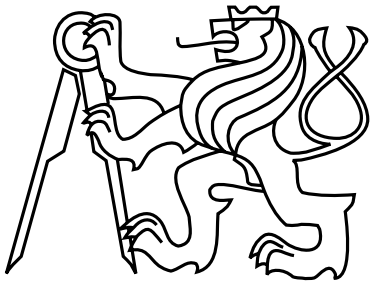
České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra měření



Diplomová práce

Plánování bezkolizní trajektorie pro mobilní roboty

Bc. Matouš Pokorný

Vedoucí práce: Ing. Michal Sojka, Ph.D.

Studijní program: Kybernetika a robotika, strukturovaný, navazující magisterský

Obor: Senzory a přístrojová technika

2. ledna 2012

Poděkování

Děkuji Ing. Michalu Sojkovi, Ph.D. za vedení diplomové práce, panu Ladislavu Čmelíkovi za pomoc s výrobou robota, kolegům s týmu Flamingos za spolupráci a cenné připomínky a hlavně mé rodině a blízkým, kteří mě po celou dobu studia podporovali.

Prohlášení

Prohlašuji, že jsem práci vypracoval samostatně a použil jsem pouze podklady uvedené v přiloženém seznamu.

Nemám závažný důvod proti užití tohoto školního díla ve smyslu §60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 2. 1. 2012 .....

Abstract

The mechanical concept of robot for competition Eurobot 2011 was designed and described in the first part of thesis. Chasis of robot, drives and manipulation actuators were designed. The Robot was made and team Flamingos participated successfully on the competiton Eurobot 2011 competiton.

The second part of thesis is collision-free path planning for a mobile robot for competition Eurobot. The used A* algorithm is extended with a new approach to representation of robot in the map. The robot is represented by the real shape and dimensions, not only as one point. This approach was implemented and tested in the robot software. The result is more reliable robot motion in the limited space.

Abstrakt

V první části práce je navržen a uveden mechanický koncept robota pro soutěž Eurobot 2011. Navrženo je šasi robota, pohonné jednotky a manipulační mechanismus. Robot byl dle návrhu vyroben a sestaven. Tým Flamingos se s ním úspěšně účastnil soutěže.

Druhá část práce se zabývá plánováním bezkolizní trajektorie mobilního robota pro soutěž Eurobot. Rozšiřuje dosud používaný A* algoritmus o nový přístup k reprezentaci robota v mapě. Robot je reprezentován svým reálným tvarem a rozměry, nikoliv jedním bodem. Tento přístup je implementován a otestován v kontextu programového vybavení robota. Výsledkem je spolehlivější pohyb robota v omezeném prostoru.

Obsah

Seznam obrázků

Seznam použitých zkratek

2D Two-Dimensional

3D Three-Dimensional

IRC Incrementary Rotation Coder

CAN Controller Area Network

ORTE OCERA Real-Time Ethernet

UDP User Datagram Protocol

Wi-Fi Wireless Fidelity

cpt counts per turn

scp secure copy

ssh secure shell

NFS Network File System

PPC PowerPC

PXMC Pikron eXtensible Motion Control

GNU GPL GNU General Public License

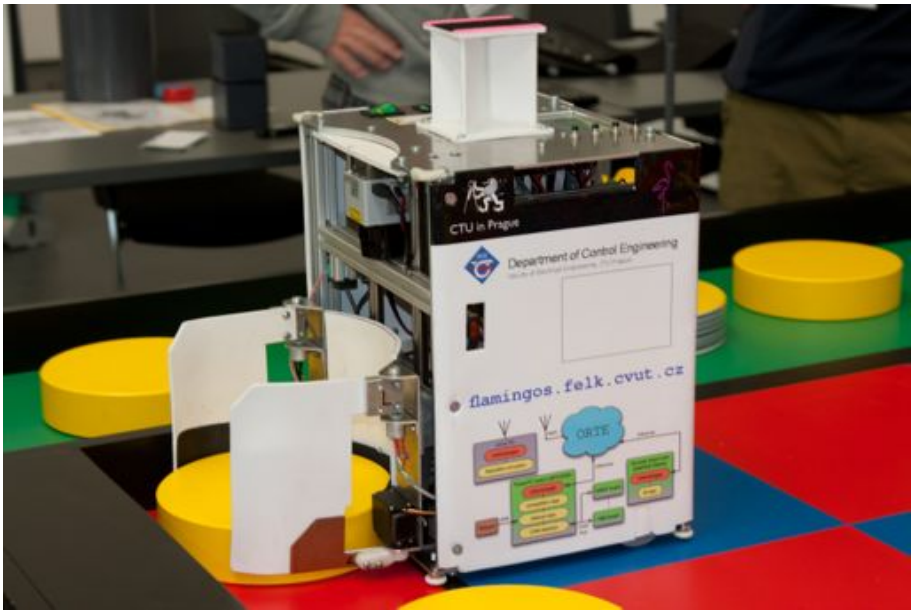
GPIO General Purpose Input/Output

PNG Portable Network Graphics

IP Internet Protocol

Kapitola 1

Úvod



Obrázek 1.1: Robot týmu Flamingos pro soutěž Eurobot 2011

S bouřlivým vývojem techniky v posledních desetiletích se roboti dostávají do života každého z nás. Pomáhají lidem v nebezpečném prostředí, při chirurgických operacích, v domácnosti a průmyslu. Pole působnosti robotů se rychle rozrůstá. Robotika je již dnes chápána jako jedno z hlavních vědních a průmyslových odvětví. Specialisté v tomto oboru mají vědomosti z mnoha disciplín, z kybernetiky, mechaniky, elektrotechniky a dalších. Projekt robotické soutěže Eurobot, který je popsán v této práci, pomáhá se vzděláváním specialistů v oboru. Studenti, účastnící se projektu, získají mnoho cenných teoretických a praktických vědomostí a zkušeností.

Diplomová práce ve dvou částech popisuje mechanickou a kybernetickou část mobilního robota pro robotickou soutěž Eurobot 2011. V první části je uveden mechanický koncept robota a v druhé rozšíření algoritmu plánování bezkolizní trajektorie.

V části týkající se mechaniky je navrženo šasi robota, pohonné jednotky a manipulační mechanismus. Do pohonných jednotek byly vybrány vhodné motory vzhledem k zadání a zkušenostem ze soutěží v minulých letech.

V druhé části, která se zabývá rozšířením algoritmu plánování bezkolizní trajektorie, je navržen a popsán nový přístup k reprezentaci robota v mapě prostředí. Tento přístup je do dosud užívaného algoritmu implementován. Výsledný software je otestován v simulacích a na reálném robotu.

Mechanický koncept robota, který byl navržen je popsán v kapitole 3. Pohonné jednotky jsou popsány v sekci 3.2, manipulační mechanismus v sekci 3.4, podvozek a ovládací panel tvořící společně šasi robota v sekci 3.1 a 3.3. Na základě návrhu byl robot vyroben a sestaven za pomoci kolegů z robotického týmu Flamingos, bývalého člena robotického týmu CTU Dragons (dnes Flamingos) Jana Bendy a pana Ladislava Čmelíka z Katedry řídicí techniky.

Rozšíření stávajícího algoritmu plánování trajektorie popisuje kapitola 4. Sekce 4.1 a 4.2 popisují teoretické předpoklady, problém a návrh řešení popisuje sekce 4.3. Implementace algoritmu je uvedena v 4.4, ověření funkce a výsledky pak v 4.5.

