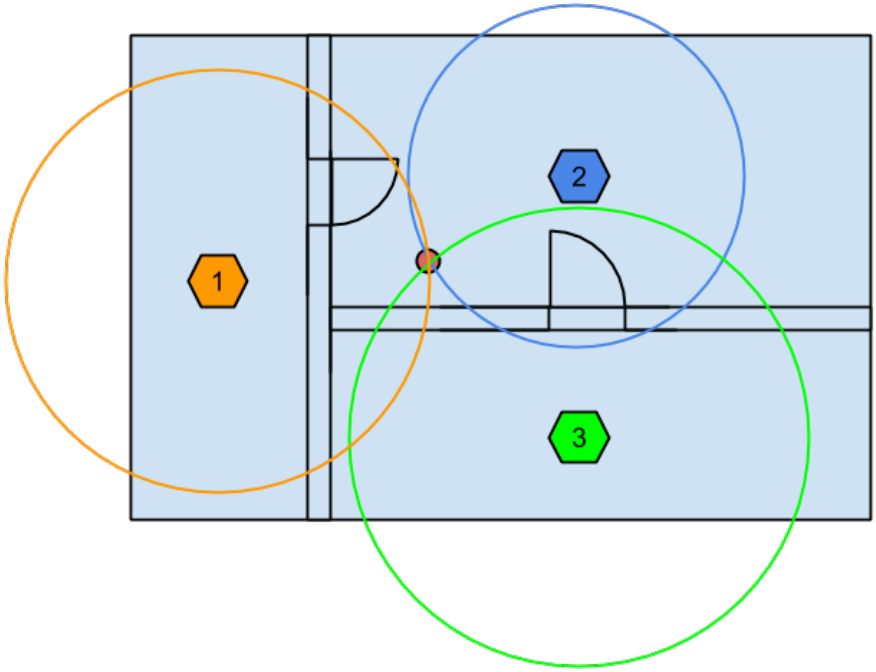


Figure 4.1. Demonstration of trilateration principle. Points 1, 2 and 3 are BLE APs which measure RSSI.

To verify this approach, I measured the signal characteristics of BLE. Since this measurement was done in Autumn 2013 when Myo was still not released, I used a heartbeat sensor from Ciclosport, model HRMBLE (can be seen on Figure 4.2). This sensor has BLE connectivity, so even if the characteristic measured on this sensor was not the same as Myo, it should serve well for a general overview of BLE signal characteristics.

Measurements were done in my flat, (Prague, CZ) which could be considered as a real life environment. Several WiFi networks and Bluetooth devices were present within these surroundings (no BLE device except the heartbeat sensor). I used my smartphone Nexus 4 with the BlueScan application for measuring RSSI values. There were two groups of measurements made.

The first group was meant to get some knowledge about RSSI values in relation to the rotation of the heartbeat sensor. I made 3 sets of measurements, one set for each axis. Graph of this group of measurements can be seen on Figure 4.3. The second group of measurements was made to get an overview of how the BLE signal dealt with signal obstacles (wall or human). You can find measured data in Appendix C.

The result of the measurement was that the localization approach suggested above, is not applicable. Mainly because I couldn't find any reasonable correlation between RSSI

Figure 4.2. Ciclosport heartbeat sensor used for measurements.

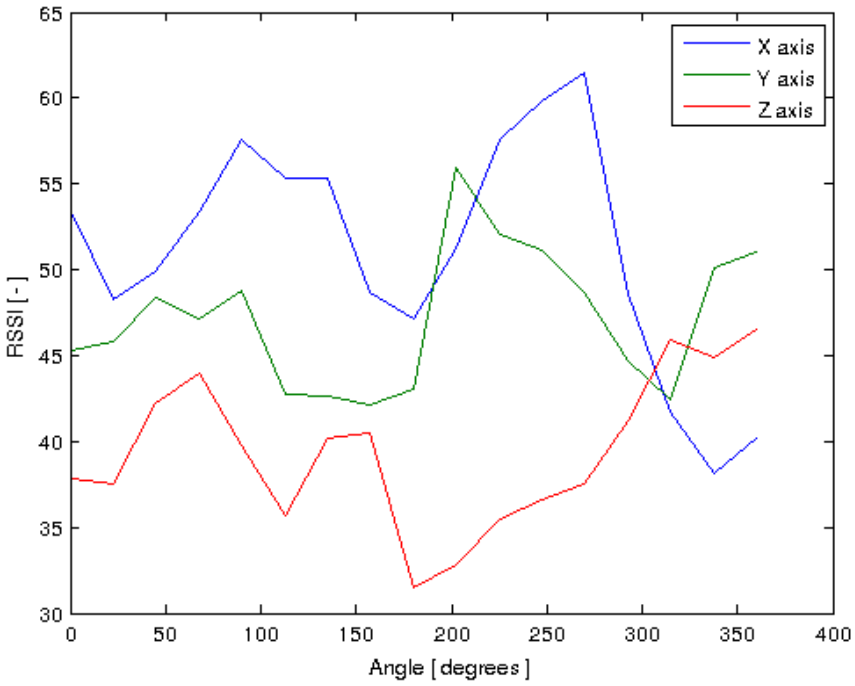


Figure 4.3. Graph of dependency of RSSI value on rotation.

and distance. Also the RSSI value seems to be unpredictable in situations concerning obstacles. This measurement also supports statement that using RSSI of BLE signal is applicable only for Proximity sensing as mentioned in [14]. It is possible that with further research a solution can be found.

4.2.2 RTLS system for WiFi

As my original idea was rejected, different approach had to be found. Since my thesis does not aim on implementing indoor localization but implementing hand gesture control, I decided to use a system which was developed at Czech Technical University as

bachelor thesis [12] by Martin Cihlář entitled: Real-Time Location System using WiFi standard. Its principle is explained in Section 2.4.3.

As it is stated in Section 2.4.3, accuracy of this system is around 2 meters. This accuracy is enough for my use case, since it is enough to localize room in which is user located. Disadvantage of this approach is that user needs to carry with him WiFi device, in this case it will be smartphone, which will decrease user comfort, but still will fulfill requirements set in Aim of the thesis chapter.

4.3 Platforms

4.3.1 Android

When I started to think about which platform to be utilized, I had to take in consideration the previously mentioned approaches and selected technologies. In the end I decided to implement this system on Android platform, because Myo has SDK for Android and also because the localization system I choose was developed on Android.

4.3.2 KNX - Switchboard

Usage of the KNX standard was specified already in the assignment of this thesis. Because the Testing phase occurred in the Center of Assistive Technologies CTU (CAT) on Czech Technical University (CTU), the special KNX element called Switchboard was used. This element was developed as bachelor thesis [15] on CTU and it was specially tailored for CAT Smart Home.

This element is a computer which is accessible through local wireless network and which is connected directly to KNX through KNX over IP. Switchboard has two main functions.

Firstly it serves as logger. Since it is „always on“ and it is connected to a KNX bus, it can listen to all events which are happening on the bus and store them for logging purpose or for more advanced analysis.

Secondly it serves as gateway for outside communication. It receives simple messages in the format *[address, value]* and sends relevant commands to the addressed element inside the KNX network. So it is easier to implement communication with this KNX network, since you only need to know how to send text data over socket.

According to the thesis [15], the Switchboard use Calimero API. It is Java library for direct communication with KNX network. This library is possible to use directly in smartphone application, which will be feature step of this application.

Chapter 5
Implementation

In this chapter I will describe the technical side of the system which was discussed in my Analysis chapter and which I have implemented. I named the application *Home Myo*.

