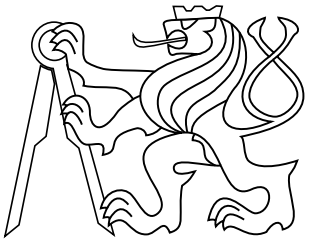
diplomová práce

Optimalizace lokalizačního systému

Bc. Martin Cihlář



Květen 2016

Vedoucí práce: Ing. Jaromír Hrad, Ph.D.

České vysoké učení technické v Praze
Fakulta elektrotechnická, Katedra telekomunikační techniky

Poděkování

Rád bych touto cestou poděkoval vedoucímu práce Ing. Jaromíru Hradovi, Ph.D., za ochotu zajistit mi všechny klíčové prvky při řešení práce, také za odborné vedení a řadu cenných připomínek.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

V Praze dne 27. května 2016 .....

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra telekomunikační techniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Martin Cihlář**

Studijní program: Komunikace, multimédia a elektronika
Obor: Sítě elektronických komunikací

Název tématu: **Optimalizace lokalizačního systému**

Pokyny pro vypracování:

Optimalizujte lokalizační systém využívající stávající infrastrukturu bezdrátové lokální sítě pro účely logistiky, zdravotnictví a krizového managementu. Při řešení se zaměřte zejména na zpřesnění lokalizačního algoritmu a na možnost monitorování pohybu většího počtu klientských zařízení z dohledového centra.

Seznam odborné literatury:

[1] Kurečka, A: *Vývoj embedded modulu pro podporu lokalizace průzkumného vozidla.* Ostrava, 2011. Bakalářská práce. VŠB - Technická univerzita Ostrava.
[2] Garcia-Valverde, T.; Garcia-Sola, A.; Hagras, H.; Dooley, J.A.; Callaghan, V.; Botia, J.A.: *A Fuzzy Logic-Based System for Indoor Localization Using WiFi in Ambient Intelligent Environments.* IEEE Transactions on Fuzzy Systems, vol.21, no.4, pp.702-718, 2013.

Vedoucí: Ing. Jaromír Hrad, Ph.D.

Platnost zadání: do konce letního semestru 2016/2017

L.S.

prof. Ing. Boris Šimák, CSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 21. 12. 2015

Abstrakt

Tato diplomová práce se zabývá problematikou lokalizace mobilních zařízení v prostředí budov, vhodnou pro využití např. ve zdravotnictví či krizovém managementu. Cílem práce je návrh kompletního lokalizačního systému na bázi WiFi technologie. Východiskem jsou poznatky předchozí bakalářské práce [25], zaměřené rovněž na lokalizační systémy. Navrhovaný lokalizační systém využívá lokalizační metodu *Localization Fingerprinting*. Základem systému je algoritmus na bázi fuzzy logiky, která je obecně popsána odvozovacím modelem. Popis algoritmu pro dynamické generování tohoto modelu je také součástí práce. Navrhovaný lokalizační systém sestává z pěti prvků. Jsou to dvě mobilní aplikace pro OS Android, z nichž první je navržena pro uživatele a druhá pro administrátora. Dále je to již zmíněný fuzzy odvozovací model. Dalším prvkem je webové dohledové prostředí pro zobrazení výsledků lokalizace. Všechny tyto prvky využívají databázi a sdílené úložiště na virtualizovaném serveru s OS Ubuntu, což je zároveň poslední prvek systému. V závěru práce je vyhodnocena úspěšnost celého navrženého systému na základě jeho testování ve dvou rozdílných prostorech.

Klíčová slova

FIS, fluktuace signálu, fuzzy logika, Localization Fingerprinting, lokalizační systém, mobilní aplikace, OS Android, simulované žíhání, RSSI, RTLS.

Abstrakt

This diploma thesis deals with the issue of locating of mobile devices within the buildings, suitable for usage in a healthcare or in a crisis management. The purpose of the work is the design of a complete localization system based on the WiFi technology. This system is based on the knowledge from previous bachelor thesis also focussed on localization systems. The proposed localization system uses Localization Fingerprinting method. The basis of this system is an algorithm based on the fuzzy logic which is generally described by an inference model. The description of the algorithm for generating of this model is also part of the work. The proposed localization system is formed by the five elements. There are two mobile applications for OS Android. The first one is designed for an user and the second is for an administrator. The next element is the early mentioned fuzzy inference model. There is also the supervision web interface for displaying the localization results. All of these elements use a database and the shared storage on the virtual server with OS Ubuntu which is also the last element of the system. Finally, the two different places were used to avaluate the system localization performance.

Keywords

FIS, signal fluctuation, fuzzy logic, Localization Fingerprinting, localization system, mobile application, OS Android, simulated annealing, RSSI, RTLS.

Obsah

Zkratky

AP	*Access Point (přístupový bod)*
CA	*Certification Authority (certifikační autorita)*
FIS	*Fuzzy Inference System (fuzzy odvozovací systém)*
kNN	*k-Nearest Neighbours (algoritmus nejbližších sousedů)*
OVF	*Open Virtualization Format (universální formát pro virtuální stroj)*
RSSI	*Received Signal Strength Indicator (indikátor přijaté úrovně signálu)*
SA	*Simulated Annealing (algoritmus simulovaného žíhání)*
SSID	*Service Set Identifier (identifikátor bezdrátové sítě WLAN)*
SSL	*Secure Sockets Layer (protokol pro zabezpečení síťové komunikace)*
TLS	*Transport Layer Security (protokol pro zabezpečení síťové komunikace)*
WLAN	*Wireless Local Area Network (lokální bezdrátová síť)*

1. Úvod

Za poslední dvě dekády se problematika lokalizace vyvinula v plnohodnotný vědecký obor. Prvotním hnacím motorem tohoto vývoje bylo zdokonalení bezdrátové datové komunikace mezi dvěma zařízeními a miniaturizace jejich fyzické velikosti. Na základě těchto dvou předpokladů už nebylo složité sledovat pohybující se objekt v prostředí a díky tomu mohly vzniknout první komerční produkty. Dalším klíčovým mezníkem bylo uvedení chytrých mobilních zařízení, zejména chytrých telefonů. Tím se lokalizace otevřela daleko většímu trhu, resp. libovolnému člověku, bez nutnosti nakupovat speciální lokalizační zařízení. Nyní obor lokalizace čeká další milník, a tím je internet věcí. Na jeho základě mohou vzniknout různorodé aplikace využívající právě pozice osob či zařízení. Jak už předchozí věty napověděly, hlavním zaměřením této práce jsou lokalizační systémy využívané ve vnitřních prostorech budov. Mohou to být budovy škol, muzeí, nemocnic, ale i firemní objekty, jako jsou například sklady.

Nejčastějším příkladem pro lokalizaci v rámci budov (angl. *indoor localization*) je prostředí nemocnice. Nemocniční areál je obecně rozsáhlé a členité prostředí, kde se může vyskytnout potřeba lokalizace pacienta nebo nemocničního vybavení. S rozvojem robotiky se do nemocnic dostává automatizace, jako je například produkt EVOcart™[26]. Jeho účelem je zmenšit zatížení personálu rutinními pracemi, jako je převážení odpadu, špinavého prádla apod. Právě taková zařízení mohou být cílem lokalizace. Samozřejmě to nemusejí být jen roboti, může to být nějaký drahý přístroj, jehož poloha může být takto kontrolována. Je jen otázkou času, kdy budou pomocí robotů převáženi i pacienti. Lokalizace nemusí být využívána pouze pro účely sledování, ale naopak může být použita i pro navigaci, ať už v nemocničním areálu, muzeu či obchodním centru. Další aplikací může být například krizový management. To by v praxi mohlo znamenat, že se pro konkrétní budovu uchovávají pozice všech přítomných osob. V případě nějakého neštěstí (např. požáru) by záchranáři mohli na základě posledních lokalizačních dat předvídat, kolik osob je v objektu, popřípadě kde byly osoby lokalizovány při vypuknutí požáru.

Jak už bylo zmíněno, s příchodem internetu věcí a v souvislosti s tím i chytrých budov, nabývá *indoor* lokalizace na důležitosti. Možné aplikace jsou přiblíženy na následujících dvou případech. Prvním případem je standardní firemní kancelář. Zde se například může regulovat klimatizace dle počtu osob v zasedací místnosti, v závislosti na přítomnosti zaměstnance (zaměstnanců) lze také automaticky regulovat osvětlení a žaluzie. Druhým případem je firemní areál, konkrétně sklad. Na základě lokalizace vysokozdvižného vozíku zde může být automaticky regulováno např. osvětlení ve skladu, což by mohlo vést k úspoře energie.

1.1. Motivace a cíle této práce

Motivace pro vznik této diplomové práce je založena na několika skutečnostech. Byla to především snaha o vytvoření nového typu lokalizačního systému, který staví na poznatcích z předchozí bakalářské práce [25]. Dále se jednalo o vytvoření aplikace, která by mohla být součástí trendu posledních let, a to internetu věcí a chytrých budov.

Hlavním cílem této práce je vytvoření lokalizačního systému na bázi lokální bezdrátové sítě, který by byl schopen lokalizovat osobu či zařízení v rámci budovy. Základem navrhovaného systému je optimalizace předešlého lokalizačního systému. Ta spočívá v tom, že jsou vyhodnoceny a eliminovány problémy spojené s předchozí verzí lokalizace. Dalším cílem je nalezení nového algoritmu pro zlepšení přesnosti lokalizace. Všechny poznatky posléze vedou k návrhu a implementaci nového systému, který by měl být co nejjednodušší na reálné nasazení a poskytovat základní funkcionalitu v podobě lokalizace uživatele s chytrým telefonem.

1.2. Členění textu

Diplomová práce je rozdělena do následujících kapitol: Kapitola 2 uvádí problematiku lokalizace a lokalizačních metod v rámci lokalizace v budovách. Obsahuje také shrnutí poznatků z předešlé bakalářské práce a návrhy principů pro eliminaci klíčových problémů. Kapitola 3 se zabývá teoretickým pozadím nového lokalizačního algoritmu na základě Fuzzy logiky a způsobem jejího využití. Kapitola 4 popisuje postup vytváření výpočetního jádra celého systému, jímž je fuzzy odvozovací systém. Závěrem kapitoly jsou popsány dvě metodiky pro optimalizaci tohoto jádra. Důležitou částí práce je i kapitola 5, která popisuje navrhovaný systém nejen jako celek, ale věnuje se i návrhům a implementaci jeho jednotlivých součástí. Kromě toho zahrnuje i testování úspěšnosti lokalizace. Poslední kapitola 6 shrnuje dosažené výsledky.

2. Lokalizace v prostředí WLAN

2.1. Lokalizace v prostředí budov

Asi nejznámějším pojmem v oboru lokalizace je pojem *Global Positioning System* (GPS), jehož aplikace se vyskytují téměř všude. Například u automobilových navigací, u navigací pro pěší turistiku, pro lodní a letecký průmysl apod. Nicméně, všechna tato využití mají společnou jednu věc, jedná se o lokalizaci venkovní. Ideálním případem tedy je, když mezi pozemní GPS stanicí a třemi až čtyřmi satelity je přímá viditelnost. To však nelze ve vnitřních prostorech budovy zajistit. Jelikož je čas šíření signálu pro GPS stěžejním parametrem, výsledná přesnost lokalizace je v rámci budovy velmi malá, za předpokladu, že vůbec dojde k navázání spojení mezi pozemní stanicí a satelitem. Z tohoto důvodu se začalo využívat jiných dostupných technologií pro *indoor* lokalizaci. Mezi ně lze řadit Bluetooth, RFID, WiFi, ZigBee, DVB-T, ultrazvuk či infračervené záření.

Různé druhy lokalizačních systémů mají i různou přesnost, která je daná stupněm lokalizace. Výčet jednotlivých stupňů je takovýto [27]:

- lokalizace na úrovni zóny - je známá zóna, kde se uživatel nachází, zóna může být definována jako místnost, více místností nebo například část budovy,
- lokalizace na úrovni místnosti - je známa místnost, kde se uživatel nachází,
- lokalizace na úrovni části místnosti - pro velké pokoje je možno určit například část, kde se uživatel nachází,
- lokalizace na základě průchodu - pozice uživatele je dána průchodem přes kontrolní bránu, ta může být dána např. dveřmi vybavenými senzory
- lokalizace vztažená k bodu - výsledkem lokalizace je popis polohy uživatele vzhledem ke známému bodu,
- lokalizace precizní - poloha uživatele je dána konkrétními souřadnicemi v daném prostoru.

Na základě výše uvedených stupňů lokalizace je odvozena i přesnost systému, která vyjadřuje, s jakou odchylkou od skutečné pozice je systém schopen určit pozici uživatele. To znamená, že pro lokalizaci na úrovni zóny, na kterou se tato práce zaměřuje, nelze očekávat výsledek s centimetrovou přesností.

2.1.1. Lokalizační techniky

Lokalizační techniky vycházejí z měření charakteristik signálu. Buď to může být čas trvání od vyslání informace k jejímu příjmu, nebo úhel, v jakém byl signál přijat

přijímací anténou, anebo to může být úroveň přijatého signálu.

1. **TOA** (*Time of Arrival*) je technika využívající doby šíření signálu mezi vysílačem a přijímačem. Doba je počítána od pevně stanoveného (a shodného pro obě zařízení) času t_0. Je tedy nutné, aby obě zařízení byla přesně časově synchronizována, a to jak síťové prvky (AP), tak lokalizovaná zařízení (chytré telefony, WiFi přívěsky, apod.). Daleko větším nedostatkem je citlivost na různou dobu šíření signálu. Jelikož se jedná o vnitřní prostory budovy, je šíření signálu ovlivněno překážkami. Nastává tak jev vícecestného šíření, kde signál může být přijat z různého směru, s různou úrovní, ale také s různým časovým zpožděním. Tato metoda je tedy vhodná pro venkovní případy, kde není signál ovlivňován tolika překážkami. Samotný výpočet pozice probíhá pomocí algoritmu trilaterace (někdy také nesprávně nazývané triangulací), což je známý způsob určování průniku několika kružnic.

2. Na podobném principu funguje i **TDOA** (*Time Distance of Arrival*). Zde však není nutné synchronizovat i lokalizované zařízení. Technika je založena na tom, že zařízení pošle informaci, kterou přijmou všechny přístupové body. Na základě rozdílných časů příchodu zprávy ke každému AP a jejich známé polohy je vypočtena poloha zařízení. Rovněž tato technika je citlivá na dobu šíření signálu. I zde se využívá algoritmu trilaterace.

3. **AOA** (*Angle of Arrival*) je technika založená na měření úhlu, ve kterém byl signál přijat vůči referenčnímu bodu. Pro účely *indoor* lokalizace je tato technika opět stěží použitelná, protože se signál šíří více směry. Není tedy zaručena stálost směru, ze kterého signál přichází. Pro výpočet pozice se využívá algoritmus triangulace.

4. **RSSI** (*Received Signal Strength Indicator*) je technika založená na měření úrovně signálu v daném bodě budovy. Jelikož se úroveň signálu se vzdáleností mění jak přirozeným způsobem, tak působením prostředí, výsledkem takového měření je RSSI charakteristika celého prostředí. Pro každý AP je tato charakteristika samozřejmě jiná. Charakteristiku z trénovací fáze lze poté porovnávat s konkrétním měřením ve fázi lokalizace. Algoritmů pro výpočet pozice je pro tuto techniku více. Může to být metoda nejbližšího souseda, resp. její vylepšená varianta k nejbližších sousedů (kNN). Mohou být využity neuronové sítě, genetické algoritmy, metoda clusterování dat a další algoritmy, které dokáží porovnávat dvě různá měření a hledat v nich možné podobnosti.

2.1.2. Lokalizační metoda *Localization Fingerprinting*

Metoda pozičních otisků [28] je asi nejlepším řešením pro lokalizaci uvnitř budov. Základem této metody jsou dvě fáze. První je tzv. *offline*, která se někdy nazývá trénovací fází. Druhá fáze je tzv. *online*, kterou lze označit jako fázi vlastní lokali-

zace (kdy je lokalizováno zařízení). Hlavním cílem první fáze je vytvoření trénovací množiny dat, která charakterizuje signálové pokrytí daného prostoru. Tvorba dat se provádí tak, že je v daném prostoru (kde se bude posléze lokalizovat) určeno několik měřících bodů, kde dochází ke skenování RSSI hodnot od okolních přístupových bodů. Skenování provádí administrátor nebo pověřená osoba, nikoliv však lokalizovaný uživatel, který tím tak nemusí být obtěžován. Tímto způsobem se projde celý prostor a výsledná trénovací množina posléze slouží k porovnávání s hodnotami RSSI, které se skenují ve fázi vlastní lokalizace. Počet RSSI hodnot, resp. počet AP od kterých jsou tyto hodnoty přijímány, může být libovolný, měly by však být alespoň tři. Ideální počet je čtyři až šest. Při více RSSI hodnotách už dochází k velké variabilitě výsledků, a tedy horší rozlišitelnosti správného výsledku. K výpočtu výsledné pozice slouží RSSI technika již výše popsaná. Lokalizační systém lze tedy obecně popsat několika prvky, a to zařízením, které vytváří trénovací množinu dat, zařízením (identifikátorem), který je lokalizován, metodou lokalizace a popřípadě serverem, který tvoří jádro systému.

2.2. Výstupy předešlé práce

Výstupem předchozí práce [25] byly dvě uživatelské aplikace na mobilní zařízení, které tvořily lokalizační systém na bázi metodiky *Localization fingerprinting*. Šlo o precizní typ lokalizace, tzn. že byla schopna určit konkrétní souřadnice v prostoru. Nicméně, její přesnost byla cca 3 metry, což je samozřejmě pouze průměrná hodnota. Z tohoto pohledu je průměrná přesnost lokalizace irelevantní, resp. nedá se použít takovým způsobem, jak by se asi od precizní lokalizace očekávalo. Tento závěr je dán zejména technologií, která je pro lokalizaci použita, a to WiFi.

Tato technologie není pro precizní lokalizaci vhodná, její přesnost lze sice zlepšit na úroveň okolo $1,5$ metru, ale k tomu je potřeba buď laboratorních podmínek, anebo zbytečně složitého algoritmu. Tyto závěry platí samozřejmě pro prostředí budov, kde není dosažena přímá viditelnost PC-AP. Na základě toho bylo provedeno přehodnocení cílů lokalizace a aktuální lokalizační systém je navržen jako lokalizace na úrovni zóny. Jelikož zůstal požadavek pro využití pouze WiFi technologie, jiná možnost nepřipadala v úvahu. Tato změna však není krokem zpět. Z čistě praktického hlediska je ve většině případů vyžadována pouze lokalizace na úrovni zóny, není například nutné sledovat pacienta, jestli leží na posteli, anebo se pohybuje po místnosti. Je pouze důležité, aby byla známa místnost (zóna), kde se momentálně nachází. Další věcí je to, že je zaručena větší úspěšnost lokalizace jako takové, protože zóna je daleko lépe určitelná (odvoditelná) než konkrétní souřadnice.

2.2.1. Rozdílnost WiFi adaptérů

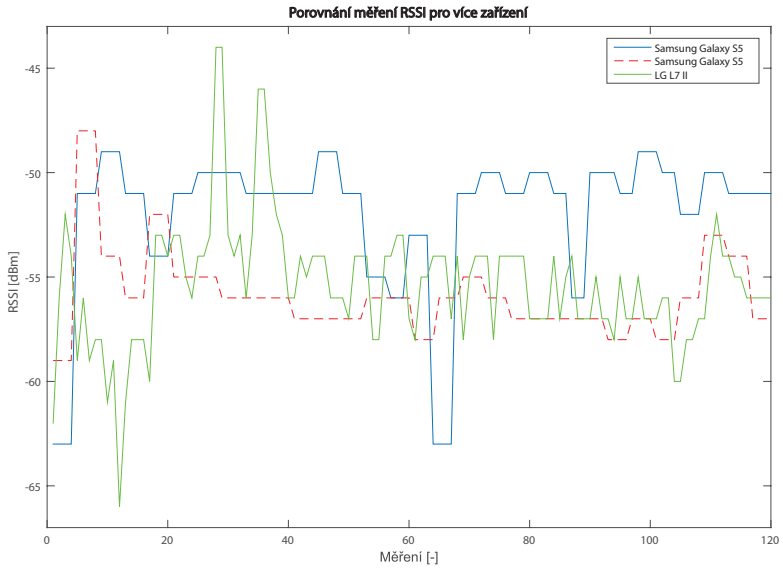
Důležitým zjištěním z předešlé práce je rozdílnost jednotlivých WiFi adaptérů. Tento problém se vyskytuje jak při porovnání zařízení odlišných výrobců, tak i při porovnání zařízení stejného výrobce. Bohužel nemusejí být shodné ani WiFi adaptéry ve stejné modelové řadě. Z provedených testovacích měření se hodnota RSSI

mezi několika zařízeními lišila dokonce i o 20 dBm na totožném místě v prostoru a při stejné orientaci zařízení. Důsledek je takový, že přesnost lokalizačního systému na bázi WiFi technologie je zcela degradována. Konkrétně se jedná o narušení samotného principu, který je založen na tom, že první zařízení měří hodnoty RSSI (trénovací fáze) vztažené k prostoru a druhé zařízení (fáze lokalizace) porovnává své naměřené hodnoty s hodnotami trénovací fáze. Kvůli rozdílnosti v RSSI mezi zařízeními však nedochází ke správnému porovnání.

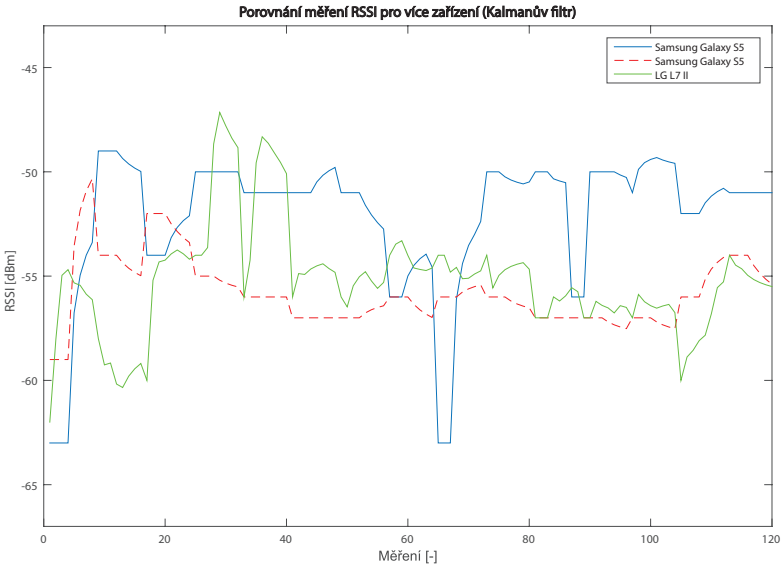
Pro eliminaci tohoto problému je využita metoda *Signal Strength Difference* (SSD) [1]. Princip spočívá v tom, že namísto zpracování konkrétní hodnoty RSSI od daného AP se zpracovává rozdíl dvou hodnot RSSI od dvou přístupových bodů. Jednotlivé rozdílové RSSI tedy budou definovány jako $\Delta_{AP1-AP2}, \Delta_{AP2-AP3}$ apod. Pokud se měří signál od N přístupových bodů, bude dané měření definováno $N-1$ rozdílovými hodnotami RSSI. Pro dosažení požadované přesnosti navrhovaného systému bude využito pět rozdílových hodnot, tedy šest přístupových bodů.

