

OSTRAVSKÁ UNIVERZITA V OSTRAVĚ
PŘÍRODOVĚDECKÁ FAKULTA
KATEDRA INFORMATIKY A POČÍTAČŮ

Řízení autonomního robota s využitím neuronových sítí

DIPLOMOVÁ PRÁCE

Autor práce: Bc. Adam Bartoň
Vedoucí práce: doc. RNDr. PaedDr. Eva Volná, Ph.D.

2016

UNIVERSITY OF OSTRAVA
FACULTY OF SIENCE
DEPARTMENT OF INFORMATICS AND COMPUTERS

Control of autonomous robot using neural networks

THESIS

Author: Bc. Adam Bartoň
Supervisor: doc. RNDr. PaedDr. Eva Volná, Ph.D.

2016

ČESTNÉ PROHLÁŠENÍ

Já, níže podepsaný/á student/ka, tímto čestně prohlašuji, že text mnou odevzdané závěrečné práce v písemné podobě i na CD nosiči je totožný s textem závěrečné práce vloženým v databázi DIPL2.

Ostrava dne

.....
podpis studenta/ky

(Zadání vysokoškolské kvalifikační práce)

ABSTRAKT

Cílem práce je návrh metody řídicí autonomního robota s využitím umělé neuronové sítě. Teoretická část práce popisuje problematiku řízení autonomního robota z pohledu navigace a současné mobilní roboty řízené neuronovými sítěmi. Výstupem praktické části je robot sestavený prostřednictvím Lego Mindstorm EV3 řešící problém vyhnutí se překážkám v prostoru a aplikace pro návrh řídicí neuronové sítě, generování a filtraci trénovací množiny, umožňující nastavení experimentu a jeho spuštění. V experimentální části je robot testován v několika prostředích.

Klíčová slova:

řízení robota, umělá neuronová síť, filtrace trénovací množiny, adaptivní rezonanční teorém, leJOS, Lego Mindstorm EV3

ABSTRACT

The aim of this thesis is to design method of controlling an autonomous robot using artificial neural networks. The theoretical part describes control issues from the perspective of autonomous robot navigation and the current mobile robots controlled by neural networks. The outcome of the practical part is assembled Lego Mindstorms EV3 robot solving the problem of avoiding obstacles in space and the application for the design of the control neural network, the generation and filtration of the training set and allowing experiment settings and launch. In the experimental part the robot is tested in several environments.

Keywords:

robot control, artificial neural network, training set filtering, adaptive resonance theorem, leJOS, Lego Mindstorm EV3

Rád bych poděkoval doc. RNDr. PaedDr. Evě Volné, Ph.D. za cenné rady, věcné připomínky a vstřícnost při konzultacích a vypracování práce.

Prohlašuji, že předložená práce je mým původním autorským dílem, které jsem vypracoval/a samostatně. Veškerou literaturu a další zdroje, z nichž jsem při zpracování čerpal/a, v práci řádně cituji a jsou uvedeny v seznamu použité literatury.

V Ostravě dne

.....

(podpis)

OBSAH

ÚVOD.....	9
CÍL PRÁCE.....	10
1 PROBLEMATIKA ŘÍZENÍ AUTONOMNÍHO ROBOTA Z POHLEDU	
NAVIGACE.....	11
1.1 Autonomie.....	12
1.2 Reprezentace informací.....	12
1.3 Řídící architektura.....	14
1.3.1 Centralizované řízení	14
1.3.2 Reaktivní řízení.....	15
1.3.3 Hybridní řízení.....	17
1.4 Shrnutí.....	18
2 ŘÍZENÍ S VYUŽITÍM NEURONOVÝCH SÍTÍ.....	20
2.1 Robot G. N. Tripathi a V. Rihani	21
2.2 GYROBO.....	22
2.3 Trilobot.....	22
2.4 Inteligentní autonomní vozidlo	23
2.5 Roverbot.....	24
2.6 EV3 OpenWorm.....	25
2.7 Shrnutí řízení robotů neuronovými sítěmi	26
3 MODEL AUTONOMNÍHO ROBOTA.....	28
3.1 Řídící neuronová síť.....	30
3.2 Generování trénovací množiny	32
3.3 Filtrace trénovací množiny pomocí adaptivní rezonanční teorie	35
3.3.1 Algoritmus ART1 pro filtraci trénovací množiny	36
3.3.2 Vliv parametru vigilance	39
4 APLIKACE PRO EXPERIMENTÁLNÍ STUDII.....	41
4.1 Připojené zařízení.....	42
4.2 Nastavení experimentu.....	42
4.3 Spuštění experimentu.....	43
4.4 Generování a filtrace TM, RTM, ERTM	44
4.5 Vytvoření neuronové sítě a její adaptace	45

4.6	Postup vytvoření experimentu.....	45
5	EXPERIMENTÁLNÍ STUDIE	47
5.1	Trénovací množina a řídicí neuronová síť	48
5.2	Definice experimentů a kritérií	50
5.2.1	Experiment č. 1	51
5.2.1	Experiment č. 2	52
5.2.1	Experiment č. 3	53
5.2.1	Experiment č. 4	54
5.2.1	Experiment č. 5	55
5.3	Shrnutí experimentů	57
	ZÁVĚR	59
	RESUMÉ	61
	SUMMARY	62
	SEZNAM POUŽITÉ LITERATURY.....	63
	SEZNAM POUŽITÝCH SYMBOLŮ.....	65
	SEZNAM OBRÁZKŮ	66
	SEZNAM TABULEK.....	67
	SEZNAM PŘÍLOH.....	68

ÚVOD

V dnešní době se člověk setkává s celou řadou robotických zařízení. Stále je však předmětem bádání jejich, a ne jen fyzických zařízení, programová část, tedy umělá inteligence. Zajímavým tématem je lidský mozek jako předloha pro takovýto inteligentní systém.

Z inspirace strukturou lidského mozku vychází umělá neuronová síť. Stejně jako mozek i umělá neuronová síť obsahuje množství jednoduchých výpočetních jednotek, neuronů, vzájemně provázaných spojením umožňujícím přenášení signálu z neuronu do neuronů. Tyto lokální interakce mezi jednotlivými neurony pak mají za následek realizaci globální funkce. Umělé neuronové sítě se dnes využívají k řešení rozličných problémů s dobrými výsledky ve srovnání s jinými rozhodovacími algoritmy.

Typů umělých neuronových sítí je celá řada, v této práci se zaměříme na sítě dopředné. Prostřednictvím nich pak budeme definovat chování robotického zařízení, jež by mělo být schopno na základě definovaného chování odvodit i vhodnou reakci na zcela neznámé situace. Práce volně navazuje na předešlou bakalářskou práci na téma Emergence chování autonomního robota. Pro sestavení fyzického robotického zařízení využijeme Lego Mindstorms EV3.

CÍL PRÁCE

Cílem této práce je navrhnout metodu vycházející z umělých neuronových sítí, která bude řídit robota v reálném čase.

Dílčí body práce:

- Analýza problematiky řízení autonomního robota z pohledu navigace.
- Rešerše problematiky řízení mobilního autonomního robota s využitím neuronových sítí.
- Sestavit robota vhodného pro pohyb v prostoru řešící problém vyhnutí se překážkám s využitím Lego Mindstorms EV3.
- Vytvořit aplikaci umožňující:
 - Návrh topologie, parametrů adaptace a trénovací množiny neuronové sítě pro adaptaci chování robota v daném prostředí.
 - Variabilní nastavení parametrů experimentu a spuštění experimentu.
- Experimentální studie chování robota, stanovení kritérií pro zhodnocení výsledků a vyhodnocení experimentů.

1 PROBLEMATIKA ŘÍZENÍ AUTONOMNÍHO ROBOTA

Z POHLEDU NAVIGACE

Z hlediska problematiky autonomie lze robotické systémy rozdělit na řízené, ovládané, autonomní a inteligentní. [1] Řízené robotické systémy jsou ovládány přímo lidským operátorem, který provádí úkony robotu, a nejsou tedy schopny samy o sobě činit akce. Ovládaným robotickým systémům jsou předem definovány akce, jež vykonávají v daném sledu. Naopak autonomním systémům jsou definovány postupy, na jejichž základě dosahují úkolů a stanovených cílů. Za inteligentní systémy považujeme ty, které jsou schopny stanovovat si vlastní cíle. Tato práce se zaměřuje na autonomní roboty a takový to systém lze definovat:

„Robot je autonomní systém, existující ve fyzickém světě, který může detekovat prostředí a konat akce k dosažení cíle.“ [2]

Slovo autonomní tedy vyjadřuje, že takovýto systém může konat své vlastní rozhodnutí bez zásahu nebo kontroly člověka v reálném světě, nikoliv ve virtuálním (v tomto případě se jedná o simulaci). Vnímání okolního prostředí systému zajišťují různorodé snímače poskytující data. Aby takový to systém mohl konat rozhodnutí je zapotřebí existence kognitivního prvku, který zabezpečí „myšlení“ robota.

Komponenty robota

Podle definice robota pak můžeme odvodit, které komponenty musí robot mít k dispozici, aby nabyl své podstaty. Především to jsou tedy:

- **fyzická konstrukce** existující v reálném universu, tj. tělo robota, díky němuž se může pohybovat, manipulovat s předměty apd.,
- **snímače**, zařízení pro vnímání okolního prostředí poskytující informace o robotovi samotném a prostředí okolo něj,
- **efektory**, jež vykonávají akce v prostředí, umožňují pohyb robota, interakci s okolím apd.,
- **řídící systém** definující systém rozhodování, resp. autonomii systému. [1]

1.1 Autonomie

Řídicí (kontrolní) systém, tedy poskytuje nástroj, který utváří autonomii robotického systému, s využitím dostupných informací, dat k plánování akcí a vykonávání akcí. Jedná se o „mozek“ robota, který nemusí být vždy jen jeden (pro různé snímače mohou existovat různé kontrolní systémy).

„Autonomie je schopnost konat vlastní rozhodnutí a akce. V robotice to znamená, že robot vytváří vlastní rozhodnutí sám a stejně tak je vykonává sám bez zásahu lidského operátora.“ [2]

Řídicí architektura poskytuje principy organizace kontrolního systému a umožňuje definovat chování robota. Využívá algoritmy k řešení problémů, cílů a to krok za krokem. Nástrojem k realizaci řídicí architektury jsou programovací jazyky, zatímco sama architektura popisuje jen kostru.

Modularita

Modularita říká jakým způsobem je kontrolní systém robota rozdělen na komponenty, moduly a jak tyto komponenty navzájem interagují a komunikují za cílem vytváření výsledného chování. Architektura se tak může skládat z několika modulů. Přístupy k modularitě při řešení architektury jsou popsány v kapitole Řídicí architektura.

1.2 Reprezentace informací

Reprezentace popisuje informace, jejich podobu, způsob jakým jsou uloženy, formu. Jakým způsobem jsou tyto informace reprezentovány a které informace jsou ukládány, řídí kontrolní systém.

Navigační mapa

Typickou reprezentací prostředí, ve kterém se robot pohybuje, je navigační mapa, která představuje část reálného světa. Existuje spousta možností a způsobů uchování informací v podobě mapy např.:

- **odometrická mapa** – robot uchovává přesné informace o cestě, např. v podobě otočení ve stupních a délce pohybu v cm apd.; jedná se o velmi exaktní mapu a je vhodná pouze pokud se prostředí nemění,
- **orientace podle cesty** – robot si uchovává sekvence pohybu v prostředí, např. cestu od jednoho bodu do druhého pomocí popisu pohybu a odbočení; závisí taky na neměnnosti prostředí, ale nepotřebuje příliš mnoho detailních informací,
- **orientační mapa** – mapa je tvořena sekvencí pohybu v prostředí s uchováním dalších informací, např. odbočení jsou podmíněné popisem okolí, mapa tedy říká co dělat v každé situaci a nezáleží, na jaké odbočce se robot aktuálně nachází (jedná se o typologickou mapu spojující jednotlivé cesty),
- **metrická mapa** – mapa je zakreslována na základě získaných informací o prostředí; např. vzdálenost od zdi k míči a velikost míče apd., ačkoliv je ukládáno větší množství informací je možné zachytit i změny v prostředí a dosáhnout lepšího plánování budoucích akcí. [2]

Artefakty

Krom samotné mapy představující prostředí, jeho strukturu, může robot reprezentovat sebe sama, jeho kapacity, možnosti, plány, cíle. Dále může reprezentovat objekty v prostředí a akce, které vykonává v prostředí; stejně jako specifické úkoly. [2] Je ovšem obtížné uchovávat a získávat takovýto model světa neboť aktualizace modelu vyžaduje množství času a prostředků.

Různé architektury řeší reprezentaci modelu světa různým způsobem vzhledem k požadavkům na zpracování, údržbu, konstrukci a reprezentaci modelu. Uchopení informací universa může být klíčové k úspěšnému řešení globálního cíle robota. To, jakým způsobem bude robot reprezentovat okolní svět a kolik informací bude uchovávat (zdali vůbec nějaké) závisí na každé konkrétní úloze a problému, který robot řeší, viz Řídící architektura, a také na množství samotných cílů, senzorů a informací vyskytujících se v prostředí.

1.3 Řídící architektura

Autonomní robot, jako pohyblivá „myslící“ entita, musí být schopen zpracovat data snímačů, definovat stav okolního světa, plánovat budoucí akce, vykonávat akce a to s omezenými zdroji a v reálném čase. Tyto úkony popisuje řídící architektura definující elementy nutné k dosažení požadovaného chování a to zejména v otázkách:

- **programovatelnosti** – robot by neměl být uzpůsoben pouze na jeden konkrétní úkol a měl by být schopen dosáhnout více úloh, skupiny úkolů, definovaných jedním globálním cílem,
- **autonomii a adaptability** – robot by měl být schopen uzpůsobit potřebné úlohy, úkony a jeho chování pro splnění definovaného globálního cíle,
- **reaktivity** – měl by být schopen se rozhodovat ve vhodném časovém rozpětí a reagovat na náhle události vyžadující okamžitou nebo téměř okamžitou reakci,
- **robustnosti** – architektura by měla být schopná zvládnout nenadálé problémy ve funkcionalitě, prioritách úkolů, chování, aktivit, řešit konflikty apd.,
- **rozšiřitelnosti** – implementace nových funkcností by neměla negativně ovlivnit chod architektury, měla by být snadno rozšiřitelná a pochopitelná. [3]

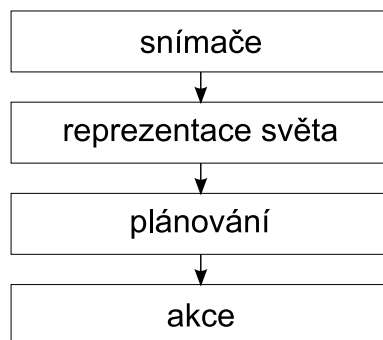
Některé architektury vycházejí z imitace chování živočichů, rostlin a biologických procesů; jsou tedy orientovány na systém typu stimul-reakce. Další architektury jsou naopak více orientovány na umělou inteligenci, reprezentaci symbolů, vnitřního světa a uvažování. Jiné se snaží kombinovat výhody z obou výše zmíněných přístupů.