5.1 Architecture of the system

As can be seen on Figure 5.1, the system consists of 3 parts — the User, the KNX network and the Real–Time Location System using WiFi (RTLS).

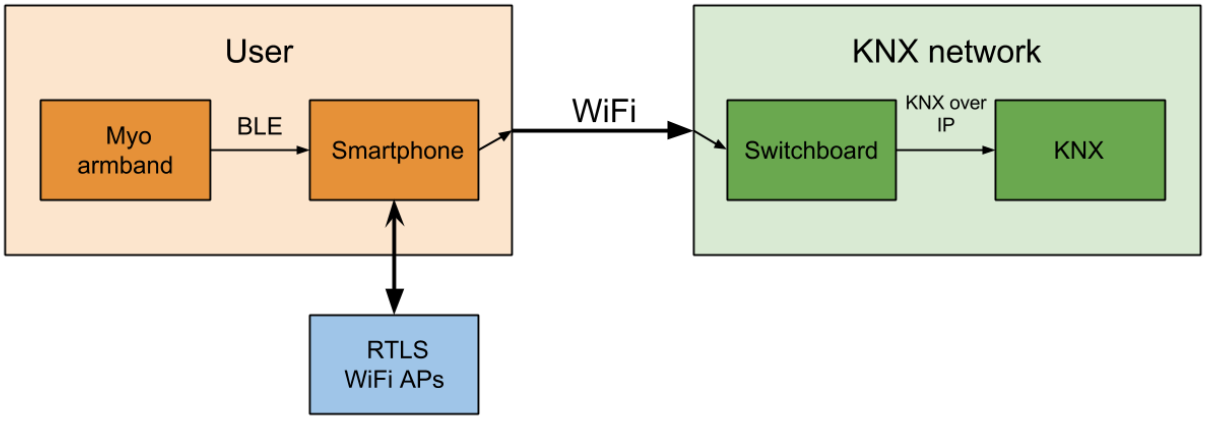


Figure 5.1. Home Myo architecture.

The User part has two elements — the Myo armband and the Smartphone application. Through the Smartphone application, any user can configure the settings and switch the Background Service, which listens for gestures detected by Myo. When a gesture is received, a query on the user location is made to the RTLS system. Based on the user location, the recognition process of Combo is started. After Combo is detected, a corresponding Command is sent over WiFi to the Switchboard. Then the Switchboard will process the Command and sent appropriate KNX commands into the KNX network.

5.1.1 Combos

Myo is capable of recognizing 5 gestures. That is quite a small amount for controlling the entire Smart Home, especially if you take into consideration that in a CAT Smart Home, which is middle size 2+1 flat, there are more than one hundred possible group addresses to control over KNX. The first step for expanding the amount of possible controlled group addresses was integrating the localization system. With user locality

context, I was able to control more elements, because I could map executed gestures for each room and therefore control those elements specified within that room. But it was still not enough to cover a potentially large number of commands in a Smart Home.

Therefore I came up with the idea of Combo. It is a predefined sequence of gestures which has an assigned Command and after performing the sequence of gestures, the Command is executed. Also I created two types of Combo. The first type is linked to some specific room and it is active only in that room. The second type is a general Combo which is always active and does not depend on user location.

During the development phase, I had to decide the maximum number of gestures Combo could contain. I tested several numbers and I decided that 3 gestures will be the maximum. It has two main reasons. Firstly, longer sequences are more difficult for the user to remember. If we have, for example, 3 rooms and in every room 3 or 4 specified Combos, it can be hard for the user to remember what Command is triggered with which Combo. Secondly it is sometimes difficult to perform Combo longer than 3 gestures, because of the false positive gesture detection of Myo. In the end I believe that this length is providing enough capacity to cover potential big amount of Commands.

5.2 Architecture of smartphone application

Smartphone application is divided into two parts — User Interface and Background Service. These parts share common „middle man“ Singleton object *AppData*, which maintains all data of application except application settings.

The User Interface part applies to application management. It has an Android GUI which enables the user to manage all application settings and combos. Also it is the place where the user can switch on or off the Background Service.

The Background Service is a main part of the application. As the name suggests, it runs in the background of a smartphone. It listens to Myo gestures and detect Combos. It is also part of an application which communicates with the external world. Using an Adapter module, it sends Commands through Switchboard to the KNX network. In addition, through the Tracker module, it communicates with the RTLS service.

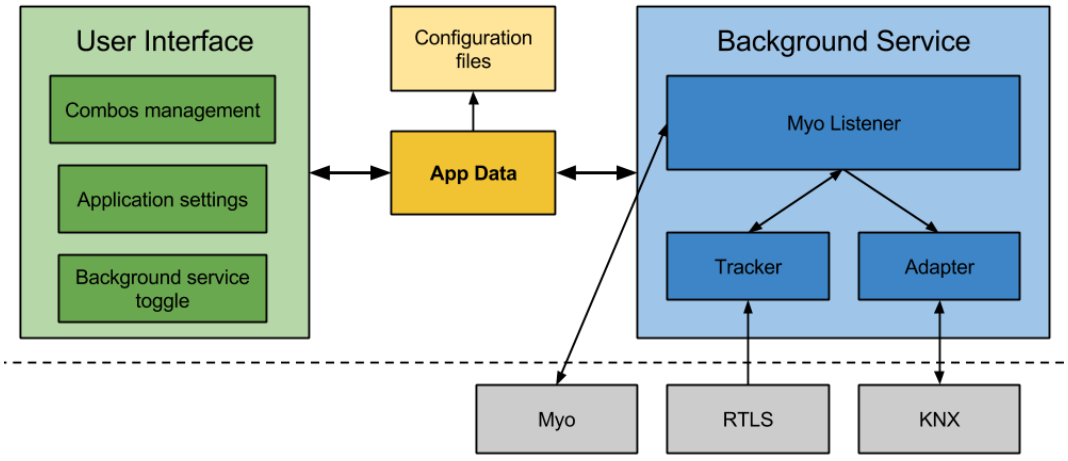


Figure 5.2. Home Myo Smartphone application architecture.

The application was designed with flexibility in mind, because I am aware that the current architecture is closely adapted to CAT Smart Home and is not usable in a general KNX Smart Home. Therefore, I created the Tracker and Adapter modules, which use Factory design patter with interfaces and thus integrating new approaches should be very easy in the future. (for example using direct communication with KNX network through Calimero API)

5.3 Used tools

5.3.1 Android SDK

As described in my Analysis chapter, I decided to use the Android platform for developing Smartphone applications. Therefore in my application I used Android SDK.

Using Myo, the demands upon a smartphone are quite high. Most importantly it has to have BLE connectivity, which these days usually is found only on high–end smartphones. BLE connectivity also supports Bluetooth 4.0, so smartphones with this connectivity are compatible with Myo. It is important which version of Android the smartphone runs on. Because native support for BLE connectivity was added in Android version 4.3 Jelly Bean (API level 18) [16] and Myo SDK is most likely using this support, Myo requires, at minimum, API level 18 [17], the smartphone for our purpose has to run, minimally, on Android 4.3.

Since I used Nexus 4 with Android 5.0 Lollipop for development and testing of this application, the API level is 21, but the application should run also on lower API levels (as low as API level 18).

5.3.2 Myo SDK

For development I used the Myo Development Kit, this comes complete with final hardware, which is not different from the consumer's version. The only difference is in the software equipment.

Myo is complete from the hardware perspective, from the software side there is still very active development. While developing the application there were two Myo SDK updates and one Myo firmware update. The application uses Myo SDK version 0.10 and my Myo armband runs firmware version 1.1.5.

