UNIVERZITA KOMENSKÉHO V BRATISLAVE

FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

Myš v bludisku s počítačovým videním

Bakalárska práca

2012

Daniel Skalický

UNIVERZITA KOMENSKÉHO V BRATISLAVE

FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

Myš v bludisku s počítačovým videním

Bakalárska práca

Študijný program:	Aplikovaná informatika
Študijný odbor:	9.2.9. Aplikovaná informatika (2511)
Školiace pracovisko:	Katedra aplikovanej informatiky
Školiteľ:	Mgr. Pavel Petrovič, PhD.
Evidenčné číslo:	b133c491-e5bc-46aa-97e2-2982fff375cb

Bratislava, 2012

Daniel Skalický

58168525

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Daniel Skalický
Študijný program: aplikovaná informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: 9.2.9. aplikovaná informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský

Názov: Myš v bludisku s počítačovým videním

Cieľ: Cieľom práce je navrhnúť a implementovať riadiaci systém pre robota do súťaže Istrobot, kategória Myš v bludisku. Robot je vybavený rôznymi senzormi a počítačovým videním, ktoré mu významne uľahčí navigáciu a mapovanie. Študent v práci preskúma relevantné algoritmy počítačového videnia, zvolí vhodný a empiricky svoj výber overí a v práci vyhodnotí.

Literatúra: 1. Gálik M., Koyšová Z.: Mouse in Maze with robot E-Puck, 2011. 2. Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.-C., Floreano, D. and Martinoli, A. (2009) The e-puck, a Robot Designed for Education in Engineering. Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions, 1(1) pp. 59-65.

Kľúčové slová: robotika, počítačové videnie, istrobot, e-puck

Vedúci: Mgr. Pavel Petrovič, PhD.

Spôsob sprístupnenia elektronickej verzie práce:
bez obmedzenia

Dátum zadania: 17.10.2011

Dátum schválenia: 20.10.2011

doc. RNDr. Mária Markošová, PhD.
garant študijného programu

Skalický
študent

Pavel Petrovič
vedúci

Poďakovanie

Týmto by som sa chcel poďakovať školiteľovi mojej bakalárskej práce Mgr. Pavlovi Petrovičovi, PhD. za cenné rady a usmernenia, ochotné konzultácie a motivujúce vedenie počas tvorby tejto bakalárskej práce.

Čestné prehlásenie

Čestne prehlasujem, že túto bakalársku prácu som vypracoval samostatne s použitím uvedenej literatúry.

….....

Bratislava 1.6.2012 Daniel Skalický

1. Abstrakt

SKALICKÝ, Daniel: *Myš v bludisku s počítačovým videním* [bakalárska práca]. Univerzita Komenského v Bratislave. Fakulta matematiky, fyziky a informatiky. Katedra aplikovanej informatiky. Odbor Aplikovaná informatika. Školiteľ: Mgr. Pavel Petrovič, PhD. Bratislava: Fakulta matematiky, fyziky a informatiky UK, 2012. 52 str.

Táto bakalárska práca pojednáva o riešení robotickej súťažnej úlohy Myš v bludisku s použitím reálneho výukového robota E-puck. Využíva metódy odometrie pre navigáciu, lokalizáciu a tvorbu pravdepodobnostnej mapy bludiska. Aplikuje algoritmus následovania pravej, resp. ľavej steny pre vyriešenie bludiska, ako aj upravený Flood-fill algoritmus pre efektívne nájdenie cieľa a nájdenie najkratšej cesty bludiskom zároveň. Prispôsobuje a aplikuje metódy počítačového videnia pre účely doprednej analýzy prostredia pred robotom. Robot spolu s dovtedy vytvoreným softvérom sa zúčastnil robotickej súťaže ISTROBOT 2012, ktorá otestovala jeho schopnosti oproti ostatným súťažiacim. Táto práca vylepšuje doterajšie výsledky robota E-puck v úlohe Myš v bludisku a otvára a testuje ďalšie, doteraz nevyužité možnosti robota v tejto úlohe. Práca zároveň ostáva otvorená pre ďalšie pokračovanie a zlepšovanie, o ktoré má jej autor úprimný a osobný záujem.

Kľúčové slová: robotika, počítačové videnie, myš v bludisku, E-puck, ISTROBOT

1. Abstract

SKALICKÝ, Daniel: *Mouse in maze with computer vision* [bachelor thesis].
Comenius University in Bratislava. Faculty of mathematics, physics and informatics. Department of applied informatics. Field: Applied Informatics. Supervisor: Mgr. Pavel Petrovič, PhD. Bratislava: Faculty of mathematics, physics and informatics, 2012. 52 pages.

This bachelor thesis disserts upon a solution of a robotics contest's discipline Mouse in maze (also referred to as Micromouse), with the use of a real educational robot, E-puck. It uses the methods of odometry, for the purposes of navigation and localization of the robot, as well as for creating a probabilistic map of the maze. It applies the algorithm of following the left or the right hand wall to find the goal of the maze, as well as the modified Flood-fill algorithm to effectively find the goal and the shortest route in the same time. It modifies and applies the methods of computer vision for the purpose of forward environment analysis, in front of the robot. The robot, together with the (to that moment) implemented program, took part in the ISTROBOT 2012 robotics contest, which tested it's abilities against other contestants. This work improves previous results of the E-puck robot in the Mouse in maze task, and opens and tests other, not yet used possibilities of the robot in this task. The work remains open to be continued and improved, on what the author of this work has an honest and personal interest.

Keywords: robotics, computer vision, mouse in maze, E-puck, ISTROBOT

2. Predhovor

V dnešnej dobe patrí robotika medzi rozšírené, a zároveň aplikované odvetvia. S robotikou sa dá stretnúť na rôznych miestach a pri rôznych aplikáciach. Na akademickej pôde slúži predovšetkým na vzdelávacie a výskumné účely, v priemysle sa výrobná hala bez robotickej automatizácie takmer nezaobíde. V domácnosti roboty odbremeňujú človeka od jednoduchých činností alebo mu slúžia ako zaujímavé hobby plné výziev a súťaží na rôznych odborných úrovniach.

Narazíme však aj na veľmi špecifické a zložité aplikácie robotiky a umelej intelgencie, napríklad vo vesmírnom výskume, pri vojenských, bezpečnostných či záchranných účeloch, alebo pri činnostiach, ktoré človek ako živý organizmus nedokáže plniť efektívne, či dokonca nemôže vykonávať vôbec. Vo svete informatiky a informačných technológií je preto dôležité venovať sa aj robotike, nielen z technického, ale aj z informatického a programátorského uhla pohľadu.

Táto práca sa snaží analyzovať a vysvetliť základnú problematiku robotiky, počítačového videnia, navigácie, mapovania a lokalizácie robota v čiastočne neznámom prostredí, pre účel riešenia optimalizačného problému Myš v bludisku. Ďalej sa pokúsi navrhnúť a implementovať riešenie pre daný problém súťažnej úlohy Myš v bludisku za použitia reálneho výukového robota E-puck.

Robot sa počas tvorby tejto práce zúčastní aj na robotickej súťaži ISTROBOT 2012, kde bude mať možnosť otestovať svoje schopnosti s druhými robotmi a súťažiacimi. Účasť na robotickej súťaži zároveň poskytne príležitosť na zhodnotenie a vyvodenie záverov ku dovtedajšiemu návrhu a implementácií riešenia, ako aj príležitosť na vylepšenie návrhu a implementácie.

3. Obsah

1. Úvod

1.1 Motivácia práce

Robotiku ako tému tejto práce som si vybral z niekoľkých dôvodov. V prvom rade preto, že sa priamo dotýka aplikovanej informatiky a rozširuje ju o ďalšie aspekty. Okrem informatických problémov a úloh dáva možnosť stretnúť sa aj s výpočtovou elektronikou a elektrotechnikou, ktorá dodnes stála a ešte stále stojí aj za najzložitejšími výpočtovými zariadeniami.

Ďalším dôvodom a výzvou je fakt, že robotika spája informatiku a výpočtové technológie s reálnym a spojitým svetom, a dáva tak možnosť počítaču interagovať s realitou. Ako som zo skúseností získaných počas tejto práce zistil, táto úloha nie je jednoduchá. Fyzický, spojitý a do veľkej miery nepredvídateľný svet skrýva mnohé úskalia pre diskrétne a deterministické zariadenie, akým je model dnešného počítača.

1.2 Ciele práce

Cieľom tejto práce je analyzovať základnú problematiku robotiky a počítačového videnia pre účely súťažnej úlohy Myš v bludisku, zhromaždiť a zhodnotiť východiská pre túto úlohu, špecifikovať, navrhnúť a implementovať riešenie reálneho robota. Cieľom je aj otestovať implementáciu a vyvodiť závery a návrhy na zlepšenie a pokračovanie práce, ako aj účasť na súťaži ISTROBOT 2012.

1.3 Štruktúra práce

Kapitola Analýza a východiská práce obsahuje analýzu problematiky potrebnej pre tvorbu tejto práce, ako aj zhromaždené a zhodnotené východiská a existujúce podobné projekty.

Kapitoly Špecifikácia požiadaviek na softvér a Návrh softvérového riešenia sa z navrhujú softvérovú aplikáciu pre robota E-puck, ktorá rieši problém Myš v bludisku.

Implementácia na reálnom robotovi a Dosiahnuté výsledky a ciele pojednávajú o procese tvorby a testovania aplikácie, ktorá vznikla podľa návrhu z predošlých častí práce a o výsledkoch implementácie a súťaže ISTROBOT.

V časti Záver je zhrnutá a zhodnotená celá práca ako jeden celok.

2. Analýza a východiská práce

V tejto kapitole sa približujem východiská a teoretickú základňu, ktorú som pri tvorbe celej práce používal.

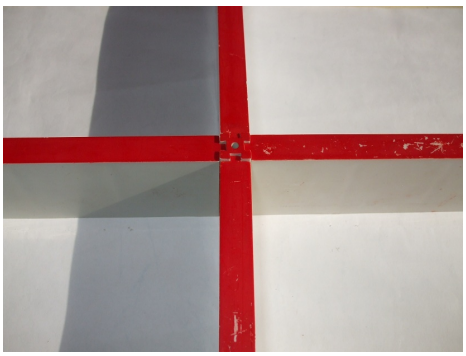
2.1 Súťaž ISTROBOT – definícia úlohy robota

ISTROBOT je robotická súťaž, ktorá sa koná každoročne v jarných mesiacoch od roku 2000 na Fakulte elektrotechniky a informatiky STU. Súťažiaci sa môžu zapísať do niekoľkých disciplín, ako napríklad Stopár (sledovanie čiary a prekonávanie prekážok na trati), Myš v bludisku (robot musí autonómne prejsť neznámym bludiskom zo štartu do cieľa za čo najkratší čas), Voľná jazda (predvádzanie kurióznych a unikátnych robotov) a na tohtoročné novinky – V sklade kečupu (dvaja roboti sa snažia ukoristiť čo najviac plechoviek rajčinového pretlaku) a Freescale race challenge (pretekanie autonómnych robotických áut na predom neznámej trati). Súťaž je otvorená pre všetky vekové kategórie, nemá obmedzenia pre účastníkov, iba spoločné pravidlá a pravidlá jednotlivých disciplín.

2.1.1 Úloha Myš v bludisku

Prostredie, v ktorom bude robot v rámci tejto práce pracovať a pre ktoré bude jeho program prispôsobený, má podľa [2] nasledovné vlastnosti.

Súťažné bludisko pre kategóriu Myš v bludisku sa podľa pravidiel súťaže ISTROBOT skladá z bielych stien, ktoré odrážajú infračervené žiarenie, zvuk, a na ich vrchnej strane sa nachádza červený náter; podlaha je čierna a drevená (alebo z podobného materiálu) a infračervené žiarenie naopak pohlcuje. Steny sú na podlahu pripevnené pomocou mrežových bodov, teda plastových prvkov bludiska, ktoré spájajú jednotlivé steny. Vždy po 90 stupňoch sa môže na mrežový bod napojiť ďalšia stena. Tieto mrežové body sú ukotvované do podlahy na presne upravených miestach, ktoré tvoria mriežku bludiskových buniek.



Obrázok č.1 *Mrežový bod bludiska*

Podľa súťažných pravidiel je bludisko zostavené tak, že okrem jednoho mrežového bodu vychádza z každého iného vždy aspoň jedna stena. Tento jeden vynechaný bod vytvorí miestnosť v bludisku s rozmerom 2 x 2 políčka, ktorá predstavuje cieľ bludiska. Bludisko je po obvode uzatvorené stenami a má tvar štvorca.