1.3.1 Centralizované řízení

Centralizované, též hierarchické, řízení robota je schéma založené na generování modelu světa sestaveném pomocí dat ze vstupů snímačů či obsluhy (uživatele). Schéma obsahuje několik základních modulů: snímače reprezentují schopnost robota vnímat své okolí (vstupní data), na základě dat ze snímačů je sestaven model světa představující prostředí (okolní svět), modul plánování představuje proces nalezení optimální cesty k dosažení požadovaného cíle. Na základě plánu následují akce, které robot vykonává dle plánu. Po

vykonání všech vrstev následuje zastavení robota a vyhodnocení nové situace, resp. zahájení procesu od počátku. [4]

Centralizované schéma je jedním z nejjednodušších a nejstarších modelů architektury řízení robota. Architektura využívá vysokoúrovňový přístup pro plánování budoucích akcí, tzn., že množina nízkoúrovňových příkazů, jež mohou sledovat více cílů oproštěných od velké komplexity prostředí, sestává popis vysokoúrovňových požadavků či omezení. Je tedy nutný přesný model popisující samotné prostředí pro plánování k dosažení globálního cíle. Vysokoúrovňový přístup navíc způsobuje prodlevu (způsobenou nutnými výpočty a požadavky na paměť) a je tedy nevhodný pro měnící se prostředí. [5] [6]



Obrázek 1: Centralizované řízení

1.3.2 Reaktivní řízení

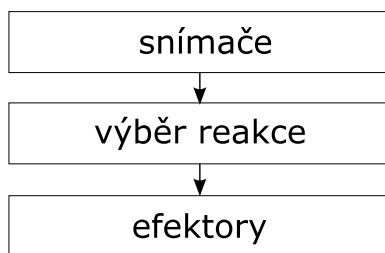
Naopak reaktivní řízení je navrženo pro řešení problémů v neznámém prostředí a je založeno na vytváření akcí na základě dat ze snímačů, tzn., že modul plánování není přítomen. V nízkoúrovňovém přístupu nejsou vysokoúrovňové požadavky zachyceny prostřednictvím nízkoúrovňových akcí, není tedy nutné vytvářet model prostředí a robot využívá namísto toho model reakcí, což má za následek velmi dobrou odezvu v měnícím se nebo neprozkoumaném prostředí. [4]

Na obrázku 2 je znázorněna reaktivní architektura, kde pomocí modulu snímače jsou získána data a následně vybrána vhodná reakce z množiny předem definovaných reakcí (výstupů) na příchozí vstup ze snímačů a robot vykoná akci efektorů na základě vybrané reakce.

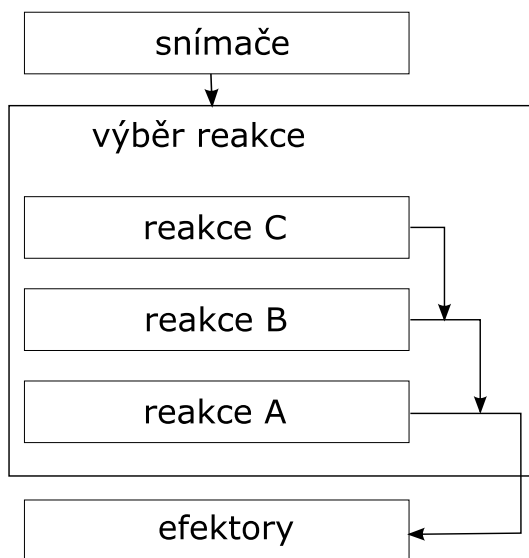
Rozdělení množiny reakcí do menších částí může zvýšit výkonnost celku. Obrázek 3 zobrazuje právě takové to rozdělení, kdy je kontrolní systém rozdělen do několika vrstev reakcí tvořící množinu chování, přičemž dosažení jedné vede k aktivaci druhé (zvyšuje se citlivost) a zároveň je dále distribuována jistá část informace ze snímačů do okolí vybrané reakce. Nejvyšší modul (reakce A) zároveň vytváří celkový výstup pro efektory. Jedná se tedy o přístup s důrazem na výběr reakce dle určitých parametrů či priority. [7]

Na obrázku 4 je architektura, která vychází z biologických poznatků a popisuje „motor“ řídící kontrolu mnoha rozdílných aktivit. Každý modul reakce z množiny reakcí může vytvářet výstup v podobě vektoru, tedy potenciál oblasti. Tyto jednotlivé výstupy jsou kombinovány a následně jejich součet tvoří celkový výstup, tj. superpozice všech oblastí. Poskytuje tedy možnost zachycení více modulů reakcí současně v celkovém výstupu. [8]

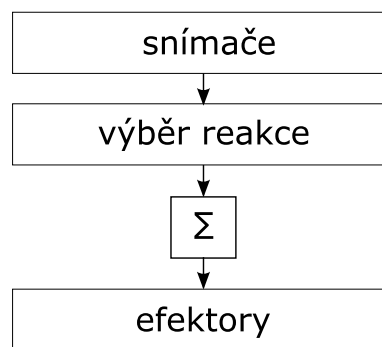
Reaktivní přístup tedy přináší zejména: rychlou reakci v neprozkoumaném prostředí, absenci nutnosti modelování prostředí, kratší dobu odezvy, větší stabilitu (pokud selže jeden modul chování z množiny chování, zbylé moduly fungují nadále). Ovšem moduly chování jsou nízkoúrovňovým popisem a nezachycují tedy vysokoúrovňové požadavky. Tudíž nelze některé cíle naplnit popisem chování robota v daných situacích či chováním jako takovým. Dále pak chování robota jako celek je daleko těžší předvídat, uvažíme-li interakci mezi prostředím a robotem skrze moduly chování, což může zároveň emergovat chování [9], které nebylo definováno žádným předpisem z množiny popisující chování a vzniká tak nenadefinované chování existující mimo množinu chování.



Obrázek 2: Architektura A (behavior-based overall arch.)



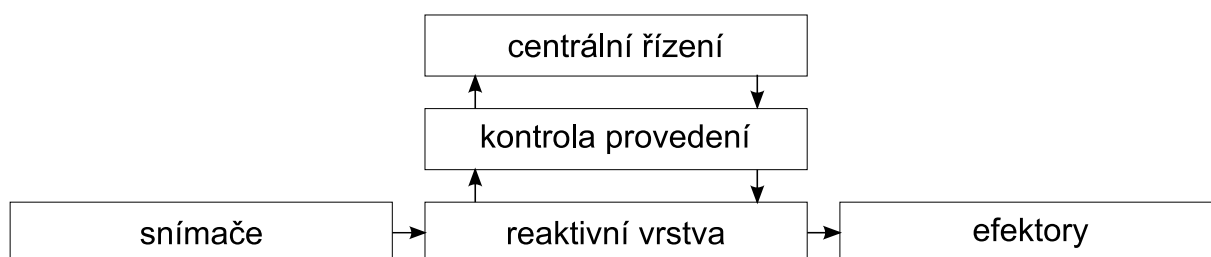
Obrázek 3: Architektura B (subsumption arch.)



Obrázek 4: Architektura C (motor schema arch.)

1.3.3 Hybridní řízení

Řešením problémů, jež přináší centralizované a reaktivní řízení robota je hybridní řízení kombinující prvky obou přístupů. Základní hybridní architektura by měla obsahovat vrstvu: centrálního řízení řešící vysokoúrovňové požadavky - plánování, vytvoření mapy prostředí, zpracování dat snímačů, vrstvu kontroly provedení spravující komunikaci vyšší a spodní vrstvy a reaktivní vrstvu vytvářející akce efektorů podle požadavků vyšších úrovní. U hybridní architektury existuje mnoho přístupů realizace. Obrázek 5 představuje jednu z typických základních architektur. [4]



Obrázek 5: Hybridní architektura

1.4 Shrnutí

Důležité při hledání vhodné řídicí architektury pro autonomního robota je nutné hledat kompromisy a uvažovat skutečnosti, že:

- Plánování vyžaduje výpočetní výkon.
- Reakce musí být rychlá.
- K plánování jsou důležité informace.
- Plánování umožňuje vyhnout se špatným rozhodnutím.
- Prostředí se mění a výsledek plánování se stává nepřesným v časovém kontinuu.

V centralizovaném řízení má robot k dispozici informace ze snímačů, bázi uložených znalostí a na základě těchto informací vytváří plán akcí. Prohledává tak potencionálně vhodné plány a snaží se predikovat. To je také důvodem proč tento přístup „strategického myšlení“ vyžaduje množství paměti a času. Není tedy praktické tento typ robota konfrontovat s neznámým či dynamicky se měnícím prostředím vyžadujícím okamžité reakce a akce, jejichž nepřítomnost může být fatální pro samotného robota.

Reaktivní přístup (reaktivní ovládání) naopak staví na rychlosti odezvy samotného systému spojením vstupů s výstupy tak, aby robot mohl velmi rychle reagovat na změny ve svém okolí. Nevýhodou je ovšem nepřítomnost vnitřní paměti, reprezentace okolního světa a úplná absence nebo jen částečná schopnost adaptace novým situacím. Robot se nedívá do minulosti ani do budoucnosti.

Jedním z reaktivních přístupů kontroly robota je Behavior-Based, který se snaží modelovat složité chování a myšlení. Architektura obsahuje vrstvy, které se od sebe minimálně liší a popisují modely chování (výstup) na základě informací ze snímačů a navíc mohou poměrně rychle zasílat informace sobě navzájem. Pokud tedy systém, robot potřebuje plánovat dopředu, činí tak v systému chování, který zasílá informace jednotlivým modelům chování. Robot tedy myslí a jedná ve stejnou chvíli a myšlení je distribuováno přes několik modulů chování současně.

Hybridní přístup se snaží spojit to nejlepší z centralizovaného a reaktivního řízení do jednoho celku. Jedna část se tedy zabývá plány a druhá okamžitými odpověďmi na vzniklou situaci. Problémovou doménou je tedy komunikace těchto dvou částí „mozku“ a řešení jejich konfliktů, očeš se stará třetí část (vrstva) spravující vzájemnou komunikaci a řešící vzniklé konflikty mezi jednotlivými částmi. Tyto systémy lze rozdělit na deterministické – založené na algoritmech, stavech, hledání cesty, apd. a systémy s umělou inteligencí – použití neuronových sítí, pravděpodobnostní plánování, apd.

Pro každý problém se ovšem hodí jiný přístup. Následuje tabulka 1 popisující několik vybraných specifikací a srovnání reaktivního a centralizovaného přístupu. Do tabulky není zahrnut hybridní přístup, vzhledem k tomu, že může dosahovat diametrálně odlišných evaluací s každou jeho rozdílnou implementací či přístupem k realizaci.

Tabulka 1: Porovnání vlastností architektury

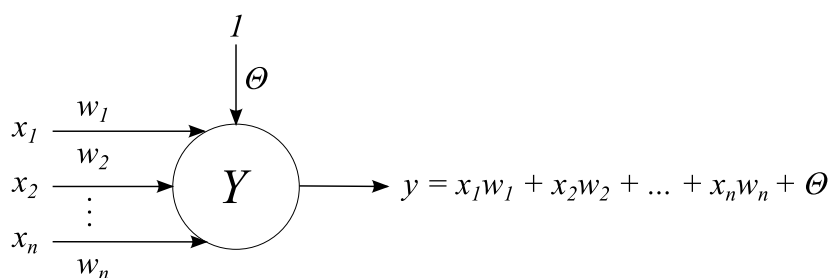
Vlastnosti architektury	Reaktivní přístup	Centralizovaný přístup
Rychlost odezvy	Velmi dobrá	Velmi špatná
Jednoduchost aplikace	Velmi dobrá	Velmi špatná
Plánování	Velmi špatné	Velmi dobré
Zaměření na cíl	Špatné	Velmi dobré
Schopnost učení	Špatné	Velmi dobré
Přizpůsobivost prostředí	Velmi dobré	Velmi špatné
Robustnost	Špatná	Dobrá
Rozšiřitelnost	Velmi dobrá	Velmi špatná

2 ŘÍZENÍ S VYUŽITÍM NEURONOVÝCH SÍTÍ

Umělá neuronová síť, dále jen UNS, je matematickým modelem vycházející ze zpracování informací nervovým systémem. Základním prvkem biologické předlohy UNS je buňka sloužící ke sběru, zpracování a přenosu signálu – neuron. Tato buňka se skládá z těla neuronu, do něhož vstupují informace prostřednictvím dendritů, a vystupují pomocí axonu. Výstupy jsou pak zároveň vstupy do dalších neuronů v síti. Přenos signálu je tvořen elektrickými signály, jež aktivují neuron až po dosažení jistého potenciálu. Na základě toho to popisu byl definován formální neuron. [10]

Formální neuron

Obrázek 6 znázorňuje formální neuron s n vstupy, každému vstupu x_i je přidružena váhová hodnota w_i . Díky této synaptické váhové hodnotě můžeme vyjadřovat inhibici či excitaci jednotlivých vstupů, tj. zvětšovat, zmenšovat význam vstupního signálu. Při dosažení prahové hodnoty Θ fiktivního vstupu určuje výstupní signál přenosová funkce. Tato funkce také převádí hodnoty do definovaného oboru hodnot. Nejčastěji se užívá sigmodiální přenosová funkce, která vymezuje výstupu hodnoty z intervalu $[0,1]$. Zapojením jednotlivých neuronů do sítě pak získáváme struktury tvořící provázanou síť. [10]



Obrázek 6: Formální neuron

UNS se ovšem podobně jako biologická neuronová síť, dále jen NS, pro dosažení požadované výstupní funkce musí naučit tuto funkci realizovat. Toho docílíme právě úpravou jednotlivých synapsí, tzn. váhových hodnot mezi neurony. Způsoby jak docílit úpravy vah lze rozdělit na dva typy: učení s učitelem a bez učitele. U učení s učitelem síti předkládáme vzory z trénovací množiny a snažíme se váhové hodnoty upravit tak, aby bylo minimalizováno jisté kritérium, nejčastěji rozdíl mezi reálným výstupem UNS a předlohou z trénovací množiny. Naopak učení bez učitele, kdy není znám vhodný výstup, si UNS

sama vzory třídí. Může pak na daný vstup přizpůsobit topologii nebo reagovat na typického zástupce dané třídy. [11]

Cílem práce není popsat neuronové sítě, a proto se zaměříme na některá vybraná řešení pro mobilní roboty.

2.1 Robot G. N. Tripathi a V. Rihani

V roce 2012 byla publikovaná práce G. N. Tripathi a V. Rihani na téma plánování pohybu autonomního mobilního robota s využitím neuronových sítí. [12] Prezentovaný robot se skládal z motoru pro pohyb kupředu a druhým pro rotaci (úpravu úhlu) robota. Měření vzdáleností bylo realizováno pomocí snímače GP2Y0A21YK0F, obsahující PSD (position sensitive detector) a IRED (infračervenou diodu). Pro vyhodnocení pohybu robota byla využita dopředná neuronová síť. Robot byl sestaven s cílem pracovat v reálném čase.

Infračervený snímač detekoval prostředí v rozmezí $+90^\circ$ až -90° , samotné snímání pak bylo prováděno pro úhly: -90° , -45° , 0° , 45° a 90° . Realizovaná neuronová síť tedy obsahovala 5 vstupních neuronů (práce však neuvádí přesnou topologii neuronové sítě). Přítomnost překážky byla reprezentovaná hodnotou 1 a nepřítomnost hodnotou 0. Trénovací množina se skládala z 5 vstupů a jednoho výstupu určujícího akci: vpravo, vlevo, kupředu, stop. Algoritmus učení byl zvolen Back-Propagation, dále jen BP, s přenosovou funkcí hyperbolické tangenty a síť trénována prostřednictvím programu Easy NN-plus.

Autonomní robot byl testován v prostředí bludiště s cílem vyhnout se překážkám. Neuronová síť byla schopná vyhodnocovat správně okolí robota a činit rozhodnutí o pohybu dle trénovací množiny.

Shrnutí

Byla použita vlastní konstrukce robota, práce však nezmiňuje parametry neuronové sítě. Zvolená trénovací množina obsahovala 14 vzorů sestavených expertně. Práce dále zmiňuje, že použití neuronové sítě zlepšilo výkon robota.

2.2 GYROBO

P. K. Kim a S. Jung v roce 2012 publikovali článek o jednokolovém diskovitém robotu využívajícím neuronovou síť. [13] Dva aktuátory jsou použity pro vyvažování a třetí pro pohyb kola vpřed. Neuronová síť jako pomocný regulátor byla použita z důvodu, že GYROBO je vysoce nelineární systém a neuronová síť nevyžaduje dynamický model soustavy (síť dokáže pracovat s nelineární nejistotou v systému). Ve skryté vrstvě neuronů byla použita Gaussova funkce a jako trénovací algoritmus BP. Samotná síť je použita pro generování kompenzačního výstupu, tj. minimalizaci chyby, lineárních regulátorů.

Pro balancování byla použita síť s parametry: parametr učení = 0.0005, 3 vstupními, 4 skrytými a 2 výstupními neurony. Úhel natočení se zapojením UNS zůstává téměř stejný a GYROBO byl schopen balancovat svépomocí (při použití pouze PD regulátoru tohoto stavu nebyl schopen). Stejně tak při pohybu vpřed neuronová síť redukovala významně oscilace oproti PD regulátoru.