Myo SDK provides complete management of the Myo armband. It shields a developer from any low level interaction. There is a singleton class called *Hub* which is used for connecting, disconnecting and scanning for Myo. It also allows you automatically connect to nearby Myo or to specific Myo MAC address. Moreover it manages Myo listeners.

Listening to Myo actions is very easy. The only thing which is needed is to override abstract class *AbstractDeviceListener* and its events method (for example onConnect, onPose etc) and register this listener with *Hub* singleton.

5.3.3 AndroidAnnotations

For better development of Android application, I used an annotation driven framework *AndroidAnnotations* [18]. It allows users to enhance components such as Activities,

Services or custom classes (under the name Bean). The main features are Dependency Injection, Event binding (in Activities), Thread management, Resource Injection and usage of typesafe SharedPreferences.

Before compilation of the application, the Android Annotations module is called, which searches for annotations in code. Based on annotations and attributes or methods which they are assigned to, it generates subclasses of annotated classes with generated methods for all above stated features. Because of this there is no performance issue, since the whole work is done already on computer before the application is installed to a smartphone.

In application I used framework version 3.2.

5.3.4 Simple XML framework

Since I decided to use XML files for storing data of this application, I used Java framework called *Simple* [19] for XML serialization and deserialization. It is mostly annotation driven framework, which can map XML entities to Java classes. In application I used version 2.7.1.

5.3.5 Pinned Section ListView

For Combos overview I used an already developed ListView called *Pinned Section Listview* [20]. It enables you to make sections in ListView, which always stay on top of the ListView until another section replaces it. Version 1.6–beta.2 was used for this application.

5.3.6 Graphical elements

In application I used several icons for easier interpretation of presented Combos and Commands. Myo gesture icons are provided by ThalmicLabs in their *Branding and UI Guidelines* [1]). Also I used icons from *Android Action Bar Icon Pack* [2]) and last but not least I used icons from *Icons8* [3]).

5.4 Configuration of application

As I already mentioned previously, for storing data in this application I decided to use the XML format. The main reason was that the Commands configuration file has to be provided from user and since I was using Switchboard, uses XML configuration file [15], work with XML was inevitable and therefore I decided to use XML completely for all data storing.

Now that I have finished the implementation, I think that perhaps a different approach to storing data would be more suitable. For example, using the SQLite database which is easier to work with regarding manipulation and the XML use just for import of data into database.

[1]) `https://developer.thalmic.com/downloads`
[2]) `https://developer.android.com/design/downloads/index.html`
[3]) `http://icons8.com/`

5.4.1 AppData singleton

As mentioned in Section 5.2 concerning the Architecture of the smartphone application, there is „middle man“ object standing between two main parts of this application. This object serves to maintain actual data between both parts.

In the beginning, I planned that every part would load the data by itself and when a change in data was done in the User Interface, the change would be stored into the XML file and then the Background Service would be notified that a change was done, so it could load the updated XML file. But later on I came up with a better approach of using Singleton, which completely serves as a proxy to data. I believe that this approach is more suitable, because it shields the two parts of the application from data access, which makes it possible to easily change the data structure or the storing format (for example from XML to SQLite). Moreover it makes it easier to share the data between the parts of the application because no notification is needed, since each change in data will be reflected immediately everywhere.

I am aware of possible difficulties with concurrency, because User Interface and Background Service are running on different threads. The style of usage of this application lowers probability of such a problem. Because when you are configuring the application, you are, most likely, not controlling the Smart Home. Also the access rate to data is quite low.

5.4.2 Tree configuration file

The Tree Configuration file is stored in a private application folder on an External Storage of the Android device. I chose this folder, because when the application is uninstalled, the data are removed from this folder. The Tree Configuration file consists of Combos data. It has entries for every Room and in every Room the Combos are saved. You can find an example of such a configuration file in Figure 5.3.

```
<rooms>
    <room id="0">
        <combo id="1" command_id="1">
            <name>Whole flat (on/off)</name>
            <myo-pose type="FIST"></myo-pose>
            <myo-pose type="FINGERS_SPREAD"></myo-pose>
            <myo-pose type="FIST"></myo-pose>
        </combo>
        <combo id="2" command_id="3">
            <name>All lights (on/off)</name>
            <myo-pose type="FIST"></myo-pose>
            <myo-pose type="WAVE_OUT"></myo-pose>
        </combo>
    </room>
    <room id="1">
        <combo id="3" command_id="26">
            <name>Lights</name>
            <myo-pose type="FIST"></myo-pose>
        </combo>
        <combo id="4" command_id="28">
            <name>Blinds</name>
            <myo-pose type="WAVE_OUT"></myo-pose>
        </combo>
    </room>
</rooms>
```

Figure 5.3. Example of Tree configuration file.

5.4.3 Rooms configuration file

The Rooms configuration file is located in a folder called *HomeMyo* in the root of External Storage. It is one of the two configuration files that the user has to provide before this application can run. This file is related to RTLS system and defines rooms which are in the Smart Home and which is RTLS able to detect. If some other localization service would be used, the structure of this file would differ. An example of the Rooms configuration file can be seen in Figure 5.4.

```
<?xml version="1.0" encoding="utf-8"?>

<rooms>
    <room id="1">
        <name>Living room</name>
        <mapping>1</mapping>
    </room>
    <room id="2">
        <name>Kitchen</name>
        <mapping>2</mapping>
    </room>
    <room id="3">
        <name>Room</name>
        <mapping>3</mapping>
    </room>
    <room id="4">
        <name>Bathroom</name>
        <mapping>4</mapping>
    </room>
    <room id="5">
        <name>Corridor</name>
        <mapping>5</mapping>
    </room>
</rooms>
```

Figure 5.4. Example of Rooms configuration file.

5.4.4 Commands configuration file

The Commands configuration file is located along with the Rooms configuration file in folder *HomeMyo* in the root of External Storage. This file is a copy of the Switchboard configuration file [15] and it specifies which Commands are available in the KNX network. In the application several fields from this configuration file are used:

- name — Name of Command
- graddress — Group Address
- knxtype — Data type which elements receive on this address
- type — Element type

Example of such a configuration file can be seen in Figure 5.5. If in future Calimero API will be implemented, this file will be replaced with exported ETS project.

5.4.5 SharedPreferences

Even AppData is used to store all application data, there is one exception — application settings. These Settings are stored in Android's SharedPreferences. For type–safe usage of SharedPreferences I used AndroidAnnotations interface, which is converted into automatically generated class.

```
<doc>
  <elements>
    <emp id="1">
      <name>3. patro-centrální funkce-vyp S 304c</name>
      <type>CENTRAL_FUNCTION</type>
      <knxtype>BOOL</knxtype>
      <graddress>3/0/13</graddress>
      <DPT>DPT_Switch(DPST-1-1)</DPT>
      <ID>P-03D3-0_GA-118</ID>
    </emp>
    <emp id="2">
      <name>3. patro-stínění-V 4.2.109 stop</name>
      <type>BLINDS</type>
      <knxtype>BOOL</knxtype>
      <graddress>3/2/1</graddress>
      <DPT>DPT_UpDown(DPST-1-8)</DPT>
      <ID>P-03D3-0_GA-15</ID>
    </emp>
  </elements>
</doc>
```

Figure 5.5. Example of Commands configuration file.

5.5 User Interface

As mentioned earlier, the User Interface part of application is meant for application management. It serves to three purposes:

- Combos management — overview of all combos, show detail of combo and add, edit, delete combo.
- Application settings — setting of KNX connection, Myo pairing, and other settings.
- Background Service switch — turning on and off listening to Myo gestures.