Súťažné bludisko môže mať rozmer 9 x 9 alebo 16 x 16 buniek. V prvom prípade sa cieľ nachádza v rohu bludiska, ktorý je po uhlopriečke oproti počiatočnému políčku, v druhom prípade sa cieľ nachádza presne v strede bludiska. Robot štartuje v jednom z rohov bludiska.

Stena bludiska má dĺžku 18 cm, šírku 1.2 cm a výšku 5 cm. Chodba, ktorá vznikne umiestnením takýchto stien do mriežky bludiska je teda široká 16.8 cm. Rozmery bludiska počítajú s 5% odchýlkou.

V bludisku existuje viac ciest zo štartovného do cieľového políčka, vchod do cieľovej miestnosti je však len jeden. Cesty nemusia byť rovnako dlhé. Vždy existuje aj cesta zo štartu do cieľa podľa pravidla sledovania pravej alebo ľavej steny bludiska, tieto ale nemusia byť rovnako dlhé a spravidla nie sú najkratšie.

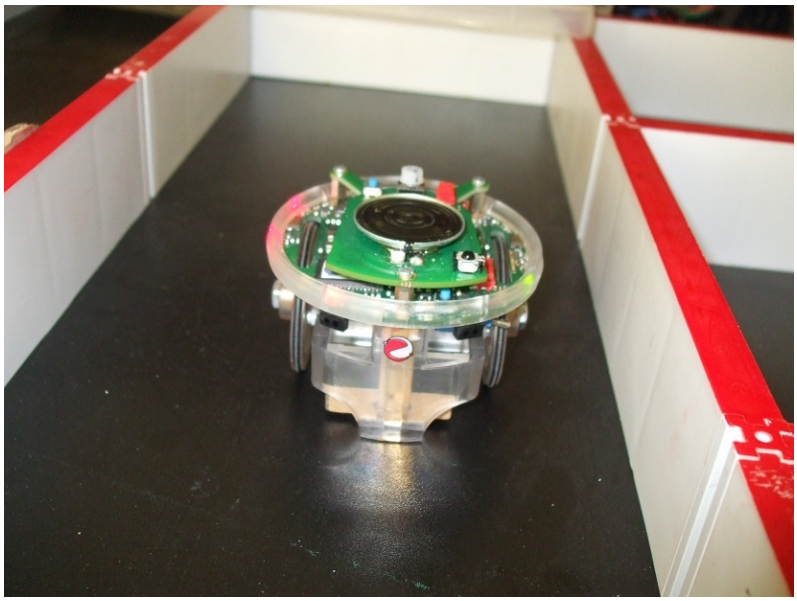
Robot má počas súťaže časové obmedzenie maximálne 5 minút v bludisku. Za tento čas sa môže najviac 10-krát pokúsiť prejsť cestu štart-cieľ. Tieto prechody sú označené ako súťažné pokusy a najkratší čas súťažného pokusu je rozhodujúcou mierkou úspešnosti robota. Súťažný pokus sa začína merať od opustenia štartovného políčka a končí sa vchodom do cieľovej miestnosti. Po dokončení takejto trasy sa robot môže samostatne vrátiť na cieľ a pokúsiť sa nájsť výhodnejšiu cestu alebo ho môže premiestniť súťažiaci za cenu dotykovej penalizácie.

Ak má robot v bludisku problémy, napríklad keď sa zasekne o prekážku (nie však keď urobil chybné rozhodnutie, napríklad na križovatke), súťažiaci mu môže pomôcť za cenu dotykovej penalizácie.

Dotyková penalizácia má hodnotu +3 sekundy ku času pokusu, v prípade prenesenia robota na štart je penalizácia platná aj pre všetky nasledujúce súťažné pokusy.

2.2 Výukový robot E-puck

E-puck je výukový robot, ktorý bol použitý pri implementácií riešenia tejto práce. Jeho návrh začal na Ecole Polytechnique Fédérale de Lausanne v roku 2004, kedy vzniklo niekoľko kusov prvej verzie robota. V roku 2006 bola dokončená druhá verzia robota. Pri jeho charakteristike vychádzam z [7].



Obrázok č.2 *Robot E-puck v bludisku*

2.2.1 Konštrukcia

Robot má konštrukciu kruhového pôdorysu s priemerom 70 mm a výškou 50 mm. Jeho telo tvorí jednoliata plastová kostra, na ktorej sú uchytené ostatné funkčné, mechanické a elektronické časti. V spodnej časti robota sa nachádza batéria, tesne nad ňou sú umiestnené motory a kolesá, vo vrchnej časti sa nachádzajú plošné obvody.

Táto základná konštrukcia sa dá rozšíriť o prídavné veže alebo nadstavby, ktoré pridávajú napríklad ďalšie senzory pre robota (napríklad kompas, ktorý by mal byť

umiestnený ďalej od zdrojov elektrického rušenia robota).

Robot má dve kolesá zabezpečujúce pohyb, ktoré sa otáčajú po rovnakej pevnej osi otáčania, nie sú však prepojené a každé je poháňané separátnym motorom. Motory používajú pevnú prevodovú redukciu 50:1. Vzhľadom ku konštrukcí tejto pohonnej sústavy sa okrem kolies robota dotýka podlahy vždy aj jeden z koncov robota.

2.2.2 Mikroprocesor

Robot používa 16 bitový mikroprocesor dsPIC30F6014A firmy Microchip. Tento mikroprocesor má podľa [13] následovné vlastnosti:

Názov vlastnosti	**Hodnota pre dsPIC30F6014A**
Taktovacia frekvencia (interný oscilátor)	7,37 MHz, 512kHz
Programová pamäť	144 kB (Flash)
Operačná pamäť	8 kB (RAM)
EEPROM pamäť pre firmvér	4 kB
Časovače	5 16 bitových časovačov
Periférna digitálna komunikácia	2 UART moduly, 2 SPI a 1 I2C zbernice
Analógová komunikácia	1 A/D prevodník, 16 * 12 bitov s vzorkovaciou frekvenciou 200 kHz

Tabuľka č.1 *Vlastnosti mikroprocesora dsPIC30F6014A*

2.2.3 Senzory

E-puck má široké spektrum senzorov, ktoré snímajú rôzne vlastnosti okolitého prostredia. Základná senzorická výbava robota je súčasťou hlavného plošného obvodu, ďalšie senzory sa dajú pripojiť pomocou prítomných portov elektronických rozhraní.

Jedným zo senzorov robota sú infračervené detektory vzdialenosti prekážok, v prípade robota E-puck ide o model senzorov TCRT1000. Sú rozmiestnené po celom obvode robota, nie však pravidelne – 6 z celkových 8 senzorov sa nachádza v prednej polovici robota a len zvyšné 2 senzory smerujú dozadu. Tieto senzory majú obmedzenú detekčnú vzdialenosť a takisto obmedzenú presnosť aj spoľahlivosť. Ako som sa presvedčil z informačných hárkov tohto hardvéru [17] a aj zo skúšobných programov a starších verzí softvéru, ktorý je popísaný v ďalšej kapitole, na absolútnu presnosť sa nedá spoľahnúť. Hlavné faktory, ktoré ovplyvňujú dáta zo senzorov, sú:

- okolité osvetlenie (alebo iné zdroje rušivého infračerveného žiarenia),
- výrobné odlišnosti senzorov (niektoré senzory podávajú viditeľne odlišnú informáciu za rovnakých podmienok ako ostatné),
- elektornické rušenie iných častí robota (pohyb motorov a funkcia kamery majú zaznamenaný vplyv na výstup senzorov),
- prirodzená odchýlka senzorov, ktorá je však pre senzor za stabilných okolitých podmienok veľmi malá,

Ďalším senzorom je CMOS VGA farebná kamera, ktorá je umiestnená v prednej časti robota a smeruje čelne vpred. Jej plné rozlíšenie je 640 x 480 pixelov. Objem dát, ktoré kamera generuje pri takomto rozlíšení, sa ale nedá naraz spracovať procesorom robota ani uložiť v operačnej pamäti. S kamerou sa teda typicky pracuje použitím menších výrezov celkovej plochy snímača, s podvzorkovaním výsledného obrazu alebo s použitím kompaktnejšieho čiernobieleho farebného modelu.

Okrem týchto dvoch podstatných senzorov má robot vstavané aj ďalšie – 3 všesmerové mikrofóny a 3D akcelerometer, k dispozícii je aj prídavný senzor pre sledovanie pozemnej čiary a kompas, ktorý však nemá plne funkčné elektronické rozhranie pre pripojenie ku portu robota a bolo by ho treba skonštruovať, resp. overiť a v prípade potreby upraviť existujúce.

2.2.4 Aktuátory

Pod týmto pojmom sa rozumejú všetky prvky robota, ktoré sú schopné vykonávať nejakú činnosť alebo akciu a ovplyvňovať tak okolité prostredie, alebo robota samotného.

Medzi hlavné, ktoré táto práca bude využívať, patria motory poháňajúce kolesá robota. Robot používa dva krokové elektromotory, každý poháňa samostatne jedno koleso. Princíp práce krokového motora spočíva v diskrétnom rotačnom pohybe po malých krokoch, ktoré sú definované mechanikou motora. Presnosť motorov, podobne ako presnosť infračervených senzorov nie je dokonalá, záleží najmä na ich korektnom používaní. Aj napriek tomu sa ale časom prejavia prirodzené odchýlky motorov

spôsobené ich mechanickou činnosťou a prevodovkou, napríklad pri snahe o presnú lokalizáciu a navigáciu robota, ako som zistil počas implementácie softvéru, ktorý táto práca navrhuje. Motory majú vstavanú pevnú prevodovku s prevodovým pomerom 50:1, maximálna technická rýchlosť robota je 13 cm/s čo predstavuje rýchlosť 1000 krokov za sekundu.

Okrem motorov sa do tejto skupiny funkčných prvkov robota môže zaradiť aj modem pre Bluetooth komunikáciu a schopnosť infračervenej dátovej komunikácia.

Robot disponuje množinou LED diód v nasledovnom zložení: 8 červených diód, ktoré sú umiestnené v hornej časti robota po jeho obvode, 1 celotelová zelená dióda (v skutočnosti je týchto diód niekoľko, sú však ovládané jedným výstupným portom) a 1 čelová oranžová dióda. Tieto prvky majú napríklad signalizačné využitie.

Robot môže prehrávať krátke zvukové sekvencie pomocou vstavaného reproduktora a hardvérového D/A prevodníka.

2.3 Prehľad vývojových nástrojov a softvéru

2.3.1 Vývojové nástroje

Na tvorbu, ladenie a testovanie softvéru pre robota E-puck existuje viac nástrojov, z ktorých niekoľko popisujem.

Pre platformu operačného systému Windows:

MPLAB IDE od firmy Microchip – ide o integrované vývojové prostredie pre programovanie mikroprocesorov od firmy Microchip. Aplikácia poskytuje funkcie pre písanie a zjednodušenie písania zdrojového kódu, ako v jazyku C, tak aj v Assembleri. Takisto obsahuje nástroje pre kompiláciu, linkovanie, archiváciu a zostavenie zdrojového kódu do konečného strojového kódu pre konkrétny procesor.

MPLAB C30 od firmy Microchip – balíček nástrojov pre MPLAB IDE, ktorý obsahuje kompilátor, linker, archiver a assembler na preklad zdrojového kódu pre rodinu procesorov dsPIC30f a iné. Tieto nástroje sú proprietárne, ale základná verzia, ktorá neobsahuje optimalizáciu (s cieľom zmenšenia objemu strojového kódu) je dostupná pre študentské účely.

Tiny PIC bootloader pre procesory firmy Microchip aj pre rodinu dsPIC30f.

Tento program má dve časti – jedna malá časť je natrvalo uložená v pamäti robota a je aktívna vždy pri spustení robota. Vykonáva bootavaciu činnosť – vždy pri spustení robota počká nastaviteľný čas, pričom čaká na komunikáciu druhej časti softvéru, ktorá je spustená na vývojovom počítači. Ak nič nezachytí, spustí vykonávanie používateľského programu uloženého v programovej pamäti robota.

Druhá časť softvéru je spustená na strane vývojového počítača ako jednoduchá oknová aplikácia. Môže spustiť nahrávanie nového programu do pamäti robota cez Bluetooth (na túto komunikáciu čaká prvá časť bežiaca v robotovi po jeho spustení či reštarte) alebo slúži ako terminál pre sériovú komunikáciu s používateľským programom, ktorý je spustený na robotovi.