Shrnutí

V této práci byla neuronová síť využita pro vyvažování pohybu jednokolového robota. Experimenty s PD regulátorem a neuronovou sítí ukázaly, že schopnost generalizace neuronové sítě může být silným nástrojem. V tomto případě však robot nebyl kontrolován samotnou neuronovou sítí.

2.3 Trilobot

Koncept navigačního systému pro mobilního robota se zabývala práce B. Markoski a spol. s využitím samo-učící neuronové sítě pro plánování pohybu robota. [14] Robot byl vybaven dvěma nezávislými koly a ultrazvukovým snímačem krátkého dosahu, 4 světelnými snímači, snímačem teploty, 8 dotykovými snímači, pasivním infračerveným pohybovým detektorem a digitálním kompasem. Cílem robota bylo vyhnout se překážkám. Kontrolován byl bezdrátově pomocí algoritmu v počítači.

Samotná neuronová síť byla tvořena dvourozměrným prostorem neuronů spojených se svými sousedy a její funkce byla založena na biologické předloze pohybu iontů

v membránách neuronů během excitační a inhibiční fáze. V práci je využit zjednodušený vzorec tohoto procesu:

$$\frac{dx_i}{dt} = -Ax_i + I_i + \sum_{j=1}^k w_{ij} [x_j]^+, \quad (1)$$

kde x_i je aktivace i-tého neuronu; I_i představuje vnější excitaci i-tého neuronu podle: E pro cíl, $-E$ pro překážku a 0 pro ostatní případy, kdy E je konstanta s velkou hodnotou; A je nezáporná konstanta pasivního poklesu neurální aktivity; w_{ij} synapse i-tého a j-tého neuronu, definována jako monotónně klesající funkce euklidovské vzdálenosti pro pozici neuronu; $[f]^+$ je excitační funkce zajišťující rozšíření signálu pouze pro kladnou aktivitu vůči ostatním neuronům. [14]

Aktivita sítě se formuje do vrcholu (definovaný cíl), tj. pozitivní oblast, a nižší záporné oblasti představují překážky. Následně se vhodná cesta generuje z aktuální polohy nalezením sousedů s nejvyšší aktivitou. Tímto způsobem je cesta generována v reálném čase bez nutnosti znalosti okolí a optimalizace cílové funkce.

Shrnutí

Navržený robot snímá své okolí pomocí kolekce senzorů, z nichž byl schopen reprezentovat své okolí ve dvou-dimenzionální mapě, která byla zároveň vstupem samo-učící neuronové sítě. Robot potřebuje ke svému pohybu definovaný cíl a sám pak sestavuje optimální cestu v reálném čase s uspokojivým výsledkem pohybu s cílem vyhnout se překážkám v prostředí.

2.4 Inteligentní autonomní vozidlo

Článek publikovaný v roce 2014 od Umar Farooq a kolektivu se zabývá řešením návrhu inteligentního autonomního vozidla schopného navigace v neznámém prostředí bez střetu s překážkami. [15] Robot se skládal ze čtyř kol, dvou předních a jednoho zadního ultrazvukového snímače, digitálního kompasu, GSM modemu a GPS přijímače.

Navigace je realizována prostřednictvím dvou kontrolérů (neuronových sítí) z nichž jedna zabezpečuje vyhnout se překážkám a druhá dosažení cíle. Oba kontroléry používají dvou

vrstvou dopřednou neuronovou sít' s algoritmem učení BP. Sít' určena pro vyhnutí se překážkám je tvořena 3 vstupními neurony (levý, pravý a zadní ultrazvukový snímač), 5 neuronů ve skryté vrstvě, 2 neurony ve výstupní vrstvě a sít' řešící dosažení cíle 2 vstupními neurony (vzdálenost mezi aktuální pozicí a cílem, rozdíl mezi orientací robota a cíle), 4 skrytými neurony a 2 výstupními. Při volbě topologie byla zamýšlená minimální komplexita systému. Obě sítě generují výstup pro kontrolu motorů vozidla. Sít' zabezpečující bezkolizní pohyb má vyšší prioritu existuje-li aktuální hrozba srážky s překážkou v blízkosti, je-li překážka daleko či žádná neexistuje je uvažován výstup sítě dosažení cíle. Data využita pro trénování sítě byla pořízena expertně na základě řízení robota lidským operátorem.

Robot byl testován v různých prostředích a dosahoval uspokojivých výsledků. Ovšem dosažení cíle robota bylo limitováno přesností GPS přijímače. Použité AT89C52 mikrokontroléry využívali linearizovanou verzi přenosové funkce tangens pro neuronovou sít'. Práce předjímá, že by bylo se škálovatelností neuronové sítě nutno použít i více mikrokontrolérů, použít DSP procesor nebo RAM založenou na neuronových sítích.

Shrnutí

Navigační problém byl v tomto případě řešen dvěma sítěmi, každá z nich se pak svým výstupem podílela na korekci pohybu robota. Robot ke svému pohybu potřebuje stanovený cíl a je schopen operovat v reálném čase. Aproximace přenosové funkce generovala chybu, a proto se zvyšujícím se počtem neuronů by se zhoršoval výsledek neuronového kontroléru.

2.5 Roverbot

Zaměříme-li se na Lego Mindstorms můžeme zmínit článek publikovaný Julio C. S. Reynoso. [16] Robot byl sestaven prostřednictvím stavebnice první generace Lego Mindstorms s programovatelnou kostkou RXC. U daného robota byla použita programovatelná kostka, dva dotykové snímače, světelný snímač a dva motory. Roverbot využíval malý Java virtuální stroj leJOS pro Lego Mindstorm RXC.

Zvolená dopředná neuronová sít' byla tvořena 3 vstupy pro jednotlivé sensory a dva výstupy pro motory. Cílem projektu bylo rovněž zkoumat emergentní chování robota, a

proto byla trénovací množina tvořena jen čtyřmi vzory, kdy přítomnost překážky detekovaná snímačem byla vyjádřena jako hodnota 1, nepřítomnost jako hodnota 0 a pohyb kupředu jako hodnota 1, dozadu jako hodnota 0 (pro každý motor).

Vytvořený program se zkompilem nahrál do RXC kostky a po spuštění byl realizován algoritmus BP. Nutno zmínit, že průběh 500 cyklů trval okolo 5 minut. Robot plnil očekávané chování.

Shrnutí

Zvolený přístup umožňoval docílit u robota komplexního chování, ovšem za cenu času, který byl potřeba k výpočtům. Samotný program bylo třeba vždy nahrát do kostky po úpravě.

2.6 EV3 OpenWorm

Neméně zajímavým projektem je OpenWorm [17], který mapoval synapse 302 neuronů háďatka obecného (*Caenorhabditis elegans*) a přenesl je do softwarové simulace. Následně byla virtuální reprezentace tohoto půdního červa aplikována na robota zkonstruovaného pomocí Lego Mindstorm EV3. Neurony hmatu červa byly nahrazeny daty z ultrazvukového snímače, který sbíral data každých 100ms (pokud snímač detekoval objekt v blízkosti 20cm vyslal signál), 95 neuronů pohybu bylo rozděleno podle svalů na levé a pravé straně a kumulováno do dvou výstupů (motorů robota).

Výsledná pozorování ukázala, že chování robota je podobné chování červa. Stimulace hmatu robota zastavila jeho pohyb (ultrazvukový snímač), stejně jako aktivování zadních a předních dotekových snímačů určovala pohyb robota vpřed nebo vzad. Stimulace snímače jídla aktivovala jeho pohyb vpřed.

Ačkoliv je těžké pochopit, jak propojení všech 302 neuronů pracuje, ukázalo se, že není třeba chování programovat ani učit. Zmapovaný systém emergoval chování červa. Výzkum se nyní snaží zodpovědět otázku, zda je zmapovaný červ v jiném těle stále háďatkem obecným a zda je naživu.

Shrnutí

Projekt OpenWorm zmapoval propojení neuronu půdního červa a přenesl jeho vnímání a výsledné chování do robotického těla. Ačkoliv simulace samotná není přesná vzhledem k programovému zjednodušení, neuronová síť napodobila mozek daného červa bez potřeby učení či programování. Je otázka na kolik lze zasahovat do sítě, respektive využívat abstrakce referenčního systému, bez vzniku vedlejších efektů či výraznější změny funkcionality.

2.7 Shrnutí řízení robotů neuronovými sítěmi

Jak lze vidět, neuronové sítě poskytují uplatnění při řešení mnoha problémů. Použitý typ neuronové sítě (struktura uzlů, topologie, algoritmus učení) významně závisí na řešeném problému. Mimo tento problém je UNS schopna řešit problémy příbuzné, se kterými se nesešla při procesu učení. Významnou se tedy stává odolnost vůči chybám a schopnost generalizace. Není možné ovšem zaručit správnost poskytnutého řešení, je zde přítomná (ve výstupu) jistá chyba.

Výhodou je samotné propojení výkonných uzlů do flexibilní a spolehlivé struktury, tedy neuronové sítě. Nejzápadnějším je pak nalezení váhových hodnot synapsí neuronů a to buď na základě předložených dat (učení s učitelem) nebo nalezení závislostí v datech (učení bez učitele). Při změně podmínek pak stačí provést pouze adaptaci sítě na nové podmínky, narážíme tak ale na problém času potřebného pro tuto adaptaci.

Nevýhodou je při vytváření složitějších struktur množství propojení samotných neuronů a neexistující postup pro tvorbu takovýchto struktur. Důležitým aspektem učení je rovněž samotná trénovací množina, která je ve většině případů sestavována expertně na základě předešlých zkušeností. Jak již bylo zmíněno další nevýhodou NS je její samotná nepřesnost a je nevhodné je používat pro uchovávání dat či řešení exaktních úloh.

Nejrozšířenějšími UNS jsou sítě vícevrstvé, jež mohou zpracovávat složitá data pro netriviální problémy. Umělé neuronové sítě se nejčastěji využívají v oblastech: klasifikace, zpracování obrazu, optimalizačních úloh, predikci časových řad, apd. Samotné sítě lze

rozdělit podle způsobů propojení uzlů: perceptron – jediný neuron pro řešení lineárně separabilních problémů; vícevrstvé sítě – řeší obecné problémy, výstup neuronu je zároveň vstupem neuronů další vrstvy; rekurentní sítě – v síti je přítomná zpětnovazební smyčka; samo-organizující se mapy – kdy jsou neurony uspořádány do mřížky (výstupní neuron reprezentující třídu výstupního vektoru je ten, jehož váhy jsou nejbližší vstupnímu vektoru) a další. [10]

Na základě získaných poznatků bude sestaven mobilní pásový robot se dvěma motory a sadou snímačů prostřednictvím Lego Mindstorms. Jako vhodná UNS bude volena dopředná síť a její funkci resp. globální cíl, tj. vyhnutí se překážkám v prostoru, rozložíme nízkourovňovým popisem na jednotlivé vzory v trénovací množině. Jednou z oblastí problémů je právě sestavování takovéto trénovací množiny pro UNS. Budeme se tedy zabývat právě problémem tvorby trénovací množiny a to přesněji jejím automatickým generováním a filtrací. Dále se pokusíme realizovat takovou síť, která by byla schopná realizovat reálný výstup jako rychlost pro jednotlivé motory robota.

3 MODEL AUTONOMNÍHO ROBOTA

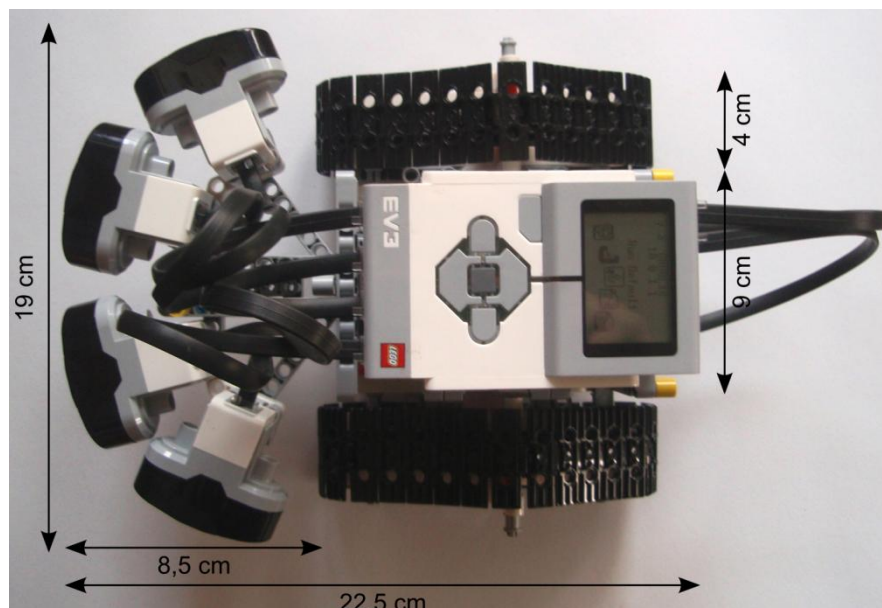
Navržený model robota je sestaven ze stavebnice Lego Mindstorm EV3. Jedná se o třetí generaci robotické řady stavebnic vydané v roce 2013. Programovatelná kostka (řídící jednotka) je vybavena 178x128 LCD displejem, procesorem TI Sitara AM1808(ARM926EJ-S core) 300 MHz, 64MB RAM a 16MB flash pamětí, Micro SDHC slotem, USB rozhraním, které umožňuje připojení přes WiFi dongle, 4 vstupní, 4 výstupní porty a v neposlední řadě podporuje řídící kostka technologii Bluetooth.

Z důvodů technických omezení stavebnice budeme hovořit o robotovi jako o modelu autonomního robota. Řídící program bude ovládat robota vzdáleně z počítače.

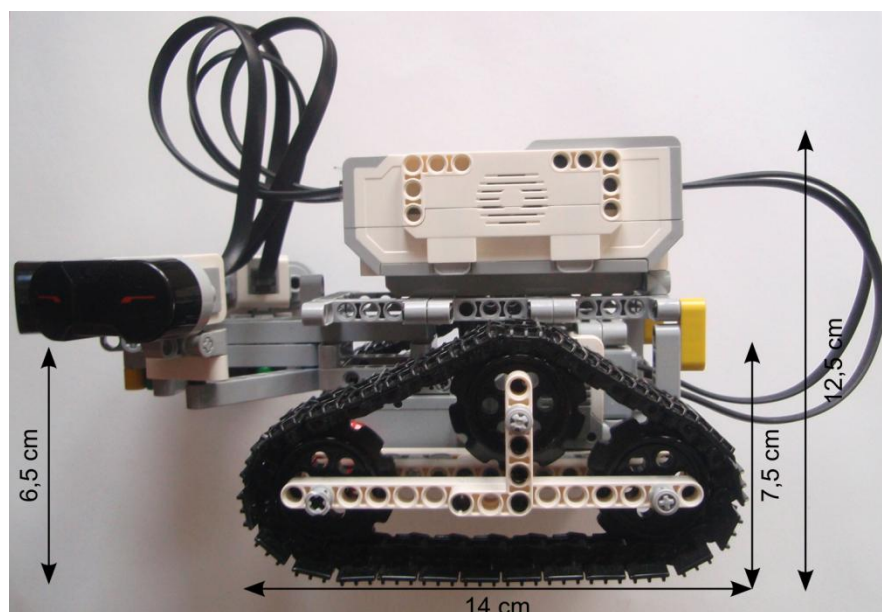
Použitá stavebnice je ze sady The Education EV3 Core Set, jež se skládá z: EV3 programovatelné kostky, 2 velkých motorů, středního motoru, 2 dotykových snímačů, barevného snímače, gyroskopu, ultrazvukového snímače, kabelů, USB kabelu, akumulátoru, dalších technických prvků a lego částí. [18] V návrhu robota byly použity 3 dodatečné infračervené snímače.