5.5.1 Combos management

For Combo management there are 3 Activity classes:

- MainActivity — it is an entry point to the application and also the place of an overview of all Combos. A screenshot can be seen in Figure 5.6.
- DetailActivity — it is a place where all details about Combo can be seen — information about assigned Command and Room. A screenshot can be seen in Figure 5.8.
- AddActivity — this activity is used for adding a new Combo or editing already existing Combos. A screenshot can be seen in Figure 5.7.

5.5.2 Settings of application

In the Settings of this application which can be seen in Figure 5.10, it is possible to change several things:

- KNX settings — IP address and port number of Switchboard
- Localization — turn off or on localization
- Myo — pair Myo armband with application (can be seen on Figure 5.9) or set time interval for locking Myo after unlock gesture is performed

Figure 5.6. Overview.

Figure 5.7. Add combo.

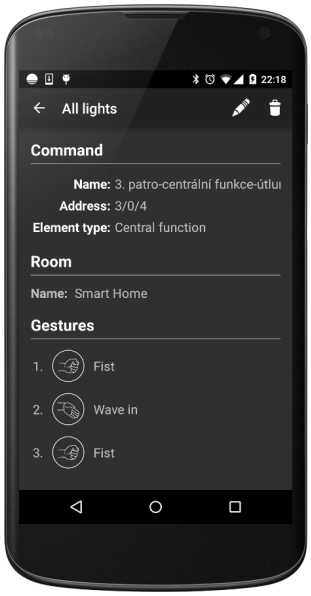


Figure 5.8. Combo detail.

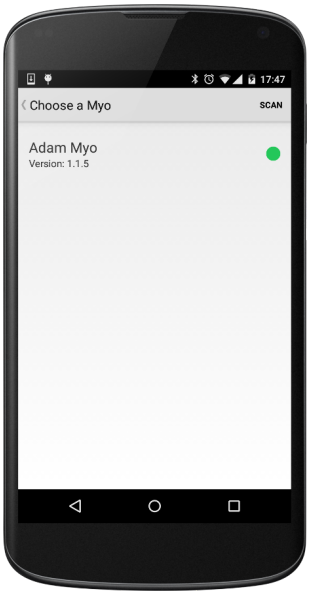


Figure 5.9. Scanning for Myo.

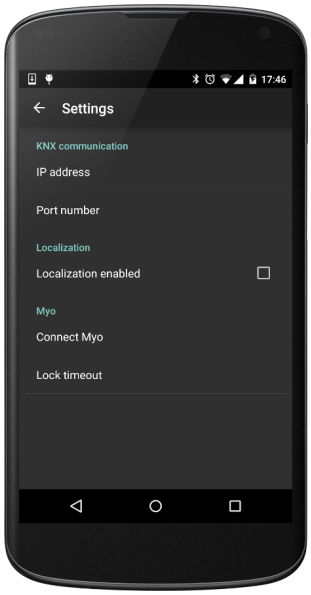


Figure 5.10. Settings of application.

5.6 Background Service

The Background Service is the part of the application where combo detection and command execution takes part. It has three modules — Tracker for localization, Adapter for communicating with KNX network and MyoListener for gesture recognition and combo detection.

5.6.1 Tracker

The Tracker module is defined by *ITracker* interface, which has one method *getLocation()* that returns *Room* object of room in which is user located. Tracker instances are created through a static factory method *TrackerFactory.createTracker()*

As previously mentioned, I used the RTLS system for CAT Smart Home. RTLS runs as a background service, which updates the user location every 10 seconds. I created *CatTracker* which implements *ITracker* interface and manages information provided from the RTLS system. Communicating between *Home Myo* and RTLS is done through service binding. Background Service exposes binding interface which RTLS binds to and then shares its data through this communication link.

5.6.2 Adapter

Adapter module is defined by *IAdapter* interface, which has two methods *sendTelegram(ITelegram telegram)* and *getState(ITelegram telegram)*. Method *sendTelegram(ITelegram telegram)* will take data from *telegram*, properly convert them into a format received by the KNX network and then send it. Method *getState(ITelegram telegram)* is used for retrieving the current state of element defined in *telegram*. Adapter instances are created through static factory method *AdapterFactory.createAdapter()*.

Since Combo defines only the KNX element and not what action should be executed with the element, the application has to discover what is the current state so it can decide what to do with it based on the data type of the element. To be more clear, here is an example: Combo can define that the state of all lights in entire Smart Home should be changed, but it does not say if lights should be turned off or turned on. Therefore the application asks for current state of all lights and then sent command with the opposite value. Retrieving current status is also done through Switchboard, by sending a command *[read, address]* and then listening for reply.

Currently in *Home Myo* only boolean values are supported, since most of the CAT Smart Home uses this data type. Other data types will follow as future steps.

To use Switchboard as „middle man“ for communication with the KNX network, I created *CatAdapter* class which implements *IAdapter* interfaces. As commands for Switchboard are in simple text format *[address, data]* [15], I implemented the simple Socket connectivity with sending text based commands assembled from *telegram* data. Because the current state needs to be retrieved before sending the actual command I created an inner classes which implements *AsyncTask*. When Commands need to be sent, firstly a status query is sent and after retrieving current status, an actual command is sent with the proper value.

5.6.3 MyoListener

MyoListener is the core module of whole application. It is a class which implements Myo SDKs *AbstractDeviceListener* class and is called from *Hub* anytime Myo detects a gesture. Describing MyoListener behavior would be quite complicated and hard to understand, so I created diagram which should clearly explain the decision process of *MyoListener* and you can find it in Figure 5.11

5.6.3.1 Locking mechanism

Myo has a built in locking mechanism which unlocks Myo after a Double tap gesture and leaves Myo unlocked for several seconds. Unfortunately the unlocked period is not extended if a gesture is performed. This feature is crucial for Combo recognition, since it is impossible to perform up to three gestures in the short time which the default locking mechanism keeps Myo unlocked.

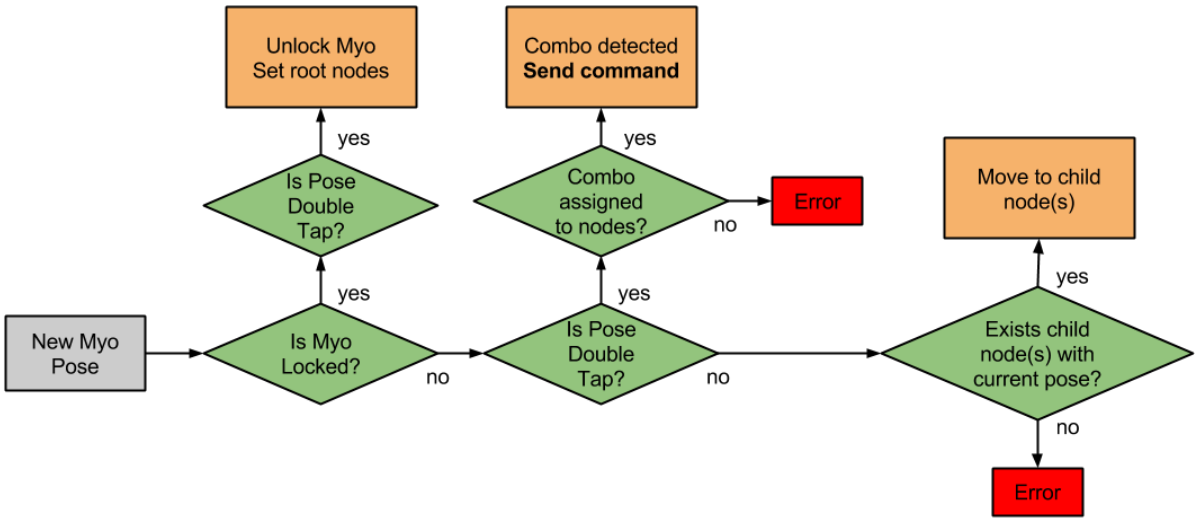


Figure 5.11. Decision diagram of MyoListener.