Epuckmonitor – tento jednoduchý nástroj pre robota E-puck slúži na testovanie a sledovanie funkcie všetkých senzorov a aktuátorov robota. Jeho súčasťou je tiež program na strane robota, ktorý je súčasťou ukážkovej demoverzie pre robota – **demoGCtronic.** Tento nástroj sa dá použiť napríklad pre skoré získavanie obrazu z robota pre analýzu jeho vlastností pred návrhom a implementáciou vlastného programu.

Webots – táto aplikácia je tiež integrované vývojové prostredie pre rôzne robotické platformy a špecifické robotické účely. Aplikácia má poskytovať hlavne funkcionalitu spojenú so simuláciou vyvíjaného softvéru. Kvôli technickým problémom ju ale nebolo možné vyskúšať. Aplikácia má komplexnú licenciu, dá sa používať bez licencie s obmedzenou funkcionalitou, so skúšobnou, časovo obmedzenou licenciou na plnú funkcionalitu alebo s plne platenou licenciou na plnú funkcionalitu. [4]

Matllab s použitím programu **ePic2** – program ePic2 je rozšírenou formou nástroja Epuckmonitor. Pochádza od tvorcov návrhu robota E-puck, vyžaduje však prostredie aplikácie Matlab a operačnú platformu, ktorá poskytuje potrebné komunikačné funkcie. [7]

ASEBA – predstavuje osobitný prístup ku programovaniu robotov. Ide o skriptovací jazyk založený na spracovaní udalostí, multiplatformové integrované vývojové prostredie na strane vývojového počítača a virtuálny stroj na strane robota. ASEBA je distribuovaná architektúra pre viacprocesorové alebo kolektívne roboty. [8]

Pre platformu operačného systému Linux:

Piklab – zastáva funkciu integrovaného vývojového prostredia podobne ako MPLAB IDE. Sadu nástrojov pre kompiláciu, linkovanie a zostavenie zdrojového kódu treba osobitne zkompilovať podľa inštrukcií v danej distribúcií operačného systému. Výstupný strojový kód z Piklabu je možné nahrávať do robotov niekoľkými hardvérovými programátormi ako aj softvérovými bootloadermi, ako napríklad cez už spomenutý Tiny PIC bootloader. [7]

Epuckuploadbt – zdrojový C++ kód, ktorý pre kompiláciu vyžaduje zkompilovanú knižnicu libbluetooth. Umožňuje nahrávanie zdrojového kódu do robota E-puck. Podobne aj program **epuckupload**, ktorý je napísaný v Perle a nevyžaduje kompiláciu ani prítomnú zkompilovanú knižnicu libbluetooth a zastáva rovnakú funkciu. [7]

2.3.2 Softvérové knižnice

2.3.2.1 E-puck library

Táto open source knižnica poskytuje základnú softvérovú vrstvu nad hardvérom robota, uľahčuje teda vývoj softvéru od programovania hardvéru. Nie je predkompilovaná a treba ju kompilovať spolu s vyvíjaným softvérom.

Knižnica sa skladá z niekoľkých samostatných modulov, ktoré sú v málo prípadoch závislé na „nižších“ moduloch knižnice. Niektoré majú viac verzií, ktoré sa líšia napríklad technickým prevedením (použité časovače). Pri jej popise vychádzam z [9] a z jej zdrojového kódu.

E_motors – poskytuje funkcionalitu pre pohyb robota. Udržuje aktuálnu rýchlosť motorov, počet prejdených krokov, spravuje prerušenia, ktoré generuje časovač (časovače) a nastavuje výstupné porty motorov v správnych intervaloch, ktoré závisia na momentálnej rýchlosti.

S touto časťou knižnice som počas prvých verzií softvéru, ktorý navrhuje táto práca mal niekoľko problémov, ktoré súviseli s presnosťou pohybu. Ako som neskôr zistil, ich pôvod je práve v tomto module a sú popísané v stati 5.3.

E_led – jednoduchým spôsobom sprostredkováva funkcie ovládateľných LED diód robota. Diódy sú pripojené na 1 bitové výstupné porty a ovládajú sa spôsobom

zapnutá / vypnutá, čo reprezentuje nastavenie 0 / 1 na výstupný port príslušnej diódy.

E_ad_conv – modul pre analógovo-digitálne konverzie a vzorkovanie vstupných signálov zo senzorov: infračervené detektory vzdialenosti, mikrofóny, akcelerometer. Používajú ho iné časti knižnice.

E_prox – modul, ktorý sprostredkováva vstupy z IR senzorov vzdialenosti.

E_I2C_* – ide o časti knižnice, ktoré realizujú komunikačný protokol zbernice I2C, ktorou je pripojená napríklad kamera.

E_po* – ide o časti knižnice, ktoré riadia činnosť kamery. Obsahuje napríklad funkcie pre zapisovanie registrov kamery, prerušenie pre horizontálnu synchronizáciu, voľbu použitia správnych funkcií podľa hardvérovej verzie kamery (E-puck používa dve verzie kamery, staršia sa už na robota neosadzuje) a nakoniec používateľské funkcie pre nastavenie kamery a zachytenie obrazu.

E_uart_char – tento modul sprostredkováva obojsmernú bezdrôtovú sériovú komunikáciu cez Bluetooth modul robota.

2.3.2.2 Knižnica OpenCV

OpenCV (Open computer vision) je knižnica s otvorenou licenciou pre spracovanie obrazu. Knižnica je multiplatformová, primárne však určená pre stolné počítačové operačné systémy.

Poskytuje pokročilé funkcie pre predspracovanie, spracovanie a detekciu v obraze alebo vo videu, 3D rekonštrukciu obrazu či učiace algoritmy pre klasifikáciu, regresiu a klasterizáciu dát. [16]

Podľa dokumentácie sa knižnica dá použiť aj na operačnej platforme ROS (Robot operating system). E-puck však natívne nepoužíva túto operačnú architektúru. Je však možné použiť softvér, ktorý umožňuje prepojenie medzi architektúrou ASEBA (popísaná v stati 2.3.1) a operačnou platformou ROS.

Z tejto zložitosti prenosu knižnice na robota E-puck usudzujem, že jej použitie pre spracovanie obrazu v tejto práci nie je vhodné ani optimálne.

2.3.2.3 Knižnica FANN

Knižnica FANN (Fast artificial neural networks) je otvorená knižnica pre rýchlu a jednoduchú prácu s neurónovými sieťami. Knižnica dokáže vytvárať viacvrstvové neurónové siete, ktoré môžu byť husto aj riedko prepojené. Takisto obsahuje trénovacie algoritmy, napríklad Backpropagation či Evolving topology training. Výhodou je možnosť uložiť a načítať celé neurónové siete. [15]

Knižnica je určená pre stolové počítačové operačné platformy, ale dokumentácia uvádza multiplatformovosť. Knižnica je napísaná v jazyku C, čo by v ideálnom prípade umožnilo aj jej kompiláciu priamo vo vývojovom prostredí pre robota.

Napriek tomu, vzhľadom ku hardvérovým možnostiam robota predpokladám len obmedzené použitie tejto knižnice na robotovi E-puck. Pre spracovanie obrazu a jeho rozpoznanie a klasifikáciu by boli potrebné niekoľkonásobne väčšie výpočtové a pamäťové zdroje, než akými disponuje robot E-puck. Na jednoduché účely, napríklad na analýzu jednorozmerných vstupov by sa však knižnica dala použiť, avšak za predpokladu jej kompilovatelnosti pre robota.

2.4 Robotika

Pre riešenie robotickej úlohy Myš v bludisku, ako aj pre iné úlohy z oblasti robotiky je potrebné vybudovať si teoretickú základňu popisujúcu štandardné spôsoby a princípy riešenia niektorých čiastkových úloh.

Pre Myš v bludisku je nevyhnutné navrhnúť alebo prispôsobiť existujúce princípy navigácie a pohybu robota. Pre efektívne riešenie tejto úlohy, teda napríklad pokus o nájdenie najkratšej cesty bludiskom je potrebné vytvoriť model lokalizácie robota v prostredí, ako aj model pre tvorbu vnútornej mapy bludiska. Takisto je nutné navrhnúť model pre analýzu tejto mapy a pre plánovanie priechodu bludiskom. V neposlednej rade, pre zvýšenie efektivity robota v bludisku, síce už nie priamo z robotiky sa dajú využiť alebo prispôsobiť existujúce princípy z oblasti spracovania obrazu, s využitím palubnej kamery robota.

V ďalších podkapitolách sa venujem analýze týchto vybraných tém, ktorých základná znalosť predchádza návrhu riešenia.

2.4.1 Odometria

V tejto časti vychádzam z [1] [5] [19]. Odometria je technika navigácie používaná v robotike, ktorá spočíva v presnom zaznamenávaní prejdenej vzdialenosti a uhla natočenia robota oproti počiatočnému natočeniu. Odometria je prostriedok, ktorý umožňuje len odhad a je vždy spojená s určitou mierou odchýlky, ktorá sa vyjadruje percentuálne v pomere ku celkovej prejdenej vzdialenosti.

Pre vytvorenie takéhoto navigačného modelu je nevyhnutné splniť základnú podmienku – robot musí mať prostriedok, ktorým dokáže zaznamenávať vzdialenosť, ktorú prešli jeho kolesá, rovnako ako prostriedok pre sledovanie, prípadne zmenu ich natočenia alebo natočania robota ako celku (ak sa napríklad jeho kolesá otáčajú po premenlivých osiach otáčania).

Použitie krokových motorov nie je jediné riešenie, ktoré sa v robotike používa pre pohyb robota alebo jeho častí v prostredí. Mal som možnosť stretnúť sa s robotmi, ktoré používali upravené servomotory, ktoré sú schopné väčšej rýchlosti a výkonu ako krokové motory, ktoré používa napríklad robot E-puck. Existujú aj iné spojité spôsoby pohybu. Tieto však majú nevýhodu oproti krokovým motorom práve pri úlohe odometrie.

Krokové motory poskytujú prirodzené riešenie merania vzdialenosti – počet prejdených krokov sa dá softvérovo zaznamenávať a potom stačia jednoduché matematické výpočty na ich prevod do ľubovoľnej miery vzdialenosti.

Iné motory, ktoré používajú spojitú mechaniku pohybu, musia byť vybavené špeciálnym hardvérovým senzorom, tzv. enkóderom, ktorý zaznamenáva rotáciu kolesa. Tento senzor väčšinou funguje na optickom princípe – na koleso je napríklad pripevnený disk s potlačou, ktorá má rôznu infračervenú odrazivosť. Pri pohybe potom senzor zaznamenáva zmenu odrazivosti povrchu kolesa, a teda generuje impulz, ktorý odpovedá vzdialenosti na obvode kolesa.

Robot E-puck používa krokové motory, odometria je preto na hardvérovej úrovni podporovaná a pri implementácií riešenia sa treba sústrediť najmä na zabezpečenie dostatočnej presnosti, prípadne na algoritmus pre korekciu chyby odometrie.

2.4.2 Pohyb robota

Robot sa môže pohybovať rôznymi spôsobmi, ktoré umožňuje jeho konštrukcia a pohybová sústava. Spôsob, akým sa robot pohybuje v prostredí je väčšinou odvodený od úlohy, ktorú má v danom prostredí riešiť alebo od obmedzení jeho konštrukcie.

E-puck má dve kolesá, ktoré ležia na rovnakej pevnej osi otáčania. Toto usporiadanie nabáda ku priamočiaremu pohybu vpred, resp. vzad (pri takomto pohybe sa kolesá otáčajú rovnakou rýchlosťou a smerom), ku otáčaniu robota na mieste (čo ešte viac uľahčuje jeho kruhový pôdorys; pri takomto pohybe sa kolesá otáčajú rovnakou rýchlosťou, ale opačným smerom), ako aj ku plynulému otáčaniu robota za pohybu – otočenie robota po kružnici (pri takomto pohybe sa kolesá otáčajú nerovnakou rýchlosťou buď rovnakým, alebo opačným smerom). [19]

Posledný spôsob pohybu je z uvedených najkomplikovanšješí z implementačného a odometrického hľadiska, poskytuje ale väčšiu dynamiku pohybu, čo môže ušetriť čas a zvýšiť efektivitu.