Hardwarové vybavení robota

Robot, obrázek 7 a obrázek 8, využívá dva velké motory, které pohání pásový podvozek, na němž je umístěna programovatelná kostka. K podvozku je rovněž připevněna senzorická hlava se 4 infračervenými snímači umístěnými ve výšce 6,5 cm od povrchu. Každý infračervený snímač vysílá infračervený signál a detekuje odraz od objektu, který se nachází před daným snímačem. Síla odraženého signálu indikuje vzdálenost od objektu. Přesnost měření závisí na velikosti objektu, jeho barvě (světlé objekty odrážejí signál lépe než tmavé), materiálu a dalších faktorech. Samotná hodnota v rozmezí 0 až 100 nedefinuje přesně vzdálenost od objektu. Nicméně hodnoty s malou odchylkou odpovídají hodnotám v centimetrech. Snímač není schopen detekovat velmi malé vzdálenosti kolem 1 centimetru od objektu. Snímač v portu S_4 je umístěn vlevo, S_3 uprostřed nalevo, S_2 uprostřed napravo a S_1 vpravo. Levý motor je připojen do portu B a pravý do portu A .



Obrázek 7: Sestavený robot - pohled shora



Obrázek 8: Sestavený robot - pohled ze strany

Softwarové vybavení robota

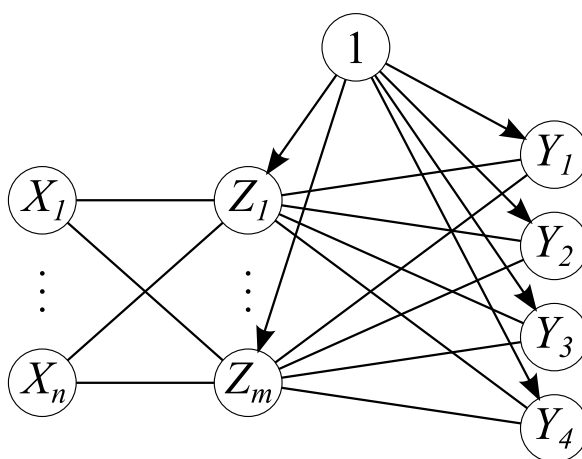
Originální Lego Mindstorms firmware programovatelné kostky byl nahrazen leJOS, jedná se o malou Java Virtual Machine, která umožňuje programovat robota v jazyce Java. LeJOS, původně TinyVM, byl vyvinut José Solórzano v roce 1999 jako open source

projekt. V roce 2013 byla vydána verze pro EV3 a zahrnuje všechny třídy EV3 API a Java Runtime systém. [19]

LeJOS zavádí operační systém na EV3 z SD karty. Programovatelná kostka po spuštění kontroluje, zda se na SD kartě nachází OS/firmware a pokud ano, zavádí systém z SD karty namísto interního. Nedochází tedy k přepisu originálního firmware a po vyjmutí karty je možno opět pracovat s originálním. Výhodou LeJOS je stabilní API, jednoduchost instalace a použití, živá vývojářská komunita.

3.1 Řídicí neuronová síť

Jako řídicí neuronová síť je zvolena třívrstvá dopředná síť. Pro zvolení vhodného počtu vrstev a neuronů v těchto vrstvách neexistuje přesný postup [20], proto je podoba této sítě zvolena na základě výsledků předešlých simulací a testování [9]. Na obrázku 9 reprezentují vstupní neurony s označením X_i , neurony skryté vrstvy jsou označeny Z_i a výstupní neurony značeny Y_i . Neurony vstupní vrstvy jsou spojeny synapsemi s neurony skryté vrstvy a neurony skryté vrstvy spojeny s neurony výstupní vrstvy. Do skrytých neuronů a výstupních neuronů vstupuje navíc váhová hodnota (práh) stále aktivního fiktivního neuronu s hodnotou výstupu rovnou jedné.



Obrázek 9: Řídicí neuronová síť

Uvážíme-li, že model robota má k dispozici čtyři infračervené snímače schopné detekovat vzdálenost od 0 do 100cm, je vhodné rozložit tento interval na několik částí, které by se daly charakterizovat např. slovy blízko, středně daleko, daleko, atd. Získáme tak větší

variabilitu pro adaptaci robota. Expanzi sítě budeme tedy provádět pouze do šířky a to na základě způsobu rozložení detekovaného intervalu snímačů do podintervalů.

U výstupů by použití stejného postupu znamenalo, že bude růst složitost adaptace sítě s tím, jak bude růst počet rychlostí (výstupů) a v závislosti s nimi počet skrytých neuronů. Opustíme-li binární výstupní vektor a nahradíme jej vektorem reálných čísel, jsme schopni zredukovat nutný počet výstupních neuronů na čtyři, jak je vidět z obrázku 9, vzhledem k použití dvou motorů – dva pro každý motor (chod vpřed a zpětný chod).

Pro transformaci vstupního signálu neuronu na výstupní využijeme sigmodiální funkci $S_\lambda(x): \mathbb{R} \rightarrow (0,1)$ vyjádřenou:

$$S_\lambda(x) = \frac{1}{1 + e^{-\lambda x}} \quad (2)$$

kde konstanta λ je přiřazena danému neuronu jako parametr strmosti. Parametr větší jak $\lambda = 3$ mění tvar sigmoidy blíže ke skokové přenosové funkci. Lze tak úpravou konstanty změnit homogenní síť na síť, jejíž neurony budou mít specifické aktivační funkce, tj. nehomogenní síť, nicméně tento parametr nebudeme adaptovat. [10]

Pro adaptaci zvolené sítě s n vstupy, m skrytými neurony a čtyřmi výstupy použijeme sestupný gradientní algoritmus Back-Propagation. Máme-li trénovací množinu $\{(\mathbf{x}_1, \mathbf{o}_1), \dots, (\mathbf{x}_p, \mathbf{o}_p)\}$ složenou z p párů n - a 4-rozměrných vektorů (vstupní a výstupní vzory) a inicializovány váhové hodnoty náhodně, po předložení vzoru \mathbf{x}_i obdržíme jako výstup sítě \mathbf{y}_i , který se však liší od výstupu \mathbf{o}_i , jež bychom očekávali. Cílem je tedy nalezení minimální chyby UNS v prostoru vah pro všechny vzory $1, \dots, p$ definovanou pro čtyři výstupní neurony jako střední kvadratickou chybu:

$$E = \frac{1}{2} \sum_{i=1}^p \sum_{j=1}^4 (\mathbf{y}_{ij} - \mathbf{o}_{ij})^2. \quad (3)$$

V průběhu iterativního algoritmu se snažíme tuto chybu minimalizovat prostřednictvím úpravy váhových hodnot w_i o přírůstek podle:

$$\Delta w_i(t) = -\alpha \frac{\partial E}{\partial w_i} + \mu \Delta w_i(t-1), \quad (4)$$

kde $\alpha \in (0,1]$ reprezentuje koeficient učení a parametr setrvačnosti $\mu \in [0,1]$ zohledňuje přírůstek v čase $t-1$ a zachovává tak trend v průběhu adaptace. [10]

Po úpravě (4) můžeme přírůstek váhy mezi vstupní a skrytou vrstvou vyjádřit jako:

$$\Delta w_j(t) = \alpha \sum_{i=1}^n [(o_i - y_i) \lambda y_i (1 - y_i) v_i] \lambda z_j (1 - z_j) x_j + \mu \Delta w_j(t-1), \quad (5)$$

kde v_i je váha odchozího spojení a z_j je výstup neuronu skryté vrstvy. Pro přírůstek váhy mezi skrytou a výstupní vrstvou úpravou (4) získáme:

$$\Delta w_i(t) = \alpha (o_i - y_i) \lambda y_i (1 - y_i) x_i + \mu \Delta w_i(t-1). \quad (6)$$

3.2 Generování trénovací množiny

Jak již bylo zmíněno, vstupem sítě je vstupní vektor $\mathbf{x} \in \{0,1\}$ a výstupem výstupní vektor $\mathbf{y} \in \mathbb{R}$. Je proto nutno zajistit, aby hodnoty požadovaného výstupu \mathbf{o}_i trénovací množiny byly z intervalu $[0,1]$. Toho docílíme normalizací hodnot:

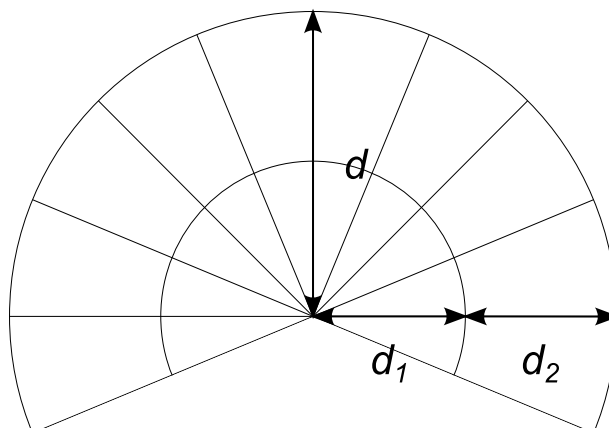
$$o_{norm} = \frac{o}{\max(o)}. \quad (7)$$

Pro získání normalizované hodnoty o_{norm} požadovaného výstupu je nutno znát jednak hodnotu o a její maximální hodnotu $\max(o)$, které může nabývat. Zpětně pak přepočítáme reálný výstup neuronové sítě na hodnoty výstupu pomocí vztahu:

$$o = o_{norm} \max(o). \quad (8)$$

Trénovací množinu, dále TM, neuronové sítě lze automatizovaně generovat. Je však potřeba znát, na kolik částí budeme snímanou oblast rozkládat. Princip generování množiny popíšeme na příkladu. Uvažujeme čtyři infračervené snímače a každý je schopen detekovat vzdálenost 0 – 100 cm. Tuto vzdálenost d rozdělíme na dvě stejně velké části,

viz obrázek 10. První část d_1 pojmenujeme blízko a druhou část d_2 jako daleko. Přítomnost překážky bude reprezentovat hodnota 1 a volný prostor hodnota 0.



Obrázek 10: Snímaná oblast

Nyní se zaměříme pouze na tu část trénovací množiny, která popisuje vstupy. Tyto vzory nazveme vstupní vzory úplné trénovací množiny. Máme-li s snímačů a i intervalů, tvořící celkovou snímanou vzdálenost d , pak počet možností, které mohou nastat je 2^{si} . Hodnota si rovněž reprezentuje počet vstupních neuronů UNS. Daná trénovací množina, na základě konfigurace robota a rozdělení snímané oblasti na dvě části, obsahuje 256 vzorů, které však obsahují také vzory ne zcela vyhovující. V tabulce 2 je ukázka prvních osmi vstupních vzorů trénovací množiny.

Tabulka 2: Vstupní vzory TM

Vzor	I1	I2	I3	I4	I5	I6	I7	I8
1	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0
4	1	1	0	0	0	0	0	0
5	0	0	1	0	0	0	0	0
6	1	0	1	0	0	0	0	0
7	0	1	1	0	0	0	0	0
8	1	1	1	0	0	0	0	0

Trénovací množina však obsahuje i vzory, které připouští možnost existence překážky ve vzdálenosti blízko a zároveň daleko, viz tabulka 2. Je tedy potřeba specifikovat skupinu vstupů I_i náležící danému snímači a v případě, že tato skupina obsahuje více než jednu hodnotu 1, takovýto vzor nezařazovat do množiny. Prvních osm vstupních vzorů redukované trénovací množiny, dále RTM, zobrazuje tabulka 3.

Tabulka 3: Vstupní vzory RTM

Vzor	Snímač1		Snímač2		Snímač3		Snímač4	
	I1	I2	I3	I4	I5	I6	I7	I8
1	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0
4	0	0	1	0	0	0	0	0
5	1	0	1	0	0	0	0	0
6	0	1	1	0	0	0	0	0
7	0	0	0	1	0	0	0	0
8	1	0	0	1	0	0	0	0

Redukovaná množina obsahuje 81 vzorů. Jak již bylo zmíněno, snímače zaznamenávají hodnotu 0 až 100 cm, z technických důvodů však budeme považovat i hodnotu 100 jako překážku (nelze říci, že se za ní překážka nenachází) a proto, je nutné vyloučit skupiny, jež obsahují výlučně hodnoty 0. Při sestavování trénovací množiny je tedy dobré dbát na rozložení intervalů a jejich množství. Zpět k našemu příkladu. Po provedení exkluze skupin s nulovými hodnotami se redukovaná trénovací množina, dále ERTM, zmenší na 16 vzorů, jak lze vidět v tabulce 4.

Tabulka 4: Vstupní vzory ERTM

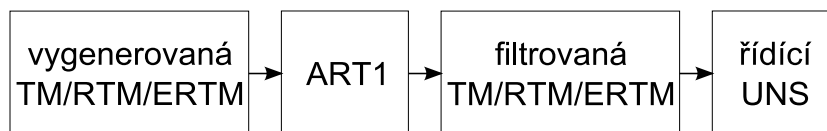
Vzor	Snímač1		Snímač2		Snímač3		Snímač4	
	I1	I2	I3	I4	I5	I6	I7	I8
1	1	0	1	0	1	0	1	0
2	0	1	1	0	1	0	1	0
3	1	0	0	1	1	0	1	0
4	0	1	0	1	1	0	1	0
5	1	0	1	0	0	1	1	0
6	0	1	1	0	0	1	1	0
7	1	0	0	1	0	1	1	0
8	0	1	0	1	0	1	1	0
9	1	0	1	0	1	0	0	1
10	0	1	1	0	1	0	0	1
11	1	0	0	1	1	0	0	1
12	0	1	0	1	1	0	0	1
13	1	0	1	0	0	1	0	1
14	0	1	1	0	0	1	0	1
15	1	0	0	1	0	1	0	1
16	0	1	0	1	0	1	0	1

Ačkoliv jsme tímto způsobem schopni z trénovací množiny odstranit více či méně nevhodné vstupy a významně ji redukovat, stále bude výsledná množina růst v závislosti na počtu snímačů a rozložení snímané oblasti. Provedeme proto další filtraci. Nicméně provedení exkluze nulových skupin závisí čistě na přístupu k řešení experimentu, avšak významně dodatečně redukuje RTM.

3.3 Filtrace trénovací množiny pomocí adaptivní rezonanční teorie

Filtrací TM odstraníme nutnost definování požadovaných výstupů pro celou (úplnou) množinu, jež by bylo časově náročné. Zároveň jsme schopni využitím heuristiky samoorganizujících sítí pro předzpracování TM nalézt typické vzory a pokrýt tak celý prostor řešení. Mimo jiné využijeme i jedné ze silných stránek neuronových sítí a to generalizaci. Jak již bylo testováno v bakalářské práci, neúplné množiny snižují pravděpodobnost lidské chyby při definování TM a celkově dosahují lepších výsledků oproti úplným trénovacím množinám [9].

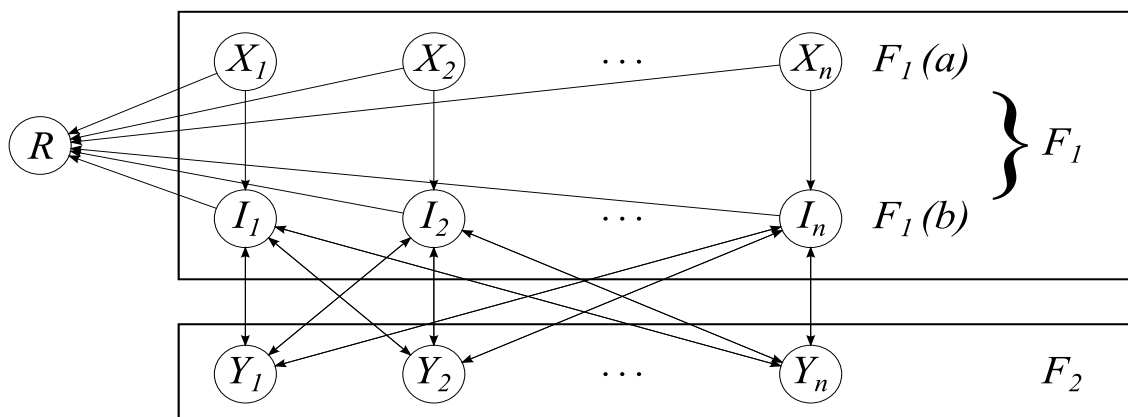
Pro filtraci TM využijeme neuronovou síť ART1 (Adaptive Resonance Theory), která pracuje s binárním vstupem. Předložený vstup je porovnán na základě parametru vigilance $\rho \in [0,1]$ s již uchovanými vzory v síti. Liší-li se tento nový vzor významně, je uložen do sítě jako nová třída, aniž by byly znehodnoceny dříve uložené informace, tato schopnost se nazývá plasticita sítě. ART1 je tedy schopna vytvářet skupiny na základě předložených vzorů a uživatel má možnost ovlivnit míru relativní podobnosti vzorů ukládaných do stejné skupiny. Jedná se tedy o učení bez učitele. Životní cyklus, kterým TM prochází, znázorňuje obrázek 11. V kroku filtrace pomocí ART1 navíc upravíme vstupní TM tak, aby byly vzory pseudonáhodně promíchány.