Fortunately Myo SDK lets you disable this default built in locking mechanism and implement your own mechanism. After disabling the default mechanism all detected gestures are provided to MyoListener and it is up to the developer how to implement the locking mechanism. I used the behavior of the default mechanism as a template, therefore Myo is unlocked with a Double tap gesture and it is kept unlocked for defined amount of seconds, which can be configured in the application settings. The only difference from the default mechanism is that if a gesture is detected and it is a valid gesture which leads to some Combo, the locking timer is reset, so there is enough time for performing following gesture.

5.6.3.2 Feedback

As stated in 2.3.3, Myo has haptic feedback. I use this for communicating with user. Myo supports three lengths of vibration — short, middle, long. Here is overview of events and my feedback.

- Myo locked or unlocked — middle vibration.
- Gesture detected and leads to some Combo — short vibration.
- Gesture is detected, but does not lead to some Combo — three short vibrations.
- Confirmed Combo has not assigned Command — three short vibrations.
- Error (localization system does not function or there is no wireless connection etc.) — five short vibrations.

5.6.3.3 Combo detection

Combo detection is key feature in *MyoListener*. For Combo detection I use a tree data structure, an example of such a structure can be seen in Figure 5.12. Every node of a tree is associated with a Myo gesture. This tree data structure is created from the Tree configuration file described in Section 5.4.2. Every room has its own tree for room specific Combos, plus the entire Smart Home has its own general tree for Combos which are not linked to any specific room.

MyoListener maintains two actual nodes, one is for the tree which is linked to the room where the user is actually located and the second actual node is for the general tree.

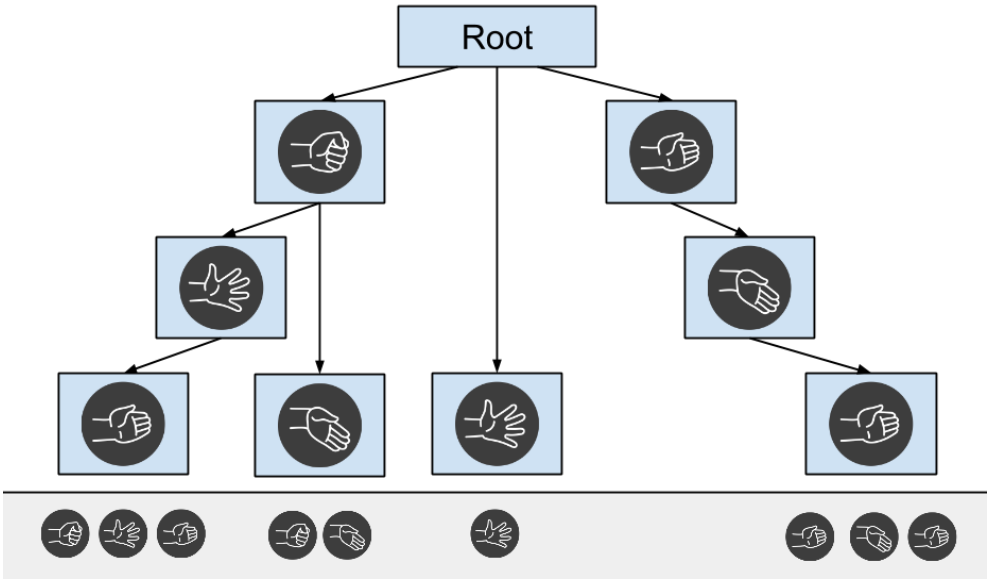


Figure 5.12. Combo tree data structure example.

The room–dependent tree is set, based on user location, when the user performs the unlocking gesture (Double tap gesture). If the user relocates himself while performing a Combo, the relocation won't be taken in consideration.

If Myo is locked, actual nodes are empty. As can be seen in Figure 5.11, after unlocking Myo, the root nodes of the actual room tree and the general tree are set as an actual node. Afterwards, when a gesture is detected, *MyoListener* will look up actual node's children and if there is a child with same gesture associated as detected gesture, this child is set as actual node. If there is no such a child, error is raised and actual nodes are reset.

When the user thinks that he has finished performing a Combo which he intended, he has to perform a confirmation gesture which is the Double tap gesture. Afterwards *MyoListener* will check the actual nodes if there is assigned some Command. If so, the Command is sent to the KNX network, if not then an error is reported. This confirmation mechanism serves for expanding the Combo range, because if *MyoListener* executes the first Command it finds, it would not be possible to use all the length of the Combos. To make it more clear: If you have the Combo sequence „Fist" and to this Combo you have assigned some Command, immediately after a Fist gesture would be recognized the Command would be sent. But then if you would like to add a new Combo with gesture sequence „Fist, Wave out" this Combo would be impossible to reach because the previous Combo would be always triggered first. Moreover this mechanism serves for decreasing unintentional Combo triggering.

Chapter 6
Testing

6.1 Set up

Whole testing phase happened in the Center of Assistive Technologies (CAT) at the Czech Technical University in Prague. In CAT there is a model Smart Home located in room 304, which uses KNX standard. The flat consists of 4 rooms — living room (304a), kitchen (304b), sleeping room (304c) and bathroom (304d). The layout of the flat can be seen on Figure 6.1.

Home Myo application ran on my Nexus 4 smartphone with Android 5.0 and Myo armband with firmware 1.1.5. During the testing I was connected with smartphone to a local WiFi network, to which Switchboard is also connected.

6.1.1 RTLS

RTLS use for estimating user location, strength of the WiFi signal. For this WiFi Access Points are needed. In my testing I used 4 WiFi APs from the company MikroTik. Their position inside the CAT Smart Home can be seen in Figure 6.1, they are marked as black crosses with the last part of APs serial number for their identification.

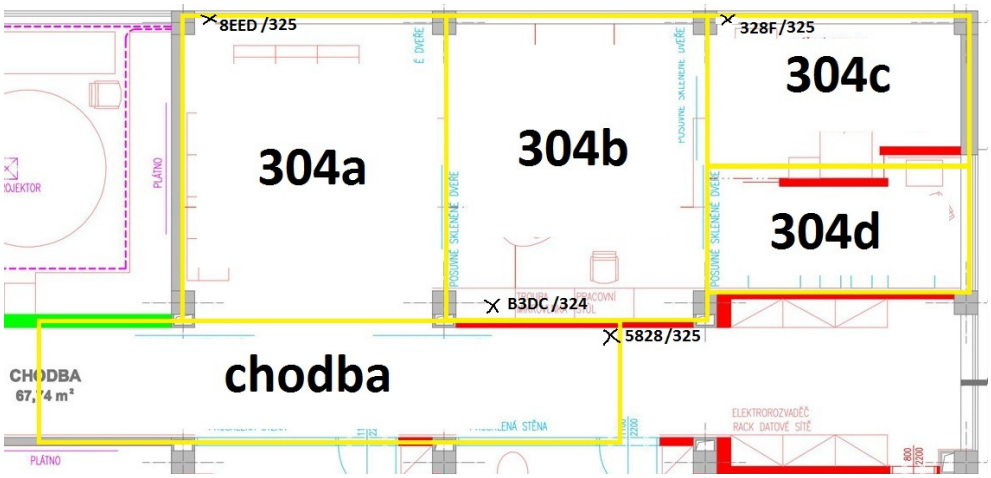


Figure 6.1. Layout of CAT Smart Home and location of RTLS WiFi APs

Before RTLS can work, sampling phase needs to be done. The sampling is done using an RTLS application called „RTLS_demo“. A network of sample points is created, where every sample point has saved strength values of WiFi APs and location where the sample point was taken, estimated by user. This network has to cover whole flat, a denser network means higher accuracy.