2.4.3 Mapovanie prostredia

Vychádzam z [5] [6] [20]. Mapovanie prostredia mobilným robotom je komplexný problém, ktorý úzko súvisí s lokalizáciou robota. Existuje viac prístupov ku tvorbe mapy, ktorých voľba záleží najmä od účelu mapy.

Používajú sa dve typické reprezentácie mapy – maticová a topologická. Maticová reprezentácia predstavuje mapu ako dvoj (prípadne viac) rozmerný priestor, v ktorom sú umiestnené objekty mapy. Topologická reprezentácia uvažuje o miestach prostredia a ich vzájomných vzťahoch, ako napríklad ich vzdialenosť, a podobá sa tak na graf. Reprezentácia mapy môže znázorňovať prázdne priechodné miesto, nepriechodné miesto (prekážky) alebo obe naraz.

Tvorba mapy sa začína vždy rovnakým krokom – senzorickým perceptom okolitého prostredia. Robot pre tvorbu mapy musí byť schopný pozorovať prostredie vhodnými senzoromi. Najčastejšie používané senzory pre túto úlohu sú ultrazvukové, laserové alebo infračervené detektory vzdialenosti prekážok. Používajú sa aj mechanické detektory prekážok alebo je možné použiť kameru a spracovanie obrazu pre účely spojené s mapovaním prostredia.

Spomenuté senzory vnášajú do algoritmov nepresnosť. Preto aj mapovanie musí s týmto faktom počítať. Typické prístupy preto bývajú pravdepodobnostné, často spojené s určitým formálnym modelom alebo s robotickou umelou inteligenciou, ako napríklad Markovove probabilistické modely, Bayesovské programovanie robotov, či metódy Monte Carlo pre rozpoznávanie, učenie sa a pochopenie senzorických vzorov robotom.

2.4.4 Lokalizácia robota v prostredí

Vychádzam z [5] [11]. Lokalizácia robota býva spojená s navigáciou robota, resp. odometriou, ak zastáva rolu navigácie. Spája dohromady navigáciu a mapovanie, pretože spracováva navigačné dáta a na základe výpočtov aproximuje polohu robota na mape.

Lokalizácia môže byť prevedená priamočiarím prístupom, ako je napríklad použitie GPS prímača. Lokalizáciu v tomto prípade vykoná za robota pozičný systém GPS. Tento prístup sa však nedá použiť v interiéroch ani v GPS nepriestupných oblastiach a v priemernom prípade má odchýlku až niekoľko desiatok metrov.

Pri použití odometrie pre navigáciu robota, lokalizácia má matematické riešenie, ktoré spočíva v dead-reckoningu. Vstupom tejto metódy je posledná známa poloha a zaznamenané vzdialenosti a zmeny kurzu robota od tejto polohy. Aplikovaním geometrických a goniometrických vlastností trojuholníka, ktorý vznikne vhodnou reprezentáciou odometrických dát je možné vypočítať novú polohu robota.

Dlhodobá presnosť lokalizácie založenej na odometrií sa však nedá zaručiť, odometrické odchýlky sa akumulujú spolu s prejdenou vzdialenosťou a táto chyba sa priamo prenáša do lokalizácie a do tvorby mapy.

Ak sa robot pokúša rozpoznať svoje okolie senzoricky a spojiť si ho s miestom na známej mape, algoritmus sa stretá s problémom perceptuálnej indiferentnosti – niektoré miesta na mape môžu „vyzerať“ rovnako pri použití daného senzoru (napríklad pri použití senzorov pre detekciu vzdialenosti prekážok je nemožné vo štvorcovej miestnosti určiť, v ktorom z rohov sa robot práve nachádza, ak nepozná iné referenčné informácie či body).

2.4.5 Analýza mapy

Keď si robot vybuduje mapu prostredia, resp. keď svoje prostredie pozná dopredu, môže sa rozhodnúť ho analyzovať a využiť tak nazbierané informácie pre riešenie svojej úlohy.

V prípade úlohy Myš v bludisku je prostredie dopredu čiastočne známe – najmä jeho topológia a tvar, nie však konfigurácia prekážok prostredia.

Bludisko, v ktorom sa odohráva táto úloha sa dá analyzovať rôznymi prehľadávacími algoritmami. V prípade, že robot preskúmal celé bludisko, prehľadávaním do šírky sa dá nájsť najkratšia cesta, ktorá je dôležitá pre efektívne riešenie úlohy. Ak robot ešte nepozná celé bludisko a aj napriek tomu ho chce preskúmavať čo najefektívnejšie, môže použiť heuristické metódy a algoritmy prehľadávania, ktoré analýzou okolia robota v mape poskytnú rozhodovaciu informáciu o výhodnosti ďalších krokov robota. Robot si potom môže vybrať najlepší smer, ktorým sa bude ďalej pohybovať.

Príkladom takéhoto algoritmu je modifikovaný Flood-fill algoritmus [3]. Robot ohodnotí každé políčko mapy jeho manhattanskou vzdialenosťou od cieľa (pri úlohe myš v bludisku je pozícia cieľa známa) aj keď nepozná konfiguráciu stien. Robot sa potom naviguje pravidlom zmenšovania vzdialenosti, teda vždy na políčko s menšou vzdialenosťou od cieľa. Pri tomto procese si tiež značí pozície stien do mapy.

Keď sa robot dostane do situácie, v ktorej mu stena nedovolí pohyb na políčko s nižšou vzdialenosťou, prebehne aktualizácia ohodnotenia políčok – políčko, na ktorom sa robot práve nachádza, dostane ohodnotenie o jedna väčšie ako je minimálne ohodnotenie jeho otvorených susedných políčok (tie, ktoré neoddeľuje stena). Po tejto zmene sa robot znovu vie pohybovať pravidlom zmenšovania vzdialenosti.

Pri aktualizácií ohodnotenia treba udržovať konzistenciu mapy vzdialeností. To znamená, že každé políčko mapy okrem cieľového musí mať ohodnotenie o jedna väčšie, než najmenšie ohodnotenie otvoreného suseda. To sa dá zabezpečiť prechodom mapy do šírky od momentálneho políčka. Cieľom je upraviť ohodnotenie políčka, ak pravidlo porušuje a vložiť do fronty všetky otvorené susedné políčka, ktoré pravidlo porušujú. Potom opravný algoritmus vyberá políčka z fronty a pridáva ďalšie, až kým nie je fronta prázdna.

Keď sa robot dostane takýmto spôsobom do cieľa, algoritmus zároveň našiel najkratšiu cestu z počiatočného políčka do cieľa. Stačí sa už len vrátiť na začiatok a nasledovať cestu so zmenšujúcou sa vzdialenosťou s upravenými hodnotami po preskúmaní pozícií stien.

2.4.6 Počítačové videnie

Počítačové videnie sa dá zjednodušene rozdeliť na dve základné úlohy – spracovanie a analýza statického obrazu, ktorá obvykle rieši rozoznávanie a identifikáciu objektov v obraze a spracovanie videa, teda sekvencie obrázkov, ktorá obvykle rieši problém detekcie rozdielov medzi jednotlivými obrazmi, napríklad s cieľom abstrahovať časovú informáciu alebo sledovať zmeny pozorovaného objektu.

Klasická reťaz udalostí spracovania obrazu vyzerá nasledovne [17]:

- **Zachytenie obrazu** – získanie optickej informácie vzorkovaním spojitého reálneho sveta, určenie rôznych atribútov obrazu, ako napríklad požadované rozlíšenie alebo rozmery obrazu.
- **Predspracovanie** – táto časť zahŕňa odstránenie nežiaducej informácie z obrazu, ako je napríklad šum alebo optická deformácia obrazu spôsobená šošovkou, alebo vylepšenie vlastností obrazu, ako je napríklad kontrast alebo jas. Patria sem aj algoritmy opravujúce väčšie chyby obrazu, ako aj detektory hrán a iné transformácie obrazu.
- **Segmentácia** – patria sem metódy prahovania, segmentácie založenej na oblastiach, porovnávacie metódy a pokročilé povrchové a hraničné detekcie.
- **Reprezentácia tvarov** – identifikácia regiónov, popis a reprezentácia tvarov na základe kontúr alebo na základe regiónov.
- **Rozoznanie objektov –** reprezentácia znalostí, štatistické rozpoznávanie vzorov, syntaktické rozpoznávanie vzorov, optimalizačné techniky.
- **Rozhodovanie –** patria sem riadiace stratégie, metódy rozpoznávania vzorov pri porozumení obrazu, sémantické označovanie a sémantická segmentácia obrazu.

Počítačové videnie je rozsiahla tematická oblasť, ktorá prevyšuje rozsah tejto práce. Pri návrhu softvéru pre úlohu Myš v bludisku budem vychádzať z tejto reťaze

udalostí, jej prevedenie však bude zjednodušené a bude používať intuitívne riešenia čiastkových úloh prispôsobené schopnostiam robota.

2.5 Existujúce práce s robotom

Pred touto prácou už existovala jedna práca (z FMFI UK), ktorá sa priamo zaoberala robotom E-puck v súťažnej úlohe Myš v bludisku na súťaži ISTROBOT [10]. Išlo o študentský projekt v rámci vyučovacieho predmetu Algoritmy pre AI robotiku, ktorý je súčasťou študijného programu Aplikovaná informatika na FMFI UK.

Projekt je datovaný rok pred touto prácou, teda na letný semester vyučovacieho roku 2010/2011. Je s ním však spojené obmedzené množstvo dokumentácie. Okrem tejto dokumentácie pri jeho stručnom popise vychádzam aj z krátkeho stretnutia s jeho autormi.

Princíp programu, ktorý vznikol v rámci projektu je nasledovný:

- Robot používa infračervené senzory na preskúmanie políčka bludiska, na ktorom sa práve nachádza. Prejde po uhlopriečke políčka a v dvoch protiľahlých rohoch sa zastaví, aby zmeral výstup infračervených senzorov vzdialenosti. To mu poskytne jednoznačnú informáciu o všetkých 4 stenách daného políčka, ktorú si robot uloží do reprezentácie mapy.
- Prehľadávaním do šírky hľadá najbližšie nepreskúmané políčko, presunie sa k nemu a pokračuje bodom vyššie. Ak sú všetky políčka preskúmané, nájde prehľadávaním do šírky najkratšiu cestu na počiatočné políčko a vráti sa na začiatok.
- Nájde vo vnútornej mape pozíciu cieľa a opäť prehľadávaním do šírky nájde najkratšiu cestu na toto miesto.
- Prejdením tejto cesty sa robot dostane do cieľa, zároveň je to najkratší čas, ktorý môže dosiahnuť.

Robot teda najprv preskúma celé bludisko a potom nájde najkratšiu cestu zo štartu do cieľa. Takéto riešenie je síce algoritmicky jednoznačne dobré, ale v súťažnom prostredí má robot veľký problém stihnúť vôbec preskúmať celé bludisko za čas 5 minút – kvôli jeho nízkej rýchlosti.

Hlavný problém, s ktorým sa tento projekt stretol, bola nepresnosť pohybu robota. Je pravdepodobné, že tieto chyby pohybu boli spôsobené problémom s knižnicou robota, ktorý som objavil a popisujem v stati 5.3. Robot si teda z dlhodobého hľadiska nebol schopný udržať presný kurz a stávalo sa, že ho vychýlila kolízia s prekážkou. V takejto situácií robot nemal náhradnú stratégiu a nenávratne sa stratil.

Toto riešenie sa pokúšalo využiť aj prídavný kompasový senzor pre zlepšenie orientácie aj v prípade, že sa robot stratí. Daný senzor senzor bol pripojený na zbernicu I2C. Podľa slov autorov však kompas nefungoval spoľahlivo.

Tento senzor som nakoniec určil ako zdroj počiatočného problému s kamerou robota. Obraz z kamery bol s pripojeným kompasom úplne nepoužiteľný. Odpojenie kompasu problém odstránilo.

3. Špecifikácia požiadaviek na softvér

Táto kapitola vyslovuje špecifické požiadavky na navrhovaný softvér, ktoré priamo súvisia s platformou robota E-puck ako aj so súťažnou úlohou Myš v bludisku.