Výsledky z dalšího měření RSSI jsou na obrázku 1, resp. 2, kde jsou porovnána tři zařízení, jež měřila úroveň signálu ve stejný čas a na téměř stejném místě. Jelikož se nejednalo o totožné místo, mohou být odchylky měření samozřejmě způsobeny právě touto chybou měření. Obrázek však slouží k vysvětlení jiného problému. Z obrázku 1 je zřejmý schodovitý tvar průběhu hodnot u obou telefonů značky Samsung v porovnání s telefonem LG. Schodovitý tvar je způsoben tím, že po jistou dobu má úroveň signálu stejnou hodnotu. RSSI data se měřila v intervalu jedné vteřiny, což by měla být minimální možná doba, za kterou se RSSI hodnoty mění. Jinými slovy, OS Android má přednastavený systémový parametr, který definuje interval, v němž se provádí měření RSSI všech okolních WLAN sítí. Tento parametr se pro každé zařízení bohužel může lišit, ať už proto, že je od jiného výrobce, nebo že jde o jinou verzi OS Android. Způsob, kterým aplikace získává RSSI hodnoty, tedy není přímo spouštění skenování okolních WLAN, ale pouze přístup k těmto výsledkům, které se automaticky mění v závislosti na hodnotě parametru (interval měření). Pro testovací telefony Samsung je tento parametr nastaven na čtyři vteřiny, proto se po první čtyři vteřiny skenování RSSI hodnota nemění, oproti telefonu LG, kde se mění každou vteřinu. Jelikož se takto skenuje určitý počet hodnot, které se poté průměrují (pro lepší výsledek), byly v případě telefonů Samsung naměřeny za osm vteřin pouze dvě různé hodnoty, pro LG jich bylo naopak osm. Tento nepoměr různých hodnot samozřejmě ovlivní výsledný průměr.

Z tohoto důvodu je nutné před samotným měřením provést kalibraci zařízení. Princip kalibrace je založen na tom, že se provede minutové kalibrační měření, ze kterého se zjistí systémový parametr měření RSSI. Dle toho je poté přizpůsoben interval skenování daného zařízení při zachování stejného počtu naměřených hodnot pro všechna zařízení. V praxi to znamená, že je za minutu zjištěna hodnota systémového parametru, například pět vteřin. Průměr se počítá z pěti hodnot, tzn. že je skenování dlouhé dvacet pět vteřin pro konkrétní zařízení. Pro jiné zařízení se může samozřejmě lišit v závislosti na systémovém parametru. Tento výsledek ovlivní také interval stanovení pozice zařízení, ne však jeho přesnost.



Obrázek 1. *Porovnání měření RSSI pro více zařízení*



Obrázek 2. *Porovnání měření RSSI pro více zařízení (Kalmanův filtr)*

2.2.2. Fluktuace signálu

Největší slabinou lokalizace na bázi WiFi signálu je jeho proměnlivost v čase. Pokud se jedná o lokalizaci uvnitř budov, je fluktuace ještě výraznější v důsledku vícecestného šíření signálu. Bohužel je to přirozený jev, který nelze zcela eliminovat, a proto je nutné alespoň ho minimalizovat. Prvotní verze lokalizačního systému fungovala na základě měření několika hodnot úrovně signálu, které byly posléze zprůměrovány. Je to asi ten nejjednodušší algoritmus na minimalizaci fluktuace, který je však stále z velké části ovlivněn extrémy naměřených hodnot. To znamená, že výsledek může být často ovlivněn výpadkem signálu, resp. jeho velmi nízkou hodnotou. Z to-

hoto důvodu je nutné, aby takovéto extrémy byly z naměřených dat odfiltrovány. Rozhodně je špatné řešení, pokud by tyto extrémní hodnoty byly zcela odfiltrovány, tedy úplně smazány. Výsledná hodnota by poté byla odvozena z menšího počtu hodnot, než je žádoucí.

V současném systému je pro minimalizaci fluktuace použit Kalmanův filtr. Jedná se o adaptivní filtr, který popisuje stavy diskrétního systému v závislosti na aktuálním měření. Je založený na postupné iteraci, kde se v každém kroku změní charakteristika filtru na základě vstupních hodnot. Hlavním přínosem tohoto principu je to, že se historické hodnoty měření nemusejí ukládat. Samotná filtrace je rekurentní a je složená ze dvou částí, a to z predikce (časová aktualizace, vztahy (1), (2)) a z filtrace (aktualizace měření, vztahy (3), (4), (5)), [23].

$$\widehat{x}_k^- = A\,\widehat{x}_{k-1} + B\,u_{k-1} \tag{1}$$

$$P_k^- = A\,P_{k-1}\,A^T + Q \tag{2}$$

První část (predikce) predikuje novou stavovou hodnotu $\widehat{x}_k^-$ (střední hodnotu měření) a kovariační matici P_k^- z předcházejícího stavu $\widehat{x}_{k-1}$, resp. P_{k-1}. Pro první iterací platí, že se hodnoty $\widehat{x}_{k-1}$ a P_{k-1} musejí inicializovat. Hodnoty $\widehat{x}_{k-1}$ a P_{k-1} jsou však i předešlými hodnotami, které jsou na základě druhé části algoritmu (filtrace) filtrovány na výsledné hodnoty. Před samotnou filtrací dochází ještě k aktualizaci charakteristiky filtru ($\widehat{x}_k$, P_k) novou změřenou hodnotou z_k.

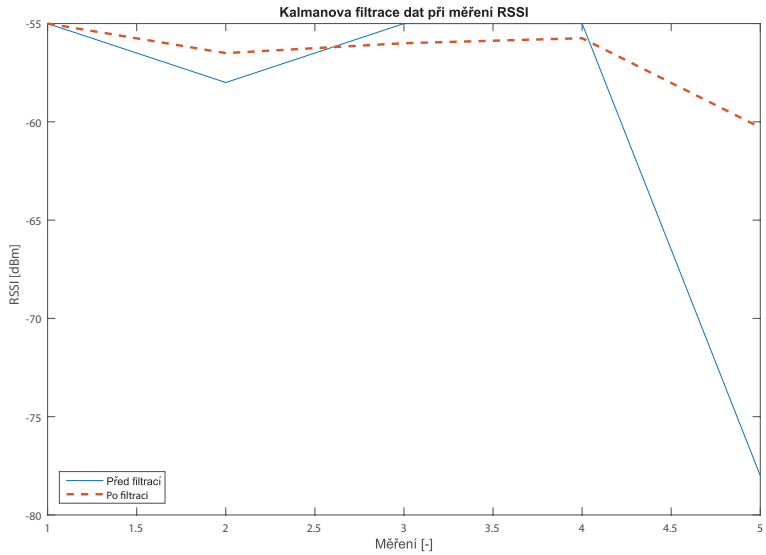
$$K_k = P_k^-\,H^T(H\,P_k^-\,H^T + R)^{-1} \tag{3}$$

$$\widehat{x}_k = \widehat{x}_k^- + K_k(z_k - H\,\widehat{x}_k^-) \tag{4}$$

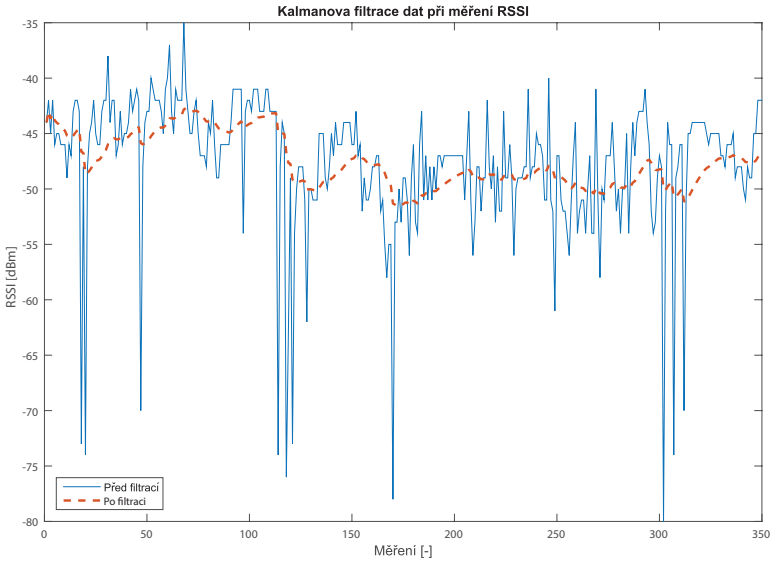
$$P_k = (I - K_k\,H)\,P_k^-\,A^T + Q \tag{5}$$

Matice A je stavová matice, která udává vztah mezi $\widehat{x}_{k-1}$ a $\widehat{x}_k$, B je kontrolní matice vstupu. Q je matice obsahující kovariační šum a R je matice obsahující šum měření, kde oba typy šumu jsou na sobě nezávislé, jsou bílé a s normálním rozložením pravděpodobnosti. Matice K je zisk Kalmanova filtru, který minimalizuje výslednou kovariační chybu danou maticí P_k. Matice H definuje vztah mezi aktuálním stavem a naměřenou hodnotou.

Praktická aplikace Kalmanova filtru je znázorněna na obrázku 3, resp. 4. V prvním případě byla filtrace použita na pět naměřených RSSI hodnot, ve druhém případě na 350 hodnot. Jak je vidět na druhém obrázku, Kalmanova filtrace odfiltrovává extrémní hodnoty fluktuace signálu, což je hlavní pozitivum. Lokalizační systém používá pro stanovení jedné RSSI hodnoty však jen několik hodnot, zpravidla se jedná o jednotky. Na obrázku 3 lze vysvětlit hlavní rozdíl mezi průměrováním a filtrací hodnot. Výsledek průměru surových dat je v tomto případě $-60,2\,dBm$, což je zároveň poslední hodnota Kalmanovy filtrace (čárkovaně). Plná čára spojuje hodnoty naměřené před filtrací (surová data). Pokud se ale vypočte průměr z filtrovaných dat, výsledek bude $-56,1\,dBm$, tedy hodnota, která není v takové míře závislá na extrémech naměřených dat. Proto bude v navrhovaném systému využit tento způsob skenovaní RSSI hodnot.



Obrázek 3. *Filtrace RSSI dat Kalmanovým filtrem*



Obrázek 4. *Filtrace RSSI dat Kalmanovým filtrem*

2.2.3. Časová náročnost vytváření mapy pokrytí

Nepříjemnou částí lokalizačního systému na bázi RSSI je tvorba kvalitní mapy pokrytí prostoru, kde bude lokalizace probíhat. Pro předchozí verzi lokalizačního systému (na bázi *localization fingerprinting*) bylo charakteristické, že prostor o rozloze například 400 m^2 může být definován 200 trénovacími záznamy, což při 15 vteřinách na jeden záznam spotřebuje necelou hodinu, nemluvě o stavu baterie zařízení.

Obecně lze časovou náročnost minimalizovat třemi způsoby [2]. Buď se sníží počet trénovacích hodnot, nebo se sníží čas tvorby trénovacího záznamu, anebo se využije určitý typ modelace potřebných hodnot. Poslední možnost je samozřejmě nejpří-

hodnější, protože při správném postupu má nejmenší vliv na chybovost lokalizace. Většina těchto algoritmů je však založena na teoretické modelaci prostředí, která ale zcela nereflektuje skutečné pokrytí. Ostatní algoritmy se snaží využít fragment trénovacích dat pro modelování zbylých hodnot. Navrhovaný lokalizační systém je však založen na odlišných principech, které nevyžadují takové množství trénovacích záznamů. Není tedy třeba v daném prostředí definovat matici dat o $M\ x\ N$ pozicích (cca $1,5\,m$ vzdálených od sebe), kde bude měření probíhat. Stačí v definovaných zónách (pokoj apod.) provést měření nezávislé na konkrétní pozici, tzn. že se v pokoji stanoví například čtyři pozice, ve kterých budou naměřena trénovací data. Trénovací data lze pro danou zónu naměřit i pouze z jedné pozice, v takovém případě však nemusí být RSSI charakteristika tak kvalitní, a proto se takový postup nedoporučuje. Dosavadní zkušenosti ukazují, že je dobré měřit trénovací data tak, aby pro cca $2,5\,m^2$ byla naměřena jedna, popř. dvě hodnoty, přičemž stačí, když se druhé měření provede ze stejné pozice jako to první. V takovém případě se však doporučuje pootočit se při druhém měření např. o 90 stupňů, aby se minimalizovala chyba měření způsobená interferencemi přítomnou osobou.

2.2.4. Omezená dynamika lokalizace vůči změnám prostředí

Při použití lokalizační metody *localization fingerprinting* může být problémem proměnné prostředí. Tím je míněno to, že v trénovací fázi se naměří trénovací množina dat, která charakterizuje dané prostředí. Ve skutečnosti ale tato charakteristika v daném prostoru nezůstává stejná. Mění se v závislosti na denní době, tzn. že v pracovní době, kdy je prostor plný lidí, může být charakter prostředí trochu jiný. Závisí také na denní době jako takové, přes noc nebo přes den může být v prostoru větší teplo a vlhko, což má také vliv na charakteristiku signálového prostředí. Toto jsou výkyvy zejména střednědobého charakteru, řádově jednotky hodin. Dalším typem jsou krátkodobé výkyvy. Ty jsou způsobeny zejména dynamikou prostředí, tzn. pohybem osob, otevírám dveří apod. Posledním typem jsou dlouhodobé výkyvy, které jsou způsobené přestěhováním velkých objektů, rekonstrukcí kanceláře apod.

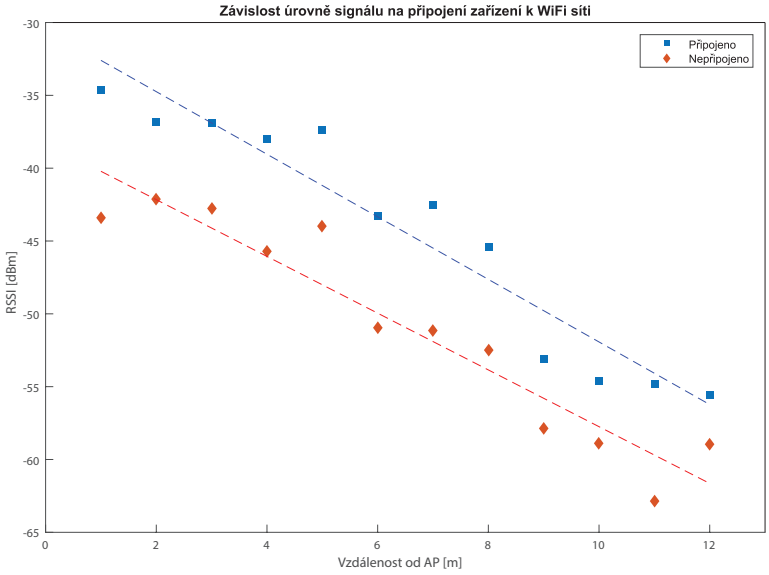
Dlouhodobé výkyvy lze jako jediné jednoduše vyřešit tím, že po velké změně v prostředí (která není tak častá) se znovu vytvoří trénovací množina dat. Dynamické výkyvy se velmi těžko modelují. Naopak výkyvy střednědobého charakteru se mohou buď modelovat, anebo ještě lépe průběžně kontrolovat. To by mohlo být provedeno například pomocí kontrolních sond, které by neustále skenovaly okolní RSSI data a na základě odlišností vůči normálnímu stavu by se trénovací množina mohla dynamicky přepočítávat právě na základě naměřených odlišností. Tento návrh není součástí této práce, ale je to možná cesta ke zlepšení výkonu lokalizačního systému.

2.2.5. Rozdíl RSSI hodnot při (ne)konektivitě s AP

Jak už bylo zmíněno v [25], po připojení zařízení k WiFi síti konkrétního AP může docházet ke zvyšování výkonu AP a tím i naskenované RSSI hodnoty. Tento jev ovlivňuje korektní tvorbu trénovacích dat či klientův pokus o lokalizaci. Na obrázku

5 je znázorněn rozdíl mezi připojeným a nepřipojeným zařízením k bezdrátové síti. V obou případech je WiFi adaptér pouze zapnutý, ale zařízení není připojeno ke skenovanému SSID. V takovém případě může být ale připojeno k jinému SSID než je lokalizační SSID, které je proměřováno, avšak za podmínky zmíněné níže.

Dosavadní lokalizační systém byl decentralizovaný, proto nebylo nutné, aby bylo zařízení připojené k lokální bezdrátové síti, ale stačilo, aby mělo zapnutý WiFi adaptér. Navrhovaný systém je však centralizovaným řešením, tzn. že ke zobrazení pozice, distribuce konfiguračních souborů apod. je zapotřebí server. To lze vyřešit tak, že v daném prostředí budou vytvořeny dvě bezdrátové sítě s různými SSID. Jedna veřejná SSID za účelem poskytování standardní konektivity, tedy k připojování uživatelů k internetové síti a také ke komunikaci zařízení se serverem. Druhá neveřejná SSID čistě za účelem vysílání, ke které by se však zařízení nepřipojovalo. Obě tyto sítě by byly vysílány dvoupásmovými vysílači (jedna WLAN síť = jedno pásmo), takže by nedocházelo ke zvyšování výkonu lokalizačního SSID, pokud by k němu bylo lokalizované zařízení připojeno.



Obrázek 5. *Naměřená úroveň signálu na připojení zařízení k WiFi síti*

2.2.6. Útlum RSSI okolím

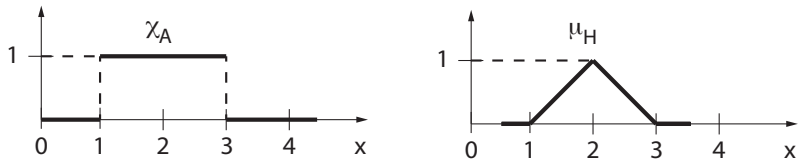
Měřená hodnota RSSI je také ovlivněna orientací samotného zařízení v prostoru. V neposlední řadě je úroveň signálu ovlivněna také polohou zařízení vůči osobě, která ho obsluhuje. Mobilní zařízení se může nacházet například v ruce před osobou, v kapse či batohu. Eliminace těchto vlivů vyžaduje komplexnější řešení, které ve většině případů přímo ovlivňuje výhody navrhovaného systému. Například by tím mohla být ovlivněna přesnost lokalizace, protože by bylo nutné modelovat orientaci osoby vůči statickým přístupovým bodům. Z těchto důvodů se orientací osoby a zařízení v prostoru práce nebude zabývat, Vychází se z modelového scénáře, kdy uživatel drží zařízení v ruce před sebou, anebo je zařízení volně položeno na stole.

3. Fuzzy logika

Fuzzy logika může být považována za jeden z důsledků vývoje umělé inteligence. Hlavním důvodem jejího rozvoje byla snaha o vytvoření převodní charakteristiky mezi přirozeným lidským úsudkem a exaktním zpracováváním instrukcí výpočetního stroje. Počátkem 60. let může být za průkopníka tohoto oboru považován L. A. Zadeh. Ten vyřešil již zmíněnou transformaci pomocí matematické teorie nepřesnosti, pod jiným názvem také známé jako teorie fuzzy množin či fuzzy logika (z angl. *Fuzzy Set Theory*, resp. *Fuzzy Logic*) [11].

3.1. Fuzzy množina

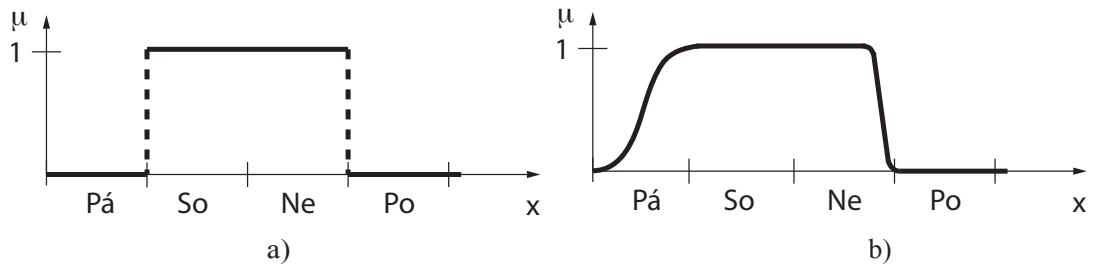
Samotný pojem *fuzzy logika* může být volně přeložen jako neurčitá logika, což je znázorněno i na obrázku 6. Vlevo je standardní logické vyjádření hodnot na intervalu $x = (1, 3)$, které jasně definují pravdu, tedy $\chi_A = 1$ a mimo interval $\chi_A = 0$. Oproti tomu fuzzy logika vpravo definuje pravdu $\mu_H = 1$ pouze pro $x = 2$. Pro ostatní hodnoty intervalu $x = (1, 3)$ může μ_H (oproti standardní logice) nabývat i neurčitých hodnot v intervalu $\mu_H = (0, 1)$. Tyto hodnoty vyjadřují míru (stupeň příslušnosti, viz podkapitola 3.2), s jakou vstupní proměnná x náleží do konkrétní fuzzy množiny (intervalu x).



Obrázek 6. Porovnání logik: vlevo binární, vpravo fuzzy

Zde je nutno vysvětlit pojmy *fuzzy množina a univerzum* a také rozdíl mezi klasickým pojetím množin a fuzzy pojetím množin. Velmi názorné přiblížení rozdílu těchto množin je uvedeno v [12]. Předpokladem je univerzum *víkend*. Pokud bude tato množina pojata klasicky, s jistotou do ní mohou být zařazeny dny sobota a neděle, viz obr. 7a. Na základě toho by výrok: „O víkendu pojedeme na hory“, znamenal, že se tak stane nejdříve v sobotu ráno. Oproti tomu fuzzy univerzum striktně neodděluje jednotlivé prvky univerza (dny v týdnu).