6.1.2 Combos

For testing I created model sets of Combos for controlling the CAT Smart Home. I created two sets, the first is meant for usage with a localization system and the second for usage without a localization system.

Since the CAT Smart Home is a model flat, in my Combos set I used mostly general Combos for demonstrating the main principle. My System is fully adjustable, everything which can be controlled through KNX can also be controlled through Home Myo application.

6.1.2.1 Combos with localization

The overview of Combos set which takes in consideration the location of the user can be seen in Figure 6.2. Complete Tree configuration file for this set can be found in Appendix B.2. In this set I used a similar approach to that which KNX uses for creating group addresses. I divided gestures by function, so it would be easier for any user to remember Combos.

Gesture sequence	Room	Name	Group address
	Global	Toggle lights in whole Smart Home	3/0/10
	Global	Toggle whole Smart Home	3/0/0
	Hall	Toggle door to Smart Home	3/2/1
	304a	Toggle all lights in room	3/0/11
	304a	Toggle first row lights in room	3/1/11
	304a	Toggle all sockets in room	3/0/24
	304a	Toggle door to hallway	3/2/1
	304b	Toggle all lights in room	3/0/12
	304b	Toggle all sockets in room	3/0/23
	304c	Toggle all lights in room	3/0/13
	304c	Toggle all sockets in room	3/0/21
	304d	Toggle all lights in room	3/0/14
	304d	Toggle all sockets in room	3/0/22

Figure 6.2. Testing set of Combos using localization.

The first gesture determines what type of elements will be controlled. „Spread finger“ gesture controls lights, „Wave out“ gesture controls doors, „Wave in“ gesture controls sockets and „Fist“ gesture is controlling commands which effect elements in whole flat. These gestures are not Room dependant, they are always active.

The first gesture determines the function of the group of elements. If no other gesture is supplied and Combo is confirmed, all elements in that function group and in the current room will be toggled (except global command). So if „Finger spread“ gesture is performed in the kitchen, all lights in that room will be toggled. But if a second gesture is performed, the user can target specific elements in that room. For example, if gesture sequence „Finger spread, Wave out“ is performed in room 304a (Living room), then only the first row of lights in that room will be toggled and not all lights in the room.

6.1.2.2 Combos without localization

The second Combo set does not count on the users location, the overview can be seen in Figure 6.3. Complete Tree configuration file for this set can be found in Appendix B.2. Generally I used same assigning principle as in the first Combo set, just extending the gesture sequence with a gesture that specifies the target room. There are four available gestures and four rooms, therefore for every room one gesture. Only the „Fist“ gesture was used for global commands, but these global Combos use a 3–gesture sequence, so it won't collide with the room Combos which use only a 2–gesture sequence.

Gesture sequence	Name	Group address
	Toggle lights in whole Smart Home	3/0/10
	Toggle whole Smart Home	3/0/0
	Toggle all lights in 304a room	3/0/11
	Toggle first row lights in 304a room	3/1/11
	Toggle all sockets in 304a room	3/0/24
	Toggle door to hallway	3/2/1
	Toggle all lights in 304b room	3/0/12
	Toggle all sockets in 304b room	3/0/23
	Toggle all lights in 304c room	3/0/13
	Toggle all sockets in 304c room	3/0/21
	Toggle all lights in 304d room	3/0/14
	Toggle all sockets in 304d room	3/0/22

Figure 6.3. Testing set of Combos without using localization.

6.2 Evaluation

I am happy to say, that system works! Short footage from testing can be found in electronic attachment.

Figure 6.4. Photo from testing Home Myo.

6.2.1 Myo

The Myo armband consists of very young technology, which has some flaws, but generally I am amazed how it works. Moreover, a big increase in reliability can be expected as more software development will be done by Thalmic Labs on its algorithms. Wearing the Myo armband does not limit the user in any way. I could wear the Myo all day without a problem. Also the battery endurance is quite good, it lasts the whole day.

Regarding gesture recognition. Myo does a great job in gesture recognition. It has very high rate of true positive detections, but also unfortunately it has a quite high rate of false positive detections. The biggest issue I found was that after performing my intentional gesture, when my hand was returning back to a resting position, usually the „opposite“ gesture was detected. For example when I performed the „Wave out“ gesture, then when my hand was coming back, the „Wave in“ gesture was detected. The same thing happened to me several times with the „Fist“ gesture, the „Spread fingers“ gesture was detected when returning my hand to the rest position. Even though it is a problem of Myo and its firmware, I believe that this issue could be improved in my application. I could implement a short time lock, after performing a gesture all gestures would be ignored for short time (for example about 500ms), so the hand could come back to a resting position.

Another issue I discovered was, that „Double tap“ gesture for unlocking Myo is misclassified very easily. Generally two rapid movements with fingers are enough for Myo to classify it as unlocking gesture. But this issue is also known to Thalmic Labs, because they recently changed this unlocking gesture from an uncomfortable „Thumb to pinkie“ gesture to the currently used „Double tap“ gesture and (at the time of this writing) they did not have enough time to collect enough samples for their training algorithms. Therefore I expect big improvements in future firmware releases. Also there is a well known workaround in the developer community for this issue. It consists of extending the unlocking gesture with more elements. The next element in the unlocking sequence

could be another gesture, or involve IMU sensors (for example rapid hand movement after „Double tap“ gesture).

It actually takes some time to learn how to use Myo. Even though it may seem that there is nothing to learn, performing hand gestures is a learning process. Mostly it is the way the gestures should be performed so that Myo reliably detects it as you intended. It consist of learning which muscles you should clench and that takes some time.

6.2.2 Combos and its detection

Combo detection which I implemented works very well. Only problems were connected with misclassified hand gesture by Myo, which I have described in the previous Section.

6.2.3 Summary

As I already said, the system works, but it should be considered as proof–of–concept, which needs further development. The main issue which needs to be solved is the localization system. RTLS is reliable determining the room in which the user is located, but it has a very low refresh frequency of 10 seconds. This makes it unreliable while the user is moving from one room to another. In addition, the main goal should be to achieve localization using BLE and therefore eliminate the requirement of having a smartphone with the user all the time.

Chapter 7
Conclusion

New technologies are released more and more frequently. That bring us many new opportunities which a few decades ago were just fantasies. In this thesis I aimed at Smart Homes and hand gesture recognition. I gathered information about hand gestures, methods of their detection, the KNX standard and indoor localization.