- Medzi technické obmedzenia patrí veľkosť operačnej pamäte, ktorá však na väčšinu bežných účelov stačí. V prípade, že sa robot stretne s väčším množstvom dát, ktoré potrebuje naraz spracovať (ako napríklad pri počítačovom videní), musí zvoliť vhodný a efektívny prístup.
- Ďalším z technických obmedzení je programová pamäť robota – program a statické dáta sa musia do tejto pamäte zmestiť ako celok. Pri písaní samotného programu teda treba postupovať šetrne, dodržiavať aj zásady čistého a efektívneho kódu, vyvarovať sa nadbytočných alebo opakujúcich sa častí.
- Keďže robotické senzory vždy prezentujú odchýlky, program musí byť pri ich použití dostatočne robustný. Pri prechode od presných výpočtov do reálneho sveta treba počítať so situáciou, že je iný než aký ho robot očakáva a mal by mať záložné stratégie aj pre takéto situácie.
- Požiadavkou je aj testovanie, ktoré overí robustnosť a odhalí nepredvídateľné chyby programu. Ideálna je tvorba viacerých verzí tej istej časti programu, niektoré sa môžu osvedčiť ako lepšie a horšie môžu poslúžiť ako zdroj informácií pre ďalšiu verziu.
- Podstatnou požiadavkou pre testovanie programu je funkcionalita pre ladenie programu – po spustení programu nemá vývojárske prostredie (okrem špeciálneho prípadu, kedy je robot pripojený káblovým rozhraním – takéto možnosti však E-puck momentálne nemá) žiadny prístup k robotovi a teda program sa nedá sledovať či prerušovať tak, ako to je bežné pri vývoji pre počítačové stolné platformy.
- Vyžaduje sa aj spoľahlivosť a korektnosť programu, nielen robustnosť na vstupy a výstupy. Nekorektné alebo na zdroje nešetrné správanie programu môže vyústiť do nepredvídateľných situácií, ako napríklad reštartovanie mikroprocesora, „zacyklenie“ výpočtu alebo zaplnenie dynamickej časti pamäte

ako je halda a zásobník volaní, ktoré sú vo všeobecnosti ťažko lokalizovateľné (už len z dôvodu komplikovaného spôsobu ladenia programu – program sa dá s momentálnou výbavou robota ladiť len priamo za behu na robotovi a s použitím vlastných ladiacich častí programového kódu).

Externé požiadavky na program, ktoré udáva súťaž ISTROBOT a úloha Myš v bludisku sú:

- Robot musí v bludisku konať autonómne, nesmie nijakým spôsobom komunikovať.
- Robot musí dodržiavať princípy súťažnej úlohy Myš v bludisku – nesmie prekračovať steny, nesmie si robiť v bludisku žiadne orientačné značenie, nesmie meniť konfiguráciu bludiska.
- Počas súťaže ISTROBOT má robot časové obmedzenie maximálne 5 minút v bludisku a najviac 10 súťažných pokusov.

4. Návrh softvérového riešenia

V tejto kapitole je popísaný návrh softvérového riešenia úlohy Myš v bludisku pre robota E-puck s ohľadom na analýzu problematiky a špecifikáciu úlohy a softvéru pre robota. Návrh vychádza zo štruktúry programovacieho jazyka C.

4.1 Architektúra softvéru

Architektúra softvéru je modulárna – program je rozdelený do niekoľkých programových celkov, ktoré sú ďalej označené ako moduly. Tieto časti sa dajú v objektovo-orientovanom modely interpretovať ako triedy, avšak vzhľadom na programovacie jazyky pre platformu robota (C, Assembler) sa o triedach nedá priamo uvažovať.

Moduly sú navrhnuté tak, aby boli čo možno najviac samostatné a obmezili množstvo závislostí medzi sebou. Závislosti a komunikácia sa ale nedajú vylúčiť, riešenie úlohy spočíva v integrácií všetkých modulov do jedného celku.

Moduly čo možno najviac skrývajú svoje vnútorné zloženie, existencií väčšieho množstva globálnych premenných sa však zabrániť nedá. Dá sa ale použiť lokálne linkovanie premenných (kľúčovým slovom `static`), čo skryje premenné pred ostatnými zdrojovými súbormi, ktoré daný zdrojový súbor k sebe prikladajú.

Globálne premenné dodržujú zásady štruktúrovaného kódu. Na jeden modul existuje najviac jedna globálna premenná, ktorá v sebe zhromažďuje všetky dátové premenné modulu. Ak modul potrebuje viac ako jednu takúto zloženú premennú (`struct`), znamená to, že vykonáva viac logických činností a mal by byť rozdelený.

4.1.1 Modulárna dekompizícia softvéru

Bt_debug – tento modul bude zapezpečovať funkcionalitu pre ladenie softvéru, a to množinou funkcií, ktoré umožnia robotovi posielať rôzny dátový obsah premenných cez Bluetooth modul robota na terminál na vývojovom počítači v znakovej podobe. Modul bude tiež zhromažďovať iné funkcie, potrebné pre ladenie softvéru.

Tests – tento modul bude obsahovať funkcie potrebné pre unit testovanie. Samotné unit testy sa budú nachádzať v osobitných moduloch. V tomto budú

definované funkcie pre vykonanie testov na parametroch.

Prox – modul, ktorý zhromažďuje dáta z infračervených senzorov vzdialenosti a konzistentne ich poskytuje ďalej. Takisto poskytuje analytické funkcie nad týmito hodnotami.

Navig – tento modul má na starosti odometrickú navigáciu robota. To znamená vykonávanie základných prvkov pohybu robota (pohyb vpred resp. vzad, otočka na mieste, otočka po kručnici, zmena rýchlosti a iné), logovanie odometrických dát vykonaného pohybu a všetky matematické prepočty vzdialeností a uhlov pre pohyb a odometriu.

Wall_follow – modul vykonávajúci algoritmus sledovania pravej, resp. ľavej steny v bludisku ako základné riešenie úlohy Myš v bludisku. Vykonáva pokročilú analýzu dát z infračervených senzorov a na jej základe vie rozhodnúť o ďalšom nasmerovaní robota.

Mapping – tento modul udržuje aktuálnu reprezentáciu mapy bludiska, vykonáva zapisovanie informácií o výskyte steny na základe momentálnej polohy robota a jeho kurzu a sprístupňuje tieto dáta ďalším modulom.

Localization – modul, ktorý vykonáva algoritmus metódy dead-reckoning. Používa logované dáta navigačného modulu a predošlú známu polohu, aby matematicky vypočítal momentálnu polohu robota v bludisku s maximálnym rozlíšením (lokalizačné súradnice polohy v bludisku majú krokovú mierku, teda poloha sa počíta s presnosťou na krok motorov robota) a pre účely mapovania bude tento modul poskytovať prevody polohy do bludiskovej mriežky alebo do iných súradníc (napríklad pre pravdepodobnostné mapovanie robot bude vykonávať viac meraní tej istej steny a môže na to použiť presný vzdialenostný interval, ktorý vygeneruje lokalizácia).

Vision – modul zabezpečuje zachytenie obrazu vhodnej veľkosti a vlastností z kamery, detekovanie hrán na základe rozdielu kontrastu, metódu segmentácie založenej na hranách. Možný prístup pre segmentáciu obrazu je aj použitie prahovania – to však vyžaduje vysoký kontrast objektov a pozadia. Nakoniec výsledkom tohto modulu je pravdepodobnostné určenie výskytu stien v najbližších políčkach bludiska pred kamerou.

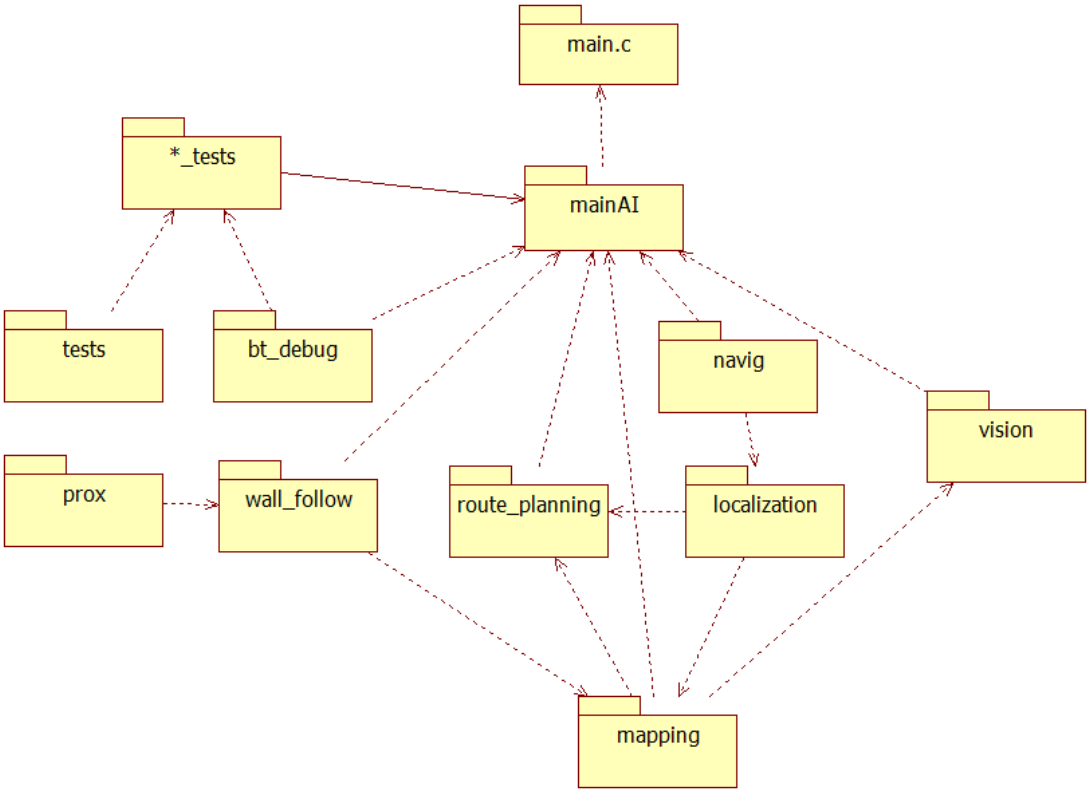
Route_planning – tento modul bude vykonávať upravený Flood-fill algoritmus, ktorého popis je v stati 2.4.5. Bude udržiavať mapu manhattanských vzdialeností všetkých políčok od cieľa, na požiadanie poskytne informáciu na ktoré susedné políčko sa má robot presunúť ďalej a zároveň sa bude tento modul starať o aktualizáciu mapy vzdialeností na základe aktuálnych informácií z modulu mapovania a z modulu lokalizácie, podľa princípu upraveného Flood-fill algoritmu.

Keďže modul vision v idálnom prípade poskytne doprednú informáciu o bludisku, súčasťou tohto modulu môže byť aj simulácia pohybu robota a dopredná simulácia algoritmu Flood-fill, ktorej cieľom môže byť napríklad včasné odhalenie slepých ciest a automatická aktualizácia vzdialenostného ohodnotenia políčok bludiska.

MainAI – hlavný modul, ktorý spája všetky ostatné do celku. V tomto module bude lokalizovaná inicializácia a vloženie všetkých modulov a použitých knižníc, hlavný cyklus riadenia robota a všetky čiastkové funkcie, ktoré vedú ku riešeniu úlohy za použitia ostatných modulov.

Súbor main.c – tento súbor bude obsahovať jedinú funkciu, a to funkciu `main`. Toto je hlavná používateľská funkcia programovacieho jazyka C, ktorej vykonávanie sa spustí pri spustení programu. Bude obsahovať len volanie funkcie pre hlavný riadiaci cyklus robota, ktorá sa nachádza v module mainAI.

Množina modulov *_tests – tieto moduly budú obsahovať unit testy všetkých testovateľných funkcií príslušného modulu, ktorého názov nahrádza *. Každý testovací modul tejto množiny obsahuje hlavnú funkciu, ktorá postupne spúšťa funkcie pre testovanie jednotlivých funkcií testovaného modulu.



Obrázok č.3 *Graf závislostí modulov*

4.2 Dátové štruktúry

V tejto časti sú popísané dátové štruktúry návrhu. Pozornosť je venovaná zloženým dátovým typom (`struct`), pre definíciu globálnych premenných modulov. Vymenované dátové typy (`enum`) sú tiež prvkom prehľadného a štruktúrovaného kódu. Ich definovanie a použitie je ponechané na programátora návrhu, avšak je veľmi odporúčané.

Modul prox – uchováva výstup každého z infračervených senzorov (typu `int`) z jedného časového momentu. To indikuje buď použitie jednorozmerného celočíselného poľa dĺžky 8 (taký je počet senzorov robota), ktoré je indexované napríklad vymenovaným dátovým typom. Druhou možnosťou je použiť štruktúru (`struct`), v ktorej bude 8 položiek typu `int`, pre každý senzor jedna.