To umožňuje s jistou neurčitostí tvrdit, že už i páteční odpoledne je součástí víkendu. Kdy je páteční odpoledne už víkend, a kdy ne, samozřejmě záleží na kontextu. Například to může záviset na profesi člověka. Pro většinu profesí už to víkend je, ale pro některé ne. Proto můžeme tuto charakteristiku vyjádřit spojitou funkcí (funkce příslušnosti, viz podkapitola 3.2), jako je na obr. 7b. Z tohoto pohledu by

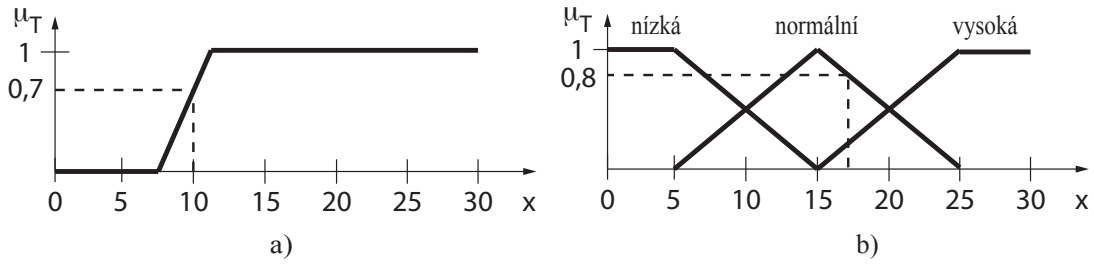


Obrázek 7. Klasická množina vlevo, fuzzy množina vpravo

výrok „O víkendu pojedeme na hory“ znamenal, že se tak může stát už v pátek odpoledne, ale ne každý toho může využít. Fuzzy množina je tedy v tomto případě den v týdnu, je definovaná funkcí příslušnosti a je podmnožinou univerza.

3.2. Proces fuzzifikace

Úlohou tohoto procesu je mapování vstupní hodnoty konkrétní proměnné na hodnotu v intervalu $\mu_T = [0, 1]$, která vyjadřuje tzv. stupeň příslušnosti [13] (z angl. *degree of membership*). Tento princip je znázorněn na obrázku 8a. Vstupní proměnnou je zde teplota, která má hodnotu $x = 10$. Stupeň příslušnosti je tedy $\mu_T = 0, 7$, což je zároveň i výsledek fuzzifikace. Křivka, která určuje jakým způsobem bude vstupní hodnota namapována, se nazývá funkce příslušnosti (z angl. *membership function*). Graf *a* v obr. 8 definuje univerzum proměnné *teplota*.

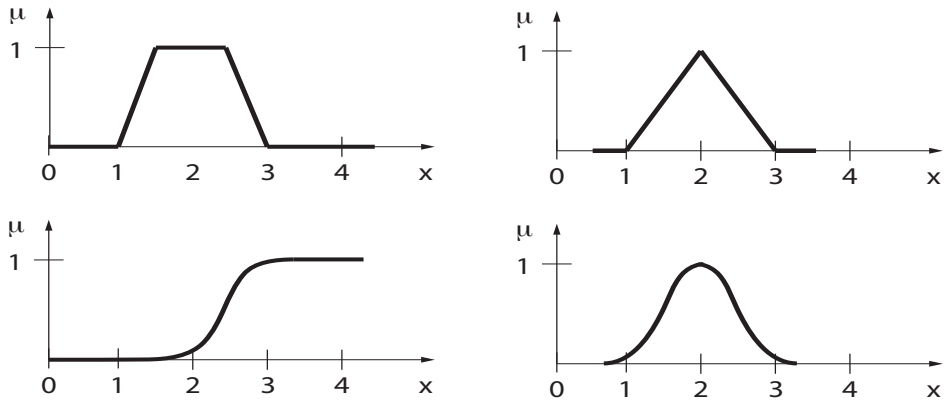


Obrázek 8. Proces fuzzifikace proměnné *teplota*

Ve skutečnosti se nepoužívá pouze jedna funkce příslušnosti pro jednu proměnnou, ale hned několik. Jelikož může teplota nabývat hodnot v širokém intervalu univerza, je dobré si tento interval rozdělit na několik menších. Každý z nich bude poté definován konkrétní funkcí příslušnosti. Na obrázku 8b je příklad takového rozdělení. Lingvistická proměnná *teplota* je tedy definována složením funkcí příslušnosti *nízká, normální* a *vysoká*. Pokud senzor teploty zašle hodnotu $17°C$, výsledkem fuzzifikace bude mapování $\mu_T = 0, 8$ a číselná podoba vstupní hodnoty $(17°C)$ se převede na výraz „normální“. Tímto způsobem se mapují všechny vstupní proměnné daného FIS (podkapitola 3.5).

Pro větší počet funkcí příslušnosti může nastat případ, kdy lze vstupní proměnnou mapovat více způsoby, protože pomyslná kolmice ze vstupní hodnoty může

protínat více než jednu funkci příslušnosti. Závisí to na počtu funkcí a jejich vzájemné pozici. V takových případech se mapuje pomocí funkce, pro kterou vychází větší hodnota μ. Pokud je průsečík v místě, kde se protíná více funkcí, záleží na návrhu systému, pro kterou funkci se algoritmus rozhodne. Jak už bylo zmíněno, interval možných hodnot dané proměnné se člení na několik menších intervalů pomocí funkcí příslušnosti. Počet těchto funkcí záleží na typu konkrétní aplikace. Při více funkcích je dosažena vysoká členitost, tzn. že malá změna hodnoty může zásadně změnit výsledné mapování hodnoty μ a její lingvistický význam. Pro vstupní hodnoty, které nejsou stálé (např. RSSI), je lepší zvolit menší počet funkcí s větší šířkou vzhledem ke fluktuaci RSSI hodnot.



Obrázek 9. Příklady funkcí příslušnosti

Funkce příslušnosti mohou mít různé tvary, např. trojúhelníkový, lichoběžníkový, gaussův, nebo tvar sigmoidy atd. Zmíněné funkce jsou na obrázku 9. K návrhu pozice funkce lze dospět dvojím způsobem. Buď je celé univerzum funkcí navrženo manuálně (symetricky či asymetricky rozmístěné), anebo se může využít algoritmizace. Předpokladem pro dynamický návrh pozic funkcí je naměřený soubor dat jednotlivých proměnných, ve kterém se na základě počtu vstupů hledá stejný počet tzv. *clusterů*. Každý je definován střední hodnotou a rozptylem a podle těchto hodnot se funkce příslušnosti rozmísťují vůči sobě.

3.3. *If-then* pravidla

Nejdůležitější součástí fuzzy algoritmu je jeho rozhodovací schopnost. Ta je založena na databázi tzv. *If-then pravidel*, díky kterým je algoritmus schopen odvodit konkrétní výsledek. Databáze tedy obsahuje nejméně dvě pravidla, aby měl algoritmus aspoň nějaké podklady pro rozhodování. Podrobnější popis využití těchto pravidel fuzzy algoritmem je popsán v podkapitole 3.5. Základní strukturu pravidla (fuzzy výroku) tvoří *podmínka (předpoklad)* a *důsledek (fuzzy implikace, tedy podmíněný fuzzy důsledek)*, viz výraz (6), resp. (7) [15].

$$\textbf{if } x \textit{ is } A \textbf{ then } y \textit{ is } B \tag{6}$$

Zmíněný výraz znamená: pokud je vstupní proměnná x fuzzifikována do fuzzy množiny A, pak výstupní proměnná y bude fuzzy množina B. Výstupem těchto pravidel je tedy vždy fuzzy množina. Konkrétní číslo je určeno až v procesu defuzzifikace.

pokud *teplota je normální* **pak** *topení je nezměněno* (7)

Počet vstupních proměnných není omezen, proto by pravidlo mohlo vypadat jako výraz (8). Klíčové slovo *is* může být trochu zavádějící, proto je nutné zmínit jeho význam v části podmínky a v části důsledku. Pokud je použito u podmínky, je tím myšlena rovnost „==“. V případě použití u důsledku je tím myšlena rovnost „=“.

pokud *teplota je normální AND pocitová teplota je nízká* **pak** *topení je zvýšeno* (8)

Vstupní proměnné se spojují logickými operátory *AND, OR*, které platí pro klasickou logiku. Pravidla jsou tedy i snáze pochopitelná. Nicméně, tyto operátory jsou vhodné pouze pro klasickou binární logiku 0 a 1. Jak už bylo zmíněno, fuzzy logika, konkrétně hodnota stupně členství, může nabývat i hodnot v intervalu [0,1]. Pro samotný výpočet je proto nutné operátory *AND, OR* převést na funkci *min()*, resp *max()*, což je zjevné z tab. 1, 2.

Návrh pravidel je jedním ze základních kroků při návrhu fuzzy systému. Předpokladem pro návrh pravidel je znalostní databáze vstupních a výstupních hodnot, které jsou vždy spárované. Na základě těchto párů je možno vytvořit *if-then* pravidla, čímž dochází k natrénování fuzzy systému.

Tabulka 1. Převod operátoru AND

x	y	$x\ AND\ y$	$min(x,y)$
0	0	0	
0	1	0	
1	0	0	
1	1	1	

Tabulka 2. Převod operátoru OR

x	y	$x\ OR\ y$	$max(x,y)$
0	0	0	
0	1	1	
1	0	1	
1	1	1	

3.4. Proces defuzzifikace

Proces defuzzifikace je opakem fuzzifikace. To v praxi znamená, že převádí agregovaný graf funkcí příslušnosti výstupní fuzzy množiny na jeden konkrétní skalár. Pojem agregovaný graf je vysvětlen v následující podkapitole. Prozatím postačuje fakt, že se jedná o součet dílčích výsledků fuzzy algoritmu. Defuzzifikace může být popsána různými metodami, zde však budou zmíněny pouze tři z nich.

1. **Metoda maxima funkce**

   Jak se dá předpokládat, tato metoda vybere maximální hodnotu $\mu_D(x)$ z agregovaného grafu a odpovídající skalár $x*$ prohlásí za výsledek celého fuzzy algoritmu, viz obrázek obr. 10a. Je si nutno uvědomit, že je tento postup aplikovatelný pouze na grafy s jedním globálním maximem.

2. **Metoda těžiště**

   Jelikož je k dispozici spojitý agregovaný graf, je možno použít metodu těžiště. Ta je založena na podílu momentu rotace a celkové plochy grafu, viz obrázek 3*b*.

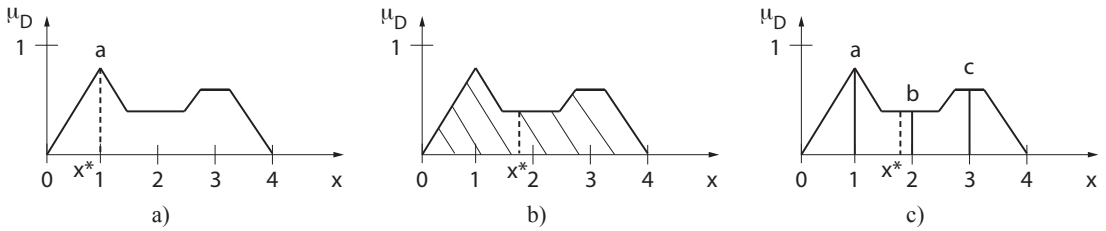
$$x* = \frac{\int \mu_D(x) \cdot x \cdot dx}{\int \mu_D(x) \cdot dx} \tag{9}$$

3. **Metoda váženého průměru**

   Jedná se o prostý vážený průměr n funkcí příslušnosti z agregovaného grafu (obr. 10c).

$$x* = \frac{\sum_{i=1}^{n} \mu_{Di}(x) \cdot x_i}{\sum_{i=1}^{n} \mu_{Di}(x)} \tag{10}$$

Pro úplnost je nutno dodat, že výstupní fuzzy množina je dána také několika funkcemi příslušnosti, jako tomu je i u vstupních proměnných, viz obr. 8b.

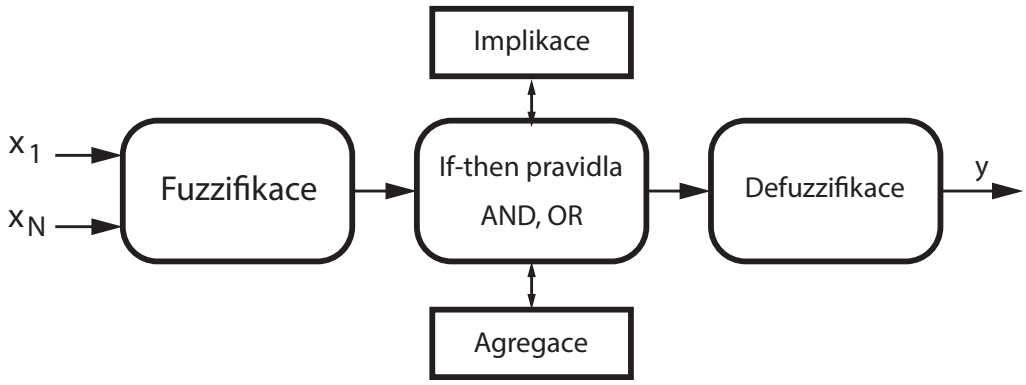


Obrázek 10. Metody defuzzifikace: a)Maximum, b)Těžiště, c)Vážený průměr

3.5. Základní model FIS

Celou kapitolu fuzzy logiky lze zahrnout pod zkratku FIS, *Fuzzy Inference System*. Z pohledu obecné aplikace se tento odvozovací systém jeví jako *blackbox*, který má typicky několik vstupů a jeden výstup. Jinými slovy, na základě konkrétních pravidel dokáže mapovat vstupní hodnoty, reprezentující několik univerz, na jednu konkrétní hodnotu, reprezentující fuzzy množinu. Na obrázku 11 je znázorněno blokové schéma takového FIS. První blok reprezentuje *fuzzifikaci*, druhý *If-then*

pravidla a třetí *defuzzifikaci*. Zmíněné bloky byly popsány už dříve. Ve schématu jsou doplněny ještě dva bloky, *agregace* a *implikace*. Těchto pět bloků kompletně pokrývá funkčnost odvozovacího systému.



Obrázek 11. *Schéma Fuzzy Inference System*

Průběh odvozovacího procesu bude přiblížen na částečném výpočtu názorného příkladu. Předpoklady fuzzy množin jsou na obrázku 12, kde jsou dva vstupní parametry − *teplota* a *pocitová teplota*, a dále výstupní parametr *topení*. Cílem úlohy FIS je zde regulace topení podle aktuální teploty v prostoru a podle pocitové teploty člověka, který může být například nemocný, a má tak jiné vnímání tepla než zdravý člověk. Pro tento FIS mohlo být navrženo několik *if-then pravidel*. Zde budou brány v potaz pouze dvě, viz výrazy (11) a (12).

pokud *teplota je normální AND pocitová teplota je nízká* **pak** *topení je zvýšeno* (11)

pokud *teplota je vysoká AND pocitová teplota je vysoká* **pak** *topení je sníženo* (12)

Základní funkčnost je taková, že se postupně vybírají jednotlivá pravidla a pro každé pravidlo probíhá fuzzifikace, aplikování logických operátorů a implikace. Po získání všech dílčích výsledků dochází k jejich agregaci a posléze k defuzzifikaci na konečný výsledek.

Popis procesu FIS modelu k obr. 13

1. Fuzzifikace - po vybrání prvního pravidla se vstupní hodnoty $teplota = 16$ a $pocitováteplota = 11$ fuzzifikují na stupeň příslušnosti $\mu_T = 0,9$, resp. $\mu_{PT} = 0,4$.

2. Aplikace operátoru - jelikož má pravidlo logický operátor *AND*, dojde k vybrání minimální hodnoty stupně členství $min(\mu_T, \mu_{PT})$, což je v tomto případě $\mu_{PT} = 0,4$.

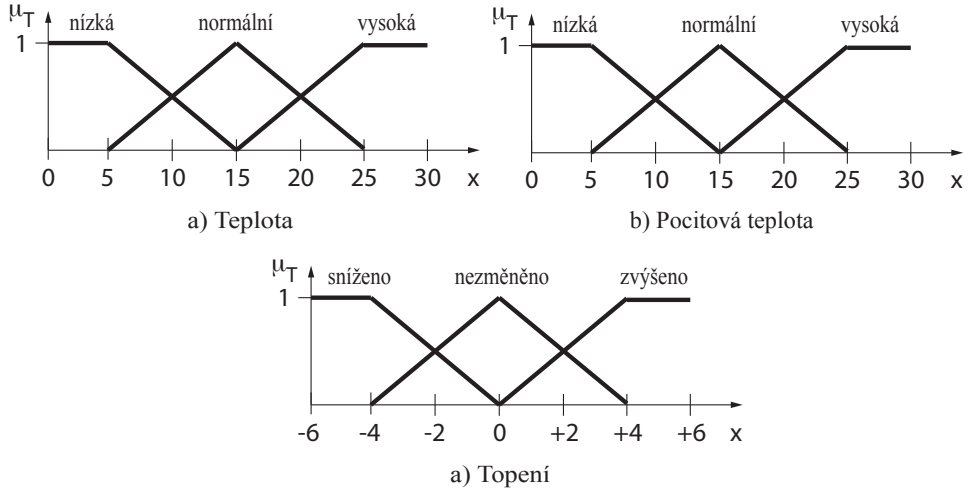
3. Implikace - je nový pojem, který znamená, že hodnota z předešlého kroku je použita na oříznutí výšky konkrétní fuzzy množiny, která reprezentuje výstupní proměnou a je součástí pravidla (v části důsledku). Tímto se získá první dílčí výsledek fuzzy odvozování.

4. Analogicky se zpracuje druhé pravidlo a po něm všechna zbývající.

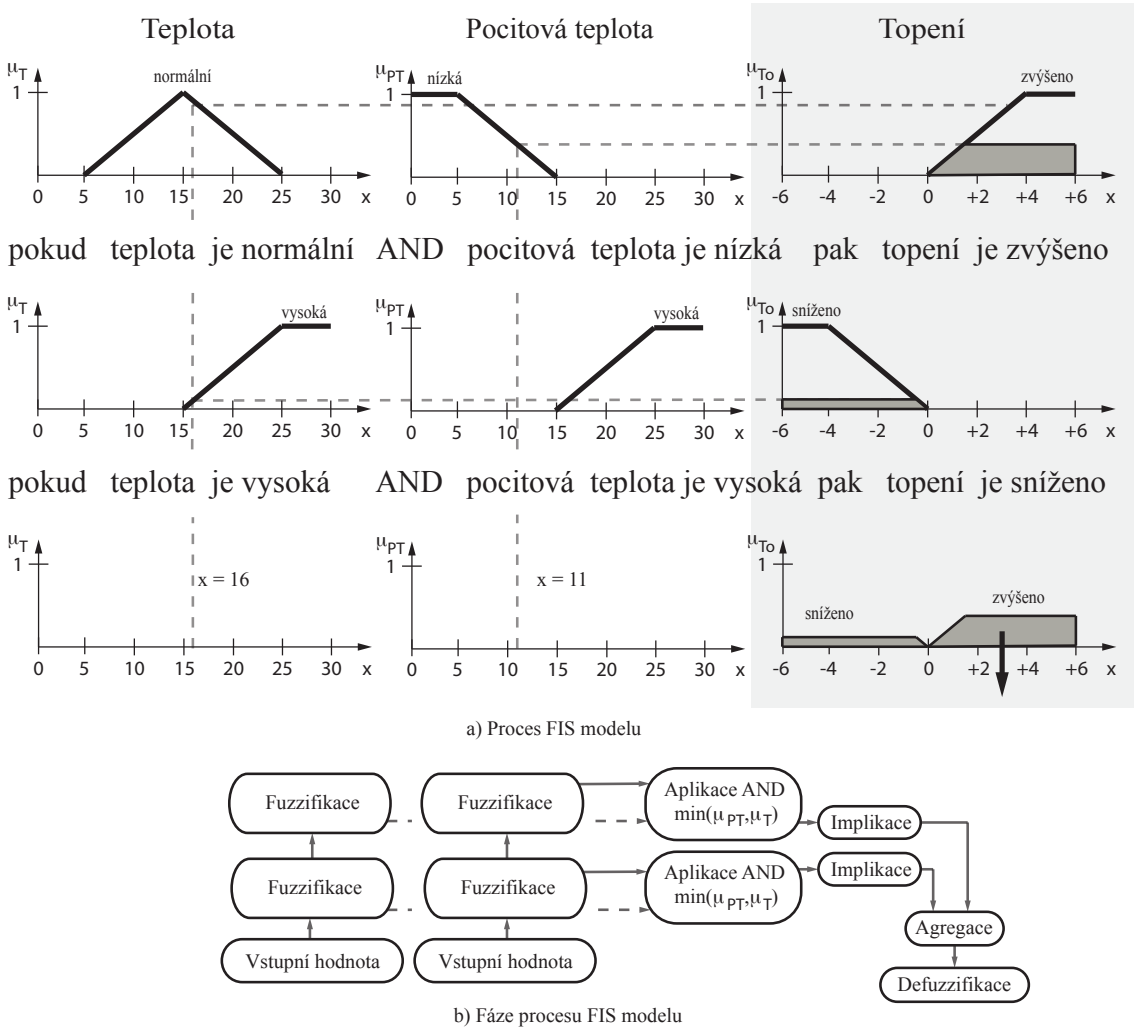
5. Agregace - poslední důležitý pojem, jehož význam je takový, že skládá všechny dílčí výsledky do jednoho výstupního univerza. Toto provádí metodou maxima, protože i pro jednu množinu (*sníženo, zvýšeno, nezměněno*) může existovat více dílčích výsledků hodnoty stupně příslušnosti, a proto se bere v potaz pouze varianta s maximální hodnotou. Výsledkem je agregovaný graf, který reprezentuje celý fuzzy rozhodovací proces.

6. Defuzzifikace - pro regulaci je nutné získat konkrétní hodnotu, proto je posledním krokem právě defuzzifikace. V tomto případě mohla být například použita metoda váženého průměru. Výsledkem procesu pak může být, že se teplota zvýší o dvě jednotky.

Pro úplnost je nutno dodat, že fuzzy logika může být popsána dvěma odvozovacími systémy (modely). Prvním je Sugeno model, jenž na základě vstupních fuzzy množin odvodí výsledek, který je popsán buď konstantou, anebo lineární rovnicí. To je oproti druhému modelu výhoda, zejména z pohledu výpočetní náročnosti, anebo z pohledu matematických analýz. Druhý model, který je využit pro tuto práci, je Mamdani. Je charakterizován stejnými vstupními fuzzy množinami jako model předchozí, ale jeho výsledkem jsou fuzzy množiny. To umožňuje lepší popis náhodných jevů, jako je fluktuace RSSI apod. Je také srozumitelnější díky jazykovému popisu (viz pravidla (11) a (12)), který je blíže lidskému chápání, než popis lineární rovnicí.