Obrázek 11: Životní cyklus TM

ART1 síť se skládá ze dvou vrstev a kontrolního neuronu R , obrázek 12. Vrstva F_1 obsahuje výpočetní neurony a to neurony X_i vstupní vrstvy $F_1(a)$ a neurony I_i vrstvy rozhraní $F_1(b)$. Vrstvu F_2 tvoří neurony tříd Y_i . Synapse mezi vrstvou F_1 a F_2 jsou

orientovány zdola-nahoru a synapse mezi F_2 a F_1 orientovány shora-dolů. Takovým to provázáním můžeme kontrolovat míru podobnosti informace uložené ve stejné skupině, konkrétně záleží na míře podobnosti váhové hodnoty synapse shora-dolů a vstupního vektoru. Rozhodování zajišťuje neuron R (reset), jež dělí stavy neuronů v F_2 do tří možností: aktivní, neaktivní – přípustný pro kompetici a ve stavu inhibice – nepřípustný pro další kompetici. [21]



Obrázek 12: ART1

Binární vstup je předložen neuronům X_i vstupní vrstvy $F_1(a)$ a informace je šířena daným neuronům ve vrstvě $F_1(b)$. Následně je z neuronů I_i šířen přes váhová spojení zdola-nahoru do neuronů Y_i a zpět přes spojení shora-dolů do neuronů I_i . Vrstva $F_1(b)$ tedy kombinuje vstupní signály a informaci z vrstvy F_2 (9) pro porovnání podobnosti vstupu (10) vzhledem k váze dané skupiny (jejímž reprezentantem je neuron Y_i) jež byla vybrána jako kandidát pro uložení informace. Jak již bylo zmíněno, vrstva F_2 obsahuje neurony tříd (kompetitivní neurony), které jsou vybírány jako kandidáti pro uložení informace prostřednictvím jejich váhových hodnot. Není-li umožněno, na základě R , uložení informace, daný neuron je inhibován a vybrán nový kandidát. V opačném případě je daný neuron vybrán jako reprezentant dané třídy a nastává adaptace váhových hodnot (11) (12). Mezi vrstvami $F_1(a)$ a $F_1(b)$ se vyskytuje fenomén emergence krátkodobé paměti a synapse mezi F_1 a F_2 pak reprezentují dlouhodobou paměť. [22]

3.3.1 Algoritmus ART1 pro filtraci trénovací množiny

Vzhledem k tomu, že vstup neuronové sítě jsme zvolili jako binární, využijeme schopnost ART1 sítě vytvářet tyto skupiny (klastrovat) pro filtraci trénovací množiny řídicí

neuronové sítě. Pro vytvoření filtrované trénovací množiny upravíme základní algoritmus ART1 [21] následně:

1. Inicializace váhových hodnot synapsí zdola-nahoru \mathbf{b}_{ij} , shora-dolů \mathbf{t}_{ij} a parametru vigilance $\rho \in [0, 1]$ ($\rho = 0$, pak všechny vstupní vzory jsou rozpoznány stejně; $\rho = 1$, pak žádný vektor není rozpoznán jako podobný vyjma zcela totožného vektoru):

$$b_{ij} = \frac{1}{(1 + n)},$$

$$t_{ij} = 1,$$

$$\rho \in [0, 1],$$

kde n je počet neuronů vstupní vrstvy, respektive počet prvků vstupního vektoru trénovací množiny.

2. Předložení vstupu $\mathbf{x} = [x_1, \dots, x_n]^T$ vstupní vrstvě F_1 .
3. Aktivace neuronů ve vrstvě F_2 (rozpознаvací vrstva) a výpočet odezvy podle:

$$y_j = \sum_{i=1}^n b_{ij} x_i, \quad (9)$$

kde y_j je odezva j -tého neuronu vrstvy F_2 a x_i je i -tý prvek vstupního vektoru \mathbf{x} .

4. Výběr neuronu s největší odezvou $y_{max} = \max(y_j)$.
5. Výpočet podobnosti δ vstupního vzoru se zpětným váhovým vektorem \mathbf{t}_{imax} neuronu určeného jako y_{max} :

$$\delta = \frac{\sum t_{imax} x_i}{\sum x_i}, \quad (10)$$

6. Test, zda je $\delta \geq \rho$, pokud ano nastává rezonance vstupu s očekáváním, kandidát je akceptován a následuje bod 7., jinak je kandidát y_{max} odmítnut a jeho výstup je

nastaven na hodnotu -1 , tzn. je v inhibici a nebude testován jako možný kandidát v dalším testu. Vybrán bude další vhodný kandidát (případné vytvoření nové třídy). Algoritmus pokračuje bodem 3.

7. Adaptace vah b_{imax} a t_{imax} neuronu y_{max} podle vztahů (Weber Law):

$$b_{imax} = \frac{L t_{imax} x_i}{L - 1 + \sum t_{imax} x_i}, \quad (11)$$

$$t_{imax} = t_{imax} x_i, \quad (12)$$

kde L je konstanta (excelling unit) obvykle se hodnota volí $L = 2$.

8. Test na postupné přidávání vstupních vektorů do sestavované redukované trénovací množiny:

- a. Pokud se v dané skupině sestavované RTM zatím žádný vektor nenachází, uloží aktuálně předložený vektor x jako typického zástupce skupiny.
- b. V případě, že je skupina neprázdná a zároveň aktuálně předložený vektor x na vstupu nese větší množství informace než vektor uložený jako typický zástupce dané skupiny definované neuronem vrstvy F_2 v RTM s dosavadní největší nesenou informací, nahradí jej.

9. Kontrola ukončovací podmínky (dosažení specifického množství kroků, apd.). Pokud je podmínka splněna, skok na bod 11.

10. Ukončení potlačení neuronů, jež byly inhibovány a pokračování od bodu 2.
















11. Zastavení algoritmu.

Stěžejním krokem modifikovaného algoritmu ART1 je krok 8., v němž se sestavuje na základě klasifikace sítě redukovaná trénovací množina. Jaké zvolit pravidlo stanovující vektor s větším obsahem nesené informace je na uživateli. V této práci budeme akceptovat do RTM vektor obsahující více kritických stavů, tj. prvků s hodnotou 1.

3.3.2 Vliv parametru vigilance

Jak již bylo zmíněno, ART1 síť vyžaduje parametr vigilance ρ , jež definuje stupeň podobnosti vyžadovaný pro přiřazení vzoru do stejné třídy. Tento parametr je definován uživatelem. Tabulka 5 zobrazuje vliv ρ na přiřazení vzorů $\mathbf{p} = [p_1, \dots, p_{16}]^T \in \{0,1\}$ s různým stupněm pseudonáhodného poškození do skupin.

Tabulka 5: Vliv parametru vigilance

Vzor trénovací množiny				Zařazení vzoru do skupiny		
Číslo	Varianta	Grafické znázornění	Stupeň poškození	$\rho = 0,3$	$\rho = 0,5$	$\rho = 0,7$
1	a		0,00%	1	1	1
	b		12,50%	1	1	1
	c		31,25%	1	1	2
2	a		0,00%	1	1	2
	b		12,50%	1	1	2
	c		6,25%	1	2	3
3	a		0,00%	2	3	4
	b		12,50%	2	3	5
	c		12,50%	2	3	4
4	a		0,00%	3	2	6
	b		12,50%	3	2	6
	c		12,50%	3	2	6
5	a		0,00%	2	3	5
	b		12,50%	1	1	5
	c		12,50%	1	4	7
Počet skupin, do nichž byla TM rozdělena:				3	4	7

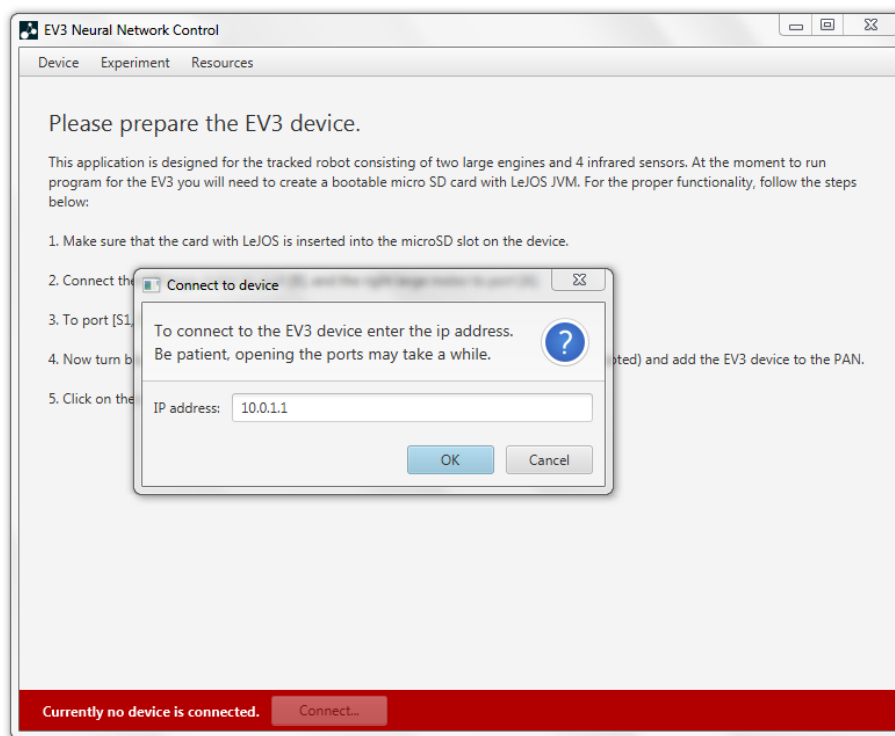
Při zvolení nízké hodnoty parametru vigilance může dojít k zařazení zcela odlišných vzorů do stejné skupiny a počet skupin je tak relativně nízký. Naopak velká hodnota ρ má za následek vytváření přílišného počtu skupin. V této práci budeme parametr vigilance volit tak, aby byla hodnota z intervalu $[0,5; 0,7]$. Nedochází tak k vytváření velkého množství skupin či případné nevhodné klasifikace vstupního vzoru.

4 APLIKACE PRO EXPERIMENTÁLNÍ STUDII

Byla vytvořena aplikace v jazyce Java, obrázek 13, umožňující kontrolovat experiment se sestaveným robotem. Mimo jiné lze ovládat prostřednictvím aplikace jakéhokoliv robota sestaveného pomocí Lego Mindstroms EV3 za předpokladu, že splňuje následující:

- SD karta s LeJOS (verze 0.9.1-beta a výše) vložená do Micro SD slotu v programovatelné kostce EV3,
- levý velký motor připojen do portu B a pravý velký motor připojen do portu A,
- do portů S1, S2, S3 a S4 připojeny infračervené snímače,
- zařízení EV3 připojeno do sítě PAN prostřednictvím Bluetooth.

Aplikaci, dokumentaci aplikace a zdrojové kódy lze najít na přiloženém DVD. Samotná aplikace se skládá z menu, hlavní obrazovky a stavového řádku zobrazujícího aktuálně připojené zařízení s možností odpojení/připojení na základě IP adresy.



Obrázek 13: Aplikace pro řízení EV3

4.1 Připojení zařízení

Na této obrazovce (Device) se nachází název aktuálně připojeného EV3 zařízení k aplikaci, stav akumulátoru (v %), možnost vypnutí a zapnutí zvuků vydávaných zařízením při změně stavu, nastavení defaultní rychlosti motorů (ve stupních/s) a zrychlení (ve stupních/s/s), v neposlední řadě aktuální hodnoty naměřené snímači (v cm) a popis pro ovládání zařízení pomocí klávesnice.

Pomocí kláves W/I lze pohybovat robotem vpřed, S/K pro zpětný chod, A/J otočení vlevo a D/L otočení vpravo. Při zvolení vyšší rychlosti mohou pásy na hladkém povrchu podkluzovat. Rychlost lze nastavit z intervalu 0 až 600 stupňů/s.

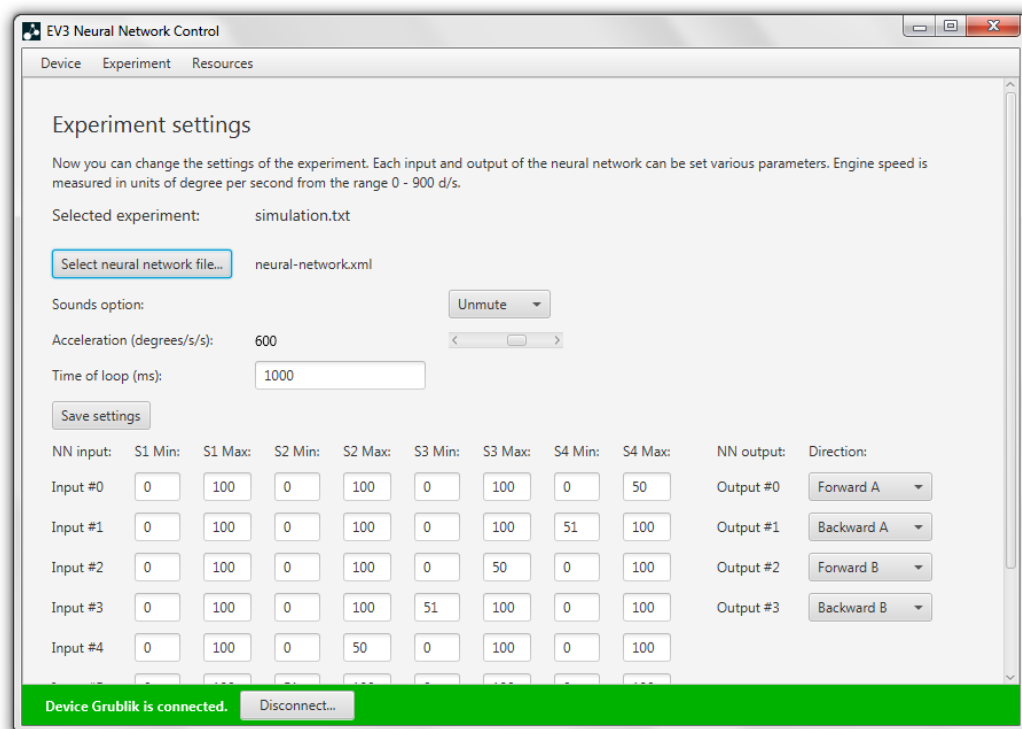
4.2 Nastavení experimentu

Po zvolení možnosti vytvoření experimentu (Experiment/Create) je potřeba aby uživatel vybral adaptovanou neuronovou síť. Na základě její topologie je vytvořena obrazovka pro úpravu parametrů experimentu viz obrázek 14. Důležitou položkou je čas potřebný na jeden cyklus v ms (time of loop) tzn. sejmutí vzdáleností snímači, předložení neuronové sítě a úprava rychlostí motorů (doporučeno volit hodnotu větší jak 500 ms). Dále lze upravit zrychlení motorů a vypnutí či zapnutí zvuků.

Níže na obrazovce následuje tabulka s úpravou vzdáleností pro jednotlivé vstupy neuronové sítě, které reprezentují řádky tabulky. Sloupce tabulky definují minimální a maximální hodnotu snímačů pro daný vstup UNS. Po nastavení intervalů reaguje vstup na překážku za předpokladu presence předmětu ve vzdálenosti od snímače v daném intervalu (vstup pro UNS nabude hodnotu 1). Uživatel tak má plnou kontrolu nad charakterem vstupu neuronové sítě a může jej libovolně upravovat. Hodnotu snímačů je nutné volit z intervalu 0 až 100 cm.