Kapitola 2

Robotická soutěž Eurobot 2011

Robotická soutěž Eurobot [12] je určena mladým lidem, kteří se zabývají robotikou. Jedná se o soutěž autonomních robotů, kteří plní dané úkoly. Téma soutěže se každý ročník mění, aby nově se účastnící týmy nebyly znevýhodněni a aby soutěž byla stále atraktivní. Převážná většina soutěžních týmů, v hlavní kategorii Eurobot, pochází z univerzit. Doplňkové kategorie Starter (od ročníku 2012 se tato kategorie nazývá Junior) se účastní hlavně středoškolské týmy, aby získaly nové vědomosti a zkušenosti důležité pro start v hlavní kategorii Eurobot. Soutěže se neúčastní pouze školy a univerzity, ale i nadšenci, robotické kroužky a kluby. Tým musí mít minimálně dva členy, jejichž věk nepřesahuje hranici třiceti let (neplatí pro supervizora týmu). Rozdíl mezi kategoriemi je v „inteligenci robota“. Robot v kategorii Eurobot musí být zcela autonomní, úkol plní bez pomoci člověka. V kategorii Starter je naopak vyžadováno řízení robota člověkem, člen týmu ovládá pohyb a chování robota pomocí dálkového ovladače. Ostatní pravidla jsou v zásadě shodná pro obě kategorie.

Soutěž Eurobot (kategorie Eurobot) probíhá ve dvou kolech. První kolo je národní, koná se na úrovni státu. Naše české kolo se konalo v Praze v termínu 6. – 8. května 2011 (hlavní soutěž 7. května, Robotický den [11]). Dva nejúspěšnější týmy a jeden tým, který obdrží cenu poroty, za zajímavé řešení nebo prostě třetí v pořadí, postupují dále do mezinárodního kola soutěže. Je snaha, aby se toto mezinárodní kolo konalo pokaždé v jiné zemi. Ročník 2011 mezinárodního kola hostilo Rusko v termínu 22. - 27. června v Astrachani (u hranic s Kazachstánem, nedaleko Kaspického moře).

Smyslem soutěže je především vzbudit zájem o vědu a techniku, rozvoj dovedností v technických disciplínách a v oblasti řízení projektů, výměna zkušeností a vědomostí mezi týmy a v neposlední řadě navázání nových přátelství [12]. Konstrukce robota je nejen práce, ale i zábava. Samotná soutěž je pak velkou společenskou událostí pro každého fanouška robotiky. Z

vlastní zkušenosti mohu říci, že účast v soutěži mi přinesla mnoho zkušeností a dovedností, které z pouhé výuky nelze získat.

2.1 Téma a pravidla soutěže Eurobot 2011

Ročník 2011 nese název Chess'Up! – „Šachovaná“. Téma je inspirováno hrou šachy. Reálně však připomíná spíše deskovou hru dáma. Dva soupeřící roboti mají přidělenou barvu svého „dresu“, která koresponduje s jednou barvou na šachovnici. Úkolem je manipulovat s herními prvky po hrací ploše tak, aby na konci soutěžního kola, které trvá 90 sekund, bylo na čtvercích šachovnice, v barvě daného robota, co nejvíce herních prvků. Herní prvek musí celou plochou podstavy ležet uvnitř čtverce šachovnice.

Obrázek 2.1: Hřiště Katedry řídicí techniky

Herní prvky představují pěšce, dámu a krále. Z těchto základních herních prvků je možné dále skládat figury, věže. Povoleny jsou kombinace pěšec + král / královna, 2 pěšci + král / královna. Za platnou je věž prohlášena, pokud jsou herní prvky vyskládány na sobě a vzájemně spojeny magnety ve správném pořadí, tj. jižní pól směřuje dolů a severní pól nahoru. Magnet je umístěn vždy ve středu podstavy válce, z kterého je herní prvek vyroben. Pěšec má magnety dva, ve spodní a horní podstavě. Král a královna mají magnet pouze ve spodní podstavě válce, protože musí být umístěni vždy na vrcholu věže. Král a královna jsou dále označeny čárovým kódem po obvodu podstavy válce. Kódy odpovídají standardu Code 39 a obsahují slova QUEEN a KING– (pomlčka doplňuje slovo na stejnou délku). Čárové kódy jsou orientovány svisle a jejich výška je 50 mm. Na každém poli šachovnice je možné umístit vždy jen jednu figuru, buď samotný herní prvek, nebo platnou věž. Každý herní prvek

a každá věž je bodově ohodnocena, vítězí tedy tým, jehož robot získá během kola vyšší součet bodů za postavené figury. Herní prvky a hřiště je zobrazeno na obrázcích 2.2 a 2.1.

Obrázek 2.2: Herní prvky a figury

V soutěži Eurobot je kladen velký důraz na bezpečnost a fair-play. Roboti se během hry nesmí vzájemně omezovat v pohybu. Pro herní strategii to znamená, že je možné během hry přesouvat figury i z polí šachovnice cizí barvy. Na hřišti je proto vymezená ochranná zóna, což je prostor o něco menší než čtverec šachovnice, kam robot může umístit jednoho pěšce, který je zde chráněn před další manipulací. Tuto figuru již dále během hry nesmí přesouvat žádný robot, ani ten, který ji tam umístil. Tímto je možné si pojistit alespoň minimální bodový zisk.

V oficiálních pravidlech jsou dále uvedeny podrobné technické specifikace a omezení na soutěžní roboty, jako například obvod robota při startu a během hry, výška robota, umístění a užívání lokalizačních systémů, pravidla bezpečnosti a fair-play, přesné specifikace herní plochy a všech prvků a tak dále. Podrobnosti zde nebudou uvedeny, je možné je nastudovat z oficiálních dokumentů. V dalším textu budou zmíněna některá další pravidla a detaily tak, aby byly v kontextu s konkrétním částí textu.

Český překlad pravidel je uveden v dokumentu v [27] a doplnění pravidel v dokumentu [13].

Kapitola 3

Mechanický koncept robota

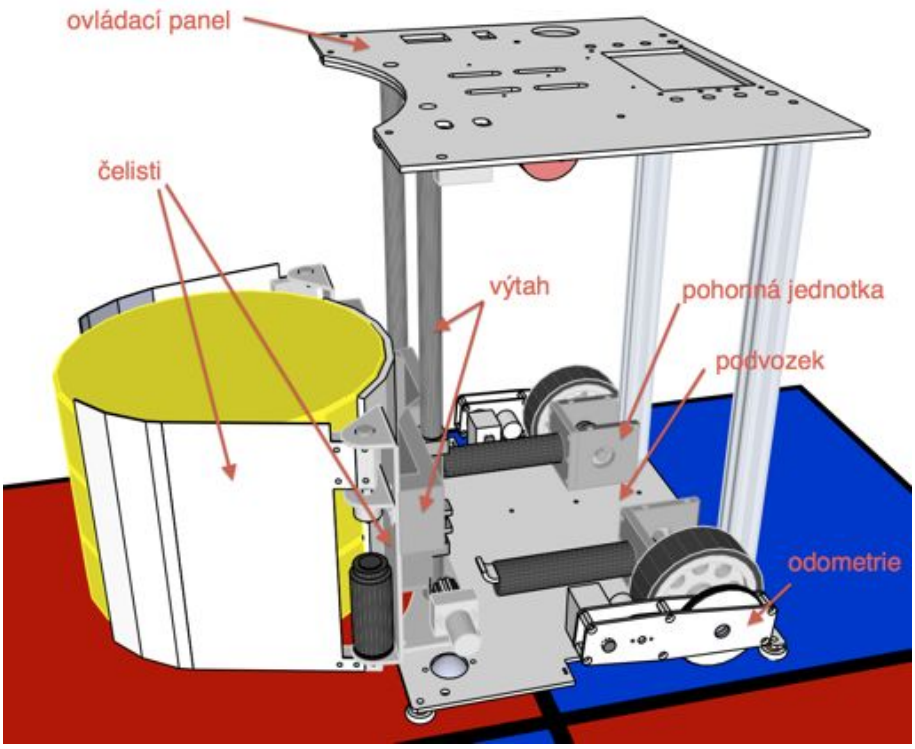
Soutěžní úkol je vždy založen na identifikaci a manipulaci s předměty. Robot plnící takovýto úkol musí být mobilní s manipulačním mechanismem. Mechanika našeho soutěžního robota se skládá z pohonných jednotek, manipulačního mechanismu – výtah a čelisti, odometrie a šasi, které tvoří ovládací panel s podvozkem. Společně jsou spojeny profilovanými hranoly. 3D model navrženého soutěžního robota je na obrázku 3.1.