Obrázek 12. Fuzzy množiny vstupů a výstupu



Obrázek 13. Proces FIS

3.6. Fuzzy logika a lokalizace

Fuzzy logika má velmi dobré předpoklady pro uplatnění v navrhovaném lokalizačním systému. Jeho cílem je lokalizace na úrovni zóny, při níž je potřeba porovnávat výsledky z lokalizační fáze s výsledky fáze trénovací. Fuzzy logika to umožňuje díky návrhu odvozovacího systému. Ačkoli se zdá být složitý a časově poměrně náročný, ve skutečnosti tomu tak není. Návrh odvozovacího systému je univerzální (vyjma vstupních definic fuzzy množin) pro libovolné prostředí. Jeho kladem je také to, že největší výpočty probíhají na samém počátku, tedy v době generování odvozovacího systému. Na základě vygenerovaného popisu se ve fázi lokalizace provádějí pouze jednoduché výpočty, které zatěžují mobilní zařízení jen částečně. Tzn. že je zařízení schopno normálního bezproblémového provozu i když odvozuje svoji pozici. Dalším přínosem fuzzy logiky je malá časová náročnost vytváření trénovací množiny dat. V jednotlivých zónách stačí provést několik měření v závislosti na

jejich velikosti. Dostačující je jedno měření na cca $2,5\ m^2$. Není nutné v přesně definované matici procházet celý prostor. Zajímavou vlastností fuzzy logiky je také její vztah k fluktuaci signálu. Jelikož se nejedná o přesnou standardní logiku, je schopna se daleko lépe vypořádat s proměnlivými vstupními hodnotami RSSI.

4. Návrh odvozovacího systému

Tato kapitola se zabývá návrhem a optimalizací klíčové součásti lokalizačního systému, jíž je fuzzy odvozovací systém (FIS). Jak už je zřejmé z předcházejí kapitoly, pro správnou funkčnost fuzzy systému je nutné navrhnout fuzzy množiny (charakterizované funkcemi příslušnosti) pro vstupní a výstupní proměnné, a také je nutné vytvořit odpovídající pravidla, která mezi těmito proměnnými vytvoří konkrétní vztah. Návrh systému je rozdělen do několika fází, které jsou podrobněji popsány v odpovídajících podkapitolách níže.

První fáze je věnována tvorbě trénovací množiny dat. Nezávisle na jejích datech jsou ve druhé fázi navrženy fuzzy množiny a univerza pro jednotlivé proměnné. Díky těmto počátečním krokům je možno vygenerovat databázi *If-then* pravidel na základě WM-metody (viz kap. 4.3). Po této posloupnosti úkonů je fuzzy systém plně funkční. Nemusí však dosahovat potřebné přesnosti, proto je dobré návrh optimalizovat. Pro optimalizaci je využita metoda simulovaného žíhání. Tato optimalizace je uplatněna nejprve ke zlepšení WM-metody a poté také ke zpětnému návrhu fuzzy množin vstupních proměnných. Po optimalizaci je fuzzy odvozovací systém schopný reflektovat konkrétní prostředí.

Pro návrh a optimalizaci jednotlivých komponent lze použít několik metod. Co se týče fuzzy množin, resp. jejich funkcí příslušnosti, ve většině případů se postupuje tak, že je na základě expertních znalostí vytvořena skupina fuzzy množin, které jsou poté algoritmicky optimalizovány. Toho se dosáhne například využitím neuronových sítí [6], algoritmu fuzzy kNN [7], genetických algoritmů [10], nebo pomocí optimalizace rojem částic [8], popř. metodou fuzzy c-means [9]. Stejné metody jsou používány i pro vytváření *If-then* pravidel.

4.1. Definice vstupních a výstupních fuzzy množin

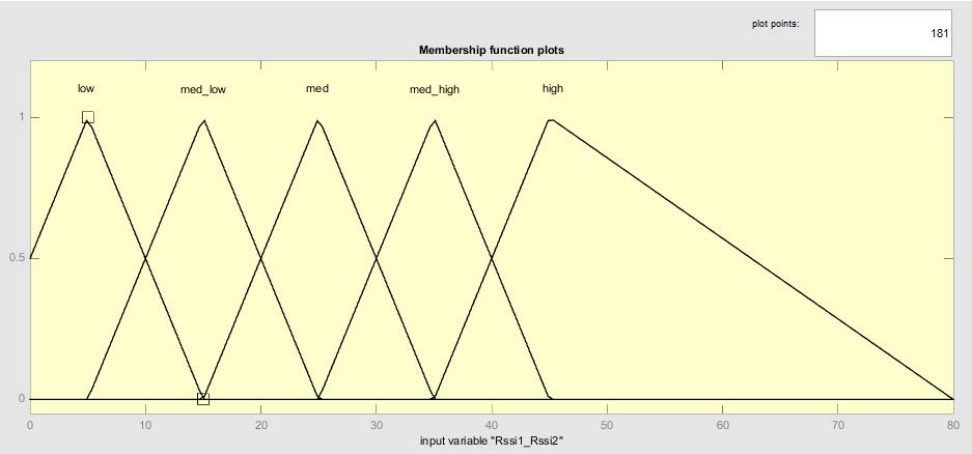
Jedním z cílů kapitoly 3 bylo přiblížení pojmu fuzzy množina a její důležitosti v celém fuzzy odvozovacím systému. Fuzzy množina definuje jednak vstupní proměnné a jednak výstupní proměnné. Od charakteru fuzzy množin se odvíjí výsledek generace *If-then* pravidel, proces fuzzifikace a defuzzifikace. Soustava fuzzy množin poté definuje celé univerzum dané proměnné, tedy interval hodnot, jakých může proměnná nabývat. Univerzum může být cíleně navrženo tak, aby nepokrývalo celý možný rozsah hodnot proměnné, ale není to příliš obvyklé. Konkrétní počet univerz poté definuje množinu (počet) všech proměnných. Pokud nebude vysloveně nutné rozlišit vstupní a výstupní hodnotu proměnné, budou oba termíny v této podkapitole vyjádřeny pouze termínem hodnota proměnné.

Jak už bylo také zmíněno, každá fuzzy množina je charakterizována svou funkcí příslušnosti. Hlavním problémem tedy je, na základě čeho se stanoví počet, tvar

a pozice funkcí příslušnosti u konkrétní proměnné. Zde je nutné podotknout, že všechny parametry, resp. celý odvozovací systém, může být navrhnut i pouze na základě expertní znalosti, což je velmi zajímavá vlastnost fuzzy logiky. Proces návrhu systému může tedy v reálných případech probíhat tak, že jsou praktické znalosti daného pracovníka předávány návrháři fuzzy systémů a na základě této vazby se posléze generuje výsledný systém. Například se může jednat o způsob parkování, tedy vazba řidič-návrhář, a z toho vyplývající automatické parkování. Ve většině případů se fuzzy logika nasazuje právě pro regulaci, např. automatická převodovka, automatické parkování apod. Tento způsob návrhu těží z fuzzy vlastnosti, jíž je míra rozlišovací schopnosti. Co tím je myšleno, ukazuje následující příklad. Při podélném parkování do řady se řidič rozhoduje na základě svých zkušeností s konkrétním typem dopravního prostředku. Vstupními proměnnými může být rychlost, pozice a vzdálenost od překážek. Výstupními proměnnými může být ovládání rychlosti a směru. Na základě těchto vstupů se řidič neřídí přesnými hodnotami proměnných, tzn. neuvědomí si, že je přesně $1,53\,m$ od překážky a jede rychlostí $1\,km/h$, ale pouze to, že je blíže k překážce a jede velmi pomalu. Ze svých zkušeností ví, že kdyby zatočil příliš vlevo, mohl by narazit do překážky. Na základě toho tedy nemění rychlost, ale trochu (ne přesně o 10°) změní směr vlevo. To je příklad části jednoduchého fuzzy regulátoru, jehož výsledkem je vždy zaparkovaný dopravní prostředek, aniž by bylo nutné specifikovat jeho druh, velikost apod. Dalo by se říci, že každý člověk je fuzzy regulátor, protože ačkoli má každý tyto meze reakce na událost vzhledem k univerzu někde jinde (ve kterém okamžiku zatočí vlevo), přesto každý dokáže zaparkovat. Pojmy *velmi pomalu, trochu vlevo* apod., jsou tzv. lingvistické výrazy dané fuzzy množiny (funkce příslušnosti).

Počet funkcí příslušnosti je v navrhovaném systému stanoven na základě charakteru jeho vstupních proměnných, tedy hodnot RSSI, resp. jejich rozdílem. Bohužel tyto hodnoty podléhají nepředvídatelné míře fluktuace, proto by bylo nevhodné celé univerzum rozdělit na velký počet intervalů. Pro velké počty by totiž klesala rozlišovací schopnost algoritmu (správná fuzzifikace), protože by se mohly hodnoty vstupní proměnné snadno pohybovat mezi sousedními fuzzy množinami. Na druhou stranu je však potřeba zaručit určitou míru rozlišitelnosti mezi těmito množinami. Na základě praktické zkušeností a literatury bylo stanoveno pět funkcí příslušnosti, viz obrázek 14.

Pokud jde o tvar funkce příslušnosti lze konstatovat, že ovlivňuje přesnost systému menší měrou, než počet nebo pozice funkce. Jelikož jsou vstupní hodnoty částečně ovlivněné fluktuací signálu, je rozdíl mezi gaussovou funkcí a funkcí ve tvaru trojúhelníku víceméně smazán. Hlavním rozdílem mezi funkcemi je pouze o trochu odlišná hodnota stupně příslušnosti pro stejnou hodnotu vstupní proměnné. Z pohledu lokalizace je tedy důležitější, aby se hodnota proměnné fuzzifikovala do správné fuzzy množiny. Pro správnou fuzzifikaci je nutné stanovit i správnou polohu funkce vzhledem k univerzu a vzhledem k ostatním funkcím. K tomuto účelu je využit optimalizační algoritmus, který přepočítá původní rozmístění funkcí na jiné. Na počátku bude mít fuzzy systém vždy takto definované fuzzy množiny. Nové pozice jsou vždy vypočítány tak, aby jejich hodnota ovlivnila (zlepšila) přesnost algoritmu jako celku. V jistém smyslu tedy jde o větší rozlišitelnost mezi výsledky.



Obrázek 14. Navržené fuzzy množiny pro vstupní proměnnou $\Delta Rss1\_Rssi2$

4.2. Trénovací množina

Trénovací množina dat je využívána ke generaci pravidel a poté i k jejich optimalizaci. Je také využívána pro optimalizaci fuzzy množin. V rámci této práce je navrhovaný fuzzy systém definován jednou výstupní proměnnou *Zone*, která vyjadřuje název zóny, kde bylo měření provedeno. Dále je definován pěti vstupními proměnnými $\Delta Rss1\_Rssi2$, $\Delta Rss2\_Rssi3$, $\Delta Rss3\_Rssi4$, $\Delta Rss4\_Rssi5\Delta Rss5\_Rssi6$, jež reprezentují jednotlivé rozdílové hodnoty od šesti přístupových bodů. Každý záznam v databázi tedy bude mít tvar

$$\Delta Rss1\_Rssi2, \Delta Rss2\_Rssi3, ..., \Delta Rss5\_Rssi6, Zone. \tag{13}$$

Samozřejmě platí, že název zóny je zároveň i budoucí výsledek lokalizace. Samotné měření není závislé na přesně definovaných měřících bodech v daném prostoru, takže je možno tato data měřit ze statické pozice v konkrétní zóně. Je tedy možné do dané lokality umístit například měřící sondu, která bude sbírat potřebná data. Tento postup se však nedoporučuje, neboť by tím byla omezena množina vstupních hodnot, které se mohou v dané zóně vyskytnout. Proto je dobré proměřit danou zónu minimálně na pěti místech tvořících pomyslnou hvězdu (střed a rohy zóny).

4.3. Generování *If-then* pravidel

Druhou důležitou součástí navrhovaného fuzzy odvozovacího systému (typu FRBS, *Fuzzy rule based system*) je databáze *If-then* pravidel, která jsou popsána v kapitole 3.3. Jak už bylo zmíněno v úvodu kapitoly, existuje zde mnoho metod návrhů těchto pravidel. Stejně jako u návrhu fuzzy množin, tak i zde se může využít metodika expertní znalosti. Pro komplexnější modely s více vstupními či výstupními

proměnnými je však velmi složité navrhnout odpovídající databázi. Proto se využívají metody, které automaticky generují pravidla z číselných dat. V podstatě se jedná o trénování fuzzy systému na základě trénovacích dat. V případě navrhovaného systému jsou tato data naměřena v reálném prostředí, ve kterém je prováděna lokalizace.

Pro generování pravidel byla vybrána tzv. WM-metoda (z angl. *Wang & Mendel method*) [18]. Tato metoda byla vybrána pro svou nízkou časovou náročnost, snadnou interpretaci a dobrou účinnost. Na druhou stranu nedosahuje stejně kvalitních výsledků v porovnání s ostatními metodami, tento nedostatek lze však kompenzovat s využitím algoritmu simulovaného žíhání, viz kapitola 4.4.

Postup generování pravidel může být popsán těmito body:

1. Prvním krokem je vytvoření trénovací množiny dat, viz podkapitola 4.2.

2. Jelikož jsou známy všechny vstupy a výstupy, je nutné nadefinovat jejich fuzzy množiny, což bylo popsáno v podkapitole 4.1.

3. Dále je potřeba vytvořit množinu možných pravidel, která se mohou stát aktivními pravidly. To v praxi znamená, že z každého záznamu v trénovací množině dat je vytvořeno jedno pravidlo dle obecného předpisu (viz rovnice (6)), tedy například ve tvaru rovnice (14). Zde je důležité upozornit, že se vždy hodnota každé vstupní proměnné fuzzifikuje do příslušného jazykového vyjádření dle navržených funkcí příslušnosti (fuzzy množin).

$$\begin{aligned}\textbf{if } \Delta Rss1\_Rssi2 \textit{ is low } AND ... AND\ \Delta Rssi5\_Rssi6 \textit{ is high}\\ \textbf{then } zone \textit{ is } zone1\end{aligned} \tag{14}$$

4. Z vytvořené databáze pravidel jsou poté odstraněna duplicitní pravidla.

5. Pro všechna pravidla je následně vypočten tzv. stupeň důležitosti. Je to součin jednotlivých stupňů příslušnosti (např. $\mu_{\Delta Rss1\_Rssi2} \cdot \mu_{\Delta Rss1\_Rssi2\cdots}$) z kroku fuzzifikace do jazykových vyjádření. Tím jsou pravidla oklasifikována ve smyslu, čím vyšší stupeň důležitosti, tím je pravidlo z pohledu systému lepší.

6. V posledním kroku se vygenerovaná pravidla seskupují dle stejné podmínkové části pravidla. Na základě stupně důležitosti se z každé skupiny vybere pravidlo s nejvyšší hodnotou tohoto stupně. Ostatní pravidla skupiny se odstraní z databáze. Tento krok je důležitý pro eliminaci chybného odvození výsledku, protože fuzzy systém by nebyl schopný odvodit ze dvou stejných podmínek správný výsledek.

4.4. Optimalizace navrženého FIS

4.4.1. Optimalizace *If-then* pravidel

Optimalizace *If-then* pravidel je založena na vylepšené verzi WM-metody, jejíž součástí je SA algoritmus. Jak už bylo zmíněno v podkapitole 4.3, v posledním kroku jsou pravidla seskupena dle jejich totožných podmínkových částí a poté je v databázi pravidel ponecháno jen pravidlo s nejvyšší hodnotou stupně důležitosti. Ostatní pravidla z téže skupiny jsou vymazána, ačkoli mohla mít daleko výraznější dopad na rozhodování systému jako celku. Tím je myšlena zejména míra rozlišitelnosti mezi jednotlivými výsledky. Jinými slovy, fuzzy systém by s větší pravděpodobností preferoval pouze jeden možný výsledek oproti situaci, kdy by mohl předkládat více možných výsledků.

Principem vylepšené varianty WM-metody [18] je využití simulovaného žíhání [17]. Jednotlivé fáze tohoto generování pravidel jsou následující:

1. Kroky $1 - 4$ jsou shodné s kroky v kap. 4.3.

2. V posledním kroku se opět vygenerují skupiny dle totožných podmínkových částí. Každá skupina obsahuje několik možných výsledků pro danou jednu podmínku. Smyslem SA algoritmu je opět minimalizace pomocí MSE funkce dle vztahu (15) a pomocí stejného pseudokódu, viz algoritmus 1 a 2. Nyní se však nebudou měnit parametry konkrétních funkcí příslušnosti, ale budou se měnit parametry databáze pravidel. Tím je myšleno to, že bude vytvořena databáze pravidel dle počtu skupin, kde každá skupina je definována příslušnou podmínkovou částí. Výsledky (což jsou v tomto případě parametry) pravidel se však budou měnit podle toho, jaké možné výsledky jsou pro danou podmínku ve skupině dostupné. SA algoritmus tedy hledá takovou kombinaci výsledků, při níž by MSE funkce byla minimalizována. Po nalezení nejlepší kombinace výsledků, tedy nejlepší databáze pravidel, budou ostatní nepoužité výsledky odstraněny.

4.4.2. Optimalizace fuzzy množin

Optimalizace fuzzy množin, resp. jejich funkcí příslušnosti, je jedním ze způsobů, jak zlepšit přesnost odvozování výsledku. Jak už bylo zmíněno v podkapitole 4.1, pro navrhovaný fuzzy systém, resp. pro návrh jeho fuzzy množin, byla zvolena metoda expertních znalostí. Výsledek návrhu lze optimalizovat pomocí několika metod, zejména neuronovými sítěmi, genetickými algoritmy, nebo například tzv. simulovaným žíháním (z angl. *Simulated Annealing*, SA). K optimalizaci byla použita poslední z uvedených metod [16], [19].

Simulované žíhání je stochastická optimalizační metoda, která byla vyvinuta v 80. letech nezávisle dvěma autory [3], [4]. Je založena na fyzikálním principu žíhání kovu. To je realizováno pomalým ochlazováním kovu tak, aby bylo dosaženo

minimální energetické hodnoty materiálu (změna vazeb mezi atomy). Toto minimum poté v algoritmu SA vyjadřuje globální minimum konkrétní funkce. Může to být jakákoliv funkce, kde pro účely této práce je minimalizována MSE (z angl. *Mean square error*), viz vztah (15).

$$MSE = \frac{1}{N} \cdot \sum_{i=1}^{N} (y_i' - y_i)^2 \tag{15}$$

kde N je množství dat v trénovací množině (stejná data, kterými byla generována pravidla) a y_i' je výsledek defuzzifikace na základě vstupních hodnot z trénovací množiny (pro případ počátečního řešení $S1$). Naopak pro generaci nového řešení $S2$ jsou vstupem pozměněná data vůči stávajícímu řešení, tedy $S1$. Počáteční a stávající řešení je označeno shodně, liší se pouze okamžikem, kdy jsou generována, tzn. že počáteční slouží pouze jako inicializace stávajícího řešení.

Pojmem řešení je myšlena uložená konfigurace (hodnota) parametrů konkrétní funkce příslušnosti, pro kterou SA algoritmus hledá minimální hodnotu MSE funkce. Počáteční řešení $S1$ tedy znamená, že při zahájení SA algoritmu se načtou výchozí hodnoty parametrů funkcí příslušnosti, viz obrázek 14. Na základě těchto parametrů se vygeneruje nové řešení $S2$ a pokud je lepší než $S1$, je akceptováno a platí $S1 = S2$, kde se z $S1$ stalo stávající nejlepší řešení.

Parametr y_i je přímo výsledek konkrétního záznamu trénovací množiny, resp. jeho poslední část. Zmíněná minimalizace má globální charakter. To znamená, že nenastane případ, že by se našlo pouze lokální minimum, jako tomu je např. u gradientního algoritmu, který ukončí činnost ihned po nalezení lokálního minima/maxima. SA k tomuto účelu využívá tzv. Metropolitní kritérium přijetí (z angl. *Metropolis Acceptance Criterion* [5]) , které je založeno na pravděpodobnostním základu. Příslušný vztah (16) vyjadřuje pravděpodobnost, se kterou SA přijme odmítnuté řešení z první fáze (bude vysvětleno později), a tak umožní algoritmu překonávat lokální extrémy.

$$p(T) = e^{-\Delta E/T} \tag{16}$$

kde ΔE je rozdíl mezi dvěma hodnotami řešení, a to nového MSE(S2) a předchozího MSE(S1). T je aktuální hodnota teploty, která se v průběhu algoritmu postupně snižuje.

Pseudokód SA algoritmu je zobrazen výše. Hlavní část algoritmu 1 začíná inicializací teploty T, tedy počáteční teploty. Tato teplota simuluje postupné ochlazování kovu, dokud nedosáhne určité úrovně. Inicializace počáteční teploty může být provedena buď manuálním přiřazením hodnoty, nebo výpočtem dle vztahu

$$T = -\frac{h_{mean}}{ln(P_0)} \tag{17}$$

kde h_{mean} je prostý průměr hodnot ΔE v případě špatného kroku (řádek 11), kde celý algoritmus končí po pevně dané době $teplota\_faze = x$ (např. $x = 100$). P_0 je počáteční hodnota pravděpodobnosti přijetí, tzn. že pro P_0 blízké jedné je teplota T vysoká. To má za následek, že jsou přijímána nová řešení S2, která ale mohou být horší než je stávající řešení S1. Postupným snižováním T výsledná hodnota

Algoritmus 1 Simulované žíhání

```
1: function simulated_annealing()
2:     Inicializace teploty T
3:     Generace počátečního řešení S1
4:     while teplota_konecna < T do
5:         while teplota_faze > 0 do
6:             Generace nového řešení S2, blízko S1 //akceptování nového řešení
7:             if MSE(S2) < MSE(S1) then S1 = S2
8:             else//akceptování nového řešení
9:                 if acceptance_criterion (MSE(S1), MSE(S2), T) then S1 =
   S2
10:                else
11:                    Špatný krok
12:                end if
13:            end if
14:            snížení teplota_faze
15:        end while
16:        snížení T
17:    end while
18: end function
```

Algoritmus 2 Kritérium přijetí

```
1: function acceptance_criterion (MSE(S1), MSE(S2), T)
2:     Výpočet změny energie ΔE = MSE(S2) − MSE(S1)
3:     Výpočet pravděpodobnosti přijetí p(T) = e^{−ΔE/T}
4:     if random(0, 1) < p(T) then return true//akceptování nového řešení
5:     else return false
6:     end if
7: end function
```

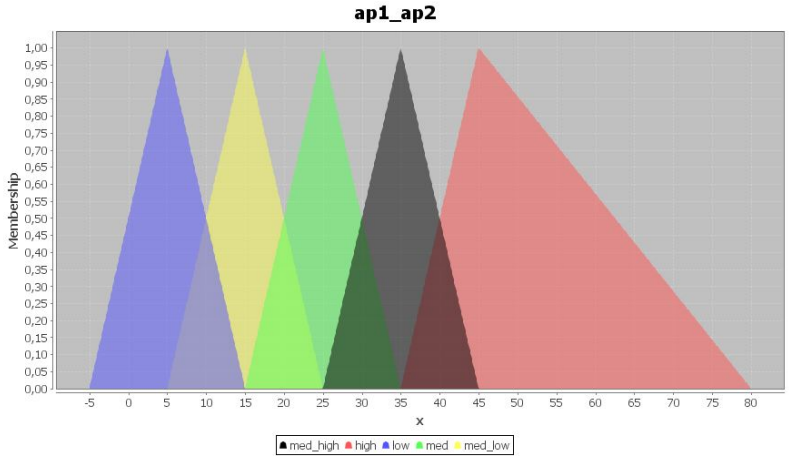
konverguje do globálního minima, protože jsou přijímány pouze nové a zároveň lepší hodnoty MSE(S2) než jsou stávající MSE(S1). Tento princip je vyjádřen vztahem (16), a který je součástí algoritmu 2.