My main goal of this thesis was to design and implement a system, which would allow the user to control the KNX Smart Home with hand gestures. For this I had to decide which technologies I would use in this system. First of all, for gesture detection I decided to use cutting edge technology, which was released just a few months ago, the Myo armband from a company called Thalmic Labs. This armband is placed on the forearm and monitors hand muscle activity using EMG technology. It is capable to recognize 5 hand gestures and track the motion of the hand. Secondly, I needed to use an indoor localization system. For that I used an application written by Martin Cihlář at CTU as a bachelor thesis. This system uses WiFi Access Points to estimate location based on signal strength fingerprinting. Last but not least, for communication with the KNX network I used a special KNX element called Switchboard which is placed in our model Smart Home in the Center of Assistive Technologies (CAT) at Czech Technical University. It serves as an easy access entry point to the CAT Smart Home KNX network. To connect all these parts I created an Android smartphone application called *Home Myo*. Myo armband is connected to this application and all detected gestures are sent to the application. The application is capable of detecting the so-called Combo, it is a sequence of gestures which have an assigned command for the KNX network. The Combos can be active in specific rooms or in the entire Smart Home. If a Combo is detected, a Command to the KNX network is sent and executed.

The application was tested in the CAT Smart Home and proved itself as fully functional, yet it should be considered more as proof-of- concept than as a ready system. To achieve a fully functional system, several improvements have to be made. First of all a different localization system needs to be used. The main disadvantage of the currently used system is its refresh frequency. The user's location is refreshed every 10 seconds which is not sufficient while the user is moving around the Smart Home. Moreover, the system is locating the user's smartphone and not directly the Myo armband itself, which makes use of this system uncomfortable. Secondly, direct communication with the KNX network needs to be implemented, so the system could be used in a general KNX Smart Home and not only in the CAT Smart Home. This would only include linking the Calimero API library to application.

I will continue to work on this system, so it could reach a deployable state and then release it as an Open source. It could serve as inspiration or foundation for other developers. There are several projects in the Myo developer community targeting Smart Home control, but as of this writing none has been released and I am not aware of any project utilizing the KNX standard.

References

[1] Official page of KNX, What is KNX. Cited on 11.12.2014.
http://www.knx.org/knx-en/knx/association/what-is-knx/index.php.

[2] ABB. *ABB i-bus ® KNX - Intelligent Installation Systems System description.*
http://www05.abb.com/global/scot/scot209.nsf/veritydisplay/e784bbf356160fcec125777e002aa94a/$file/0156_systembe_gb_06_09.pdf.

[3] Wolfgang Köhler. Simulation of a knx network with eibsec protocol extensions. Master's thesis, Technische Universität Wien, 2008.
https://www.auto.tuwien.ac.at/bib/pdf_thesis/THESIS0002.pdf.

[4] V.I. Pavlovic, R. Sharma, and T.S. Huang. Visual interpretation of hand gestures for human-computer interaction: a review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):677–695, Jul 1997.

[5] T.B. Blair and C.E. Davis. Innovate engineering outreach: A special application of the xbox 360 kinect sensor. In *Frontiers in Education Conference, 2013 IEEE*, pages 1279–1283, Oct 2013.

[6] Frank Weichert, Daniel Bachmann, Bartholomäus Rudak, and Denis Fisseler. Analysis of the accuracy and robustness of the leap motion controller. *Sensors*, 13(5):6380–6393, 2013.

[7] Thomas G. Zimmerman, Jaron Lanier, Chuck Blanchard, Steve Bryson, and Young Harvill. A hand gesture interface device. *SIGCHI Bull.*, 17(SI):189–192, May 1986.

[8] Official product page of Myo. Cited on 24.11.2014.
https://www.thalmic.com/en/myo/.

[9] Sparkfun overview of bluetooth versions. cited on 30.11.2014.
https://learn.sparkfun.com/tutorials/bluetooth-basics/common-versions.

[10] Guang yao Jin, Xiao-Yi Lu, and Myong-Soon Park. An indoor localization mechanism using active rfid tag. In *Sensor Networks, Ubiquitous, and Trustworthy Computing, 2006. IEEE International Conference on*, volume 1, pages 4 pp.–, June 2006.

[11] Fan Li, Chunshui Zhao, Guanzhong Ding, Jian Gong, Chenxing Liu, and Feng Zhao. A reliable and accurate indoor localization method using phone inertial sensors. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pages 421–430, New York, NY, USA, 2012. ACM.

[12] Martin Cihlář. Analysis of signal coverage in a given area for rtls technology in 2.45 ghz frequency band. Bachelor thesis, Czech Technical University in Prague, 2014.
https://dspace.cvut.cz/handle/10467/24199.

[13] Technology page of Quuppa. Cited on 29.12.2014.
http://quuppa.com/technology/.

[14] BlueTooth developer portal, Proximity profile. Cited on 15.12.2014.
https://developer.bluetooth.org/TechnologyOverview/Pages/PXP.aspx.

[15] Trejtnar Ladislav. Smart Switchboard for KNX Installation. Bachelor thesis, Czech Technical University in Prague, 2014.
https://dspace.cvut.cz/handle/10467/24213.

[16] Android developer portal. Cited on 26.12.2014.
https://developer.android.com/.

[17] Myo developer portal. Cited on 26.11.2014.
https://developer.thalmic.com/.

[18] AndroidAnnotations official page. Cited on 20.12.2014.
http://androidannotations.org/.

[19] Official page of Simple framework. Cited on 23.12.2014.
http://simple.sourceforge.net/.

[20] GitHub page of Pinned Section Listview. Cited on 25.12.2014.
https://github.com/beworker/pinned-section-listview.

Appendix A
List of used shortcuts

Short cut	Whole expression
API	Application Programming Interface
BLE	Bluetooth Low Energy
CAT	Center of Assistive Technologies
CTU	Czech Technical University
EMG	Electromyography
GPS	Global Postioin System
GUI	Graphical User Interfaces
HAIP	High Accuracy Indoor Positioning
HCI	Human-Computer Interaction
IMU	Inertial Measurement Unit
IP	Internet Protocol
MAC	Media Access Control
OS	Operation System
OSI model	Open Systems Interconnection model
PL	Power Line
RFID	Radio-frequency identification
RSSI	Received Signal Strength Indication
RTLS	Real-Time Location System
SDK	Software Development Kit
SQL	Structured Query Language
TP	Twisted Pair
UI	User Interface
USB	Universal Serial Bus
WiFi APs	WiFi Access Points
XML	Extensible Markup Language

Appendix B
How to get Home Myo working

To get Home Myo application working, there are two important prerequisites — a Rooms configuration file (mentioned in Section 5.4.3 and a Commands configuration file (mentioned in Section 5.4.4).

These configuration files needs to be place into a folder „HomeMyo“ into the root folder to External Storage of smartphone. To achieve that, you can either put them into SD card root folder (if your smartphone poses SD card slot) or you can connect Smartphone with USB cabel to PC and through File manager upload it to root folder.

Name of the file have to be „rooms.xml“ for Rooms configuration file and „commands.xml“ for Commands configuration file.