Modul navig – uchováva momentálnu rýchlosť robota, kurz robota v podobe uhla (počiatočný uhol je považovaný za nulový), počet prejdených krokov od poslednej zmeny kurzu a denník (ďalej označený ako log), v ktorom sú zaznamenané zmeny stavu

navigácie.

Rýchlosť a počet prejdených krokov od poslednej zmeny kurzu sú jednoduché celočíselné položky, kurz by kvôli lepšej presnosti mal byť číslo typu `float`, resp. `double`. Log môže byť prevedený ako cyklické pole záznamov, čo indikuje ďalšiu položku hlavnej štruktúry – index vrcholu logu.

Záznam logu môže byť separátna štruktúra, ktorá musí obsahovať minimálne kurz a počet prejdených krokov týmto kurzom.

Modul wall_follow – uchováva informáciu o tom, aká korekcia pohybu sa práve aplikuje na robota (aby sa nestala situácia, kedy sa dostanú do konfliktu dve rôzne korekcie polohy, ale aby vždy najprv skončila jedna korekcia a až potom sa začala vykonávať iná), ďalej informáciu o tom, ktorú stenu robot sleduje počas aktuálneho behu programu (nastaví sa na začiatku programu a potom ostáva nemenná až do skončenia) – týmto spôsobom sa zabezpečí konzistentnosť rozhodovania robota. Ako poslednú si modul uchováva informáciu o tom, na ktorej strane robota sa naposledy nachádzala nejaká stena. Táto informácia sa aktualizuje pre nové meranie senzorov a ostáva zachovaná až do ďalšieho merania pre rozhodovacie účely modulu. Všetky položky tejto štruktúry môžu byť napríklad vymenovaného dátového typu.

Modul mapping – jedinou premennou, ktorú uchováva je mapa bludiska. Mapa bludiska zachytáva pravdepodobnostnú informáciu o konfigurácií stien bludiska. To znamená, že obsahuje celočíselné hodnoty, ktoré sú úmerné pravdepodobnosti, že na danej pozícií steny ozaj stena je (pre jednu stenu robot vykonáva viac meraní, aby sa znížilo riziko chyby). Tieto hodnoty pravdepodobnosti existencie steny sú potom logicky mapované na reálne steny bludiska.

Riešenie je použiť celočíselné pole dĺžky počtu všetkých stien bludiska a nad touto reprezentáciou vybudovať mapovacie funkcie, ktoré algoritmicky prepočítajú vstupné parametre na inú logickú úroveň mapy (napríklad: parametrami sú súradnice políčka bludiska a identifikátor jednej zo štyroch stien bunky a funkcia na základe týchto informácií vypočíta index jednorozmerného poľa, v ktorom modul uchováva informáciu o tejto stene).

Modul localization – ukladá informáciu o poslednej známej polohe robota

(počiatočná poloha je v bode [0, 0]) dvojicou celočíselných premenných, identifikujúcich súradnice x a y bludiska. Táto súradná sústava bude mať mierku veľkosti jednoho kroku motorov robota, čo zabezpečí maximálnu logickú presnosť lokalizácie.

Ďalej modul uchováva informáciu o tom, či došlo ku prekročeniu ďalšej (pomyselnej) čiastkovej hranice vnútri bunky bludiska (jedno políčko bude rozdelené na abstraktnú mriežku, ktorej bunka má čiastkové, napríklad desatinné rozmery oproti celému políčku bludiska) za účelom mapovania – keďže mapovanie vykonáva viac meraní pre jednu stenu.

Takisto je treba uložiť posledný spracovaný index navigačného logu, ako aj počet spracovaných krokov momentálneho stavu navigácie (ktorý ešte nie je súčasťou logu, ale je podstatný pre určenie skutočnej momentálnej polohy robota). Tento počet potom poslúži pri ďalšom volaní aktualizácie polohy, aby lokalizačný modul vedel určiť koľko krokov z prvého nespracovaného záznamu navigačného logu už započítal do zmeny pozície.

Modul vision – keďže operačná pamäť robota nepostačuje na uchovanie celého obrazu z kamery pre spracovanie, tento modul si v jednom momente môže uchovávať len malú časť obrazu, napríklad jeden riadok obrazových bodov, prípadne aj niekoľko riadkov, závisí to od objemu dát. To znamená, že modul uchováva pole typu `unsigned char` dĺžky jednoho riadku alebo násobku tejto dĺžky (pri uchovaní viac ako jednoho riadku) pre uchovanie hodnôt jasu obrazových bodov.

Ďalej potrebuje štruktúru, v ktorej si postupne uchová analytickú informáciu o celom obraze – jednorozmerné súradnice detekovaných hrán medzi bielou stenou a čiernou podlahou pre každý spracovaný riadok obrazu. Keďže v jednom riadku obrazu sa môže nachádzať viac ako jeden kontrastný prechod, štruktúra musí počítať aj s týmto faktom, napríklad použitím dynamického spájaného zoznamu.

Modul route_planning – tento modul má štruktúru pre uchovanie mapy manhattanských vzdialeností každého políčka bludiska od cieľa. Dobrou doplnkovou informáciou je pre každé políčko aj množina otvorených susedov – políčok, na ktoré sa dá z tohto políčka prejsť – uľahčí to prehľadávanie a traverzovanie touto mapou a pritom nemusí dochádzať ku žiadnej ďalšej komunikácií s mapovacím modulom, iba pri

aktualizácií tejto vzdialenostnej mapy. Táto mapa môže byť reprezentovaná napríklad ako dvojrozmerné pole štruktúr, ktoré obsahujú celé číslo vyjadrujúce manhattanskú vzdialenosť políčka od cieľa a pole logických hodnôt dĺžky 4, ktoré predstavuje množinu otvorených susedov, pričom logická hodnota pod indexom poľa predstavuje fakt, či je daný sused otvorený alebo nie.

Takisto modul route_planning má FIFO štruktúru reprezentujúcu frontu políčok vzdialenostnej mapy, ktorá je potrebná pri opravách ohodnotenia políčok počas odhaľovania konfigurácie stien bludiska. Front môže byť reprezentovaný ako cyklické pole pre uchovanie indexov buniek vzdialenostnej mapy, pričom je uložený aj index začiatku a konca fronty. Táto statická reprezentácia má ale pevnú veľkosť a môže tak dôjsť ku pretečeniu.

Iné riešenie je použitie dynamickej štruktúry, akou je napríklad spájaný zoznam.

4.3 Funkcionalita modulov

Modul prox:

- Zaznamenanie hodnôt všetkých senzorov do dátovej reprezentácie.
- Určenie a vrátenie identifikátora senzoru, ktorý detekuje najbližšiu prekážku.
- Vrátnie hodnoty požadovaného senzoru.
- Určenie, či sa v blízkosti robota nachádza nejaké prekážka, alebo nie.

Modul bt_debug:

- Poslanie poľa znakov cez Bluetooth.
- Poslanie obsahu celočíselnej premennej na terminál cez Bluetooth. Táto funkcia vyžaduje prevod číselnej hodnoty na pole znakov, ktoré sa dá odoslať vyššie uvedenou funkciou.
- Poslanie reálneho čísla na terminál.
- Poslanie formátovaného obsahu globálnych premenných iných modulov. Použitím funkcií z predošlých bodov a použitím vytvoreného formátovania pre každú globálnu dátovú štruktúru odošle ich obsah na terminál vzdialeného počítača.
- Pozastavenie programu až do prijatia určitého znaku cez Bluetooth.

- Pozastavenie programu na určitý počet prázdnych inštrukcií procesora (aktívne čakanie).
- Testovanie, či došlo ku prijatiu určitého znaku cez Bluetooth.

Modul tests:

- Testovanie, či sú celočíselné argumenty funkcie rovné, ak nie, tak výpis skutočnej a očakávanej hodnoty.
- Testovanie, či je argument funkcie logická 1 alebo 0, možnosť výpisu preddefinovaného znakového reťazca, ak test zlyhal.

Modul navig:

- Realizácia zmeny rýchlosti priameho pohybu robota.
- Realizácia otočky robota na jednom mieste o daný uhol. Prejdené kroky počas otočky sa nulujú, aby nerušili navigáciu, ktorá počíta s logovaním iba priamočiarych prejdených krokov a kurzu.
- Realizácia otočky robota po kružnici – podobné vlastnosti ako bod vyššie s rozdielom, že sú použité iné matematické funkcie, ktoré počítajú koľko krokov a akou rýchlosťou musia prejsť motory, aby otočili robota o požadovaný uhol. Na výpočty obvodov týchto kružnicových výsekov sú použité matematické vlastnosti obvodu kružnice a geometrické vlastnosti robota (rozkolie, obvod kolies, obvodová dĺžka kroku motora).
- Funkcie pre výpočet dĺžky kružnicových výsekov, po ktorých prechádzajú kolesá robota pri otáčaní na mieste alebo po kružnici a závisia od uhla, o ktorý sa robot otáča.
- Optimalizácia uhla tak, aby bol zmenšený na ekvivalentné minimum a šetril tak robota od zbytočného otáčania – napríklad otočenie o úhol 270 stupňov je ekvivalentné otočeniu robota o úhol -90 stupňov.
- Množina funkcií označených ako „gettery“, ktoré vracajú obsah globálnej premennej tohto modulu alebo iné informácie týkajúce sa navigácie.
- Funkcie pre zastavenie a obnovenie fyzického pohybu kolies bez logovania.

- Funkcia, ktorá sleduje prejdenie daného množstva krokov priamočiarym pohybom a potom preruší pohyb.
- Funkcie pre logovanie stavu navigácie. Sú volané vždy pred zmenou stavu navigácie, teda napríklad pred zmenou kurzu robota a uložia momentálny stav, teda kurz a počet prejdených krokov týmto kurzom.Starajú sa aj o reštartovanie počítadiel krokov pre nový stav navigávie.

Modul wall_follow:

- Funkcia pre určenie, na ktorej strane robota sa nachádza stena na základe maximálnej hodnoty blízkosti zo senzorov.
- Funkcia na správu rôznych korekcií kurzu tak, aby nevznikali konflikty medzi korekciami. Určuje aj stav, kedy nie je potrebná žiadna korekcia kurzu.
- Funkcia, ktorá určuje či je robot v bezpečnom vzdialenostnom koridore pri stene, alebo sa dostal mimo tento koridor – je príliš blízko, alebo príliš ďaleko od steny. Používa senzory na detekciu vzdialenosti.
- Funkcia, ktorá určuje, či robot vchádza do pravouhlého rohu medzi stenami, alebo nie. Použije pritom senzory na detekciu vzdialenosti.
- Ďalej modul definuje funkciu, ktorá na základe týmto modulom zvolenej korekcie určí smer, ktorým sa má robot fyzicky otočiť, aby zrealizoval korekciu navrhnutú v predošlých bodoch. Veľkosť uhla otočenia v tomto smere je konštantná, čím menšia, tým jemnejšia je korekcia. V zásade ide o hodnoty menšie ako 10 stupňov.

Modul mapping:

- Funkcia pre jednorázový konštantný zápis do mapy, ktorý zvýši pravdepodobnosť existencie danej steny o konštantnú hodnotu (odvodenú napríklad od počtu vykonaných meraní danej steny). Funkcia sama zistí z modulu localization a z modulu wall_follow, pri ktorej stene sa robot nachádza.
- Funkcia pre manuálny zápis hodnoty do mapy, parametrizovaná súradnicami políčka bludiska, identifikátorom jednej zo štyroch možných stien políčka a hodnotou, ktorá má byť vpísaná na toto miesto mapy.

- Funkcia pre čítanie hodnoty pravdepodobnosti existencie danej steny, ktorá je určená parametrami – pozícia v bludisku a identifikátor steny políčka.
- Funkcie pre prechod od abstraktnej súradnej sústavy bludiska ku reálnej reprezentácií bludiska, teda k jednorozmernému poľu.
- Funkcie pre overovanie korektnosti indexov – pri prechodoch medzi abstraktnou reprezentáciou mapy ku skutočnej, zápisoch a čítaniach mapy.
- Funkcia pre zjednodušenie kurzu – pre účely mapovania stačí vedieť, ktorým zo štyroch smerov v bludisku sa robot pohybuje relatívne k mape. Zároveň sa takto čo najviac eliminuje odchýlka navigácie a lokalizácie. Kurz v rozmedzí 45..-45 stupňov je považovaný za pohyb vpred, 45..135 za pohyb doľava, 135..-135 za pohyb dozadu a -135..-45 za pohyb doprava. Táto informácia je použitá pri zápise do mapy, ktorý potrebuje identifikovať stenu, okolo ktorej robot práve prechádza.