Co se týče proměnné *teplota_faze*, je její fyzikální princip takový, že aktuální hodnotu T udrží určitou dobu konstantní, aby došlo k ustálení fyzikálních jevů v kovu. Samotná teplota T je snižována konstantní hodnotou α z intervalu $(0, 1)$, kde se nejčastěji využívá hodnoty v intervalu $(0,8; 0,95)$. Konkrétně $T_{k+1} = \alpha \cdot T_k$. Algoritmus může být ukončen celkovým snížením teploty T, anebo po několika cyklech snížení T, kdy nedošlo k přijetí nového řešení.

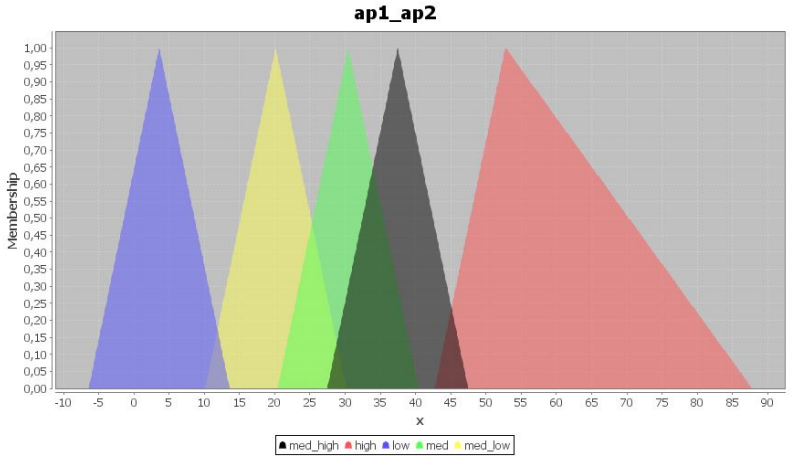
U generace nového řešení bylo zmíněno, že se využívá funkce MSE, která má jeden vstupní parametr odpovídající přímo hodnotě výsledku trénovací množiny, a druhý parametr je výsledek defuzzifikace fuzzy systému. Je nutné doplnit, že při každé generaci nového řešení S2 se vezme hodnota parametru současného řešení a přičte se k němu náhodné číslo z intervalu $(-0,5; 0,5)$. Tím je zaručeno, že se nové řešení může pohybovat dle potřeby (vpravo/vlevo). Parametry, které definují každé řešení,

jsou parametry funkcí příslušnosti. To znamená, že trojúhelníková funkce může být definována pomocí tří parametrů a, b, c, kde a, c značí body průniku s osou x a b značí polohu vrcholu vůči x. Cílem této optimalizace je tedy nalezení nových pozic funkcí příslušnosti pro danou vstupní proměnnou a danou fuzzy množinu. Výsledné optimalizované pozice by měly zajistit lepší odvozování výsledku vzhledem k fuzzy systému jako celku.

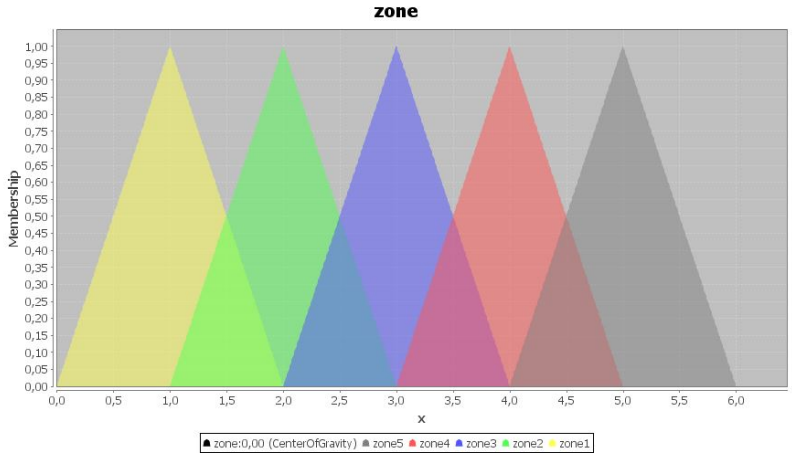
Oproti optimalizaci pravidel lze optimalizaci fuzzy množin názorně zobrazit a porovnat. Na obrázku 15 je původní návrh fuzzy množin pro vstupní proměnné. Tento návrh je výchozí pro všechny generované fuzzy systémy, tzn. že výsledek optimalizace je závislý pouze na trénovacích datech. Možný výsledek je zobrazen na obrázku 16, kde je zřetelně vidět posunutí některých fuzzy množin. V této části optimalizace byly optimalizovány pouze parametry posunutí, samozřejmě lze optimalizovat i šířku či popřípadě pozici vrcholu jednotlivých funkcí příslušnosti. Grafy jsou vytvořeny pomocí Java knihovny jFuzzyLogic, viz kap. 4.5. Pro úplnost je doplněn tvar výstupních fuzzy množin, kde každá množina definuje jednu zónu, viz obr. 17.



Obrázek 15. Výchozí návrh fuzzy množin z kap. 4.1



Obrázek 16. Optimalizované fuzzy množiny pro výchozí návrh



Obrázek 17. Výstupní fuzzy množiny

4.5. Knihovna jFuzzyLogic

Pro účely této práce je fuzzy odvozovací systém realizován na základě Java knihovny jFuzzyLogic [21]. Jedná se o open-source knihovnu, nezávislou na platformě, která je určená pro návrh a realizaci jakéhokoli fuzzy kontroléru, jehož chování je popsáno speciálním programovacím jazykem FCL (z angl. *Fuzzy Control Language*). Syntaxe tohoto jazyka je standardizována v dokumentu IEC 61131. Celkové chování fuzzy systému (kontroléru) je popsáno jedním souborem, který může vypadat následovně [22]:

```
// Block definition (there may be more than one block per file)
FUNCTION_BLOCK tipper

// Define input variables
VAR_INPUT
     service : REAL;
     food : REAL;
END_VAR

// Define output variable
VAR_OUTPUT
     tip : REAL;
END_VAR

// Fuzzify input variable 'service'
FUZZIFY service
     TERM poor := (0, 1) (4, 0) ;
     TERM good := (1, 0) (4,1) (6,1) (9,0);
     TERM excellent := (6, 0) (9, 1);
END_FUZZIFY

// Fuzzify input variable 'food'
FUZZIFY food
     TERM rancid := (0, 1) (1, 1) (3,0) ;
     TERM delicious := (7,0) (9,1);
END_FUZZIFY
```

```
// Defzzzify output variable 'tip'
DEFUZZIFY tip
     TERM cheap := (0,0) (5,1) (10,0);
     TERM average := (10,0) (15,1) (20,0);
     TERM generous := (20,0) (25,1) (30,0);
     // Use 'Center Of Gravity' defuzzification method
     METHOD : COG;
     // Default value is 0 (if no rule activates defuzzifier)
     DEFAULT := 0;
END_DEFUZZIFY

RULEBLOCK No1
     // Use 'min' for 'and' (also implicit use 'max'
     // for 'or' to fulfill DeMorgan's Law)
     AND : MIN;
     // Use 'min' activation method
     ACT : MIN;
     // Use 'max' accumulation method
     ACCU : MAX;

     RULE 1 : IF service IS poor OR food IS rancid
                 THEN tip IS cheap;

     RULE 2 : IF service IS good
                 THEN tip IS average;

     RULE 3 : IF service IS excellent AND food IS delicious
                 THEN tip is generous;
END_RULEBLOCK

END_FUNCTION_BLOCK
```

Jedná se o často využívaný příklad na fuzzy odvozovací systém, jehož cílem je odvodit výši spropitného na základě kvality obsluhy a jídla. Popis systému je umístěn ve $FUNCTION\_BLOCK$ s názvem *tipper*. V blocích VAR_INPUT a VAR_OUTPUT jsou definovány vstupní, resp. výstupní proměnné. K nim jsou definovány funkce příslušnosti, a to v blocích $FUZZIFY service$, $FUZZIFY food$ a $DEFUZZIFY tip$. Každá funkce je dána výrazem $TERM$, kde následuje lingvistický název fuzzy množiny a poté číselný popis tvaru funkce. Ten je dán výčtem bodů ve tvaru (x, y), kde x je hodnota na ose x a y je jeho stupeň příslušnosti. Pro něj se používají pouze dvě hodnoty, a to 1 a 0. Trojúhelníková funkce je tedy dána třemi body, což je případ výstupních fuzzy množin v bloku $DEFUZZIFY tip$. V tomto bloku se také definuje metoda defuzzifikace. Je jí COG (z angl. *Center of gravity*), tedy metoda těžiště, viz kap. 3.4. Knihovna podporuje kromě trojúhelníkové funkce i další, jako je lichoběžníková, prostá přímka, gaussovská křivka apod. Pro účely optimalizace funkcí příslušnosti se využívají parametry x. Posledním blokem je $RULEBLOCK No1$, který obsahuje množinu *If-then* pravidel. V závislosti na optimalizaci se zde objeví vždy konkrétní soubor pravidel. Tento soubor (*fis.fcl*) je generován na základě metod z předešlé kapitoly, zabývající se návrhem odvozovacího systému.

5. Navrhovaný lokalizační systém

5.1. Funkcionalita systému

Navrhovaný lokalizační systém je složen z množství prvků, které se od sebe diametrálně liší, a proto byl návrh také různorodou činností. Pro snadnější identifikaci bylo použito pojmenování jednotlivých prvků systému začínající vždy na klíčový výraz *FuzzyLoc*. Celý lokalizační systém se tedy nazývá *FuzzyLoc System*. Nebylo jednoduché zkombinovat všechny potřebné funkcionality do jednoho systému, nicméně mezi klíčové požadavky na systém patří:

- Vytvoření mobilní aplikace *FuzzyLoc Admin* pro správce systému, která bude schopna vytvořit trénovací množinu dat, registrovat uživatele a přístupové body do systému.

- Vytvoření mobilní aplikace *FuzzyLoc Client*, která bude skenovat okolní RSSI a na základě stažené definice fuzzy odvozovacího systému bude schopna určit svoji polohu (zónu) a výsledek zaslat na server.

- Vytvoření jednoduchého webového rozhraní *FuzzyLoc Supervisor*, určeného ke zobrazování výsledků lokalizace a definice zón, dle kterých se bude odvozovat výsledná zóna.

- Vytvoření webového serveru *FuzzyLoc Server*, který bude schopen komunikace se všemi výše zmíněnými prvky, bude obsahovat MySQL databázi k ukládání všech potřebných dat a jeho nasazení v produkčním prostředí bude jednoduché.

- Vytvoření programu *FuzzyLoc Engine* pro generování definic, dle kterých se bude lokalizovat každé zařízení.

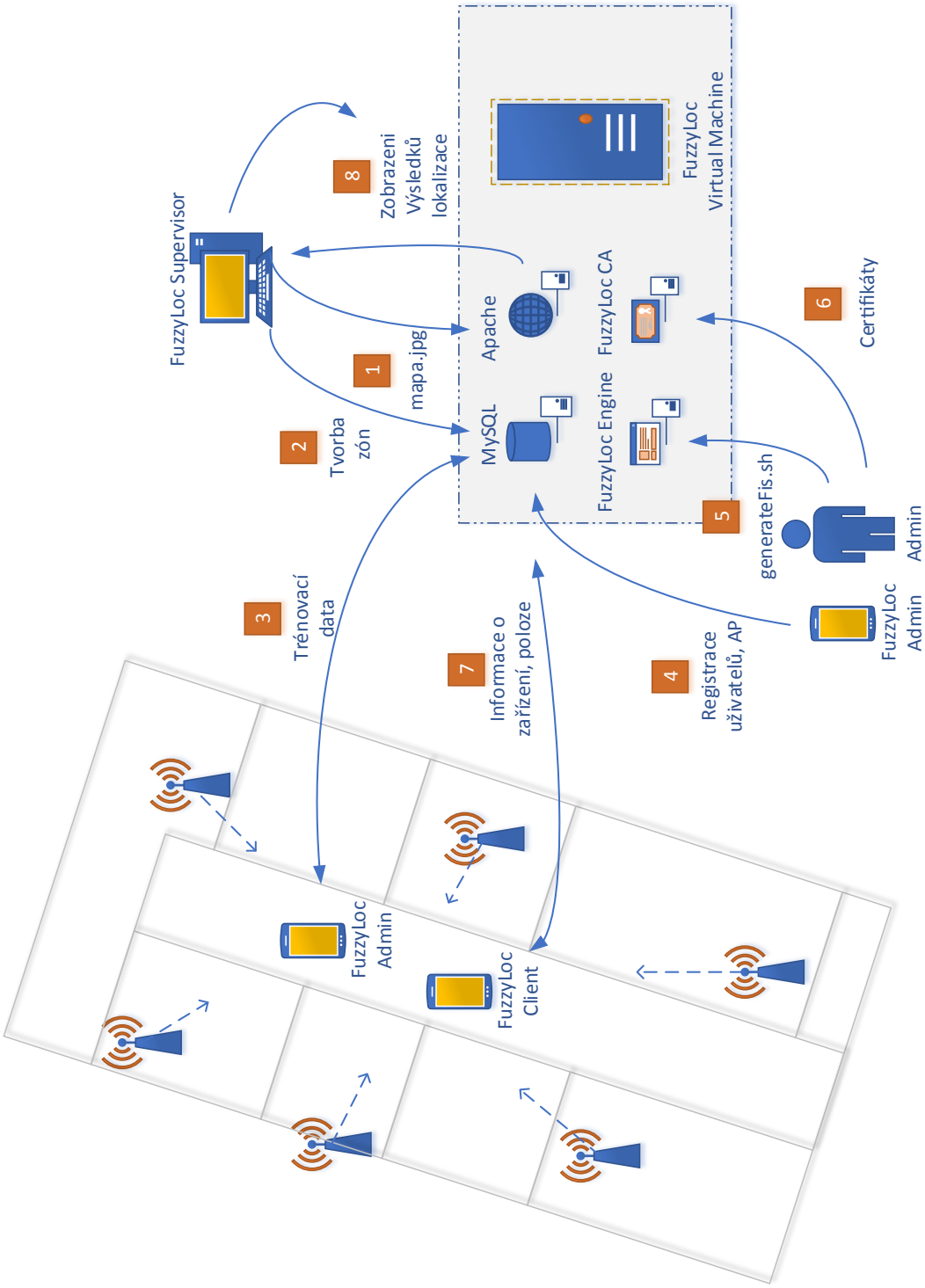
Před samotným lokalizováním je nutné do sítě organizace implementovat server. Pro snadnou implementaci bylo využito řešení virtuálního stroje, takže je snadné tento soubor importovat do virtualizovaného prostředí, např. VMware. Jedná se o standardizovaný univerzální formát OVF, takže je téměř jisté, že bude kompatibilní s virtualizovaným prostředím dané organizace. Pokud by ho nebylo možno importovat přímo, je ho možné překonvertovat na jiný typ souboru. Dále je doporučeno změnit nastavená hesla serveru.

Výchozí IP adresa stroje je nastavená na `192.168.0.20`. Pokud to nebude vyhovovat, je nutné změnit IP adresu virtuálního stroje v jeho síťové konfiguraci

a také ji změnit v konfiguračním souboru (u hodnoty `SERVERNAME`) webové stránky v `/etc/apache2 /sites-available/default-ssl.conf`. V případě změny IP adresy je nutné vygenerovat také nový certifikát pro server, kde hodnota `CommonName (CNAME)` musí reflektovat novou adresu. Generování certifikátu může proběhnout na základě vytvořené certifikační autority FuzzyLoc CA, anebo s využitím certifikačních prostředků organizace.

Celkové fungování systému je znázorněno na obrázku 18. Jednotlivé kroky jsou popsány v textu níže a na obrázku jsou vyznačeny číslem kroku.

1. Konfigurace systému začíná krokem 1, kdy se přes webové rozhraní nahraje mapa prostoru, v němž bude posléze probíhat lokalizace. Mapa je nahrána přímo na server, takže je dostupná všem zařízením, která budou přistupovat k dohledovému webovému prostředí. Maximální povolená velikost obrázku mapy je 8MB.
2. Následným krokem je vytvoření zón na základě nahrané mapy. Jednotlivé zóny se vytváří definováním dvou rohů (levý horní a pravý dolní) zóny, takže je možné vytvářet zóny pouze tvaru čtyřúhelníku. Každá zóna musí být pojmenována unikátním jménem dle potřeb administrátora. Tyto parametry zón jsou poté uloženy na server.
3. Vytvořené definice zón jsou poté staženy do FuzzyLoc Admin a na základě toho je vytvořena trénovací množina dat. Samozřejmě musí měření probíhat tak, aby byla vybrána ze seznamu správná zóna, ta, kde se právě měří trénovací data. Soubor s daty se poté nahraje na server.
4. Administrátor dále musí provést registraci lokalizovaných uživatelů a přístupových bodů, které budou využívány k lokalizaci (minimálně jich musí být šest). To vše z jeho mobilní aplikace.
5. Administrátor poté spustí připravený skript `/home/user/FuzzyLocEngine /generateFis.sh`, kde se v závislosti na množství dat vygeneruje soubor `fis.fcl`.
6. Poté ještě musí administrátor vygenerovat certifikáty pro jednotlivá zařízení, která budou lokalizována. Tento krok není momentálně k dispozici z důvodu chyby určité knihovny v Android OS, takže je prozatím nutné pouze nainstalovat kořenový certifikát FuzzyLoc CA do klientských zařízení, anebo využít vlastních certifikačních prostředků. Kořenový certifikát se ještě musí nainstalovat do PC, odkud se bude přistupovat k FuzzyLoc Supervisor.
7. Nyní může být zahájena lokalizace zařízení, kde po prvním spuštění FuzzyLoc Client je uživatel vyzván ke schválení stažení `fis.fcl` ze serveru. Aplikace si ještě vyžádá zadání SSID a IP serveru. Uživatel poté už jen spustí lokalizaci zařízení a aplikace se přepne do režimu běhu na pozadí. Výsledky lokalizace zasílá na server.
8. Závěrečným krokem je zadání URL `https://IP-serveru` na PC s nainstalovaným kořenovým certifikátem. Po přihlášení se zobrazí dohledové prostředí.



Obrázek 18. Schéma lokalizačního systému

5.2. FuzzyLoc Server

Jak už bylo uvedeno dříve, je navrhovaný lokalizační systém centralizovaným řešením, které využívá předností serveru. Je nutno připomenout, že se samotný výpočet pozice mobilních zařízení provádí nezávisle v každém z nich. Hlavním účelem serveru je spojení všech dílčích částí systému do jednoho uceleného prostředí. Základním požadavkem na server je komunikace s mobilními aplikacemi, a to FuzzyLoc Admin a FuzzyLoc Client. Dále je nutné udržovat konfigurační data lokalizačního systému, k čemuž slouží MySQL databáze. Server je také důležitý pro přístup k výsledkům lokalizace pomocí dohledového rozhraní FuzzyLoc Supervisor. Posledním požadavkem je generování odvozovacího systému pomocí Java aplikace FuzzyLoc Engine.

5.2.1. Virtuální stroj

V závislosti na všech zmíněných požadavcích je lokalizační systém postaven na základě různorodých programových prostředí a jazyků jako je PHP, MySQL, Java, Javascript a HTML. Aby všechny tyto složky spolupracovaly správným způsobem, bylo nutné celý systém nějakým způsobem unifikovat. Důvodem jsou hlavně problémy, které by mohly nastat při nasazování systému do reálného prostředí. Stačí samozřejmě malý problém a systém by nemusel pracovat tak, jak byl navržen. Problém by mohl například nastat při spuštění FuzzyLoc Engine pro generaci odvozovacího systému, který závisí na instalaci Javy, jejího umístění, resp. nastavení správné systémové cesty ke spouštěči `javaw.exe`. Zmíněná aplikace by také nemusela mít garantovaná práva na čtení či zápis potřebných souborů. Podobný problém by mohl nastat i v případě PHP skriptů či dalších programových souborů, které určitým způsobem závisí na správné konfiguraci OS. Jistěže lze některé z těchto problémů vyřešit konfiguračními programy, které připraví například adresářovou strukturu, přístupová práva apod., rozhodně je však lepší podobným problémům předejít.

Unifikace lokalizačního systému je založena na virtualizovaném operačním systému, resp. na virtuálním stroji. Díky tomu je možné všechny potřebné úkony přednastavit už při samotném návrhu. Myšlenka nasazení lokalizace v daném prostředí je tedy taková, že lokální administrátor pouze nahraje virtuální stroj do stávajícího (nebo nového) virtualizovaného prostředí. Danému stroji přiřadí IP adresu, provede měření trénovacích dat a tato data pomocí stejné aplikace umístí na server a zahájí generování odvozovacího systému. Poté už jen zaregistruje klienty, kteří budou lokalizováni. Uvedení do chodu by tedy mělo být časově nenáročné.

Takové řešení má pochopitelně své výhody a nevýhody. Nevýhodou může být například zdrženlivost administrátora, který přesně neví, zda daný stroj není bezpečností hrozbou pro místní síť. Tato záležitost je ale vyřešena tak, že virtuální stroj není zcela uzavřený. Tím je myšlena jeho přístupnost administrátorovi, jenž může stroj spravovat a tudíž ho i zabezpečit vzhledem k firemním požadavkům. Kladným přínosem tohoto řešení je jednak zmíněná bezproblémová implementace, ale hlavně bezpečnost. Jedná se totiž o samostatné prostředí, které může být jednoduše zabezpečeno, a v případě poruchy či napadení může být snadno vypnuto, aniž

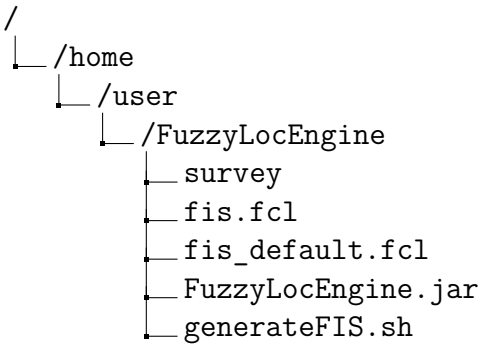
by ovlivnilo nějaký další systém. Pro virtuální stroj může být například vytvořena vlastní LAN síť a provoz může být veden přes firewall apod.