Pro soutěž Eurobot 2011 bylo potřeba zkonstruovat celého robota od základu. Robot, se kterým jsme se úspěšně účastnili ročníku 2010, byl použit jako základ demonstrační platformy pro prezentace, dny otevřených dveří a podobně. Demonstrační platforma je náplní diplomové práce kolegy z týmu Michala Vokáče (vokacmic@fel.cvut.cz). Tato skutečnost je pozitivní i negativní. Můžeme změnit některé díly dle nových poznatků, ale zároveň nemáme hotové součásti, abychom je mohli rovnou použít.

Změny se týkají hlavně pohonné jednotky. Z rozměrových důvodů byl převod řemenem nahrazen kuželovým soukolím. Bezkomutátorové motory a převodovky byly vybrány z produkční řady firmy Maxon motor o průměru 22 mm. Návrh odometrie zůstal prakticky nezměněn. Dále byla navržena dolní základna robota, tzv. baseplate, a ovládací panel, který zároveň zpevňuje celou konstrukci. Pohony a odometrie se bez úprav budou nadále používat v dalších ročnících. Ostatní mechanické části se upraví vždy podle aktuálního zadání soutěže.

Mechanismus specifický pro tento ročník, který bude manipulovat s herními prvky se skládá ze dvou celků, čelistí a výtahu. Čelisti jsou kompletně navržené a vyrobené pro tuto soutěž. Vetší část výtahu je přejata jako funkční celek z jiného zařízení.

Mechanický koncept byl vytvořen na papíře, poté konzultován s celým týmem a nakonec překreslen do podoby 3D modelu v počítači. Model slouží pro dokumentaci a vizualizaci, na které lze ověřit funkce mechanizmů. Z 3D modelu se také generují data pro výrobu



Obrázek 3.1: 3D model soutěžního robota

mechanických součástí. Jako modelovací software byl použit volně dostupný program Google SketchUp [29]. Tento neprofesionální návrhový program byl zvolen protože se jednoduše ovládá, je rozšířitelný pomocí zásuvných modulů, je dostupný zdarma a lze ho provozovat na všech běžných operačních systémech. Každá osoba bez znalosti technického kreslení v tomto programu může rychle vytvořit vizualizaci prakticky čehokoliv.

3.1 Podvozek

Robot je koncipován jako mobilní kolový robot. Robot se pohybuje pomocí dvou nezávisle hnaných kol a pojezdových kuliček, obrázek 3.2. Pohyb robota je řízen diferenciálně. Otáčení robota, tj. úhlová rychlost je řízena vzájemným poměrem otáček jednotlivých kol. Řízení otáček motoru využívá zpětné vazby z inkrementálního rotačního snímače (dále jen IRC), který je zabudován v těle motoru. Snímány jsou otáčky motoru před převodovkou.

Hnaná kola by bylo výhodné umístit do středu základny podvozku, jejíž tvar by se měl co nejvíce podobat kruhu. Toto uspořádání je výhodné pro bezproblémový pohyb robota po hřišti a dovoluje zjednodušit plánovací algoritmus robota. Navržené řešení je jiné vzhledem

k využití prostoru uvnitř robota. Problém plánování trajektorie takovéhoto robota je popsán v kapitole 4 této diplomové práce.

Celý podvozek, obrázek 3.2, se skládá ze základny, ke které jsou připevněny dvě pohonné jednotky, dvě jednotky nezávislé odometrie a dvě pojezdové kuličky. Základna je vyrobena z 3 mm silného duralového plechu. Zkušenost z loňského ročníku ukázala, že je potřeba volit vhodný materiál a tvarování základny (mělké výřezy, co nejméně otvorů) tak, aby nedocházelo ke krutu základny a tím i k nedokonalému kontaktu hnaných kol s povrchem hřiště.

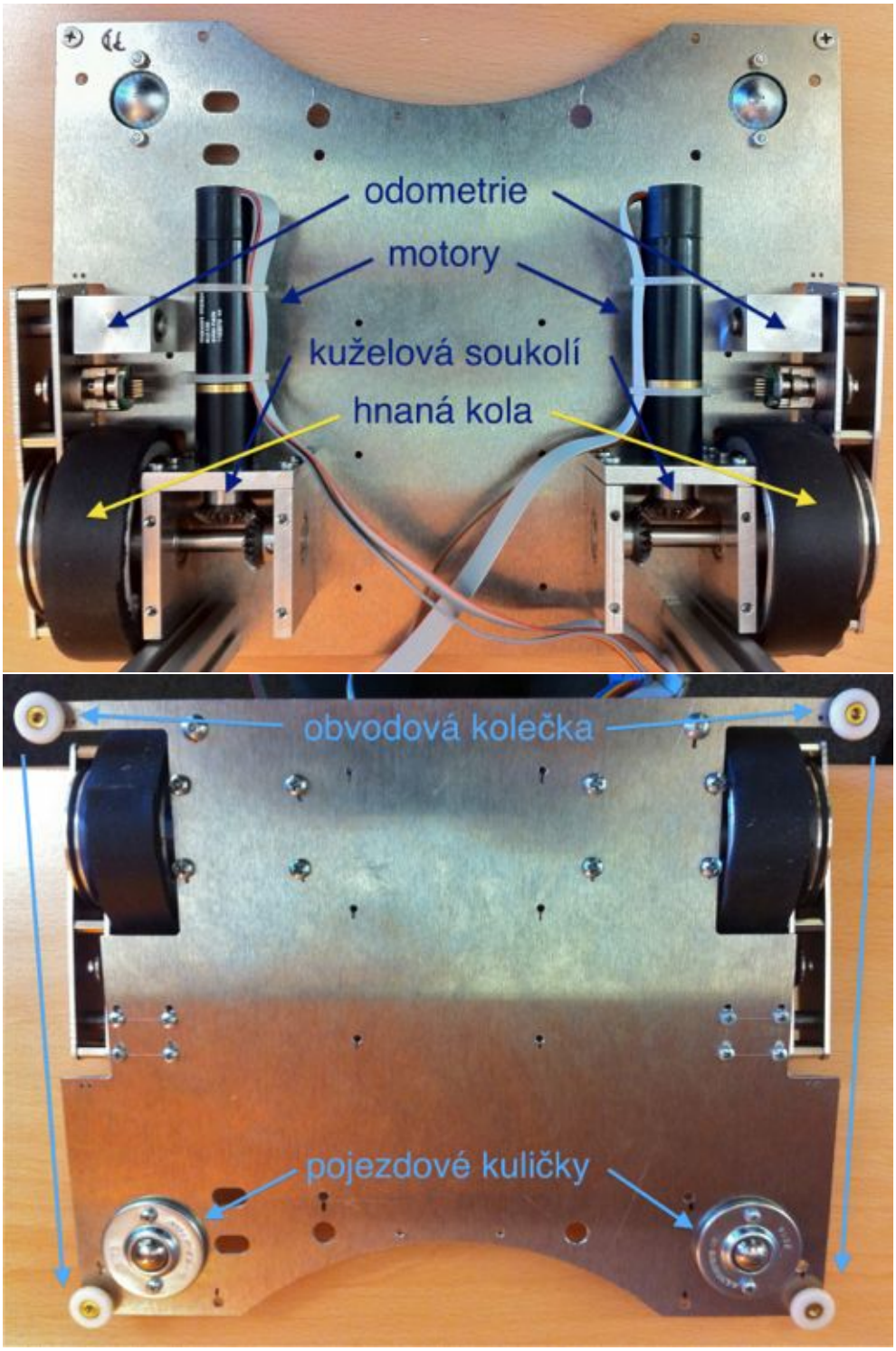
Pojezdové kuličky (kuličkové dopravní jednotky od firmy ALWAYSE [28]) umožňují pohyb v rovině (2 stupně volnosti) a slouží jako další opěrné body robota. V rozích čtvercového podvozku jsou upevněna obvodová kolečka, obrázek 3.2, ze silonu s vertikální osou otáčení. Jejich účelem je zamezit uváznutí robota, pokud dojde ke kolizi s překážkou. Když robot svým rohem zavadí o překážku, např. nerovný spoj dvou bočnic hřiště, která mu brání v otáčení, kolečko se protočí, robot se uvolní a může pokračovat v pohybu.