B.1 Rooms configuration file

To use RTLS in CAT, this configuration file needs to be uploaded

```
<?xml version="1.0" encoding="utf-8"?>

<rooms>
    <room id="1">
        <name>Living room</name>
        <mapping>1</mapping>
    </room>
    <room id="2">
        <name>Kitchen</name>
        <mapping>2</mapping>
    </room>
    <room id="3">
        <name>Room</name>
        <mapping>3</mapping>
    </room>
    <room id="4">
        <name>Bathroom</name>
        <mapping>4</mapping>
    </room>
    <room id="5">
        <name>Corridor</name>
        <mapping>5</mapping>
    </room>
</rooms>
```

B.2 Tree configuration files

Tree configuration file is created automatically therefore it is not needed to upload it anywhere. If you want to use Combos set which I created in Testing chapter, Section 6.1.2, here are configuration files which were created for that sets. To use them in application, you have to upload them into folder which is located in root folder of External Storage, the folder is placed in path „/Android/data/cz.cvut.uhlirad1.homemyo/files“

Tree configuration file for Combos using localization:

```
<rooms>
   <room id="0">
      <combo command_id="1" id="11">
         <myo-pose type="FIST"/>
         <myo-pose type="FINGERS_SPREAD"/>
         <myo-pose type="FIST"/>
         <name>All flat </name>
      </combo>
      <combo command_id="4" id="12">
         <myo-pose type="FIST"/>
         <myo-pose type="WAVE_IN"/>
         <myo-pose type="FIST"/>
         <name>All lights </name>
      </combo>
   </room>
   <room id="1">
      <combo command_id="5" id="13">
         <myo-pose type="FINGERS_SPREAD"/>
         <name>Lights in 304a</name>
      </combo>
      <combo command_id="13" id="14">
         <myo-pose type="WAVE_IN"/>
         <name>All sockets in room </name>
      </combo>
      <combo command_id="26" id="15">
         <myo-pose type="FINGERS_SPREAD"/>
         <myo-pose type="WAVE_OUT"/>
         <name>First row lights </name>
      </combo>
      <combo command_id="28" id="22">
         <myo-pose type="WAVE_OUT"/>
         <name>Doors</name>
      </combo>
   </room>
   <room id="4">
      <combo command_id="8" id="20">
         <myo-pose type="FINGERS_SPREAD"/>
         <name>All lights in room </name>
      </combo>
      <combo command_id="11" id="21">
         <myo-pose type="WAVE_IN"/>
         <name>All sockets in room</name>
      </combo>
   </room>
```

```
    <room id="2">
       <combo command_id="6" id="16">
          <myo-pose type="FINGERS_SPREAD"/>
          <name>All lights in room</name>
       </combo>
       <combo command_id="12" id="17">
          <myo-pose type="WAVE_IN"/>
          <name>All sockets in room</name>
       </combo>
    </room>
    <room id="3">
       <combo command_id="7" id="18">
          <myo-pose type="FINGERS_SPREAD"/>
          <name>All lights in room </name>
       </combo>
       <combo command_id="10" id="19">
          <myo-pose type="WAVE_IN"/>
          <name>All sockets in room </name>
       </combo>
    </room>
    <room id="5">
       <combo command_id="28" id="23">
          <myo-pose type="WAVE_OUT"/>
          <name>Doors</name>
       </combo>
    </room>
 </rooms>
```

Tree configuration file for Combo set which does not use localization:

```
 <rooms>
    <room id="0">
       <combo command_id="1" id="1">
          <myo-pose type="FIST"/>
          <myo-pose type="FINGERS_SPREAD"/>
          <myo-pose type="FIST"/>
          <name>All flat </name>
       </combo>
       <combo command_id="4" id="12">
          <myo-pose type="FIST"/>
          <myo-pose type="WAVE_IN"/>
          <myo-pose type="FIST"/>
          <name>All lights </name>
       </combo>
       <combo command_id="5" id="13">
          <myo-pose type="FINGERS_SPREAD"/>
          <myo-pose type="FINGERS_SPREAD"/>
          <name>Lights in 304a</name>
       </combo>
       <combo command_id="26" id="14">
          <myo-pose type="FINGERS_SPREAD"/>
          <myo-pose type="FINGERS_SPREAD"/>
          <myo-pose type="WAVE_OUT"/>
          <name>First row lights in 304a </name>
```

```
        </combo>
        <combo command_id="13" id="15">
           <myo-pose type="FINGERS_SPREAD"/>
           <myo-pose type="WAVE_IN"/>
           <name>Sockets in 304a</name>
        </combo>
        <combo command_id="28" id="16">
           <myo-pose type="FINGERS_SPREAD"/>
           <myo-pose type="WAVE_OUT"/>
           <name>Doors</name>
        </combo>
        <combo command_id="6" id="17">
           <myo-pose type="WAVE_IN"/>
           <myo-pose type="FINGERS_SPREAD"/>
           <name>Lights in 304b</name>
        </combo>
        <combo command_id="12" id="18">
           <myo-pose type="WAVE_IN"/>
           <myo-pose type="WAVE_IN"/>
           <name>Sockets in 304b</name>
        </combo>
        <combo command_id="7" id="19">
           <myo-pose type="WAVE_OUT"/>
           <myo-pose type="FINGERS_SPREAD"/>
           <name>Lights in 304c</name>
        </combo>
        <combo command_id="10" id="20">
           <myo-pose type="WAVE_OUT"/>
           <myo-pose type="WAVE_IN"/>
           <name>Sockets in 304c</name>
        </combo>
        <combo command_id="8" id="21">
           <myo-pose type="FIST"/>
           <myo-pose type="FINGERS_SPREAD"/>
           <name>Lights in 304d</name>
        </combo>
        <combo command_id="11" id="22">
           <myo-pose type="FIST"/>
           <myo-pose type="WAVE_IN"/>
           <name>Sockets in 304d</name>
        </combo>
     </room>
     <room id="4"/>
  </rooms>
```

Appendix C
Bluetooth Low Energy measurements

C.1 RSSI depended on rotation

Angle	Average
0	53,4736842105263
22,5	48,2631578947368
45	49,9473684210526
67,5	53,3684210526316
90	57,6315789473684
112,5	55,3157894736842
135	55,3157894736842
157,5	48,6315789473684
180	47,1052631578947
202,5	51,2105263157895
225	57,6315789473684
247,5	59,7894736842105
270	61,421052631579
292,5	48,5263157894737
315	41,6842105263158
337,5	38,1578947368421
360	40,1578947368421

Table C.1. X axis

Angle	Average
0	45,2631578947368
22,5	45,7894736842105
45	48,4210526315789
67,5	47,1578947368421
90	48,7894736842105
112,5	42,7368421052632
135	42,6315789473684
157,5	42,1578947368421
180	43
202,5	55,9473684210526
225	52,0526315789474
247,5	51,1578947368421
270	48,6842105263158
292,5	44,7368421052632
315	42,4210526315789
337,5	50,1578947368421
360	51,0526315789474

Table C.2. Y axis

Angle	Average
0	37,7894736842105
22,5	37,5263157894737
45	42,2105263157895
67,5	44
90	39,7368421052632
112,5	35,6842105263158
135	40,1578947368421
157,5	40,5263157894737
180	31,4736842105263
202,5	32,8421052631579
225	35,5263157894737
247,5	36,5789473684211
270	37,4736842105263
292,5	41,1578947368421
315	45,9473684210526
337,5	44,8947368421053
360	46,5263157894737

Table C.3. Z axis

C.2 RSSI depended on obstacles

No. of measurement	Average	Minimum	Maximum	Variance
1	74,75	68	86	36,20
2	71,05	66	80	28,89
3	75,6	66	95	69,73

Table C.4. Devices placed 2 meters from each other with human placed in middle.

No. of measurement	Average	Minimum	Maximum	Variance
1	79,1	73	90	21,04
2	83,65	70	96	44,45
3	75,85	64	90	33,61

Table C.5. Devices placed 2 meters from each other with human placed next to the heartbeat sensor.

No. of measurement	Average	Minimum	Maximum	Variance
1	67,2	63	70	3,01
2	66	61	70	11,79
3	71,8	58	88	69,22

Table C.6. Devices placed 2 meters from each other with no obstacles.

No. of measurement	Average	Minimum	Maximum	Variance
1	67,85	64	71	2,87

Table C.7. Devices placed 2 meters from each other with wall in between.