Nejdůležitějším požadavkem na virtuální stroj je univerzálnost jeho nasazení. K tomuto účelu je využit tzv. *Open Virtualization Format* (OVF). Je to kompletní balík definující virtuální stroj, a to jak samotný operační systém (či více systémů), tak doplňkové obrazy disků, certifikáty apod. OVF soubor je vytvořen na základě standardu, který se k němu váže a který zajišťuje přenositelnost mezi virtualizačními platformami. Mezi nejznámější patří například VMware, VirtualBox, Oracle VM, Microsoft Virtual Machine Manager, Red Hat a další. Velmi důležitý je také výběr diskového formátu. Zde byl zvolen VMDK (*Virtual Machine Disk*). Tento formát je nativní formát platformy VMware, kde byl i vytvořen, ale hlavně je podporován drtivou většinou výrobců. Tam, kde není nativně podporován, existují konvertory mezi jinými formáty.

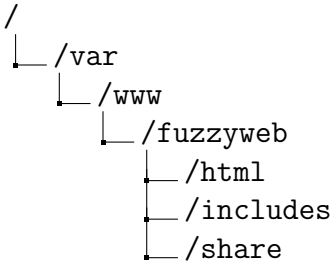
5.2.2. Webový server

Pro virtuální stroj byl zvolen operační systém Ubuntu Server 14.04.4 LTS. Důvodem pro tuto volbu byly zejména nízké nároky na hardwarové prostředky a také široká uživatelská základna. Výsledný OVF není větší než 4 GB. Minimální HW požadavky jsou: 300 MHz x86 procesor, 192 MiB RAM a 1 GB místa na disku.

Pro účely lokalizačního systému postačuje nasazení webového serveru. Jeho základem je programový balík LAMP, což je zkratka používaná pro skupinu pojmů Linux, Apache, MySQL a PHP/Perl/Python. Co se týče veškeré webové komunikace, je pro tyto účely v prostředí Apache vytvořena adresářová struktura obsahující tři hlavní složky– `/html, /includes a /share`, které jsou zastřešovány složkou `/fuzzyweb`, viz obrázek 20. Oproti výchozí konfiguraci bylo nutno předefinovat nastavení tzv. `DocumentRoot` složky, která je nyní definována v adresářové struktuře jako `/var/www/fuzzyweb/html`. Zajímavostí této složky je, že se jedná o veřejnou složku, do které přistupuje webový prohlížeč, resp. uživatel. V praxi to tedy znamená, že například na URL `https://192.168.0.20/index.html` je namapována cesta `/var/www/fuzzyweb/html/index.html`. Z hlediska bezpečnosti to tedy znamená, že je uživatel schopen zobrazit obsah adresáře `/html`, ale už není schopen zobrazit obsah u `/fuzzyweb` a výše. Totéž platí např. i u adresáře `/fuzzyweb/includes`. Díky tomu je jednak možno vytvářet několik oddělených webových stránek, ale hlavně lze využít skryté adresáře pro webovou komunikaci. Obecně je velkou snahou přesouvat co možná nejvíce souborů do skrytých adresářů a až tam je spouštět. Stejný způsob je využit i v této práci. Například pro PHP skripty platí, že jsou uloženy v adresáři `/includes`, odkud jsou volány pomocnými skripty v `/html`, které obsahují pouze příkaz `include(skryty_skript.php)`. Tím je dosaženo toho, že se skrytý skript vykoná, i když je mimo přímou dostupnost uživatele. Na druhou stranu teoreticky není možné (při správné konfiguraci serveru), aby byl PHP skript přečten jako text, na rozdíl od HTML či JavaScript souborů. Při zavolání PHP skriptu je vždy vykonán, není však zobrazen jeho obsah. Nicméně, pokud tyto skripty obsahují citlivé údaje, nebo přistupují do databáze s citlivými údaji, preventivně se z `DocumentRoot` složky přesouvají.



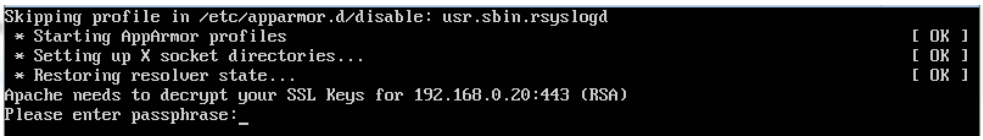
Obrázek 19. Adresář FuzzyLocEngine



Obrázek 20. Navržená konfigurace

Součástí webového serveru je i adresář pro generování odvozovacího systému, viz 19. Po spuštění skriptu `generateFIS.sh` se do tohoto adresáře nakopíruje soubor `survey` s trénovacími daty. Pomocí Java aplikace `FuzzyLocEngine.jar` se poté vygeneruje odvozovací systém do souboru `fis.fcl`, který se zkopíruje do umístění, odkud si ho stahují klientské aplikace. Soubor `fis_default.fcl` slouží jako inicializační popis FIS systému.

Nutno k serveru ještě doplnit, že na některých místech bylo nutné zadat přihlašovací údaje. Jelikož se jedná o prvotní testovací verzi systému, hesla jsou na většině míst stejná, popřípadě slabá. V produkčním prostředí se samozřejmě musí změnit. Po startu serveru se objeví důležitá položka, a to zadání hesla pro SSL protokol, resp. hesla k certifikátu, viz obr. 21. Bez zadání hesla nebude komunikace mezi prvky šifrována, a jelikož server podporuje pouze šifrované spojení, lokalizační systém nebude fungovat. Heslo je *FuzzyLocSystem*. Poté se zadává jméno uživatele serveru, což je *user*, a je to zároveň i heslo. Do rozhraní PhpMyAdmin je už. jméno *root*, heslo *FuzzyLocSystem*. Toto heslo je i u všech doposud vygenerovaných certifikátů.



Obrázek 21. Zadávání hesla pro SSL/TLS

5.2.3. PHP skripty

Skripty napsané ve skriptovacím jazyce PHP kompletně zprostředkovávají komunikaci se serverem, resp. s databází. Na obrázku 22 jsou tyto skripty umístěny ve dvou různých adresářích (1) a (2). Důvod pro toto členění je popsán výše, jde o snahu, aby bylo co nejvíce souborů vně veřejně přístupného adresáře `DocumentRoot`. V případě obrázku 22 se jedná o adresář `/var/www/fuzzyweb/html`. Význam jednotlivých skriptů je následující:

- Skupina skriptů v adresáři (1) je pouze odkazem na soubory v adresáři (2). Tzn. že jsou aplikacemi nejdříve volány skripty ve veřejné složce, které odkazují na skripty ve skryté složce.

- `conf.php` je konfigurační skript, který obsahuje přístupové údaje do databáze a definice globálních proměnných s názvy tabulek.

- `delAp.php` má za úkol na vyžádání mazat registrované přístupové body z tabulky.

- `delItem.php` na vyžádání vymaže registrované uživatele či zařízení z tabulky.

- `getItem.php` přistupuje k obsahu tabulek s registrovanými uživateli a registrovanými přístupovými body.

- `setAp.php` ukládá nově registrované AP do tabulky.

- `saveFileSurvey.php` je skript, pomocí kterého se nahrává soubor s trénovací množinou dat na server. Na základě něho je vygenerován soubor `fis.fcl`, popisující fuzzy odvozovací systém.

- `saveZones.php` je využíván webovým rozhraním pro ukládání nadefinovaných zón.

- `sendFileFis.php` umožňuje klientům stáhnout `fis.fcl` do zařízení a používat ho k odvozování pozice.

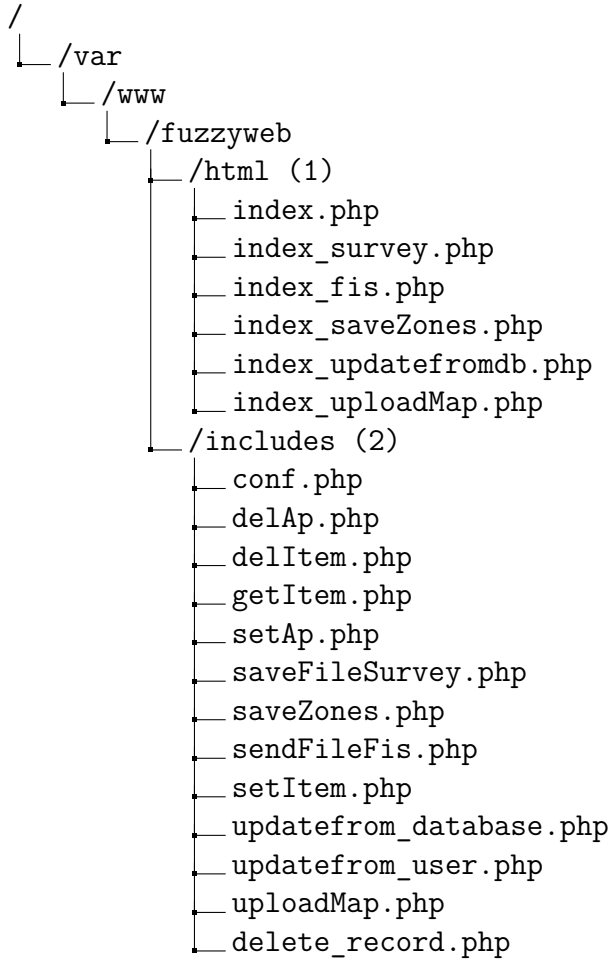
- `setItem.php` ukládá nově registrované uživatele/zařízení do tabulky.

- `updatefrom_database.php` je skript, který na základě hodnot všech tabulek databáze vygeneruje a odešle data, popisující souhrnné informace důležité k zobrazení ve webovém rozhraní.

- `updatefrom_user.php` je volán jednotlivými klienty, kteří chtějí uložit svoji aktuální polohu do tabulky databáze.

- `uploadMap.php` je skript volaný z webového rozhraní a ukládá na server mapu prostředí, kterou správce vložil při vytváření zón pro samotnou lokalizaci.

- `delete_record.php` je skript, který je volán klientskou aplikací při ukončení lokalizační služby, aby smazal lokalizační záznam konkrétního zařízení z tabulky `buffer_user`.



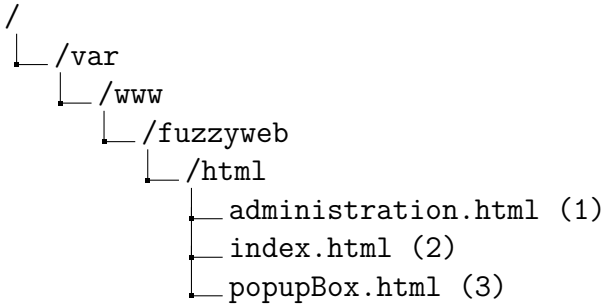
Obrázek 22. PHP adresáře

5.2.4. HTML soubory

HTML soubory jsou celkem tři (viz obr. 23). Všechny definují pouze vizuální vjem webového rozhraní a odkazují na skripty psané v JavaScript. Na rozdíl od PHP skriptů je obsah HTML a JavaScript souborů zobrazitelný komukoli, kdo si zobrazí v prohlížeči zdrojový kód stránky. Nicméně, soubor (2) je webová stránka, kterou Apache vrátí na požadavek o adresu `https://192.168.0.20/`. Ostatní dva soubory jsou pomocné soubory jednak pro administraci (1) (definice zón, nahrávání mapy na server apod.), jednak pro zobrazení vyskakovacího okna (3).

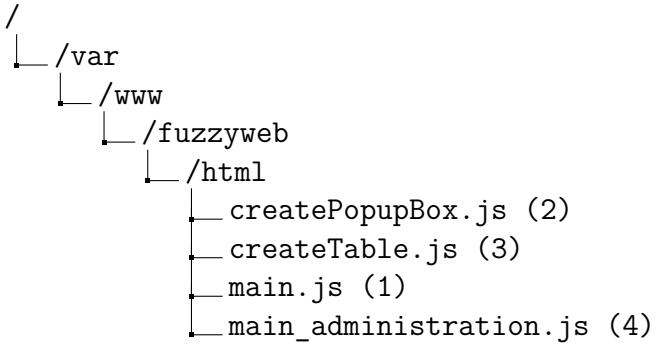
5.2.5. JavaScript soubory

Klíčová logika celého webového rozhraní je psána ve skriptovacím jazyce JavaScript. Hlavní webová stránka `index.html` využívá skripty (1),(2) a (3). První zmiňovaný je hlavní skript, který se stará o ovládací prvky, komunikaci se serverem, souřadnicový systém pro správné zobrazení mapy a jejích prvků, dále slouží k zobrazení lokalizovaných uživatelů a k další algoritmizaci. Zbylé dva jsou pomocné skripty na vytvoření tabulky lokalizovaných uživatelů, resp. vytvoření zobrazova-



Obrázek 23. HTML soubory

cího okna pro zobrazení seznamu uživatelů v dané zóně. Poslední skript (4) na obrázku 24 obsahuje hlavní logiku pro administrativní část webového rozhraní, a to `administration.html`.



Obrázek 24. JavaScript soubory

5.2.6. Databáze MySQL

Veškerá výměna informací mezi jednotlivými prvky lokalizačního systému probíhá pomocí MySQL databáze. Pro snadnější přístup k její administraci je využito webové rozhraní `phpMyAdmin`. Databáze obsahuje celkem čtyři tabulky. První z nich je `buffer_user`, tab. 3. Tato tabulka slouží ke sběru lokalizačních dat ode všech klientů. Tzn. že poté, co klientská aplikace FuzzyLoc Client odvodí svou polohu, zanese informaci o své identitě (MAC) a poloze (ZONE), v níž se nachází, právě do této tabulky. Do ní přistupuje z opačné strany dohledové rozhraní FuzzyLoc Supervision, které následně zobrazí výsledek.

Tabulka 3. $buffer\_user$

ID	MAC_USER	$ZONE$
1	d4:cb:6d:a8:1b:73	3

Další tabulka, `table_ap`, tab. 4, udržuje informace o registrovaných přístupových bodech, které lokalizační systém využívá. Tyto hodnoty jsou zadávány správcem sítě. Klientská aplikace si je vždy stáhne při svém spuštění a dle nich poté skenuje RSSI data. Totéž platí i o FuzzyLoc Admin.

Tabulka 4. *table_ap*

ID	*MAC_AP*	*AP_NAME*
1	b4:ca:6d:95:1b:32	AP1

Důležitá je také tabulka 5, `table_users`. V ní jsou registrovaní uživatelé, kteří jsou identifikováni pomocí MAC adresy a jména. První hodnota slouží k systémové komunikaci, druhá k lepší identifikaci uživatele. Tato tabulka také obsahuje položku typu uživatele. Může nabývat dvou hodnot, a to `User` a `Device`. Takto lze rozlišit, zda se jedná o lokalizaci uživatele, nebo například nemocničního vybavení.

Tabulka 5. *table_users*

ID	*MAC_USER*	*USER_NAME*	*USER_TYPE*
1	b5:d3:6c:95:1b:af	Jaroslav	User
2	e5:d3:55:95:1b:4f	Vozik	Device

Poslední tabulkou je `table_zones`, tab. 6. Obsahuje konfigurační parametry jednotlivých zón. Číslo zóny je opět systémový parametr, který je překládán do jména zóny pro její jasnější identifikaci. Poté následují čtyři hodnoty, které definují velikost zóny. Ty jsou vytvářeny při definování zón ve webovém rozhraní. Poslední parametr tabulky je aktuální počet uživatelů v zóně.

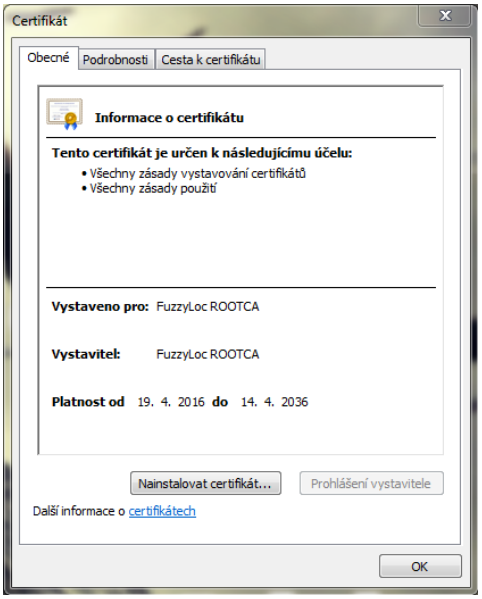
Tabulka 6. *table_zones*

ID	*ZONE _NUMBER*	*ZONE _NAME*	X1	Y1	X2	Y2	*USER _COUNT*
1	1	Ucebna	0	0	137	250	3

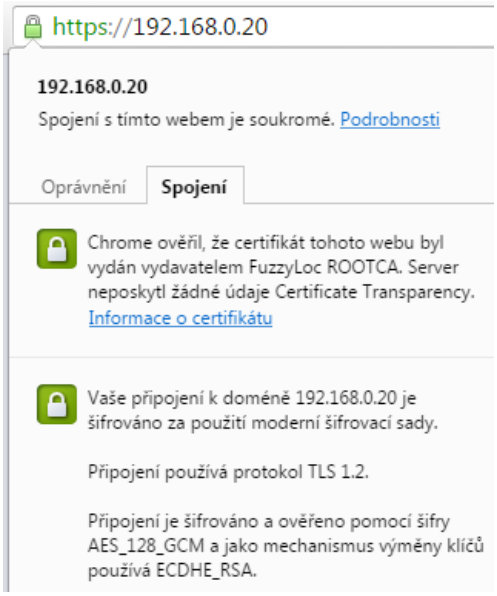
5.2.7. Správa certifikátů

Zabezpečení je už dnes nedílnou součástí veškeré TCP/IP komunikace. Nejinak tomu je i v případě navrhovaného lokalizačního systému, který se skládá ze tří různých zdrojů informace. Jsou to mobilní aplikace, webový server a webové dohledové prostředí. Základem zabezpečení je využití protokolu HTTP v součinnosti s protokolem SSL/TLS, který umožňuje šifrování komunikace mezi dvěma zdroji informace. Konkrétně se jedná o šifrování komunikace mezi mobilní aplikací a serverem, a potom mezi serverem a webovým dohledovým prostředím. Pro dosažení šifrované komunikace je nejjednodušší variantou vygenerování certifikátu pouze pro server. Tyto certifikáty se nazývají *self-signed certificates*, jelikož se podepisují samy sebou. Pokud je využita tato varianta certifikátů, webový prohlížeč naváže sice šifrovanou HTTPS komunikaci, ale zároveň uživateli ohlásí, že serverový certifikát nemůže ověřit, protože není podepsán známou certifikační autoritou (CA). Každý systém, resp. prohlížeč, disponuje seznamem důvěryhodných CA, kterými ověřuje certifikáty protilehlé strany. Tento problém lze vyřešit tím, že se do daného seznamu naimportuje kořenový certifikát, na jehož základě byl serverový certifikát

v minulosti odvozen, a na jehož základě se i ověří.



Obrázek 25. Kořenový certifikát FuzzyLoc ROOTCA



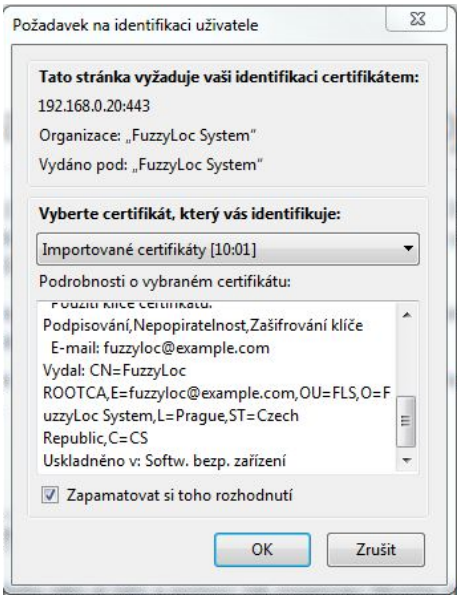
Obrázek 26. Zabezpečení v prohlížeči Chrome

Základem serverových či uživatelských certifikátů je tedy tzv. kořenový certifikát *Root CA*. Je zajímavé, že je tento certifikát také *self-signed*. Certifikační autority mají obecně charakter stromové hiearchie, což umožňuje, aby se z kořenového certifikátu odvozovaly další CA. Někdy se také označují jako *Intermediate CA*, ze kterých se může odvodit další CA na nižší úrovni, nebo mohou sloužit k odvozování serverových a klientských certifikátů a k jejich podpisu. Totéž platí i o *Root CA*. Každý certifikát je definován několika parametry, kde mezi ty nejdůležitější patří privátní a veřejný klíč, důvěryhodné CA (ze kterých byl odvozen), období platnosti a dále tzv. *Common Name*. *Common Name* je parametr, který definuje, komu byl certifikát vydán. Tzn. že v případě serverového certifikátu bude mít *Common Name* hodnotu doménového jména serveru (např. `www.domena.cz`), anebo IP adresu, na které je webový server dostupný. V případě uživatelského certifikátu zde může být například jméno uživatele, jeho e-mail apod. Nutno dodat, že hodnota tohoto parametru musí být v celém stromu unikátní. Platnost certifikátu může být různá, pro uživatele například rok, pro server tři roky a pro kořenový certifikát například deset let. Samotný certifikát a příslušný veřejný klíč se generuje na základě privátního klíče. Je tedy nutné klíč uschovat do bezpečného místa, protože v opačném případě by mohl útočník převzít naši identitu či identitu serveru. Privátní klíč se generuje pomocí šifrovacího algoritmu AES (z angl. *Advanced Encryption Standard*). Generovaný certifikát je posléze výstupem hash funkce SHA (z angl. *Secure Hash Algorithm*), do které vstupuje již vygenerovaný privátní klíč a konfigurační soubor popisující *Root (Intermediate)* CA.