Vedle tabulky pro nastavení intervalů se nachází přiřazení chodů (vpřed, zpětný chod) motorů jednotlivým výstupům neuronové sítě.

Po nastavení experimentu je potřeba jej uložit. Lze jej pak kdykoliv opět nahrát prostřednictvím menu (Experiment/Load...). Možnost Nastavení (Settings) v záložce Experiment umožňuje vrátit se na obrazovku nastavení načteného experimentu.



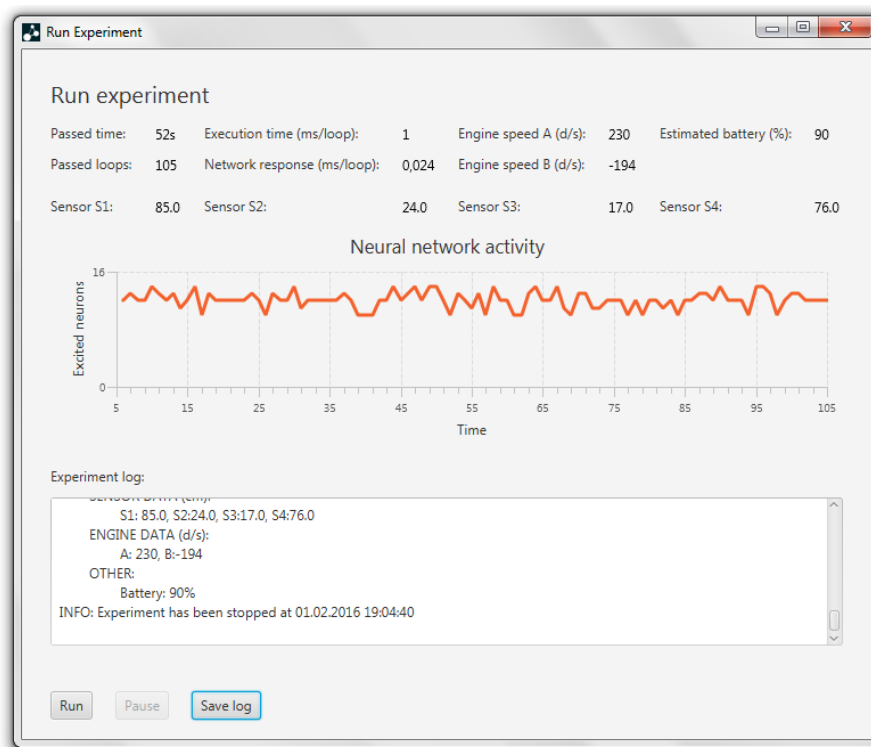
Obrázek 14: Nastavení experimentu

4.3 Spuštění experimentu

Poslední položkou, kterou lze nalézt v menu pod položkou Experiment je možnost jej spustit (Run), která zobrazí okno spuštěného experimentu, obrázek 15. Najdeme zde informaci o uběhlém čase experimentu, počtu uběhlých cyklů, čas potřebný na aktuální výpočet cyklu, čas odezvy neuronové sítě v daném kroku, hodnotu snímačů, aktuální odezvu neuronové sítě v podobě nastavených hodnot rychlostí motorů a stav akumulátoru. Stav akumulátoru se aktualizuje jednou za 100 uběhlých cyklů experimentu. Stejně tak se co 100 cyklů automaticky ukládají naměřené hodnoty experimentu.

Ve střední části okna se vykresluje aktivita UNS zobrazující počet excitovaných neuronů sítě v čase. Můžeme tak sledovat kolik neuronů je využíváno na základě vstupů.

Posledními prvky je výpis průběhu experimentu, jenž zaznamenává do souboru údaje z experimentu a tlačítka na pozastavení, znovu spuštění a uložení samotného experimentu.



Obrázek 15: Spuštěný experiment

4.4 Generování a filtrace TM, RTM, ERTM

Výběrem generování trénovací množiny v menu zdrojů (Resources/Training set/Generate) můžeme získat soubor s trénovací množinou pro neuronovou síť. Nástroj dokáže, na základě popisu v kapitole Generování trénovací množiny, generovat úplnou trénovací množinu, redukovanou trénovací množinu a redukovanou trénovací množinu s exkludovanými nulovými skupinami. Okamžitě po změně nastavení generování zobrazuje i výsledný počet vzorů v dané trénovací množině.

Trénovací množinu lze filtrovat pomocí ART1, a postupu popsaného v kapitole Filtrace trénovací množiny pomocí adaptivní rezonanční teorie, výběrem položky filtrovat v menu zdrojů (Resources/Training set/Filter). Filtrování probíhá na základě parametru vigilance jehož vliv je popsán v kapitole Filtrace trénovací množiny pomocí adaptivní rezonanční teorie a kapitole Vliv parametru . Navíc lze zvolit pravidlo definující vektor s větším obsahem nesené informace – nejvíce jedniček, nejvíce nul (the most of ones, the most of zeros).

Je nutno zmínit, že generování/filtrování velkých TM může zabrat velmi dlouhou dobu. Jakmile je trénovací množina vygenerována a následně profiltrována (volitelné) lze přistoupit k její úpravě a to konkrétně k úpravě výstupních hodnot tzn. definování rychlostí motorů pro jednotlivé chody. Ty jsou defaultně nastaveny na hodnotu 0. Prvky výstupního vektoru TM mohou nabývat hodnot 0 až 600 stupňů/s a jsou automaticky při zpracování řídicí neuronovou sítí normalizovány do intervalu $[0,1]$.

4.5 Vytvoření neuronové sítě a její adaptace

Pro vytvoření neuronové sítě stačí zvolit možnost vytvořit neuronovou síť v menu (Resources/Neural network/Create). Obrazovka umožňuje nastavit topologii sítě – počet vstupních, skrytých a výstupních neuronů, koeficient učení α a parametr setrvačnosti μ , případně vytvoření nehomogenní sítě úpravou intervalu pro náhodné přiřazení konstanty strmosti λ jednotlivým neuronům, viz kapitola Řídicí neuronová síť. Nicméně parametr setrvačnosti a koeficient učení lze upravit později při nastavení adaptace.

Samotná adaptace (Resources/Neural network/Train) vyžaduje výběr vytvořené neuronové sítě a trénovací množiny, jež musí být kompatibilní s danou neuronovou sítí. Jakmile jsou tyto položky vybrány lze přistoupit k definování podmínek ukončení adaptace tj. nastavení maximálního počtu cyklů a minimální přípustné chyby učení viz kapitola Řídicí neuronová síť. Rovněž lze dodatečně upravit koeficient učení a parametr setrvačnosti učení. Po kliknutí na tlačítko trénování (Train) se zobrazí graf vývoje chyby učení v čase. Hodnoty z průběhu učení jsou rovněž zaznamenány v souboru v adresáři s uloženou UNS. Jakmile je dosaženo podmínky ukončení adaptace, zobrazí se přehled, zda byla síť úspěšně adaptována (v tomto případě se zároveň upravené hodnoty synaptických spojení neuronů uloží), hodnota chyby a počet kroků učení.

4.6 Postup vytvoření experimentu

Pro korektní vytvoření experimentu je vhodné dodržovat následující postup použití aplikace, sníží se riziko chybného nastavení:

1. Definování rozložení snímaného intervalu snímači a na základě tohoto rozložení určení počtu vstupů pro UNS.

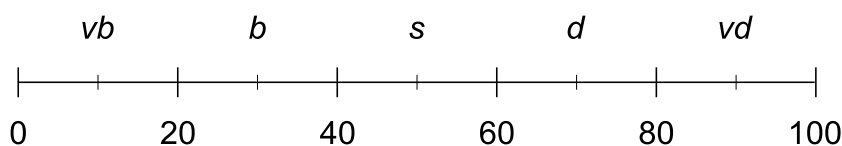
2. Nastavení generování trénovací množiny (TM, RTM či ERTM) a její vygenerování.
3. Filtrace trénovací množiny pomocí ART1 (volitelné, záleží také na velikosti TM).
4. Úprava prvků výstupního vektoru TM na požadované hodnoty rychlostí motorů, s ohledem na chod vpřed či zpětný chod, z intervalu 0 až 600 stupňů/s.
5. Vytvoření odpovídající neuronové sítě (potřeba dbát na počet vstupních neuronů shodných s počtem vstupů při generování TM a zvolení odpovídajícího počtu skrytých neuronů).
6. Adaptace vytvořené sítě pomocí (filtrované) trénovací množiny.
7. Připojení EV3 robota k aplikaci (pokud tak ještě není učiněno).
8. Vytvoření/nahrání parametrů experimentu a nastavení reakce vstupů sítě na dané hodnoty snímačů (je potřeba si uvědomit, že řádky tabulky reprezentují jednotlivé vstupy UNS a měřené hodnoty se pohybují v rozmezí 0 až 100 cm). Uložení experimentu.
9. Spuštění experimentu.

5 EXPERIMENTÁLNÍ STUDIE

Cílem experimentální studie je ověření chování robota řízeného umělou neuronovou sítí adaptovanou trénovací množinou filtrovanou prostřednictvím ART1 sítě. Pro provedení experimentů je využita vytvořená aplikace. Využijeme obecného popisu přístupu k řešení problému, viz Řídící neuronová síť. Necht' je snímaný prostor infračerveného snímače rozdělen na pět částí, obrázek 16. Tyto části vyjádříme množinou termů:

- *vb* – velmi blízko (0 – 19 cm),
- *b* – blízko (20 – 39 cm),
- *s* – středně daleko (40 – 59 cm),
- *d* – daleko (60 – 79 cm),
- *vd* – velmi daleko (80 – 100 cm).

Každý snímač má přidruжено těchto pět termů, jež jsou binární reprezentací překážky v dosahu snímače v daném intervalu. Hodnota jedna vyjadřuje přítomnost překážky v daném intervalu a hodnota nula prázdný prostor. Množina všech termů daných snímačů pak tvoří vstupní vektor řídící UNS. Hodnotu 100 cm považujeme za překážku v příslušném intervalu (nelze rozlišit překážku ve vzdálenosti 100 cm či prázdný prostor za touto vzdáleností). Nenastane tedy stav, kdy by nebyla překážka ani v jednom intervalu.



Obrázek 16: Snímaná oblast

Trénovací množina UNS bude tedy obsahovat 20 vstupů a 4 výstupy. Úplná trénovací množina pro danou konfiguraci obsahuje 1 048 576 vzorů, redukovaná trénovací množina 1 296 vzorů a redukovaná množina s exkludovanými nulovými skupinami 625 vzorů. Pro experimentální studii využijeme právě ERTM. Vliv nastavení parametrů vigilance pro tuto

trénovací množinu zobrazuje tabulka 6. Charakteristiky počtu skupin odrážejí deset filtrování ERTM pomocí ART1 sítě pro každé nastavení parametru vigilance.

Tabulka 6: Počet skupin ERTM v závislosti na parametru vigilance

Hodnota parametru vigilance	Minimální hodnota počtu skupin	Maximální hodnota počtu skupin	Průměrná hodnota počtu skupin	Střední hodnota počtu skupin
$\rho = 0,0$	6	12	8,6	7
$\rho = 0,1$	5	11	7,8	5
$\rho = 0,2$	5	10	7,8	8
$\rho = 0,3$	36	53	46,9	51
$\rho = 0,4$	36	54	46,2	47
$\rho = 0,5$	40	51	46,3	50
$\rho = 0,6$	177	188	182,6	188
$\rho = 0,7$	179	192	184,9	184
$\rho = 0,8$	625	625	625,0	625
$\rho = 0,9$	625	625	625,0	625
$\rho = 1,0$	625	625	625,0	625

Počet skupin filtrované TM je proměnlivý díky promíchání vzorů (o promíchání se stará aplikace) před předložením ART1 neuronové sítě a tento krok poměrně významně ovlivňuje podobu výsledné trénovací množiny, nicméně je důležitý pro větší variabilitu TM. Při filtrování seřazené skupiny dochází k vytváření skupin s relativně podobnými částmi vstupního vektoru, což je způsobeno výběrem vhodného vzoru z utvářené skupiny do filtrované TM. Je patrné, že daná množina s 625 vzory a parametrem vigilance větším jak $\rho = 0.5$ dosahuje uživatelsky nepřívětivého počtu skupin, vzhledem k následné nutnosti nastavit manuálně těmto vzorům výstupní hodnoty. Pro filtraci ERTM využijeme hodnotu parametru vigilance $\rho = 0.5$.

5.1 Trénovací množina a řídicí neuronová síť

Vygenerovaná ERTM má 20 vstupů, 4 výstupy a byla provedena filtrace prostřednictvím ART1 s parametrem vigilance $\rho = 0.5$. Výsledná filtrovaná množina obsahuje 51 vzorů a do TM byly zvoleny vzory s největším množstvím nesené informace, tj. největším počtem jedniček. Pro přehlednost úpravy vygenerované trénovací množiny provedeme sloučení sloupců přidružených intervalů jednotlivých snímačů do sebe samých, tzn., že každých 5 binárních vstupů u každého snímače TM reprezentujících překážku v daném intervalu nahradíme pouze jednou lingvistickou hodnotou vzdálenosti. Můžeme tak přehledněji nastavit výstupy. Zpětně pak expandujeme TM pro adaptaci neuronové sítě. Výsledná

expandovaná množina s nastavenými hodnotami je součástí přílohy. Tabulka 7 popisuje přehledně výstupy trénovací množiny, kde S_1 , S_2 , S_3 a S_4 označuje snímače v daném portu; B_F a A_F rychlost motorů pro chod vpřed, B_B a A_B rychlost motorů pro zpětný chod pro porty A, B.

Tabulka 7: Trénovací množina pro adaptaci UNS

Číslo vzoru	Ohodnocení snímačů termy				Nastavení rychlosti motoru pro chod vpřed a zpětný chod ve stupních za sekundu			
	S4	S3	S2	S1	BF	BB	AF	AB
1	vb	d	vd	b	350	0	300	0
2	d	s	s	vb	300	0	350	0
3	b	d	s	vd	350	0	300	0
4	d	s	d	s	300	0	350	0
5	b	s	b	d	250	0	200	0
6	vd	d	vd	d	500	0	500	0
7	vd	vb	s	vb	0	100	100	0
8	s	b	s	s	200	0	200	0
9	s	vd	vd	vb	300	0	350	0
10	s	vb	d	b	100	0	0	100
11	d	b	b	b	150	0	200	0
12	vd	vd	vb	s	0	100	100	0
13	vd	vb	s	vd	0	100	100	0
14	vd	vd	d	s	350	0	400	0
15	b	b	b	vb	150	0	200	0
16	d	s	d	d	350	0	300	0
17	d	vb	vd	b	100	0	0	100
18	b	d	vd	s	400	0	350	0
19	s	d	vb	vb	0	100	100	0
20	vd	vb	s	d	0	100	100	0
21	vb	vd	vd	vb	200	0	200	0
22	b	vd	vb	vd	100	0	0	100
23	b	vb	b	s	100	0	0	100
24	vd	b	d	b	150	0	200	0
25	vb	s	vb	b	0	100	100	0
26	vb	s	s	vd	300	0	250	0
27	s	d	d	d	400	0	350	0
28	d	s	s	b	250	0	300	0
29	vb	b	vd	vd	200	0	150	0
30	vd	vb	d	vb	0	100	100	0
31	b	s	d	vd	300	0	250	0
32	b	vb	s	s	100	0	0	100
33	vd	vd	s	b	250	0	300	0
34	d	vd	d	vb	300	0	350	0
35	vd	vb	s	s	0	100	100	0
36	vd	vd	b	d	150	0	200	0

Číslo vzoru	Ohodnocení snímačů termy				Nastavení rychlosti motoru pro chod vpřed a zpětný chod ve stupních za sekundu			
	S4	S3	S2	S1	BF	BB	AF	AB
37	<i>vb</i>	<i>s</i>	<i>b</i>	<i>s</i>	250	0	200	0
38	<i>s</i>	<i>b</i>	<i>vd</i>	<i>s</i>	200	0	150	0
39	<i>d</i>	<i>b</i>	<i>vb</i>	<i>vd</i>	100	0	0	100
40	<i>b</i>	<i>vd</i>	<i>d</i>	<i>b</i>	400	0	400	0
41	<i>vd</i>	<i>s</i>	<i>d</i>	<i>s</i>	300	0	350	0
42	<i>b</i>	<i>vb</i>	<i>vb</i>	<i>vb</i>	0	100	100	0
43	<i>s</i>	<i>d</i>	<i>b</i>	<i>vb</i>	150	0	200	0
44	<i>vb</i>	<i>vb</i>	<i>vd</i>	<i>b</i>	100	0	0	100
45	<i>vd</i>	<i>s</i>	<i>s</i>	<i>d</i>	250	0	300	0
46	<i>s</i>	<i>b</i>	<i>d</i>	<i>s</i>	200	0	150	0
47	<i>vb</i>	<i>vd</i>	<i>b</i>	<i>b</i>	200	0	150	0
48	<i>d</i>	<i>d</i>	<i>d</i>	<i>b</i>	350	0	400	0
49	<i>b</i>	<i>b</i>	<i>s</i>	<i>vd</i>	200	0	150	0
50	<i>d</i>	<i>vb</i>	<i>vb</i>	<i>vb</i>	0	100	100	0
51	<i>vb</i>	<i>b</i>	<i>vb</i>	<i>d</i>	100	0	0	100
52	<i>vd</i>	<i>vd</i>	<i>vd</i>	<i>vd</i>	500	0	500	0

Zvolené nastavení rychlostí pro jednotlivé chody obou motorů byly nastaveny experimentálně s ohledem na omezení maximální rychlosti na 500 stupňů/s z důvodu podkluzování pásů na hladkém povrchu při vyšších rychlostech. Na konec množiny byl přidán dodatečně vzor číslo 52, který definuje defaultní chování pro pohyb vpřed. Přidáním vzoru zamezíme tomu, aby robot, bez detekce jakékoliv překážky jel jiným směrem nežli rovně.