Nezávislá odometrie se používá pro lokalizaci robota a je koncepčně přejata z ročníku 2010. Navrhl ji bývalý člen týmu CTU Dragons (dnes Flamingos) Jan Benda. Odometrie vykazuje jen velmi malé mechanické vůle, využívá IRC AVAGO 3300 s velkým rozlišením (4096 cpt / 16384 cpt při kvadraturním dekódování) a tak se robot přesně a spolehlivě lokalizuje na herní ploše. Ročník 2010 ukázal, že i při prokluzu a smýkání hnacích kol a neplánovaných pohybech celého robota byla chyba lokalizace na konci zápasu menší než 3 cm v obou osách a 5 ° v úhlu natočení. Odometrie je tedy považována za přesnou a velmi spolehlivou a proto se na její koncepci nic nezměnilo.

Při návrhu robota tým uvažoval i o tříkolovém všesměrovém podvozku [30] jako použil konkurenční tým FELaaCZech [31] z Katedry kybernetiky FEL ČVUT. V současné době máme dobře odladěný algoritmus pro řízení mobilního robota s diferenciálním dvoukolovým podvozkem, mnoho dílů se vyrábělo a navrhovalo od začátku, takže by to bylo mnoho změn naráz, které by se nepovedlo včas odladit. Tuto myšlenku jsme tedy zavrhli.

3.2 Pohon

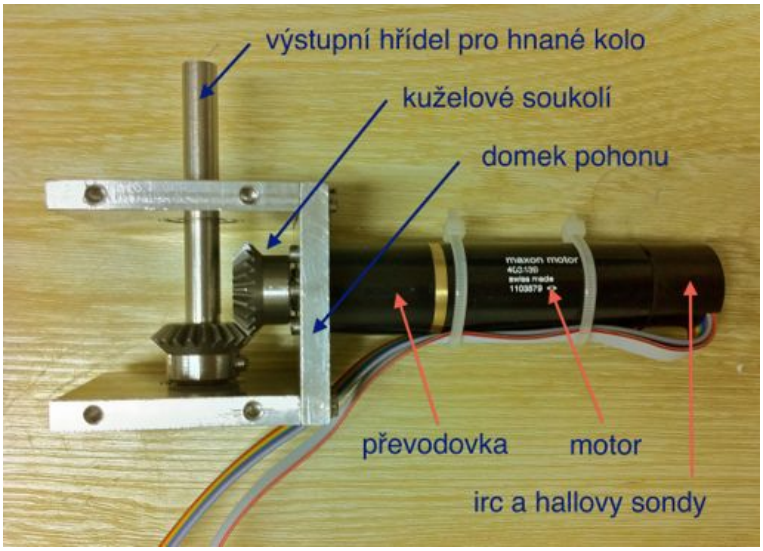
Pohon robota obstarávají dvě pohonné jednotky (diferenciální řízení). Jedna pohonná jednotka se skládá z elektrického motoru, domku převodovky s kuželovým soukolím, obrázek 3.3, které přenáší moment motoru do pravého úhlu na hnané kolo robota, a samotného hnaného kola. Motor je dodáván jako volitelně konfigurovatelná sestava sesazená v jeden kompaktní celek. V našem případě motorová sestava obsahuje samotný bezkomutátorový



Obrázek 3.2: Podvozek robota, pohled shora a zdola

motor, planetovou převodovku, IRC snímač otáček motoru před převodovku a Hallovy sondy pro snímání úhlu natočení motoru, kterých se využívá k elektronické komutaci motoru při rozběhu.

Požadavkem na pohonnou jednotku byla kompaktnost, aby tvořila jeden snadno oddělitelný celek, a univerzálnost, pohon se bude beze změny používat v následujících ročnících soutěže. Domek převodovky je tedy navržen tak, že umožňuje montáž pohonu horizontálně i vertikálně. Loňský robot (nyní demo robot) využíval převod řemenem, obrázek 3.4. Motor nebyl upevněn k řemenice s hnaným kolem, montáž a demontáž pohonu tedy byla problematická a zdlouhavá. Dále celý převod zabíral v omezeném prostoru mnoho místa (robot je dle pravidel pro ročník 2011 omezen výškou – 350 mm a obvodem pří startu – 1200 mm / během hry – 1400 mm) a řemen se musel chránit před stykem s ostatními součástmi. Výhodou převodu řemenem byla variabilita uspořádání, motor nemusel být v těsné blízkosti hnaného kola a řemen kompenzoval nepřesnosti výroby, jako např. nesouosost motoru a řemenice.



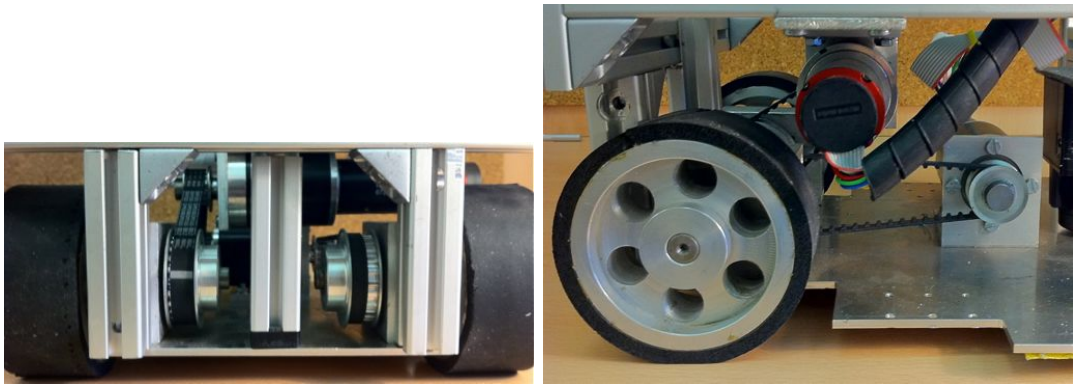
Obrázek 3.3: Navržený pohon robota

Letos byl tedy navržen převod kuželovým soukolím. Mnoho zahraničních týmu tento způsob využívá, takže jsme ho považovali za vhodný a spolehlivý (z principu by hnaná kola mohla být přímo na hřídeli převodovky, problémem jsou rozměry, celá sestava motoru je dosti dlouhá a nemusely by se vedle sebe dvě sestavy na šířku robota vměstnat). Původně se uvažovalo o plastovém kuželovém soukolí (nemusí se mazat, méně se opotřebovává), bohužel však v daných rozměrech (modul převodu) nejsou plasty schopné zajistit přenos námi požadovaného momentu na hnaná kola. Kuželové soukolí je tedy ocelové, promazané vazelínou a domek převodovky je zakrytý plastem bránícím vniku nečistot.