Pro účely této práce byla vytvořena privátní *Root CA*, viz obr. 25. Samozřejmě je možné získat *Intermediate CA* od nějaké známé certifikační autority za určitý poplatek, anebo je možné zdarma využít služeb *Let's Encrypt*. Jaký způsob certi-

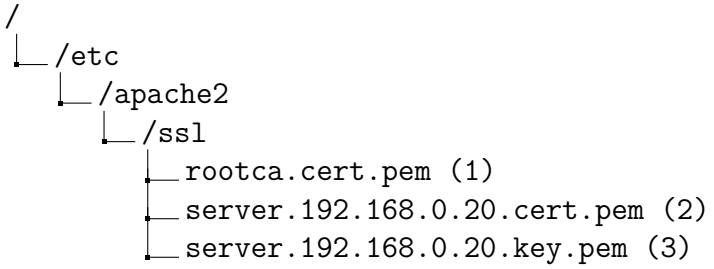
fikace se pro lokalizační systém využije, záleží už na administrátorovi samotném. K certifikaci je využit nástroj `OpenSSL`. Nejprve se tedy vygeneruje privátní klíč pro kořenový certifikát a následně kořenový certifikát, který podepíše sám sebe. Tím vznikla *Root CA*. Poté se vygeneruje privátní klíč pro server/uživatele. Pomocí tohoto klíče je nutné vygenerovat požadavek o podpis (z angl. *Certificate Signing Request*) a na základě kořenového certifikátu se tento požadavek podepíše, čímž se vytvoří výsledný server/uživatelský certifikát.

Pro šifrování TCP/IP komunikace stačí jen importovat kořenový certifikát do uživatelského zařízení a na server importovat serverový certifikát a jeho příslušný privátní klíč. Tím však zabezpečení lokalizačního systému není úplné. Je nutné také autentizovat jednotlivá uživatelská zařízení, zda mají právo přistupovat k prostředkům serveru. K tomu jsou určeny již zmíněné uživatelské certifikáty, které se instalují do zařízení společně s příslušnými privátními klíči. Po příslušné konfiguraci na straně serveru, vyžaduje server od každého klienta autentizaci pomocí osobního certifikátu, vydaného stejnou kořenovou CA. Teorie se bohužel liší od praxe tím, že tímto způsobem lze zatím ověřovat pouze webový prohlížeč na straně dohledového prostředí, viz 27. V operačním systému Android se totiž vyskytuje chyba, která prozatím znemožňuje využít autorizaci pomocí vlastního certifikátu, resp. bude se muset najít řešení, jak tuto chybu obejít. HTTPS komunikace však funguje bez problému.

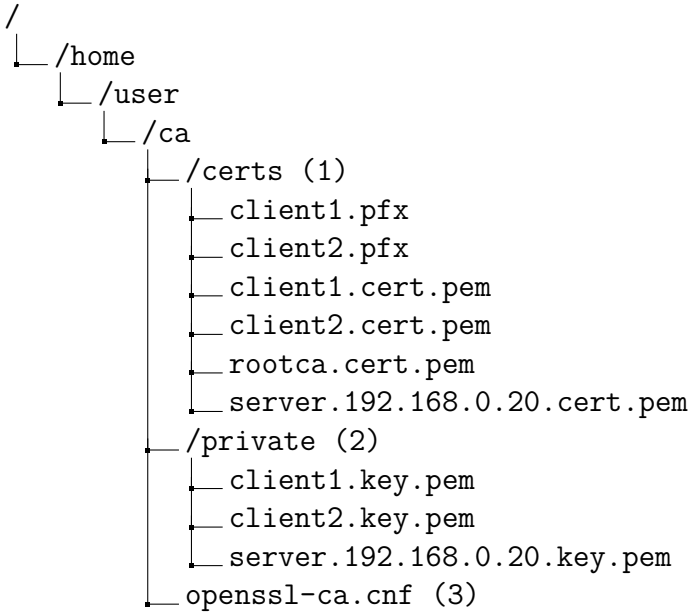


Obrázek 27. Autentizace uživatele dohledového prostředí

Níže jsou uvedeny cesty k důležitým souborům certifikace. Na obrázku 28 je konfigurační adresář webového serveru, který obsahuje kořenový certifikát (1), serverový certifikát (2) a privátní klíč serveru (3), což jsou povinné soubory konfigurace. Na obrázku 29 je domovský adresář, kde složka `certs` (1) obsahuje všechny generované certifikáty (včetně výše zmíněných) a složka `private` (2) obsahuje všechny privátní klíče. Soubory s koncovkou `.pfx` v sobě obsahují privátní klíč i certifikát. Soubor (3) je konfigurační soubor pro generování kořenového certifikátu.



Obrázek 28. SSL adresář webového serveru



Obrázek 29. CA adresář pro generování certifikátů

5.3. Mobilní aplikace

5.3.1. FuzzyLoc Client

Požadavky na aplikaci

První aplikací pro platformu OS Android je *FuzzyLoc Client*. Hlavním úkolem této aplikace je výpočet pozice zařízení a následné zprostředkování výsledku centrálnímu serveru. Aplikace je tedy určena pro uživatele, který má být lokalizován. Pro tuto a následující aplikaci (*FuzzyLoc Admin*) platí, že minimální podporovaná verze operačního systému je 4.0.3, Ice Cream Sandwich. Pro účely lokalizace je nutné, aby měla aplikace uživatelské rozhraní, kde však není požadavkem zobrazení aktuální pozice samotnému uživateli. Rozhraní plní funkci čistě konfigurační, tzn. že slouží k nastavení síťových parametrů (IP serveru, SSID sítě apod.).

Programová implementace

Aplikace je popsána sadou tříd, jejichž účel je následující:

- Třída `MainActivity` je spouštěcí třída, která vytváří pracovní adresář v interní paměti telefonu, který slouží k uložení konfiguračního souboru *fis.fcl*, viz kapitola 4.5. Po startu aplikace je přítomnost tohoto souboru vždy automaticky zkontrolována a v případě jeho absence je znovu stažen z adresáře na centrálním serveru. Další úlohou je kontrola nastavení síťové konfigurace a stažení aktuálního seznamu registrovaných přístupových bodů, podle kterých se lokalizuje. Důležitým úkolem této třídy je též aktivace úlohy na pozadí, která skenuje RSSI data. Na základě knihovny jFuzzyLogic a souboru *fis.fcl* je odvozena pozice zařízení a posléze odeslána na server.

- Třída `GetData` umožňuje stažení seznamu registrovaných AP.

- Třída `DownloadFile` umožňuje stažení souboru `fis.fcl`.

- Třída `ScanCalibration` je navržena pro provedení kalibračního procesu, ve kterém se zjistí systémový interval skenování WiFi okolí. Na základě zjištěného výsledku se nastaví charakter skenování okolí ve fázi lokalizace, viz kapitola 2.2.1.

- Třída `Notification` je přímo volána z předcházející spouštěcí třídy a obsluhuje notifikační rozhraní v hlavním menu OS Android. Díky tomu může uživatel zjistit, zda na pozadí běží lokalizační procesy, může je také zastavit nebo vyvolat konfigurační menu aplikace.

- Třída `BackgroundService` je volána také ze spouštěcí třídy a jejím účelem je obsluha procesu na pozadí, tedy skenování úrovně signálu a dalších procesů. Proces na pozadí má své vlastní výpočetní vlákno, čili zbytečně nezatěžuje hlavní vlákno operačního systému. Uživatel tedy může na mobilním zařízení dále pracovat.

- Třída `MultiScan` má na starosti sběr RSSI dat (využitím třídy `Scan`) a jejich zpracování do tvaru, ve kterém je poskytuje fuzzy logice (třída `FuzzySolver`). Získaný výsledek je poté předán třídě `SendData` k odeslání na server.

- Třída `Scan` slouží právě k přístupu k naměřeným RSSI datům v zásobníku OS Android.

- Třída `KalmanFilter` provádí filtraci (korekci fluktuace) naměřených hodnot (viz kapitola 2.2.2), které jsou poté průměrovány a reprezentují konkrétní vzorek RSSI.

- Třída `SendData` obstarává komunikaci se serverem, resp. odesílání výsledků na server. Toho je dosaženo využitím HTTP dotazu POST, který zavolá PHP skript (umístěný na webovém serveru), předá mu informace, a ty jsou poté tímto skriptem zpracovány. Informacemi jsou výsledek lokalizace, MAC adresa mobilního zařízení a popřípadě další volitelné proměnné.

- Třída `FuzzySolver` propojuje konfigurační soubor *fis.fcl* s knihovnou jFuzzyLogic a vrací výslednou pozici.

- Třída `Global` pouze udržuje globální statické veřejné proměnné.

- Třída `InitConfig` je jednou ze tří aktivit, které slouží jako rozcestník mezi jednotlivými položkami konfigurace. Těmi je jazyk, síťové parametry a parametry lokalizace.

- Třída `Language` umožňuje změnu mezi českým a anglickým jazykem.

- Třída `NetworkValues` slouží k zadávání síťové konfigurace jako IP serveru, uživatelského jména, hesla a také SSID bezdrátové sítě, která je zdrojem RSSI hodnot. Uživatelské jméno a heslo je jedna z možností autentizace vůči lokalizačnímu systému.

- Třída `ActivityFromService` umožňuje spuštění aktivity s konfiguračními možnostmi při současném vypnutí procesu lokalizace.

- Třída `ServiceFinalizer` na základě uživatelova vstupu ukončí všechny úkony na pozadí a tím vypne lokalizaci zařízení.

- Třída `AsyncDelegate` je rozhraní, které se využívá pro předání notifikace mezi ukončenou úlohou na pozadí a třídou, která o tom musí být informována.

5.3.2. FuzzyLoc Admin

Požadavky na aplikaci

Druhou mobilní aplikací je *FuzzyLoc Admin*. Oproti předcházející aplikaci má zcela jiné využití, a to k administraci systému. Je proto využívána administrátorem, nikoliv uživatelem. Jedním z požadavků na ni je možnost registrovat do databáze jednak nové uživatele, jednak přístupové body. Mnohem důležitějším úkolem této aplikace je však poskytnout rozhraní pro vytvoření trénovací množiny dat.

Programová implementace

Aplikace je popsána sadou tříd, jejichž účel je následující:

- Třída `MainActivity` je spouštěcí třída. Vytváří pracovní adresář v interní paměti telefonu, který slouží k uložení souboru `survey` (trénovací množina dat). Dále slouží jako rozcestník mezi jednotlivými funkcemi aplikace.

- Třída `CreateSurvey` slouží k generování trénovací množiny dat.

- Třída `GetData` umožňuje stažení seznamu registrovaných AP, uživatelů či seznamu nadefinovaných zón.

Obrázek 30. *Úvodní obrazovka*

- Třída `UploadFile` umožňuje nahrání souboru `survey` na server, kde je dále zpracováván.

- Třída `MultiScan` má na starosti sběr RSSI dat (využitím třídy `Scan`) a jejich zpracování do tvaru, ve kterém jsou zapisována do souboru `survey`. Získaný soubor je poté předán třídě `UploadFile` k odeslání na server.

- Třída `ScanCalibration` je navržena pro provedení kalibračního procesu, ve kterém se zjistí systémový interval skenování WiFi okolí. Na základě zjištěného výsledku se nastaví charakter skenování okolí ve fázi lokalizace, viz kapitola 2.2.1.

- Třída `Scan` slouží právě k přístupu k naměřeným RSSI datům v zásobníku OS Android.

- Třída `KalmanFilter` provádí filtraci (korekci fluktuace) naměřených hodnot (viz kapitola 2.2.2), které jsou poté průměrovány a reprezentují konkrétní vzorek RSSI.

- Třída `Global` pouze udržuje globální statické veřejné proměnné.
- Třída `Language` umožňuje změnu mezi českým a anglickým jazykem.
- Třída `NetworkValues` slouží pro zadávání síťové konfigurace jako IP serveru, uživatelského jména, hesla a také SSID bezdrátové sítě, která je zdrojem RSSI hodnot. Uživatelské jméno a heslo je jedna z možností autentizace vůči lokalizačnímu systému.
- Třída `AddAp` umožňuje registraci nového AP do lokalizačního systému.
- Třída `DelAp` slouží k odstranění AP z lokalizačního systému.
- Třída `DeleteAddAp` vykonává samotnou konektivitu s databází a definuje požadavek na smazání či přidání AP.
- Třída `AddUserDevice` umožňuje registraci nového uživatele či zařízení do lokalizačního systému.
- Třída `DelUserDevice` slouží k odstranění uživatele či zařízení z lokalizačního systému.
- Třída `DeleteAddUserDevice` vykonává samotnou konektivitu s databází a definuje požadavek na smazání či přidání uživatel či zařízení.
- Zcela poslední je rozhraní `AsyncDelegate`, které se využívá pro předání notifikace mezi ukončenou úlohou na pozadí a třídou, která o tom musí být informována.
- Třída `IOFile` je třída, která vykonává zápis trénovacích dat do souboru.

5.4. FuzzyLoc Supervisor

Posledním prvkem lokalizačního systému je *FuzzyLoc Supervisor*, což je webové rozhraní pro dohledovou a administrativní činnost. Z pohledu administrace se jedná o dva základní kroky. Prvním krokem je nahrání mapového podkladu na server ve formě obrázku. Tento obrázek slouží pro pozdější grafické zobrazení pozic jednotlivých uživatelů. Slouží však také ke druhému kroku administrace, čímž je definování jednotlivých zón. Zóny (resp. jejich počet a rozmístění) jsou navrženy administrátorem. Zóna může být libovolně velká, může tedy definovat jednu místnost či více. Může definovat i celý blok budovy, vše záleží na požadavcích na lokalizaci. Vytvoření zóny je provedeno dvěma kliky do mapy, které definují levý horní a pravý dolní roh zóny, jedná se tedy o čtyřúhelník. Po tomto zakreslení je administrátor vyzván k pojmenování zóny. Jméno by samozřejmě mělo vystihovat charakter či účel. Pro

lepší orientaci mohou mít zóny různou barvu. Na závěr je potřeba všechny informace uložit na server pomocí konkrétního tlačítka.

Druhým účelem rozhraní je samotné zobrazení uživatelů, což je zprostředkováno na základě nahrané mapy. Mapa může být přibližována nebo oddalována (stejně jako tomu bylo i v minulém případě). Uživatele je možno zobrazit dvěma různými režimy zobrazení. Prvním je celkový pohled, který každou zónu označí značkou a číslem, udávajícím počet uživatelů a zařízení v dané zóně. Pokud je potřeba zjistit, kdo se v dané zóně nachází, je možné najet kurzorem myši na značku a zobrazí se informační okno, kde je seznam přítomných uživatelů. Pro rychlejší nalezení uživatele slouží funkce, kdy se klikne na uživatele v tabulce a tím se zvýrazní zóna, v níž se uživatel nachází. Druhý režim zobrazení je individuální. To znamená, že se z tabulky uživatelů mohou vybrat tři konkrétní, kteří budou jednotlivě a s popiskem jména zobrazeni na mapě. Co se týče tabulky samotné, každý záznam zde je definován svým jménem, poté tím, zda se jedná o zařízení či uživatele a dále jménem zóny, kde se nachází. V režimu individuálním se zde objeví navíc grafická značka, zda je uživatel vybrán ke zobrazení na mapě.

5.5. Testování

Navrhovaný lokalizační algoritmus byl podroben dvěma testováním, z nichž každé probíhalo v jiném prostředí. Metodika testování je u obou prostředí stejná. Nejprve se rozmístilo šest přístupových bodů (s ohledem na využívané WiFi kanály). Prostor se poté rozdělil do několika zón, které zpravidla kopírovaly jednotlivé místnosti. Takovéto rozdělení však není nutné, do zóny může patřit několik místností, anebo naopak jedna místnost (např. chodba) může být rozdělena do několika zón. Posléze bylo v každé zóně určeno několik bodů, v nichž probíhal sběr trénovacích dat pomocí aplikace *FuzzyLoc Admin*. V každém bodě byla provedena dvě měření. Celá trénovací množina byla poté vložena do *FuzzyLoc Engine*, která vygenerovala popis odvozovacího fuzzy systému do souboru `fis.fcl`. Ten byl vložen do aplikace *FuzzyLoc Client*, která pro účely testování nezasílala data na server, ale výsledek přímo zobrazovala. Testování lokalizace probíhalo v definovaných bodech a při konkrétní orientaci osoby držící mobilní telefon. V každém bodě se vygenerovalo dvacet výsledků pozice, resp. zóny, kde systém předpokládal polohu zařízení. Výsledné ohodnocení přesnosti je tedy dáno úspěšností určení správné zóny.

5.5.1. První testovací prostředí

První prostředí je zobrazeno ve schématu 31. Na základě dostupných místností bylo definováno šest zón. Chodba je rozdělena na dvě zóny, stejně jako prostor 304*a*. Prostředí zde bylo velmi různorodé, např. co do vybavení, nábytku, otevřených prostor apod. Bylo to zároveň prostředí velmi náročné na lokalizaci, protože zde je mnoho vlivů, které ovlivňují výslednou úspěšnost. Prvním vlivem byl pohyb osob, protože měření bylo provedeno v pracovní době. S pohybem osob také souvisí otevírání a zavírání dveří, resp. situace, kdy byla trénovací data měřena při uzavřených dveřích, ale v důsledku cizího zavinění byla lokalizace odhadována

pro otevřené dveře. To se týká zejména zón 1, 3 a 5. Kromě toho v zóně 1 bohužel probíhalo měření RFID technologií, takže i to mohlo mít vliv na přesnost výsledku. Posledním vlivem, což je spíše charakteristika prostředí, je pomyslný trojúhelník tvořený lokalizačními body (obdélníkové se šipkou) (5), (6) a (8). Do zóny 2 totiž nevedou klasické vstupní dveře, pomyslný trojúhelník se dá tedy pokládat za jeden prostor, čemuž odpovídají i výsledky.

Tabulka 7. Úspěšnost stanovení správné zóny pro obě prostředí

Číslo lokalizačního bodu	Úspěšnost v prvním prostředí [%]*	Úspěšnost ve druhém prostředí [%]*
1	65	100
2	40	100
3	75	90
4	80	95
5	65	95
6	50	80
7	75	85
8	50	90
9	80	90
10	90	75
11	55	45
* 20 *měření na* 1 *lokalizační bod*		

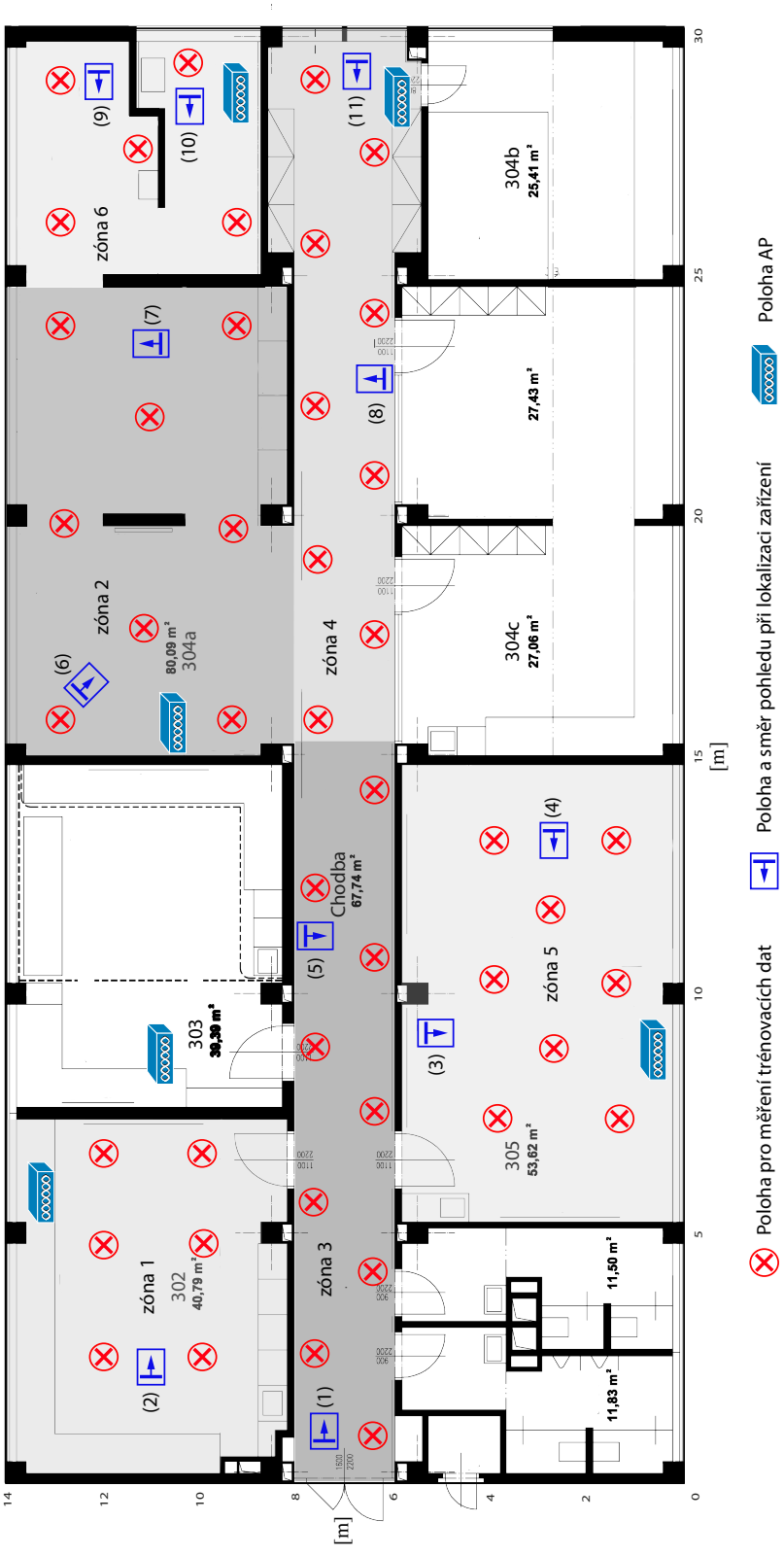
Z tabulky 7 lze usuzovat, že lokalizační systém má tendenci lokalizovat s větší úspěšností v zónách, které jsou v podstatě standardními místnostmi. To platí v případě lokalizačních bodů (3), (4), (7), (9), (10). Zde je tedy rozlišitelnost místnosti na dobré úrovni. Lokalizační bod (2) byl zmíněnými vlivy ovlivněn nejvíce, a proto se nedá považovat za důvěryhodné měřítko pro posouzení úspěšnosti systému. Chybovost ve zmíněném pomyslném trojúhelníku je způsobena hlavně velmi nízkou rozlišitelností mezi zónami. To znamená, že pokud by byly zóny odděleny příčkami, bude mít každá zóna svoji charakteristickou mapu pokrytí a systém ji dokáže lépe detekovat.