Pro topologii řídicí UNS byla zvolena následující konfigurace: 20 vstupních neuronů, 20 skrytých a 4 výstupní. Parametr učení $\alpha = 0.85$, koeficient setrvačnosti $\mu = 0.9$ a síť je homogenní tj. $\lambda = 1$ pro všechny neurony. Podmínka ukončení adaptace byla zvolena na $E \leq 0,0001$ (3) vzhledem k normalizaci TM (7), (8). Samotná adaptace trvala 1m a 49s, 14 393 cyklů. Průběh adaptace a vývoj chyby v čase je obsažen v příslušném souboru v příloze.

5.2 Definice experimentů a kritérií

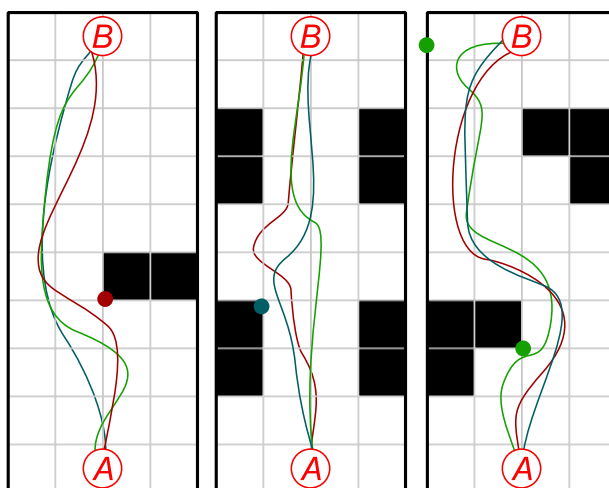
Nechť je umístěn model autonomního robota do prostoru vymezeného popisem experimentu, jež sestává ze čtverců o ploše 15 cm^2 . Cílem robota je dosažení úkolu definovaného daným experimentem bez kolize s překážkami ve vymezeném prostoru. Pro zhodnocení experimentu jsou zaznamenávány následující údaje:

- počet primárních kolizí (přímé kolize robota s překážkou, tj. najetí do překážky),
- počet sekundárních kolizí (kolize jen některé části robota jako jsou kabely snímačů, apd.),
- počet uváznutí na místě,
- potřebný čas experimentu,
- dráha pohybu modelu robota.

Akcelerace motorů je pro všechny experimenty nastavena na hodnotu 600 stupňů/s/s. Snímání prostoru probíhá jednou za 500 ms. Výstup průběhu jednotlivých experimentů obsahující data o rychlostech, snímaných vzdálenostech, době odezvy UNS atd. je součástí přílohy.

5.2.1 Experiment č. 1

Prostředí pro první experiment má rozměry 150×60 cm. Jednotlivé varianty obsahují překážky různých rozměrů ve střední části. Úkolem robota je se překážkám vyhnout a dosáhnout bodu B. Ve variantě B je testováno, zda je robot schopen držet směr se střídavou změnou snímané vzdálenosti po stranách.



Obrázek 17: Prostředí č. 1, zleva varianta A, B, C

Pro každou variantu prostředí byly provedeny tři průjezdy, viz dráhy průjezdů v obrázku 17, vyznačeny jsou rovněž sekundární kolize. Problémem v těchto prostředích bylo

otáčení robota blízko rohových částí. V několika případech při otáčení robot zavadil hranou snímače o překážku a to z důvodu, že se daný roh nacházel ve slepém místě senzorické hlavy či byly nasnímány chybné údaje. Rovněž průjezd v některých případech trval poměrně dlouho, což je způsobeno těsným rozmístěním překážek a robot tak musel operovat s velmi nízkými rychlostmi. Naměřené hodnoty jednotlivých průjezdů zobrazuje tabulka 8.

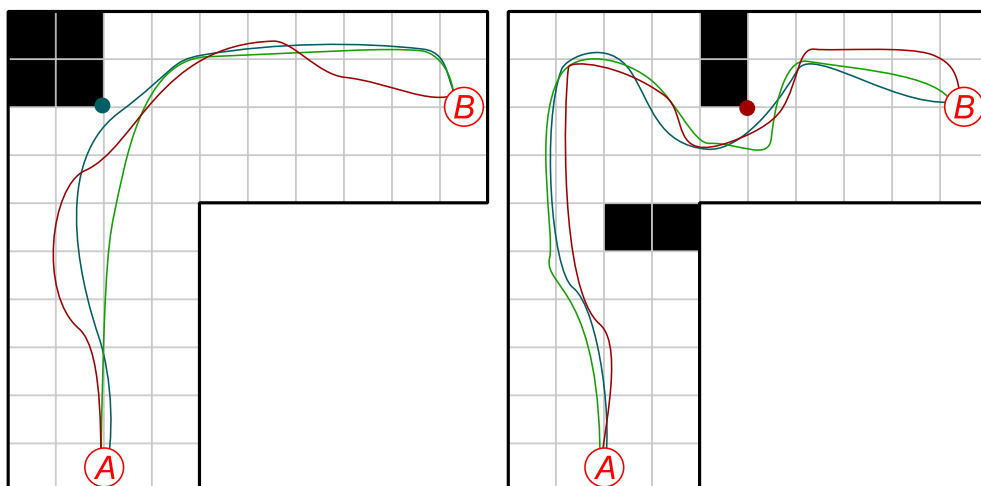
Tabulka 8: Výsledky experimentu č. 1

Číslo průjezdu	Varianta prostředí	Počet primárních kolizí	Počet sekundárních kolizí	Počet uvážnutí robota	Doba trvání experimentu
1	A	0	0	0	18s
2	A	0	0	0	17s
3	A	0	1	0	18s
4	B	0	0	0	21s
5	B	0	0	0	32s
6	B	0	1	0	31s
7	C	0	0	0	21s
8	C	0	2	0	26s
9	C	0	0	0	28s

Ve všech případech robot dosáhl cílového bodu bez přímé kolize a nedošlo k jeho uvážnutí. Z výstupu aplikace je patrné, že signál snímačů se v některých případech odráží způsobem, vedoucím k chybnému vyhodnocení měřené vzdálenosti viz případy rohových kolizí.

5.2.1 Experiment č. 2

Ve druhém experimentu je úkolem robota opět dosažení bodu B v prostředí reprezentující pravoúhlou pravotočivou zatáčku s překážkami. Zatáčka obsahuje překážky ve dvou variantách A, B.



Obrázek 18: Prostředí č. 2, zleva varianta A, B

Při průjezdu variantou A robot opět lehce zavadil rohem snímače při otáčení o roh překážky. U varianty B došlo k sekundární kolizi pásu a rohu druhé překážky při úpravě směru pohybu.

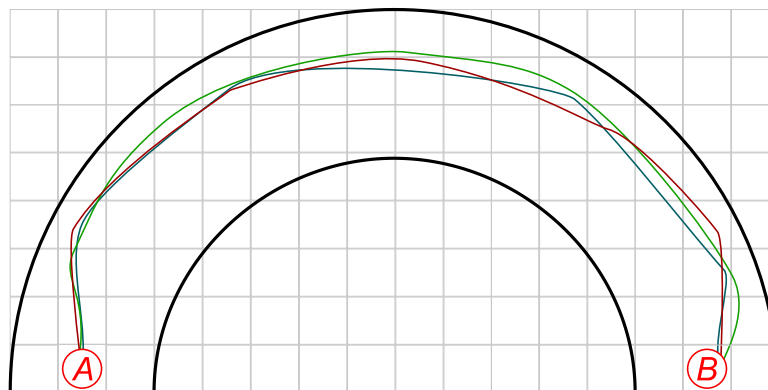
Tabulka 9: Výsledky experimentu č. 2

Číslo průjezdu	Varianta prostředí	Počet primárních kolizí	Počet sekundárních kolizí	Počet uváznutí robota	Doba trvání experimentu
1	A	0	1	0	26s
2	A	0	0	0	24s
3	A	0	0	0	22s
4	B	0	0	0	37s
5	B	0	0	0	34s
6	B	0	1	0	38s

Opět nedošlo k uváznutí robota či přímé kolizi. Z tabulky 9 lze vyčíst, že u varianty B se čas průjezdu výrazně navýšil. Toto navýšení způsobila druhá překážka, kdy byl robot nucen významně zpomalit a na místě se pomalu otáčet do vhodné pozice pro průjezd do cílové části prostředí.

5.2.1 Experiment č. 3

Cílem třetího experimentu je otestování schopnosti robota pohybovat se pod úhlem a dosáhnout cílového bodu B bez kolize se stěnami definujícími hranici prostředí. Stěny jsou umístěny tak, aby aproximovaly zobrazené zakřivení na obrázku 19.



Obrázek 19: Prostředí č. 3

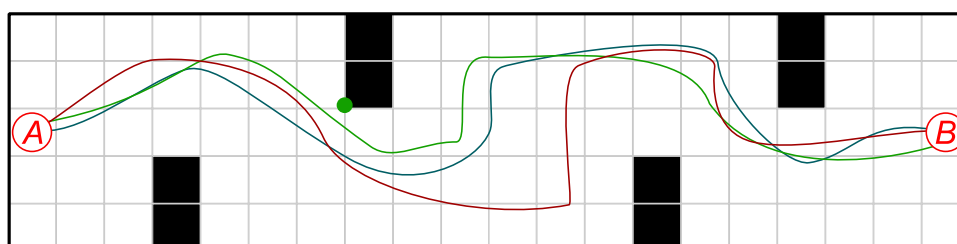
U tohoto experimentu nedošlo k žádnému typu kolize ani uváznutí. Avšak snímač na levé straně robota jen velmi těsně mýjel levou stěnu dráhy. Naměřené časy průjezdu tohoto prostředí jsou zaznamenány v tabulce 10.

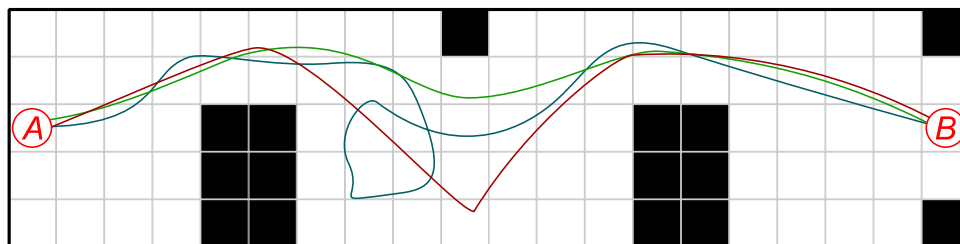
Tabulka 10: Výsledky experimentu č. 3

Číslo průjezdu	Počet primárních kolizí	Počet sekundárních kolizí	Počet uváznutí robota	Doba trvání experimentu
1	0	0	0	47s
2	0	0	0	51s
3	0	0	0	44s

5.2.1 Experiment č. 4

V předposledním experimentu je robot umístěn do prostředí o délce 300 cm a šířce 75 cm. Mezi body A a B jsou umístěny střídavě překážky tak, aby robot musel provést slalom mezi nimi. Opět je úkolem robota se bezkolizně dostat do konečného bodu.





Obrázek 20: Prostředí č. 4, shora varianta A, B

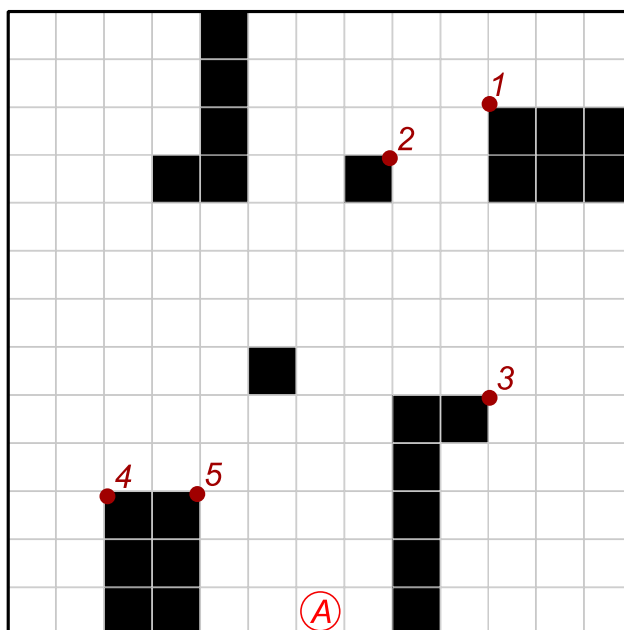
Ve variantě A došlo k sekundární kolizi snímače a rohu překážky při otáčení, ve variantě B projel robot zúženými místy a nenarazil do překážek, nedošlo rovněž k uváznutí. V jednom případě se ve variantě B robot otočil špatným směrem, nicméně byl schopen se nakonec dostat do cílového bodu. Tato otočka se projevila významně na době trvání experimentu, viz tabulka 11.

Tabulka 11: Výsledky experimentu č. 4

Číslo průjezdu	Varianta prostředí	Počet primárních kolizí	Počet sekundárních kolizí	Počet uváznutí robota	Doba trvání experimentu
1	A	0	0	0	42s
2	A	0	1	0	50s
3	A	0	0	0	57s
4	B	0	0	0	1m 45s
5	B	0	0	0	42s
6	B	0	0	0	54s

5.2.1 Experiment č. 5

Posledním experimentem je volný pohyb robota v prostředí 195×195 cm, jež obsahuje množství překážek umístěných tak, aby tvořily navíc „místnosti“, ve kterých může robot potencionálně uváznout či způsobit kolizi pohybem uvnitř.



Obrázek 21: Prostředí č. 5

Experiment v prostředí číslo 5 probíhal po dobu 30 minut (3603 cyklů snímání prostředí). V průběhu experimentu došlo k celkem 10 sekundárním kolizím, žádné primární kolizi a žádnému uváznutí. Do horní levé oblasti se robot dostal až po 7 minutách a 32 sekundách, do pravé dolní po 8 minutách a 41 sekundách od počátku spuštění experimentu. Robot prozkoumal celé sestavené experimentální prostředí.