Modul localization:

- Funkcia, ktorá prejde všetky nové záznamy logu navigácie a na ich základe prepočíta momentálnu polohu robota. Potom musí ešte zobrať do úvahy aj aktuálny stav navigácie, ktorý nie je v logu navigácie – keď sa robot pohybuje rovno, nedochádza ku logovaniu, pretože informácie sú aktuálne.
- Funkcie pre výpočet x-ového a y-ového prírastku, resp. úbytku polohy robota. Uhol a prejdená vzdialenosť pod týmto uhlom definujú preponu pravouhlého trojuholníka. Dĺžky odvesien tohto trojuholníka predstavujú zmenu polohy v x-ovej a y-ovej osi, každá odvesna pre jednu os. Dĺžka odvesny sa dá vypočítať zo základného goniometrického vzťahu:

  $\cos \alpha = a / c \quad \sin \alpha = b / c$

  kde a je odvesna priľahlá k uhlu α, b je protiľahlá odvesna k uhlu α a c je prepona.
- Modul ďalej definuje funkcie pre prevod krokovej súradnej polohy robota na abstraktné úrovne, napríklad na úroveň súradníc bludiskových políčok.
- Funkcie, ktoré signalizujú posun robota o určitú vzdialenosť v oboch osiach –

táto vzdialenosť je čiastková poloha robota v políčku bludiska a slúži ako signál, že má dôjsť ku ďalšiemu meraniu výskytu steny a zápisu do mapy (mapa je pravdepodobnostná, teda čím viac zápisov v prospech existencie steny, tým väčšia istota skutočného výskytu steny). Meranie sa vykonáva na rôznych miestach steny, na čo sa použije práve signál o prekročení pomyselnej čiastkovej hranice v rámci jednoho políčka bludiska.

Modul vision:

- Funkcia pre nastavenie kamery pre zachytenie ďaľšieho riadku, resp. riadkov.
- Funkcia pre zachytenie obrazu z nastavenej kamery do vnútorného poľa.
- Funkcia pre detekovanie hrán v čiastočnom obraze. Prechodový filter široký niekoľko obrazových bodov prejde celým čiastkovým obrazom, pritom počíta priemernú hodnotu jasu pravej a ľavej časti filtra. Porovná ich, a ak sa líšia viac ako o konštantnú hodnotu – citlivosť filtra, filter určí toto miesto ako hranu. Zároveň jeho miesto (index poľa) uloží do vnútornej reprezentácie pre hrany obrazu. Toto je intuitívny prístup, formálny postup by bol použiť niektorý zo známych operátorov pre detekciu hrán, napríklad Robertsov operátor alebo Laplacov operátor.
- Funkcia pre segmentáciu uložených hrán obrazu – táto funkcia pomyselne rozdelí obrázok do niekoľkých častí, ktoré odpovedajú miestam, kde sa očakáva hrana steny a podlahy. Ak sa v danom segmente nachádza hrana, funkcia ho označí ako stenu. Tento princíp však funguje len za podmienok, že robot sa pozerá vždy rovnakým smerom, napríklad paralelne so sledovanou stenou a je tiež len intuitívnou formou segmentácie.
- Funkcia, ktorá na základe výsledkov segmentácie vykoná zápis do mapovacieho modulu.

Modul route_planning:

- Funkcia, ktorá rozhoduje na ktoré susedné políčko sa má robot presunúť. Pri jej volaní sa najprv aktualizujú otvorení susedia vnútornej mapy vzdialeností na základe dát z mapovacieho modulu, potom prebehne krok modifikovaného algoritmu Flood-fill, ktorý vyberie otvoreného suseda s najmenšou

vzdialenosťou ku cieľu alebo prebehne oprava vzdialeností.

- Funkcia, ktorá na základe mapovacieho modulu opraví údaje o otvorených susedoch každého políčka vnútornej mapy vzdialeností. Robota v mape lokalizuje pomocou modulu localization.
- Funkcia, ktorá vykoná opravu ohodnotenia. Ak políčko, na ktorom robot práve stojí, porušuje pravidlo algoritmu, ktoré hovorí, že každé políčko vzdialenostnej mapy s výnimkou cieľového políčka, musí mať ohodnotenie o jedna väčšie ako je najmenšie ohodnotenie spomedzi jeho otvorených susedov je vložené do fronty a spustí sa aktualizácia. Počas nej sa vyberie políčko z fronty, zvýši sa jeho ohodnotenie tak, aby dodržovalo vyššie popísané pravidlo a vložia sa do fronty všetci otvorení susedia daného políčka, ktoré pravidlo porušujú. Algoritmus pokračuje až do vyčerpania fornty.
- Funkcia, ktorá rozhodne, na ktoré susedné políčko sa má robot posunúť pri spätnej ceste z cieľa do štartovného políčka.

Modul mainAI:

- Funkcia pre hlavný riadiaci cyklus robota. V prípade, kedy robot len sleduje niektorú zo stien sa opakuje cyklus popísaný pseudokódom:

```
vykonajMeranieIRSenzorov();

analyzujHodnotyIRSenzorov();

/* urči na ktorej strane robota je stena a aká
korekcia kurzu sa práve vykonáva */

otocRobotaOUhol(smerKorekcie() * VELKOST_KOREKCIE);

/* ak sa nevykonáva žiadna korekcia, smerKorekcie() je
nulový a robot sa neotáča. Miesto toho pokračuje v
pôvodne nastavenej rýchlosti priamo vpred */
```

Keď sa robot pohybuje modifikovaným Flood-fill algoritmom:

```
presunSaNaSusednePolicko(floodFillDalsiePolicko());

/* pri presune zaznamenávaj detekované steny do mapy,
```

```
ak sa na políčko nedá presunúť kvôli novej stene, vráť
robota do východzej polohy a pri ďalšom prechode cyklu
flood-fill vygeneruje nový smer pretože aktualizuje
vzdialenosti podľa novej konfigurácie stien */
if(robotJeVCieli()){
   while(not robotJeVStarte()){
      presunSaNaSusednePolicko(floodFillSpatnaCesta());
   }
}
/* a cesta pokračuje v smere ku cieľu, pričom tentoraz
prejde zostupne už po prehodnotených políčkach, ktoré
definujú najkratšiu cestu */
```

- Funkcia pre presun robota na jedno zo štyroch susedných políčok. Táto funkcia vykonáva sekvencie pohybu, ktoré presunú robota na východziu polohu nového políčka, ak sa nachádza na východzej polohe momentálneho políčka. Vykonáva pritom mapovanie v čiastkových intervaloch, ak nemôže dokončiť presun kvôli novej stene, vráti robota na východziu poloho tohto políčka bludiska.

5. Implementácia návrhu na reálnom robotovi

5.1 Výber použitých nástrojov a technológií

Pre implementáciu bolo zvolené vývojové prostredie MPLAB IDE a programovací jazyk C na platforme operačného systému Windows. Pre kompiláciu programu sú použité nástroje MPLAB C30. Pre nahrávanie nového programu do robota je použitý program Tiny PIC bootloader, ktorý zároveň slúži ako terminál pre sériovú komunikácie s robotom. Pre počiatočné testovacie účely bol zvolený program Epuckmonitor. Výber týchto nástrojov podporila knižnica E-puck library pre robota, ktorá je napísaná v jazyku C a Assembler pre kompilátor MPLAB C30 a bude tiež použitá.

5.2 Postup implementácie

Ako prvé vznikly menšie testovacie programy, ktoré ma zoznámili s knižnicou robota a predviedli jeho možnosti. Posielali dáta zo senzorov na terminál, umožňovali ďiaľkové ovládanie robota a pod.

Ďalej začal vznikať návrh softvérového riešenia robota a paralelne s ním aj implementácia, ktorá poskytla spätnú väzbu pre návrh. Niektoré navrhnuté postupy sa totiž testovaním ukázali ako nevhodné. Počas implementácie teda vzniklo niekoľko verzí modulov.

Ako prvý sme pre riešenie úlohy Myš v bludisku vybrali princíp sledovania pravej, resp. ľavej steny. Implementácia teda postupovala od získavania dát zo senzorov vzdialenosti, cez navigáciu robota až po modul pre riadenie sledovania steny (wall_follow). Táto časť bola dokončená pred súťažou ISTROBOT. V tomto čase už existoval aj prototyp modulu pre mapovanie prispôsobený princípu sledovania steny.

Po súťaži vývoj pokračoval dokončením mapovacieho modulu a návrhom a implementáciou lokalizácie robota. Systém navigácie, lokalizácie a mapovania sa ukázal byť nepresný, preto som venoval veľa času ladeniu tohto systému a objavil aj problémy zakorenené v knižnici robota spojené s hardvérom. Presnosť tohto systému sa zvýšila, ale z dlhodobého hľadiska stále dochádza ku vychýleniu robota.

Implementácia ďalej pokračovala spracovaním obrazu a modifikovaným Flood-

fill algoritmom, ktorý úlohu myši v bludisku rieši efektívne.

5.3 Nekorektná funkcia motorov

Krokový motor robota E-puck sa môže nachádzať v niekoľkých pohybových fázach. Štyri fázy sú vyhradené pre prechody motora o jeden krok vpred, resp. vzad od fázy, v ktorej sa práve nachádza. K pohybu motora o krok dochádza práve pri prechode od jeho momentálnej fázy ku následujúcej (v jednom z dvoch možných smerov rotácie). Tieto štyri fázy majú striktné poradie, opačné poradie spôsobí pohyb motora opačným smerom. Piata fáza je nulová, teda nespôsobí žiadnu akciu motora.

Fázou sa v tomto zmysle rozumie štvorica bitov, ktoré sú vo výstupnom registry procesora a tvoria výstupný port motora. Každý z dvoch motorov robota je pripojený na vlastný výstupný port.

Pre vyvolanie pohybu motora je nutné nastaviť (resp. zmeniť) výstupné bity do jednej zo štyroch kombinácií, ktoré reprezentujú fázy motora. Takisto je nutné, aby tieto výstupné hodnoty na porte ostali minimálny čas, kým sa stihne motor fyzicky otočiť – tento čas udáva maximálnu možnú technickú rýchlosť robota.

Modul e_motors z knižnice robota teda v presnom poradí a s presným načasovaním nastavuje fázy na výstupné porty motorov. Problém, ktorý som zaznamenal spočíva vo fakte, že fázy musia nasledovať za sebou v striktnom poradí. Ak toto nie je dodržané, motor so sebou trhne (ak je mu prezentovaný fázový rozdiel väčší ako jedna fáza) alebo sa pohne o krok opačným smerom (ak je mu prezentovaná predchádzajúca fáza ku fáze, v ktorej sa nachádza).

Modul zaznamenáva aj fázu, v ktorej sa motor práve nachádza. Pri žiadosti o zastavenie motorov modul nastaví na port motora nulovú fázu (aby sa predišlo nežiadúcemu priechodu elektrického prúdu cez elektroniku motora), a pritom si zaznamená fázu, v ktorej motor skončil pohyb. Pri žiadosti o pokračovanie pohybu modul na port motora nastaví ako prvú fázu tú, ktorá nasleduje v správnom smere po fáze, ktorú zaznamenal ako poslednú. Keďže to nemusí byť prvá fáza (ktorá nasleduje za nulovou, rovnako ako aj posledná fáza je nasledovníkom nulovej pri spätnom pohybe), motor môže vykonať nečakaný chybný pohyb.

Tento problém sa najviac prejavil pri častom zastavovaní a pokračovaní pohybu

motorov aj v prípade minimálnej dĺžky zastavenia, ktorá je nepostrehnuteľná pre pozorovateľa robota (ale modul stihol nastaviť nulovú fázu na port motora).

Aj z týchto dôvodov bolo nájdenie príčiny dlhodobej nepresnosti pohybu robota a jej odstránenie časovo a vedomostne náročné.

V súvislosti s modulom e_motors som objavil ešte jeden problém, ktorý rovnako súvisí s presnosťou pohybu a s fázami motora. Tento problém spočíva vo fakte, že nová fáza motora musí ostať na výstupnom porte aspoň minimálny čas. Modul nijakým spôsobom nezaručuje celkové dokončenie posledného kroku po tom, čo používateľ požiadal o zastavenie pohybu. Pri zmene fázy motora počas prerušenia programu časovačom modul najprv zaznamená, že došlo ku zmene fázy (ak k nej má dôjsť z dôvodu správneho časového momentu) a potom nastaví novú fázu na výstupný port.