Problémem kuželového soukolí je požadavek na přesnost výroby. Osy ozubených kol musejí svírat přesně pravý úhel a zuby převodu mají vůči sobě definované polohy [20]. Pohon byl tedy vyroben tak, aby bylo možné vzájemnou polohu ozubených kol jemně doladit. Když se podařilo sesazení tak, že celý pohon měl hladký chod, byla poloha všech točivých součástí fixovaná speciálním průmyslovým lepidlem na ložiska (Loctite 638). Odladění hladkého chodu nebylo jednoduché, nyní však je pohon jako jeden díl a není potřeba se převodem dále zabývat.

Hnaná kola robota zaručují bezproblémový pohyb robota po hřišti. Návrh byl prakticky přejat z minulých let [2], kola byla pouze zúžena z 35 mm na 25 mm, zachován byl průměr 70 mm (celkový průměr s pneumatikou je 80 mm). Adhezi kol s povrchem hrací plochy zajišťují pneumatiky, které tvoří 5 mm silná mikroporézní guma (prodejna Guma, v Ječné ulici v Praze) nalepená po celém obvodu kol. Guma na sebe během pohybu robota přichytává nečistoty, což snižuje adhezi pneumatik. Při vyšších zrychleních robota může dojít k prokluzu kol, je tedy dobré pneumatiky občas očistit a udržovat povrch hřiště bez prachu a nečistot.

Navržený pohon robota pro ročník 2011 je na obrázku 3.3, pohon robota užívaný v minulých ročnících soutěže Eurobot je pak na obrázku 3.4.



Obrázek 3.4: Pohon robota pro soutěž Eurobot 2010

3.2.1 Výběr a dimenzování motoru

Pohon bylo potřeba navrhnout kompletně od začátku. Hlavním požadavkem bylo zmenšení rozměrů a kompaktnost. Při návrhu jsem vycházel z [2] a [8].

Parametry zadání:

- m = 10 kg – celková hmotnost robota

- v = $2\,\mathrm{ms}^{-1}$ – dopředná rychlost
- a = $2{,}5\,\mathrm{ms}^{-2}$ – zrychlení

Dle vypočtu ideálního pohonu:

$$F = ma = 10 \cdot 2,5 = 25\,N \tag{3.1}$$

síla, kterou působí jeden motor je tedy $25/2 = 12,5N$, dále je uvažován jeden motor

$$M = Fr = 12,5 \cdot 0,04 = 0,5\,Nm \tag{3.2}$$

$$v_{ot} = \frac{60v}{2\pi r} = \frac{2 \cdot 60}{2\pi 0,04} = 477\,otmin^{-1} \tag{3.3}$$

$$P = Fv = 12,5 \cdot 2 = 25\,W \tag{3.4}$$

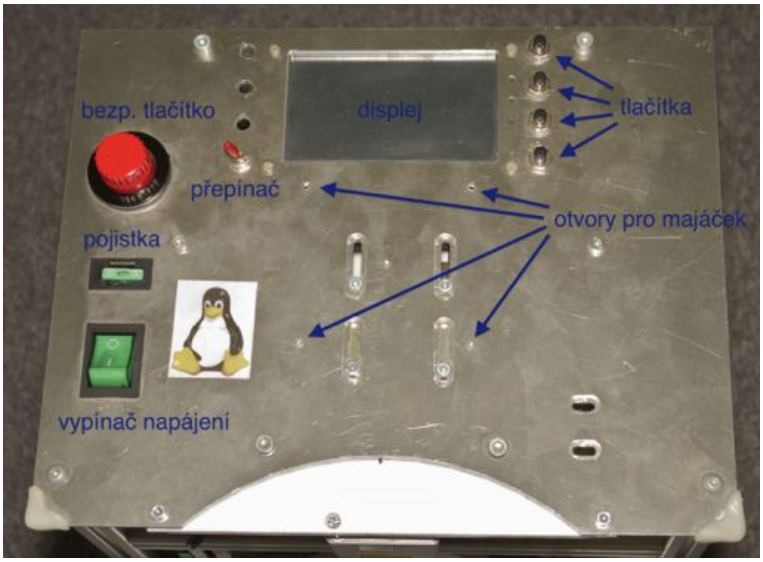
Motory byly vybrány z produkce švýcarské firmy maxon motor ag [17] (v ČR distribuuje firma Uzimex Praha, spol. s.r.o.). S těmito motory jsou na Katedře řídicí techniky dobré zkušenosti, není tedy důvod měnit výrobce. Výrobce poskytuje zdarma program maxon Selection Program (mSP[1]), který vybere optimální pohon vzhledem k zadaným parametrům. Do programu byly zadány parametry dle výpočtu $v_ot = 480\,otmin^{-1}$, $M = 0,5\,Nm$, $P = 25\,W$ a uvažovaný převod 1 : 1. Z nabízených možností byl vybrán bezkomutátorový motor, s integrovaným Hallovým snímačem úhlu natočení, typu EC o průměru 22 mm a výkonu 100 W, to je zbytečně mnoho, paradoxně však 30 W motory s vhodnou převodovku jsou podstatně rozměrově větší. Na motor navazuje planetová převodovka GP 22 HP, s převodovým poměrem 29 : 1, o průměru 22 mm určená pro zvýšenou zátěž, převodová kola jsou ocelová. Rychlost otáčení hnaných kol je řízena zpětnovazebně. Hallův snímač motoru má příliš malé rozlišení, proto je na vývod hřídele motoru připevněn IRC typu MR M 512 s rozlišením 512 cpt, tj. 14848 cpt na otáčku hnaného kola. Ve výpočtu nejsou uvažovány mechanické a elektrické ztráty, neboť jejich přesnou hodnotu nelze v našich podmínkách přesně určit. Motorová soustava je tedy vybrána tak, aby měla dostatečnou výkonovou rezervu. Při maximálních požadovaných hodnotách je zatížení motoru 67%, pro ztráty je tedy uvažováno 30% výkonu.

Výstup z program maxon Selection Program je v příloze A.

[1] http://www.maxonmotor.com/downloads.asp - do pole **Search** – **Decription** vepsat mSP a nechat nalézt tlačítkem **Go**

3.3 Ovládací panel

Ovládací panel, obrázek 3.5, slouží k ovládání a provozní diagnostice robota a nazývá se tak vrchní část robota, horní duralový kryt. Je v něm umístěn barevný displej, bezpečnostní tlačítko, hlavní vypínač, 30 A automobilová pojistka napájení z baterie, ovládací tlačítka a přepínače, montážní otvory konstrukce a majáčku. Vrchní kryt tedy neslouží pouze jako ovládací panel, ale i ztužuje celou konstrukci robota a nese držák majáčku. Ovládací panel s podvozkem spojují čtyři kusy hliníkových profilů ITEM 20 × 20 mm a vodící tyče manipulačního mechanismu.