5.5.2. Druhé testovací prostředí

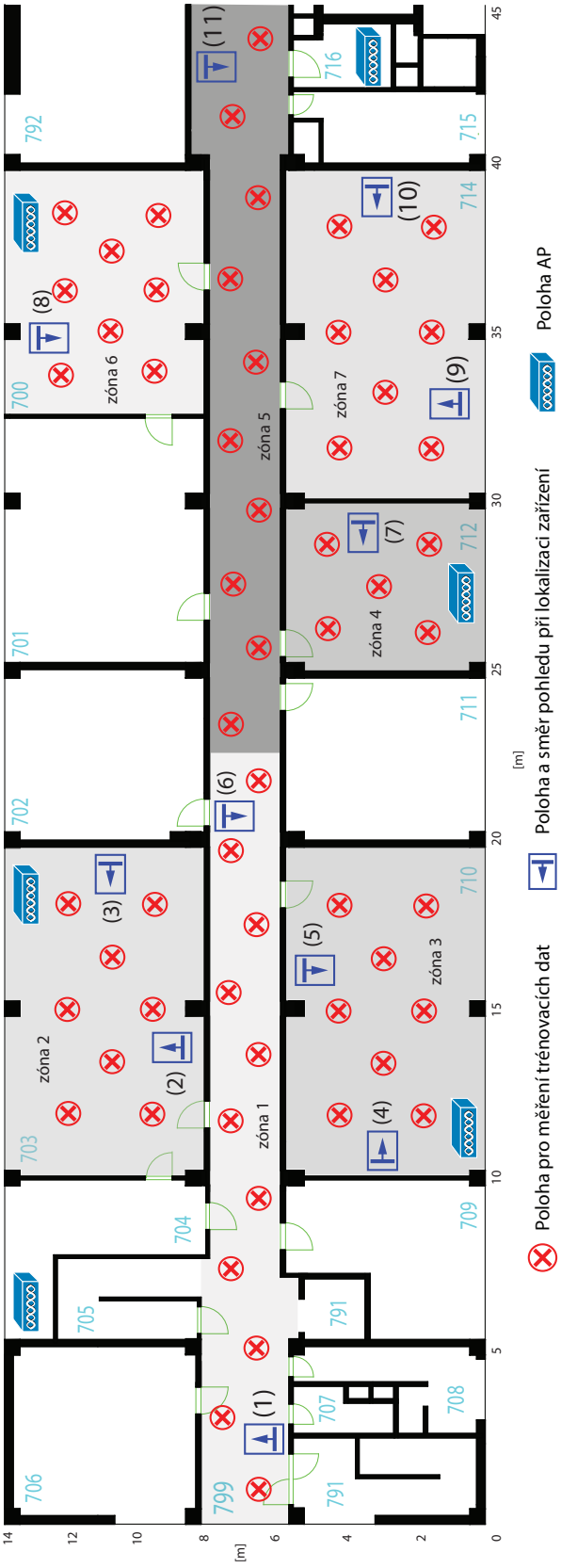
Schéma testování ve druhém prostředí je na obrázku 32. Postup testování byl shodný s předchozím. V prostoru bylo definováno sedm zón, které jsou barevně odlišeny, a chodba byla opět rozdělena na dvě menší zóny. Měření probíhalo v odpoledních hodinách, tedy spíše v klidném prostředí. Prostředí je na rozdíl od předchozího členitější, větší a odráží typický model jedné chodby s postranními oddělenými kancelářemi. To je z pohledu *room-level* lokalizace ideální prostředí, protože každá místnost je daleko lépe rozlišitelná od ostatních, a to díky specifickému útlumu signálu pro jednotlivé místnosti. Ten je ovlivněn jednak členitostí, ale také odliš-

nou tloušťkou či materiálem zdí. Může být ovlivněn i bezpečnostními dveřmi, které v předchozím prostředí nejsou. Je také důležité zmínit, že předchozí testovací prostředí (resp. místnosti, kde bylo možno testovat) bylo spíše otevřeného charakteru, a proto úspěšnost lokalizace byla na nízké úrovni.

Na první pohled se může zdát, že je v každé zóně umístěno AP, což může vést k myšlence, že právě tato konfigurace (1 zóna = 1 AP) je nezbytná pro samotnou lokalizaci. Tento předpoklad mohl platit pro minulou verzi lokalizačního systému (viz [25]), kde se bralo v potaz n signálů s nejvyšší úrovní. Nejvyšší úroveň tedy mohla mít nejvyšší váhu v samotném výpočtu. V současném systému byla využita metoda skenování rozdílových hodnot $\Delta RSSI$ mezi konkrétními přístupovými body, takže si jsou všechny rovny a všechny ovlivňují úspěšnost stejným způsobem. To dokládají i lokalizační body (9) a (10), které měly dobrou úspěšnost, aniž by bylo v dané zóně AP. Jednotlivé lokalizační body, resp. jejich úspěšnosti, jsou v tabulce 7. Je zde zjevný rozdíl mezi úspěšnostmi obou testování, což dokládá, že je lokalizační systém silně závislý na konkrétním prostředí. To v praxi znamená, že může být vhodný např. pro členité nemocniční prostředí, ale naopak pro lokalizaci v moderní otevřené kanceláři vhodný nebude.



Obrázek 31. Schéma prvního testování



Obrázek 32. Schéma druhého testování

6. Závěr

Diplomová práce se zabývá komplexní úlohou, jejímž výsledkem je systém pro lokalizaci mobilních zařízení uvnitř budov, a to na úrovni zóny. Zóna může být definována jako místnost, více místností, anebo jako část budovy. Nosným médiem je lokální bezdrátová síť, resp. technologie WiFi. S využíváním této technologie se však pojí několik klíčovým faktorů, které degradují úspěšnost lokalizace. Za dva nejdůležitější lze označit fluktuaci signálu a rozdílnost WiFi adaptérů.

Ani jeden z nich se nedá zcela eliminovat, proto byly navrženy metody, které jejich vliv alespoň minimalizují. Fluktuace signálu je minimalizována metodou Kalmanova filtru, jehož hlavní charakteristikou je odolnost vůči extrémům v naměřených RSSI hodnotách (naměřené hodnoty úrovně signálu). Lokalizační systém využívá RSSI hodnoty jako stěžejní metriku pro odvození pozice zařízení. Z toho důvodu se tyto hodnoty pro jednotlivá měření průměrují z několika vzorků. Pokud se však průměrují přímo naměřená data, je velká pravděpodobnost, že bude výsledek příliš ovlivněn nějakou extrémní hodnotou, která se mezi nimi může vyskytnout (extrémně nízká nebo extrémně vysoká úroveň signálu). Tyto extrémy samozřejmě ovlivní výsledný průměr, který by v opačném případě mohl lépe charakterizovat dané měření. Kalmanův filtr extrémní hodnoty neodfiltruje zcela pryč, naopak s nimi počítá, ale oproti standardnímu průměrování není výsledná hodnota natolik ovlivněna extrémy.

Druhý klíčový problém spočívá v rozdílnosti jednotlivých WiFi adaptérů. Tento případ se týká jak adaptérů mobilních zařízení různých výrobců, tak i zařízení stejného výrobce. Dokonce se vyskytují rozdíly i mezi dvěma shodnými zařízeními, které mohou mít jiný typ adaptéru, nebo jinou verzi operačního systému. Rozdíl mezi adaptéry způsobuje jinou naměřenou hodnotu RSSI, a to na stejném místě, ve stejný čas a při stejné orientaci dvou zařízení. Množství zařízení je přitom tak velké, že nelze vytvořit databanku s kompatibilními zařízeními. I kdyby se vytvořila, opět zde bude ovlivňující faktor, a to verze OS. Proto byla navržena metoda, při níž se už neskenují absolutní hodnoty od jednotlivých přístupových bodů, ale jejich rozdílové hodnoty, tedy $\Delta_{AP1-AP2}, \Delta_{AP2-AP3}$ apod. Rozdílová hodnota by tedy měla být pro různé adaptéry teoreticky stejná. Blíže se tomuto tématu věnuje kapitola 2.2.

Optimalizace předešlého lokalizačního systému [25] se projevila tak výrazně, že nakonec došlo k vytvoření zcela nového systému. Princip lokalizace, jímž je metoda *Localization Fingerprinting*, viz kapitola 2.1.2, se však nezměnil. Změnil se však hlavní výpočetní algoritmus, který nyní využívá fuzzy logiky, resp. fuzzy odvozovacího systému (FIS), kdežto předešlý systém využíval metodu kNN. Nově byl také navržen postup, jak generovat FIS na základě trénovacích dat, která charakterizují signálové pokrytí konkrétního prostoru v budově. Tento návrh byl posléze

i implementován do prvku *FuzzyLoc Engine*.

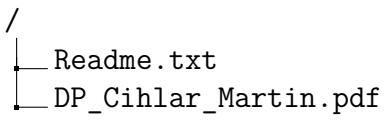
Výsledný lokalizační systém je tvořen pěti nově navrženými a vytvořenými prvky. Jeden už byl zmíněn výše, je jím *FuzzyLoc Engine*, který generuje definice (popis FIS), dle kterých se odvozuje výsledná pozice zařízení. Jedná se o Java aplikaci. Dalším je *FuzzyLoc Server*, který tvoří samotné jádro celého systému. Umožňuje komunikaci se všemi ostatními prvky, poskytuje jim úložný prostor jak v podobě sdílených složek, tak v podobě MySQL databáze. Tento server je založen na OS Ubuntu s webovým serverem Apache. V zájmu snadné implementace lokalizačního systému do produkčního prostředí je celý server virtualizovaný. To umožňuje jeho následný export do OVF souboru, který je podporován drtivou většinou virtualizačních platforem. Není tedy třeba všechny HTML, PHP, Java a JavaScript soubory migrovat na produkční server. Kromě uvedených *backend* prvků jsou to prvky *frontend*, k nimž patří *FuzzyLoc Admin*. Jedná se o mobilní aplikaci pro OS Android, která je určená správci dané sítě, jež bude využívána při lokalizaci. Hlavní účel této aplikace je tvorba trénovací množiny dat, registrace uživatelů a přístupových bodů do systému. Dalším prvkem je aplikace *FuzzyLoc Client*, která je určena pro běžného uživatele. Jejím účelem je skenovat okolní RSSI hodnoty a na základě FIS popisu odvozovat svoji polohu a tu posléze nahrávat na server. Posledním prvkem je aplikace *FuzzyLoc Supervisor*. Jedná se o webové dohledové prostředí, které zobrazuje pozice jednotlivých uživatelů. Tato webová aplikace také slouží k definici rozložení zón v lokalizovaném prostředí. Veškerá komunikace mezi prvky je šifrována SSL/TLS protokoly, které využívají certifikáty od vytvořené kořenové certifikační autority *FuzzyLoc CA*.

Vytvořený lokalizační systém byl podroben testům ve dvou různých prostředích. Tím prvním bylo spíše otevřené výukové prostředí a druhé bylo standardní patro budovy s několika kancelářemi a učebnami. Testy jsou popsány v kapitole 5.5. V tabulce 7 je porovnání úspěšností z jednotlivých testů. Z těchto výsledků lze vyvodit, že ve standardním prostředí, obecně tedy ve členitém prostředí, kde jsou místnosti standardně odděleny přepážkami a dveřmi, je lokalizační systém schopný odvozovat pozici zařízení daleko lépe, než v prostředí otevřenějším. Je to z toho důvodu, že jednotlivé místnosti jsou z pohledu signálového pokrytí unikátní, a proto je zde velká míra rozlišitelnosti jednotlivých zón, větší než v otevřenějším prostředí.

Souhrnně lze konstatovat, že práce potvrdila vhodnost zvolené cesty a použitých metod, neboť vedly k očekávaným, byť ještě zdaleka ne perfektním výsledkům. Co se týče budoucího vývoje systému, tak je zde mnoho věcí, které by se daly zlepšit a popřípadě doplnit. Za nejdůležitější krok v budoucím vývoji systému lze považovat optimalizaci samotné fuzzy logiky, resp. odvozovacího systému. Mohl by se zlepšit způsob generování FIS systému tak, aby byl zcela univerzální a ještě lépe vždy popisoval daný prostor. S tím samozřejmě souvisí i zlepšení úspěšnosti, resp. dosažení stabilní úspěšnosti v různých prostředích. Druhým krokem ke zlepšení systému by mohlo být dynamické generování trénovacích dat daného prostoru. Toho by mohlo být dosaženo například rozmístěním nízkoenergetických sond pro sběr aktuálních informací o signálovém pokrytí. Sběr těchto informací by ale mohly provádět i samotné přístupové body, které by průběžně měřily okolní RSSI data, a ta by posléze sloužila k aktualizaci FIS systému, resp. celého lokalizačního systému.

Příloha A.

Obsah přiloženého CD



Literatura

[1] A.K.M. Mahtab Hossain et al. "SSD: A Robust RF Location Fingerprint Addressing Mobile Devices' Heterogeneity". In: *Mobile Computing, IEEE Transactions on* 12.1 (led. 2013), s. 65–77. ISSN: 1536-1233. DOI: 10.1109/TMC.2011.243.

[2] Xiaoyong Chai a Qiang Yang. "Reducing the Calibration Effort for Location Estimation Using Unlabeled Samples". In: *Pervasive Computing and Communications, 2005. PerCom 2005. Third IEEE International Conference on*. Břez. 2005, s. 95–104. DOI: 10.1109/PERCOM.2005.34.

[3] V. Černý. "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm". In: *Journal of Optimization Theory and Applications* 45.1 (), s. 41–51. ISSN: 1573-2878. DOI: 10.1007/BF00940812. URL: http://dx.doi.org/10.1007/BF00940812.

[4] S. Kirkpatrick, C. D. Gelatt a M. P. Vecchi. "Optimization by Simulated Annealing". In: *Science* 220.4598 (1983), s. 671–680. ISSN: 0036-8075. DOI: 10.1126/science.220.4598.671. eprint: http://science.sciencemag.org/content/220/4598/671.full.pdf. URL: http://science.sciencemag.org/content/220/4598/671.

[5] L. William Jorgensen. "Perspective on "Equation of state calculations by fast computing machines"". In: *Theoretical Chemistry Accounts* 103.3 (), s. 225–227. ISSN: 1432-2234. DOI: 10.1007/s002149900053. URL: http://dx.doi.org/10.1007/s002149900053.

[6] C. T. Lin a C. S. G. Lee. "Neural-network-based fuzzy logic control and decision system". In: *IEEE Transactions on Computers* 40.12 (pros. 1991), s. 1320–1336. ISSN: 0018-9340. DOI: 10.1109/12.106218.

[7] J. M. Keller, M. R. Gray a J. A. Givens. "A fuzzy K-nearest neighbor algorithm". In: *IEEE Transactions on Systems, Man, and Cybernetics* SMC-15.4 (čvc 1985), s. 580–585. ISSN: 0018-9472. DOI: 10.1109/TSMC.1985.6313426.

[8] G. Fang, N. M. Kwok a Q. Ha. "Automatic Fuzzy Membership Function Tuning Using the Particle Swarm Optimization". In: 2 (pros. 2008), s. 324–328. DOI: 10.1109/PACIIA.2008.105.

[9] L. Chen a C. L. P. Chen. "Pre-shaped fuzzy c-means algorithm (PFCM) for transparent membership function generation". In: (říj. 2007), s. 789–794. DOI: 10.1109/ICSMC.2007.4413722.

[10] O. Cordón et al. *Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases*. Sv. v. 19. Advances in fuzzy systems. NJ: World Scientific, c2001.

[11] L. A. Zadeh. "Fuzzy logic and approximate reasoning". In: *Synthese* 30.3 (), s. 407–428. ISSN: 1573-0964. DOI: 10.1007/BF00485052. URL: http://dx.doi.org/10.1007/BF00485052.

[12] *Fuzzy Logic Toolbox^{TM} User's Guide*. The MathWorks, Inc. 2016. URL: http://cn.mathworks.com/help/pdf_doc/fuzzy/fuzzy.pdf.

[13] Pavel Jura. *Základy fuzzy logiky pro řízení a modelování*. Vysoké učení technické v Brně, Nakladatelství VUTIUM, Brno, 2003.

[14] Vilém Novák. *Základy fuzzy modelování, 1. vyd.* BEN-Technická literatura, Praha, 2000.

[15] T.J. Ross. *Fuzzy logic with engineering applications*. McGraw-Hill, 1995. ISBN: 9780070539174. URL: https://books.google.cz/books?id=XJxRAAAAMAAJ.

[16] Tahsin Alp Yanar a Zuhal Akyürek. "Fuzzy model tuning using simulated annealing". In: *Expert Systems with Applications* 38.7 (2011), s. 8159–8169. ISSN: 0957-4174. DOI: http://dx.doi.org/10.1016/j.eswa.2010.12.159. URL: http://www.sciencedirect.com/science/article/pii/S0957417410015228.

[17] J. Casillas, O. Cordon a F. Herrera. "COR: a methodology to improve ad hoc data-driven linguistic rule learning methods by inducing cooperation among rules". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 32.4 (srp. 2002), s. 526–537. ISSN: 1083-4419. DOI: 10.1109/TSMCB.2002.1018771.

[18] L. X. Wang a J. M. Mendel. "Generating fuzzy rules by learning from examples". In: *IEEE Transactions on Systems, Man, and Cybernetics* 22.6 (lis. 1992), s. 1414–1427. ISSN: 0018-9472. DOI: 10.1109/21.199466.

[19] Guixi Liu a Wanhai Yang. "Learning and tuning of fuzzy membership functions by simulated annealing algorithm". In: (2000), s. 367–370. DOI: 10.1109/APCCAS.2000.913511.

[20] P. Cingolani a J. Alcalá-Fdez. "jFuzzyLogic: a Java Library to Design Fuzzy Logic Controllers According to the Standard for Fuzzy Control Programming". In: (pros. 2012), s. 61–75. URL: http://jfuzzylogic.sourceforge.net/html/pdf/Cingolani_Alcala-Fdez_jFuzzyLogic_2013_IJCIS.pdf.

[21] P. Cingolani a J. Alcalá-Fdez. "jFuzzyLogic: a robust and flexible Fuzzy-Logic inference system language implementation". In: (červ. 2012), s. 1–8. ISSN: 1098-7584. DOI: 10.1109/FUZZ-IEEE.2012.6251215.

[22] "FCL Example". In: *JFuzzyLogic [online]*. (). [cit. 2016-04-06]. URL: http://jfuzzylogic.sourceforge.net/html/example_fcl.html.

[23] Greg Welch a Gary Bishop. *An Introduction to the Kalman Filter*. Chapel Hill, NC, USA: University of North Carolina at Chapel Hill, 1995.

[24] Kurečka A. *Vývoj embedded modulu pro podporu lokalizace průzkumného vozidla. Bakalářská práce.* VŠB - Technická univerzita Ostrava, 2011.

[25] Cihlář M. *Analýza pokrytí zadaného území signálem pro technologii RTLS v pásmu 2,45 GHz. Bakalářská práce.* ČVUT v Praze, 2014.

[26] "AUTOMATED GUIDED VEHICLE". In: *OPPENT - ADVANCED MOTION TECHNOLOGY [online].* (). [cit. 2016-04-26]. URL: http://www.oppent.com/en/agv/.

[27] A. Malik. *RTLS for Dummies, 1st ed.* ISBN 04-703-9868-X. Indianapolis, Wiley Pub., Inc., 2009.

[28] G. Ding et al. "Overview of received signal strength based fingerprinting localization in indoor wireless LAN environments". In: (říj. 2013), s. 160–164. DOI: 10.1109/MAPE.2013.6689973.

[29] Jose M. Alonso et al. "Enhanced WiFi localization system based on Soft Computing techniques to deal with small-scale variations in wireless sensors". In: *Applied Soft Computing* 11.8 (2011), s. 4677–4691. ISSN: 1568-4946. DOI: http://dx.doi.org/10.1016/j.asoc.2011.07.015. URL: http://www.sciencedirect.com/science/article/pii/S15684946110%2002766.

[30] G. Lui et al. "Differences in RSSI readings made by different Wi-Fi chipsets: A limitation of WLAN localization". In: (červ. 2011), s. 53–57. ISSN: 2325-0747. DOI: 10.1109/ICL-GNSS.2011.5955283.

[31] A.K.M. Mahtab Hossain a Wee-Seng Soh. "A survey of calibration-free indoor positioning systems". In: *Computer Communications* 66 (2015), s. 1–13. ISSN: 0140-3664. DOI: http://dx.doi.org/10.1016/j.comcom.2015.03.001. URL: http://www.sciencedirect.com/science/article/pii/S0140366415001115.

[32] Ngewi Fet, Marcus Handte a Pedro José Marrón. "A Model for WLAN Signal Attenuation of the Human Body". In: UbiComp '13 (2013), s. 499–508. DOI: 10.1145/2493432.2493459. URL: http://doi.acm.org/10.1145/2493432.2493459.

[33] Christian Esposito a Massimo Ficco. "Deployment of RSS-Based Indoor Positioning Systems". In: *International Journal of Wireless Information Networks* 18.4 (2011), s. 224–242. ISSN: 1572-8129. DOI: 10.1007/s10776-011-0131-7. URL: http://dx.doi.org/10.1007/s10776-011-0131-7.

[34] C. Luo, H. Hong a M. C. Chan. "PiLoc: A self-calibrating participatory indoor localization system". In: (dub. 2014), s. 143–153. DOI: 10.1109/IPSN.2014.6846748.

[35] V. Kumar, A. Kumar a S. Soni. "A Combined Mamdani-Sugeno Fuzzy Approach for Localization in Wireless Sensor Networks". In: ICWET '11 (2011), s. 798–803. DOI: 10.1145/1980022.1980196. URL: http://doi.acm.org/10.1145/1980022.1980196.

[36] Y. Chapre et al. "Received signal strength indicator and its analysis in a typical WLAN system (short paper)". In: (říj. 2013), s. 304–307. ISSN: 0742-1303. DOI: 10.1109/LCN.2013.6761255.

[37] M. Meenalochani a S. Sudha. "Fuzzy Based Estimation of Received Signal Strength in a Wireless Sensor Network". In: WCI '15 (2015), s. 624–628. DOI: 10.1145/2791405.2791543. URL: http://doi.acm.org/10.1145/2791405.2791543.

[38] José M. Alonso et al. *Computer Aided Systems Theory - EUROCAST 2009: 12th International Conference, Las Palmas de Gran Canaria, Spain, February 15-20, 2009, Revised Selected Papers*. Ed. Roberto Moreno-Díaz, Franz Pichler a Alexis Quesada-Arencibia. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. Kap. WiFi Localization System Using Fuzzy Rule-Based Classification, s. 383–390. ISBN: 9783642047725. DOI: 10.1007/978-3-642-04772-5_50. URL: http://dx.doi.org/10.1007/978-3-642-04772-5_50.

[39] A. S. Salazar, L. Aguilar a G. Licea. "Estimating Indoor Zone-Level Location Using Wi-Fi RSSI Fingerprinting Based on Fuzzy Inference System". In: (lis. 2013), s. 178–184. DOI: 10.1109/ICMEAE.2013.27.

[40] N. Hernández et al. "WiFi localization system based on Fuzzy Logic to deal with signal variations". In: (zář. 2009), s. 1–6. ISSN: 1946-0740. DOI: 10.1109/ETFA.2009.5347015.