V prípade, že používateľ pre zastavenie pohybu motora používa práve počet prejdených krokov (zmien fáz), ktorý udržuje a poskytuje tento modul, posledný krok ostane fyzicky nedokončený, aj keď logicky došlo ku zmene fázy. Tá je totiž hneď zmazaná nulovou fázou, pretože modul poskytol informáciu, že počet krokov sa zvýšil a používateľ s touto informáciou naložil tak, že požiadal o zastavenie motorov, ktoré sa vykoná okamžite.

Riešenie prvého problému je externé zabezpečenie faktu, že každý pohyb s motorom sa skončí práve v poslednej fáze sekvencie a teda modul e_motors vždy pokračuje v prvej fáze sekvencie (v oboch smeroch). Lepším riešením, ktoré by možno poskytlo ešte väčšiu presnosť motorov, by bolo navrhnúť vlastnú knižnicu s funkcionalitou pre pohyb motorov.

Riešenie druhého problému spočíva v konštantnom čakaní pred skončením pohybu na dokončenie fázy. Tu by bolo oveľa lepším riešením práve vlastný modul pre pohyb motorov.

Implikácia týchto riešení je nasledovná: presnosť otočenia sa znížila na štvrtinu – počet prejdených krokov každého motora je vždy deliteľný číslom 4 (počet fáz motora) aby pohyb skončil v poslednej fáze sekvencie.

6. Dosiahnuté výsledky a ciele

6.1 Súťaž ISTROBOT 2012

V kategórií Myš v bludisku sa tento rok na súťaž prihlásilo 19 súťažiacich. Štartovalo však len 15 a cieľ dosiahlo 9 robotov.

Robot E-puck sa umiestnil na 6. mieste s kvalifikačným časom 2 minúty a 8 sekúnd. Vylepšuje sa tak minuloročný výsledok robota, kedy robot cieľ nenašiel. Na kvalifikáciu do finále však tento čas nestačil. Do finále postúpila prvá štvorica robotov a robot, ktorého vybrala porota.

E-puck na súťaži použil algoritmus sledovania najprv pravej a potom ľavej steny. Po dosiahnutí cieľa bol ručne premiestnený na štart a dostal tak časovú penalizáciu za dotyk.V oboch variantoch algoritmu úspešne a bez problému dosiahol cieľ.

Ostatní súťažiaci, s výnimkou víťaza, používali tiež algoritmus sledovania niektorej zo stien. Víťaz použil Floodfill algoritmus, ktorý ma neskôr tiež inšpiroval pri tvorbe tejto práce. Víťazný čas 17 sekúnd však robot E-puck technicky nemôže prekonať, s maximálnou rýchlosťou 13 cm/s by za uvedený čas prešiel práve do cieľa, keby sa pohyboval po vzdušnej čiare a v súťažnom bludisku 9 x 9 políčok (ktoré bolo použité aj počas tejto súťaže).

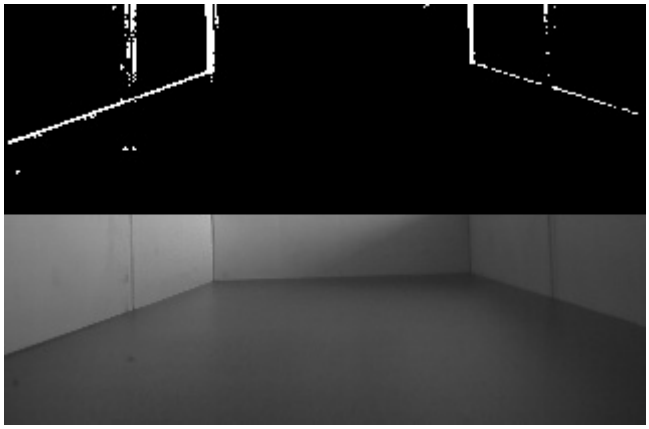
6.2 Implementácia a testovanie

V rámci implementácie a testovania bola implementovaná väčšina návrhu, niektoré časti však do dátumu odovzdania tejto práce neboli úplne dokončené.

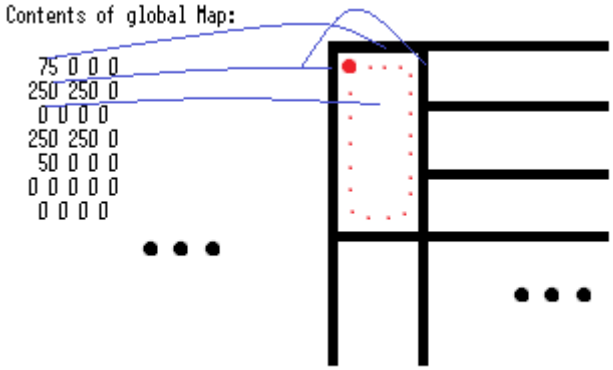
Ide o najmladšiu zmenu do návrhu, ktorou je modifikovaný Flood-fill algoritmus. Integrovanie tohto algoritmu do riešenia vyžaduje zmeny a nové verzie niektorých ďalších modulov, najmä hlavného riadiaceho cyklu ako aj spôsobu pohybu robota (nedá sa ďalej pohybovať sledovaním steny, robot musí prekonávať aj slepé miesta bez pomoci steny) a rozsiahle testovanie. Zmeny si vyžaduje aj princíp tvorby mapy, keďže robot sa počas tohto algoritmu nemusí vždy pohybovať pozdĺž celej steny.

Do dátumu odovzdania tejto práce ostal nedokončený aj modul pre spracovanie obrazu, najmä segmentácia a integrácia s mapovacím modulom.

Obrázok č.4 *Výstup prispôsobeného detektora hrán*



Obrázok č.5 *Výstup prispôsobeného detektora hrán*



Obrázok č.6 *Výstup mapovania počas sledovania steny*

7. Záver

Cieľom tejto práce bolo analyzovať problematiku robotickej súťažnej úlohy Myš v bludisku a východiská pre návrh jej riešenia. Ďalej špecifikovať a navrhnúť riadiaci systém pre reálneho robota, ktorý je schopný úlohu vyriešiť v rámci jej pravidiel. Cieľom bolo aj implementovať tento systém a otestovať ho v rámci robotickej súťaže ISTROBOT 2012.

Riadiaci systém pre robota, ktorý som navrhol a implementoval v rámci tejto práce bol schopný súťažnú úlohu vyriešiť za dodržania pravidiel súťaže s časom 2 minúty a 8 sekúnd a umiestnil sa tak na 6. mieste v kategórií Myš v bludisku na súťaži ISTROBOT.

Súčasťou návrhu sú dve verzie systému. Prvá, ktorá bola použitá aj na súťaži, spočíva v sledovaní pravej, resp. ľavej steny bludiska vzhľadom na robota. Druhá pokročilá verzia využíva modifikovaný Flood-fill algoritmus pre riadenie pohybu robota, princíp odometrie pre navigáciu, lokalizáciu a tvorbu mapy bludiska, ako aj metódy počítačového videnia pre doprednú analýzu bludiska.

Implementácia druhej verzie systému nebola v čase odovzdania tejto práce plne funkčná a integrovaná, avšak jej návrh považujem za dokončený a konzistentný. Implementácia druhej verzie riadiaceho systému je teda kandidátom pre ďalšie pokračovanie tejto práce, o ktorom si myslím, že dokáže ešte väčšmi zlepšiť výkon robota v robotickej súťažnej úlohe Myš v bludisku. S touto motiváciou mám záujem naďalej pokračovať v tejto práci a v jej zlepšovaní.

8. Zoznam použitej literatúry

[1] ANDERSON, D. P. *IMU Odometry* [online] dostupné na internete: <http://www.geology.smu.edu/~dpa-www/robo/Encoder/imu_odo/> [31.05.2012]

[2] BALOGH, R. c2000-2012. *Pravidlá kategórie Myš v bludisku* [online] dostupné na internete: <http://www.robotika.sk/contest/2012/umouse.php> [31.05.2012] FEI STU

[3] BENKOVIC, S. *The modified flood-fill algorithm* [online] dostupné na internete: <http://www.micromouseinfo.com/introduction/mfloodfill.html> [31.05.2012]

[4] CYBERBOTICS c2012 *Webots 6* [online] dostupné na internete: <http://www.cyberbotics.com/overview> [31.05.2012]

[5] DIXON, J., HENLICH, O. 1997 *Mobile robot navigation: Final report* [online] dostupné na internete: <http://www.doc.ic.ac.uk/~nd/surprise_97/journal/vol4/jmd/> [31.05.12], Imperial College, London

[6] DUCHOŇ, F., JURIŠICA, L. 2011 *Globálna navigácia robotov na báze geometrickej mapy* [online] dostupné na internete: <http://www.atpjournal.sk/buxus/docs/Duchon%202.pdf> [01.06.2012]. ATP Journal 2/2011. ISSN 1336-233X

[7] EPFL [Ecole Polytechnique Federale de Lausanne] *E-puck* [online] dostupné na internete: <http://www.e-puck.org> [31.05.2012]

[8] EPFL [Ecole Polytechnique Federale de Lausanne] *Aseba: an event based architecture.*. [online] dostupné na internete: <http://mobots.epfl.ch/aseba.php> [01.06.2012]

[9] *E-puck reference manual 2007* [online] dostupné na internete: <http://hades.mech.northwestern.edu/images/4/4c/E-puckReferenceManual.pdf> [01.06.2012]

[10] GÁLIK, M., KOYŠOVÁ, Z. 2011. *Mouse in Maze with robot E-puck* [online] dostupné na internete: <http://virtuallab.kar.elf.stuba.sk/robowiki/index.php?title=Mouse_in_Maze_with_robot_E-Puck> [31.05.2012] FMFI UK

[11] JURIŠICA, L., DUCHOŇ, F., 2010 *Klasické metódy lokalizácie mobilného robota*

v prostredí [online] dostupné na internete: <http://www.atpjournal.sk/buxus/docs/casopisy/atp_2010/pdf/online102.pdf> [01.06.2012]. AT&P journal 4/2010. FEI STU

[12] MICROCHIP TECHNOLOGY INC. c2007 *MPLAB C30 C Compiler User's Guide* [online] dostupné na internete: <http://ww1.microchip.com/downloads/en/DeviceDoc/MPLAB%20C30%20UG_DS-51284f.pdf> [31.05.2012]

[13] MICROCHIP TECHNOLOGY INC. c2009 *dsPIC30F6014A* [online] dostupné na internete: <http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en024766> [01.06.2012]

[14] MINEBEA CO., LTD c2004 *PG15S-020: PM Motor PG Type* [online] dostupné na internete: <http://www.nmbtc.com/pdf/motors/PG15S020.pdf> [31.05.2012]

[15] NEMERSON, E. *FANN reference manual* [online] dostupné na internete: <http://leenissen.dk/fann/html/files/fann-h.html> [01.06.12]

[16] OPENCV DEV TEAM c2011-2012 *OpenCV Documentation* [online] dostupné na internete: <http://docs.opencv.org/> [01.06.12]

[17] SONKA M., HLAVAC, V., BOYLE, R. 1998 *Image Processing, Analysis, and Machine Vision,* Second Edition, PWS Publishing, ISBN 0-534-95393-X

[18] VISHAY INC. 2009 *TCRT1000, TCRT1010: Reflective optical sensor with transistor output* [online] dostupné na internete: <http://www.vishay.com/docs/83752/tcrt1000.pdf> [31.05.2012]. Rev. 12-Mar-12

[19] WINKLER, Z. 2005 *Odometrie: modely kolových vozidel* [online] dostupné na internete: <http://robotika.cz/guide/odometry/cs> [01.06.2012]

[20] WOLF, D.F., SUKHATME, G.S., FOX, D., BURGARD, W. 2005 *Autonomous Terrain Mapping and Classification Using Hidden Markov Models* [online] dostupné na internete: <http://www.informatik.uni-freiburg.de/~burgard/postscripts/wolf05icra.pdf> [31.05.2012] International Conference on Robotics and Automation

9. Zoznam príloh

Príloha A: DVD médium – zdrojový kód softvéru robota, zkompilovaný systém pre riadenie robota sledovaním steny, elektronický dokument tejto práce, obrazové a video prílohy. Viď súbor Obsah.txt na DVD.