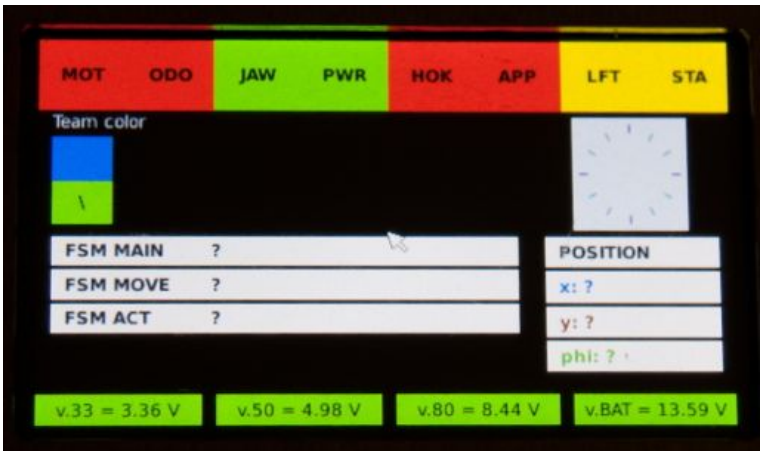
Obrázek 3.5: Ovládací panel robota

Barevný dotykový displej slouží k zobrazení provozních údajů robota, obrázek 3.6:

- Aktuální hodnoty palubních napětí – 3,3 V **v.33**, 5 V **v.50**, 8 V **v.80** , napětí baterie **v.BAT** (změna barvy signalizuje stav vybití baterie)
- **x**, **y** – poloha robota na herní ploše v kartézských souřadnicích X, Y (poloha osy otáčení robota, orientace os viz obrázek 3.7)
- **phi** – úhel natočení do směru jízdy (úhel se odměřuje proti směru hodinových ručiček od osy X)
- Tříhodnotový stav částí robota – motory **MOT**, odometrie **ODO**, čelisti **JAW**, napájení **PWR**, laserový 2D dálkoměr **HOK**, aplikace řídící robota **APP**, výtah **LFT**,

startovací spínač **STA**, barva signalizuje stav jednotky (zelená – v pořádku, červená – chyba, žlutá – připraveno, vyžaduje manuální činnost)

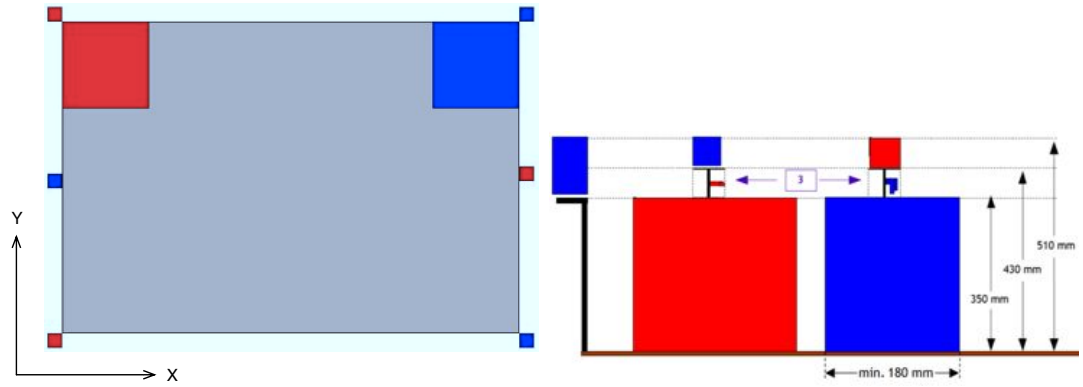
- **Team color** barva dresu robota – modrá nebo červená
- Aktuální stavy stavových automatů – pohyb **FSM MOVE**, aktuátory **FSM ACT**, hlavní automat **FSM MAIN** (herní strategie)
- Otáčející symbol signalizuje, že displej komunikuje a zobrazuje aktuální informace (zpětné lomítko na zeleném poli)



Obrázek 3.6: Informace na displeji

V soutěži Eurobot se klade velký důraz na bezpečnost, proto dle pravidel [27] musí být každý robot vybaven „velkým červeným tlačítkem“, kterým lze v případě nekorektního chování robota odpojit napájení všech aktuátorů, tzn. hnací motory, pohony manipulačních mechanismů a jím podobné. Toto tlačítku musí mít minimální průměr 20 mm a musí být snadno shora dostupné. Maximální výška horní hrany tlačítka nad hrací plochou je 370 mm (ostatní součásti robota maximálně 350 mm). Po odpojení bezpečnostním tlačítkem zůstávají všechny řídící desky napájeny, aby byly zachovány aktuální stavy robota a bylo je možné zpětně vyčíst pro další analýzu. Pro naší potřebu je v napájecím obvodu zapojen vypínač, který odpojí napájení baterie kompletně, a vratné PTC pojistky 3,75 A. Pojistky jsou dvě, jedna pro hnací motory a druhá pro ostatní aktuátory.

Základní úlohou mobilního robota v soutěži Eurobot je bezpečný pohyb po hřišti. Robot se tedy potřebuje lokalizovat na hřišti a detekovat překážku (oponenta). Pro usnadnění této úlohy je povoleno použití majáčků (aktivní nebo pasivní zařízení napomáhající lokalizaci robota), tři kusy pevných majáčků, umístěných po obvodu hrací plochy, viz obrázek 3.7,



Obrázek 3.7: Orientace os a majáčků vzhledem k hřišti, obrázky přejaty z [27]

a jeden kus lokalizačního majáčku, který se může umístit buď na vlastního nebo soupeřova robota podle toho, jakého robota je potřeba lokalizovat. Stacionární majáčky mají tvar kvádru. Rozměry podstavy jsou 80 × 80 mm a výška je 160 mm. Lokalizační majáček má tvar krychle, délka hrany je 80 mm. Majáček by měl být na robotu umístěn co nejblíže středu jeho podstavy ve výšce, kterou znázorňuje obrázek 3.7. Majáček je k podstavci připevněn pomocí suchého zipu, kde na podstavě jsou háčky. Majáčky mohou být pasivní (např. obrázky, jakési značky pro kameru) nebo aktivní, pak mají možnost mezi sebou (i drátově) a s robotem jakkoliv bezdrátově komunikovat, což je jinak robotu výslovně zakázáno. Robot kromě majáčků nesmí nijak komunikovat s žádným jiným systémem nebo člověkem.

Podstava lokalizačního majáčku našeho robota je připevněna pomocí konstrukce z plastových desek k ovládacímu panelu. Slouží pouze oponentovi, protože náš robot v letošním roce nikterak majáčků nevyužívá.