K sekundární kolizi označené jako číslo 1 došlo celkem třikrát a to vždy po otočení z daného prostoru za překážkou a vyjždění ven, kdy robot zavadil pásy o roh překážky. Číslem 2 jsou označeny celkem 2 sekundární kolize snímačů při objíždění překážky zleva. Sekundární kolize číslo 3 nastala opět dvakrát a kolidovaly snímače při vyjždění z „místnosti“. Předposlední kolizí je kolize s označením číslo 4 (kolize snímače s rohem), jež nastala pouze jedenkrát v průběhu opouštění prostoru za překážkou. Poslední kolizí je číslo 5, která nastala dvakrát – jednou kolidoval pás při objíždění vlevo a po druhé snímač při otáčení.

V průběhu experimentu byla pozorována schopnost robota couvat. Couvání probíhalo vždy jen po krátkou dráhu v případě, že robot neměl příliš prostoru k otočení na místě. Tato schopnost couvat vzad nebyla součástí trénovací množiny.

5.3 Shrnutí experimentů

V pěti prostředích s několika variantami rozmístění překážek bylo provedeno celkem 25 experimentů s využitím vytvořené aplikace. Jak lze vidět v tabulce 12, v žádném experimentu nedošlo ke kolizi primárního typu, tj. přímé najetí robota do překážky. Nicméně se objevily sekundární kolize způsobené, ve většině případů, hranou snímačů při otáčení robota. Kolize se vyskytovaly vždy u rohů překážek. Pravděpodobné vysvětlení, je že vyslaný infračervený paprsek daného snímače se nevhodně odrazil a došlo tak k chybnému měření.

V žádném z experimentů nedošlo k uvážnutí robota. Ačkoliv byla snaha potlačit chybu měření daných infračervených snímačů, rozdělením snímané oblasti do několika intervalů, byla přesto snímána chybná data. V konečném důsledku však tyto občasné chybné údaje ze snímačů měly pozitivní efekt na celkové chování robota. Proměnlivou změnou vstupů řídicí UNS i ve stavu setrvávání na stejném místě bylo docíleno toho, že se robot neustále pohyboval. Díky tomuto, i velmi nepatrnému pohybu, získal schopnost uvolnit se z potencionálního uvážnutí, jež by jinak nastalo.

Tabulka 12: Shrnutí experimentů

Číslo prostředí	Varianta prostředí	Max. počtu primárních kolizí	Max. počtu sekundárních kolizí	Max. počtu uvážnutí robota	Průměrná doba trvání experimentu
1	A	0	1	0	17,7s
1	B	0	1	0	28,0s
1	C	0	2	0	25,0s
2	A	0	1	0	24,0s
2	B	0	1	0	36,3s
3	-	0	0	0	47,3s
4	A	0	1	0	49,7s
4	B	0	0	0	67,0s
5	-	0	10	0	1 800,0s

V posledním experimentu se ukázala zajímavá schopnost robota couvat v případě nedostatečného místa pro otočení. Vzor, jenž by popisoval toto specifické chování, tj. couvání dozadu, trénovací množina řídicí UNS neobsahovala. Robot couval vždy jen krátkou vzdálenost a nehrozila tak srážka s překážkou za ním.

- Rychlost jednotlivých motorů byla upravována vhodně, vzhledem k dané situaci, v níž se robot nacházel.
- Efektivně se robot pohyboval ve větších prostorech, kde nebyla potřebná přílišná přesnost pohybu.
- Úzké prostory neumožňovaly, vzhledem k absenci couvání, robotovi se vhodným způsobem otáčet. Docházelo tak k sekundárním kolizím snímačů či pásů.
- Bylo pozorováno, že robot je schopen se pohybovat rovněž v těsné blízkosti překážky, aniž by do ní narazil.

Reálná podoba jednotlivých prostředí je zdokumentována fotografiemi v příloze. Stejně tak jsou přiloženy i záznamy průběhu jednotlivých experimentů, které generuje vytvořená aplikace. Rovněž je součástí přílohy i demonstrační video pohybu robota.

ZÁVĚR

Cíl práce, návrh metody vycházející z umělých neuronových sítí, která bude řídit robota v reálném čase, byl splněn včetně všech dílčích cílů.

Na základě provedené analýzy problematiky řízení autonomního robota z pohledu navigace a řešerše problematiky řízení mobilního robota s využitím neuronových sítí byla vytvořena aplikace EV3 Neural Network Controller pro řízení modelu autonomního robota. Z technických důvodů aplikace komunikuje s robotem vzdáleně.

Jako metoda umělé inteligence byla zvolena dopředná třívrstvá neuronová síť s adaptačním algoritmem Back-Propagation. Vybraná úloha vyhnutí se robota překážkám v prostoru byla úspěšně implementována. Experimenty prokázaly, že pro řešení problému je dostačující síť s jednou vnitřní vrstvou, což je v souladu s [20].

Dvou motorový pásový model autonomního robota se čtyřmi infračervenými snímači je sestaven ze stavebnice Lego Mindstrom EV3. Robot využívá operační systém LeJOS a komunikace mezi aplikací a robotem probíhá v síti PAN prostřednictvím technologie Bluetooth.

Vytvořená aplikace umožňuje ovládání libovolného robota se dvěma motory a čtyřmi infračervenými snímači sestaveného ze stavebnice Lego Mindstrom EV3 s operačním systémem LeJOS. Dále aplikace uživateli poskytuje možnost vytvoření umělé neuronové sítě, tj. návrh topologie a parametrů adaptace, generování trénovací množiny s variabilním nastavením a samotnou adaptaci vytvořené sítě. Rovněž lze nastavit parametry experimentu a jeho spuštění - autonomní mód robota. V neposlední řadě aplikace obsahuje funkcionalitu filtrování vygenerované trénovací množiny. Pro filtraci trénovací množiny je využita ART1 neuronová síť s modifikovaným algoritmem učení. Díky filtraci trénovací množiny je možné významně redukovat počet vzorů, které pak může uživatel snadněji upravovat a definovat tak chování robota.

Využití heuristiky samo-organizujících sítí (ART1) pro předzpracování trénovací množiny umožňuje ve vygenerované množině nalézt typické vzory a pokrýt tak celý prostor řešení. Zároveň je využita schopnost generalizace neuronové sítě, snížení možnosti zanesení uživatelské chyby a dosažení celkově lepších výsledků oproti trénovací množině úplné.

Pro ověření chování modelu autonomního robota, resp. navržené neuronové sítě adaptované vygenerovanou a filtrovanou trénovací množinou, byla vytvořena sada experimentů a stanoveny hodnotící kritéria. Robot splnil všechny cíle definované v jednotlivých experimentech bez uvážnutí v některém místě experimentálního prostředí či přímé kolize s překážkou. Nicméně se v průběhu experimentů vyskytovaly kolize některých částí robota, jimiž zavadil o překážky. Ve většině případů se tak dělo u rohů překážek při otáčení, kdy robot zavadil rohem některého z bočních snímačů či pásem. Každý průběh experimentu je zaznamenán a okomentován.

Největším omezením pro robota je nepřesnost použitých infračervených snímačů. Tato nepřesnost byla částečně potlačena rozdělením snímané oblasti daného snímače do několika intervalů. Použití přesnějších snímačů by umožnilo robotovi daleko lépe se orientovat v prostoru. Problémem aplikace je pak uživatelsky poměrně složité nastavení vyžadující preciznost a jistou míru pochopení problematiky.

RESUMÉ

Tato práce představuje metodu řídicí autonomního robota s využitím umělé neuronové sítě. Práce je členěná do pěti kapitol. Kapitola první popisuje problematiku řízení autonomního robota z pohledu navigace. V druhé kapitole se práce zaměřuje na současné mobilní roboty řízené neuronovými sítěmi a popisuje vybrané projekty. Kapitola třetí představuje navržený model autonomního robota sestaveného z Lego Mindstorm EV3 řešící problém vyhnutí se překážkám v prostoru. Dále pak kapitola popisuje navrženou řídicí neuronovou síť robota, metodu pro generování trénovací množiny, metodu využívající heuristiky samoorganizujících sítí pro filtraci trénovací množiny pomocí adaptivní rezonanční teorie a vliv parametru vigilance na filtrování trénovací množiny. Ve čtvrté kapitole je popsána vytvořená aplikace umožňující návrh řídicí neuronové sítě robota, generování a filtraci trénovací množiny, umožňující nastavení experimentu a jeho spuštění. V poslední páté kapitole je představena navržená trénovací množina, definována sada experimentů s robotem v několika experimentálních prostředích a zhodnoceny výsledky těchto experimentů.

SUMMARY

This thesis presents the method of controlling an autonomous robot using artificial neural networks. The thesis is composed of five chapters. Chapter One describes controlling issues from the perspective of an autonomous robot navigation. Chapter Two concentrates on contemporary mobile robots controlled by the neural networks and describes the selected projects. Chapter Three presents the proposed model of autonomous Lego Mindstorm EV3 robot solving the problem of avoiding obstacles in space. Furthermore, the chapter describes the proposed controlling neural network, the method for generating a training set, the method uses heuristic of the self-organising networks for filtering the training set using adaptive resonance theory and the impact of the vigilance parameter to filtering training sets. Chapter Fourth describes the application for the design of the control neural network, the generation and filtration of the training set and allowing experiment settings and launch. The final Chapter Five presents the proposed training set, the defined set of experiments with the robot in several experimental environments and assessed the results of these experiments.

SEZNAM POUŽITÉ LITERATURY

- [1] **Dudek, Gregory a Jenkin, Michael.** *Computational principles of mobile robotics.* Cambridge : Cambridge university press, 2010. 978-0521692120.
- [2] **Matarić, J., Maja.** *The robotics primer.* London : Mit Press, 2007.
- [3] **Alami, Richard, a další, a další.** An architecture for autonomy. *The International Journal of Robotics Research.* 17, 1998.
- [4] **Peréz, Carreras.** *A proposal of a behavior-based control architecture with reinforcement learning for an autonomous underwater robot.* University of Girona : Department of Electronics, Informatics and Automation, 2003.
- [5] **Hug, Rajibul, Mann, K.I., George a Gosine, G., Raymond.** Mobile robot navigation using motor schema and fuzzy context dependent behavior modulation. *Applied Soft Computing.* 8, 2008.
- [6] **Yang, Xiaoyu, Moallem, Mehrdad a Patel, V., Rajnikant.** A layered goal-oriented fuzzy motion planning strategy for mobile robot navigation. *Systems, Man, and Cybernetics.* 35, 2006.
- [7] **Brooks, Rodney.** A robust layered control system for a mobile robot. *Robotics and Automation.*
- [8] **Arkin, C., Ronald.** *Behavior- Based robotics.* Cambridge : MIT press, 1998.
- [9] **Bartoň, Adam.** *Emergence chování autonomního robota.* Ostrava : Ostravská univerzita v Ostravě, Bakalářská práce, 2014.
- [10] **Rojas, Raúl.** *Neural Networks: A Systematic Introduction.* Berlin : Springer, 1996. 978-3540605058.
- [11] **Mařík, Vladimír, Štěpánková, Olga a Lažanský, Jiří.** *Umělá inteligence.* Praha : Academia, 1997. 80-200-0496-3.
- [12] **Tripathi, G. N. a Rihani, V.** *Motion Planning Of an Autonomous Mobile Robot Using Artificial Neural Network.* arXiv preprint arXiv : 1207.4931, 2012.
- [13] **Kim, P. K. a Jung, S.** Experimental Studies of Neural Network Control for One-Wheel Mobile Robot. *Journal of Control Science and Engineering.* 2012, 2012: 3.
- [14] **Markoski, B. a al, et.** Mobile robot control using self-learning neural network. *Intelligent Systems and Informatics.* 2009. SISY'09. 7th International Symposium on. IEEE, 2009. p. 45-48.
- [15] **Farooq, U. a al, et.** Design and Implementation of Neural Network Based Controller for Mobile Robot Navigation in Unknown Environments. *International Journal of Computer & Electrical Engineering.* 2014, 6.2: 83-89.
- [16] **Reynoso, J. S. C.** A neural network for Java Lego robots. Learn to program intelligent Lego Mindstorms robots with Java. *Javaworld.* [Online] 16. 4. 2005. [Citace: 21. 11. 2015.] <http://www.javaworld.com/article/2071879/enterprise-java/a-neural-network-for-java-lego-robots.html>.
- [17] **Black, L.** A Worm's Mind In A Lego Body. *I Programmer.* [Online] 16. 11. 2014. [Citace: 22. 11. 2015.] <http://www.i-programmer.info/news/105-artificial-intelligence/7985-a-worms-mind-in-a-lego-body.html>.
- [18] LEGO.com Education 45544 – LEGO® MINDSTORMS® Education EV3 Core Set. *LEGO.com Education.* [Online] [Citace: 26. 1 2016.] <https://education.lego.com/cs-cz/lego-education-product-database/mindstorms-ev3/45544-lego-mindstorms-education-ev3-core-set>.
- [19] LeJOS, Java for Lego Mindstorms / EV3. *LeJOS.* [Online] [Citace: 26. 1 2016.] <http://www.lejos.org/ev3.php>.

- [20] **Masters, T.** *Practical Neural Network Recipes in C++*. San Francisco : Morgan Kaufmann, 1993. 978-0-12-479040-7.
- [21] **Wells, Richard B.** Introduction to Biological Signal Processing and Computational Neuroscience. *University of Idaho*. [Online] 2010. [Citace: 2. 6 2016.] <http://www.mrc.uidaho.edu/~rwwells/techdocs/Biological%20Signal%20Processing/>.
- [22] **Kriesel, David.** A brief introduction to neural networks. [Online] 2007. [Citace: 28. 1 2016.] http://www.dkriesel.com/en/science/neural_networks.

SEZNAM POUŽITÝCH SYMBOLŮ

UNS	Umělá neuronová síť.
NS	Neuronová síť.
BP	Back-Propagation.
DSP	Digitální signálový procesor.
GSM	Globální systém pro mobilní komunikaci.
GPS	Globální polohovací systém.
RAM	Paměť s přímým přístupem.
USB	Universal Serial Bus.
LCD	Liquid Crystal Display.
SDHC	Secure Digital High Capacity.
TM	Trénovací množina.
RTM	Redukovaná trénovací množina.
ERTM	Redukovaná trénovací množina s exkludovanými nulovými skupinami.
PAN	Personal Area Network
IP	Internet Protocol

SEZNAM OBRÁZKŮ

Obrázek 1: Centralizované řízení	15
Obrázek 2: Architektura A (behavior-based overall arch.).....	16
Obrázek 3: Architektura B (subsumption arch.).....	17
Obrázek 4: Architektura C (motor schema arch.).....	17
Obrázek 5: Hybridní architektura	17
Obrázek 6: Formální neuron	20
Obrázek 7: Sestavený robot - pohled shora	29
Obrázek 8: Sestavený robot - pohled ze strany.....	29
Obrázek 9: Řídící neuronová síť	30
Obrázek 10: Snímaná oblast	33
Obrázek 11: Životní cyklus TM.....	35
Obrázek 12: ART1	36
Obrázek 13: Aplikace pro řízení EV3.....	41
Obrázek 14: Nastavení experimentu.....	43
Obrázek 15: Spuštěný experiment	44
Obrázek 16: Snímaná oblast	47
Obrázek 17: Prostředí č. 1, zleva varianta A, B, C	51
Obrázek 18: Prostředí č. 2, zleva varianta A, B.....	53
Obrázek 19: Prostředí č. 3.....	54
Obrázek 20: Prostředí č. 4, shora varianta A, B	55
Obrázek 21: Prostředí č. 5.....	56

SEZNAM TABULEK

Tabulka 1: Porovnání vlastností architektury	19
Tabulka 2: Vstupní vzory TM	33
Tabulka 3: Vstupní vzory RTM.....	34
Tabulka 4: Vstupní vzory ERTM	34
Tabulka 5: Vliv parametru vigilance	39
Tabulka 6: Počet skupin ERTM v závislosti na parametru vigilance	48
Tabulka 7: Trénovací množina pro adaptaci UNS.....	49
Tabulka 8: Výsledky experimentu č. 1	52
Tabulka 9: Výsledky experimentu č. 2	53
Tabulka 10: Výsledky experimentu č. 3	54
Tabulka 11: Výsledky experimentu č. 4	55
Tabulka 12: Shrnutí experimentů	57

SEZNAM PŘÍLOH

Příloha č. 1 – CD/DVD s aplikací, dokumentací a soubory experimentů.