3.4 Manipulační mechanismus

Soutěžní úloha je každý ročník jiná, aby nově se účastnící týmy nebyly znevýhodněny. K řešení úlohy je většinou zapotřebí specifický mechanismus, který již nelze opakovaně použít v dalších ročnících. Úlohy jsou vždy založené na manipulaci a přesunu břemen. V letošním roce je potřeba přesunout a případně klást na sebe předměty, symbolizující pěšce, krále a dámu. Jsou to válce o průměru 200 mm a výšce 50 mm, jejich hmotnost se pohybuje mezi 200 g a 700 g. V případě věže může být maximální hmotnost, kterou je potřeba zdvihnout 1200 g. Manipulační mechanismus tedy musí pracovat spolehlivě a hladce a zároveň být dostatečně pevný a robustní. Mnou navržený mechanismus pracuje jako sériový manipulátor s jedním posuvným kloubem ve svislé orientaci, na kterém je umístěno chapadlo. Pracovně

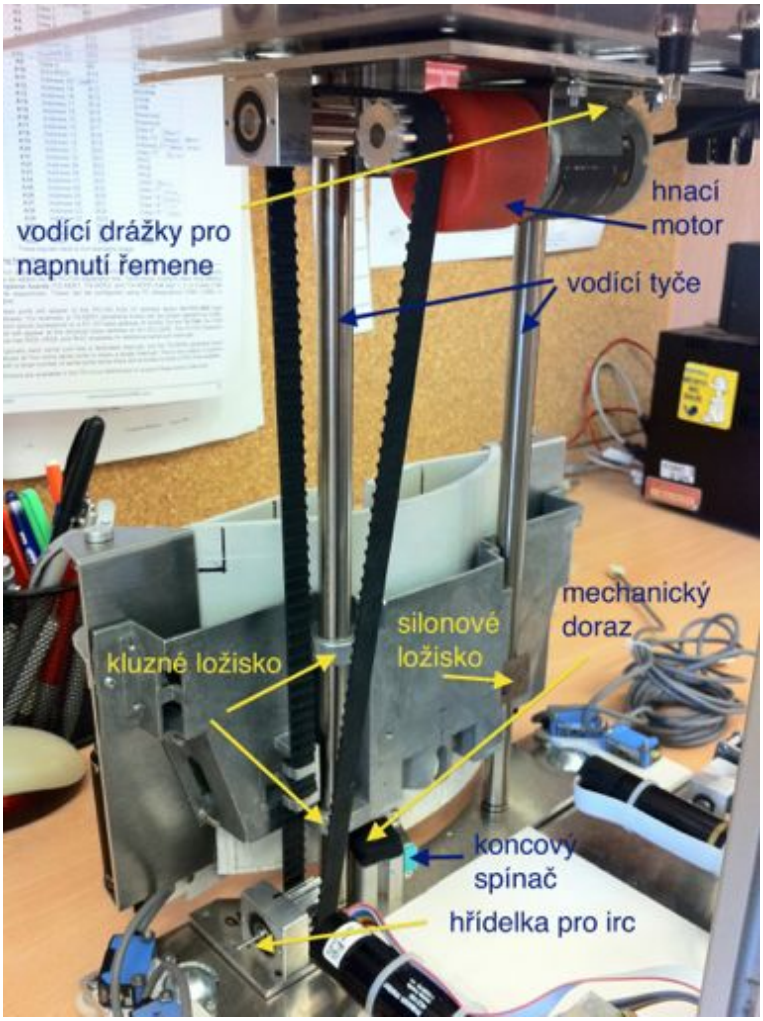
jsme ho rozdělili na dvě části – výtah a čelisti.

3.4.1 Výtah

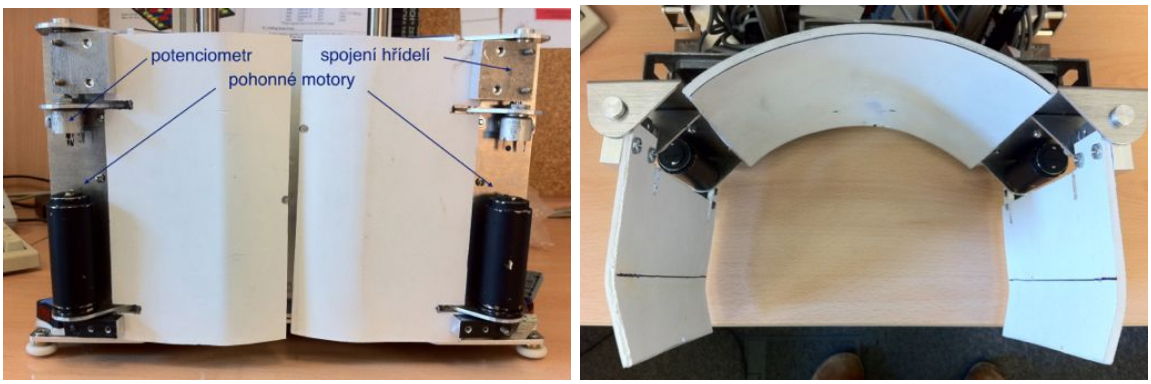
Aby bylo možné stavět věže, je potřeba herní prvky zdvihat a klást na sebe. Výška pěšce je 50 mm, takže potřebný zdvih mechanismu v případě dvojité věže je 100 mm. Námi použitý mechanismus disponuje zdvihem až 165 mm. Základem výtahu, obrázek 3.8, je lineární vedení (dvě vodící tyče a jezdec) převzaté z vyřazeného profesionálního skeneru HP 4c. Lineární vedení se pohybuje lehce i se zátěží, má jedno pevné kovové kluzné ložisko a jedno silonové ložisko otevřené, které kompenzuje nepřesnosti a umožňuje plynulý pohyb jezdce i při nedokonalé rovnoběžnosti vodících tyčí. Vodící tyče z hlazené oceli jsou upevněny do duralových čel, které jsou připevněny k podvozku a ovládacímu panelu, celý manipulační mechanismus lze tedy snadno demontovat jako samostatný díl. Pohyb mechanismu až do krajních poloh je omezen mechanickými dorazy s koncovými spínači. Posun jezdce zajišťuje polyuretanový řemen s lichoběžníkovým palcovým ozubením PowerGrip XL (Contitech 280 XL, Uzimex Praha, spol. s.r.o.), který je poháněn stejnosměrných motorem MFA/COMO DRILLS 919D (4,5 - 15 V) s konvenční převodovkou 50 : 1. Přenos síly z motoru na řemen a vedení řemene zajišťují řemenice XL s 12 zuby (průměr 19 mm). Řemen se napíná polohou motoru v montážních drážkách. Polohu jezdce výtahu odměřuje IRC AVAGO 3300 umístěný na dolní řemenici společně s dolním koncovým spínačem, který určuje minimální polohu jezdce. Při daném průměru řemenice je odměřování polohy jezdce v rozlišení 272 kroků / mm při kvadraturním vyčítání IRC.

3.4.2 Čelisti

Pro zdvih a přesnou manipulaci s herními prvky je důležité, aby robot mohl herní prvky pevně uchopit. K tomu slouží čelisti, obrázek 3.9, které jsou umístěny na jezdci výtahu. Pokud se herní prvek ocitne mezi čelistmi, čelisti obejmou herní prvek a robot s ním může dále manipulovat vodorovně i svisle. Dle dodatku pravidel [13] má každý herní prvek po obvodu základního válce zkosenou hranu pod úhlem 45 °(2 × 2 mm). Snažili jsme se tuto informaci využít, ale výsledky nebyly pozitivní, vnitřní strana čelistí je tedy vypolstrována měkkou pěnou a gumou z duše jízdního kola, aby se zlepšilo tření mezi čelistmi a herními prvky. Čelisti jsou polohovány stejnosměrnými motory Faulhaber s konvenční převodovku, které jsou řízeny na polohu pomocí P regulátoru a řídící desky eb_board [7]. Úhel natočení každé čelisti odměřuje potenciometr.



Obrázek 3.8: Část manipulačního mechanismu – výtah



Obrázek 3.9: Část manipulačního mechanismu – čelisti

3.5 Elektronika

V této sekci jsou uvedeny některé informace týkající se elektroniky robota v kontextu k mechanice.

Odometrie Elektronický modul odometrie s mikrokontrolérem typu Renesas H8S/2638 dekóduje signál z IRC snímačů a po CAN sběrnici robota pravidelně posílá číselné hodnoty. Každé číslo vyjadřuje celkový počet kroků IRC snímače za dobu běhu programu v procesoru od posledního resetu nebo přerušení napájení. Počet kroků snímačů je přímo úměrný ujeté dráze robota. Mikrokontrolér má přímo v pouzdře integrovánu periférii dvoukanálového kvadratůrního enkodéru. Každý mikrokontrolér H8S je tedy schopen kvadratůrně zpracovávat signál ze dvou IRC snímačů. Elektronická deska odometrie a řízení motorů obsahuje tedy dva mikrokontroléry H8S, protože je potřeba obsloužit celkem čtyři IRC snímače, dva snímače nezávislé odometrie a dva snímače motorů. Programové vybavení obou procesorů je až na identifikátory zpráv CAN sběrnice prakticky shodné.

Řídicí jednotka motorů Řídící signály pro silovou část elektroniky motorů generuje mikrokontrolér Renesas H8/2638 na základě informace o poloze z Hallových sond a IRC snímače. Po zapnutí elektroniky se využívá informace z Hallových sond a průběh napětí na fázích motoru odpovídá blokové komutaci. To je nejjednodušší způsob řízení napětím pravoúhlého průběhu, v podstatě kopíruje funkci mechanického komutátoru. Informace o poloze kotvy z IRC snímače se uvažuje od okamžiku, kdy se zaregistruje indexová značka snímače, která značí celou otáčku motoru a tvoří tak vztažný bod jinak relativního IRC snímače. IRC podává mnohem přesnější informaci o poloze kotvy motoru a bloková komutace může přejít v komutaci sinusovým signálem. Na fázích motoru je pak sinusový průběh napětí. Díky tomu má motor plynulejší chod, vyšší účinnost a velký rozběhový moment. Program mikrokontroléru využívá universální knihovnu pro řízení motorů PXMC [4], kterou vyvinula společnost PiKRON s.r.o. a je šířena pod licencí GNU GPL.

Řídicí jednotka výtahu Řídící deska eb_board nedisponuje kvadraturním enkodérem, IRC je tedy vyčítán softwarově ze dvou běžných GPIO pinů s nejvyšší možnou vzorkovací frekvencí (více jak 100 kHz). Inkrement a směr otáčení IRC se určuje pomocí dekódování Grayova kódu [21].

3.6 Poznámky k užívání a dalšímu vylepšení mechaniky robota

Základna podvozku Zkušenost z loňského ročníku ukázala, že je potřeba volit vhodný materiál a tvarování základny (mělké výřezy, co nejméně otvorů) tak, aby nedocházelo ke krutu základny a tím i k nedokonalému kontaktu hnaných kol s povrchem hřiště. Tento problém se novým návrhem podařilo odstranit, avšak při další revizi podvozku by se mělo uložení pohonných jednotek koncipovat jako kyvná náprava, aby byl definitivně vyloučen jakýkoliv vliv na kontakt hnaných kol s povrchem hřiště. Dalším možným řešením je odpružení pojezdových kuliček, což je konstrukčně jednodušší.

V další revizi by se mohlo též zvážit spojení pohonné jednotky s odometrií v jeden kompaktní celek. Nové řešení musí být stále spolehlivé a přesné co se týče odometrie a pohonná jednotka by měla být umístitelná na výšku i na šířku jako v tento okamžik.

Čelisti Později byly motory nahrazeny běžnými modelářskými servy Hitec HS-300. Aby se nezničily převody serv, spoj mezi servy a samotnými čelistmi byl realizován přes mechanické pojistky, které jsou příslušenstvím k servům. Pokud dojde k přílišnému krutu v pojistce, protočí se a vnější zatížení se nepřenese do převodovky serva. Takovéto pojistky je třeba užívat na všech místech, kde může dojít k mechanickému přetížení citlivých a drahých součástí. Zničení celého motoru je finančně náročné, nehledě na dobu dodání náhradního dílu. Podobným způsobem je třeba jistit všechny mechanické části, např. dorazy u výtahu.

Některé podrobné návody jako, např. práce s gravírovací frézkou Comagrav MT, jsou popsány na wiki týmu Flamingos [23].

Všechny 3D modely, konstrukční výkresy a podklady pro výrobu jsou na přiloženém CD, příloha B, nebo vždy aktuální v repositáři týmu Flamingos.

Kapitola 4

Plánování bezkolizní trajektorie

Pohyb v prostoru je základním úlohou mobilního robota. Autonomní robot musí sám řešit krizové situace během pohybu v prostředí (reálném i virtuálním), musí reagovat na své okolí. Bezkolizní plánování trajektorie je proces, při kterém je nalezena bezpečná trasa ze startovního do cílového bodu v prostředí robota. Soutěžní robot se pohybuje po hřišti, kde jsou pevně dané překážky, náhodně umístěné předměty bránící v pohybu a robot protivníka, se kterým nesmí přijít do kolize. Bezpečný pohyb soutěžního robota v omezeném prostředí je základní předpoklad splnění soutěžního úkolu.

Tato kapitola je strukturována následovně. Problém zde řešený je popsán v sekci 4.3.1. Předpoklady pro řešení problému jsou uvedeny v 4.1, 4.2. Řešení problému je popsáno v 4.3.1, 4.3.3, konkretní implementace v 4.4 a testování v 4.5.

4.1 A* algoritmus

Algoritmus A* (A star) [5] slouží k prohledávání stavového prostoru, konkrétně k nalezeni nejkratší cesty tímto prostorem, přičemž bere v úvahu určitá omezení. Vstupem algoritmu je ohodnocený graf, počáteční a cílový uzel hledané cesty. Výstupem algoritmu je pak nejkratší (či jinak optimální) cesta z počátečního do cílového uzlu nebo cesta žádná, pokud neexistuje.

Princip algoritmu vychází z algoritmu prohledávání do šířky (BF) [5], využívá však prioritní frontu O a hodnotící funkci $f(n)$. Uzly (vlastně celé cesty) jsou ve frontě řazeny vzestupně dle hodnoty hodnotící funkce $f(n) = h(n)+g(n)$. Ta vyjadřuje předpokládanou délku celé cesty ze startovního do cílového uzlu. Heuristická funkce $h(n)$ vyjadřuje odhad vzdálenosti z aktuálního do cílového uzlu. Pokud graf vyjadřuje mřížku se čtvercovými buňkami, lze volit heuristickou funkci jako Eukleidovská vzdálenost [26]. Předpokládá se, že hodnota

heuristické funkce $h(n)$ je nenulová, kladná a že heuristika je tzv. přípustná, tj. že odhad vzdálenosti je vždy hodnota menší nebo rovna vzdálenosti reálné. Funkce $g(n)$ vyjadřuje již uraženou reálnou vzdálenost (délku cesty) ze startovního do aktuálního uzlu, je to součet hodnot všech hran mezi uzly již uražené cesty.

Dále se předpokládá, že prioritní fronta neobsahuje kružnice (graf není cyklický) a počet možných cest je konečný.

A* algoritmus lze popsat následovně, přejato z [5]. Definice pojmů:

- O – množina (prioritní fronta) dosud nezpracovaných uzlů
- C – množina již zpracovaných uzlů
- q_{goal} – cílový uzel, konec cesty
- q_{start} – startovní uzel, počátek cesty
- n_{best} – první uzel v prioritní frontě O, nejvhodnější kandidát k expanzi
- $N(n_{best})$ – množina uzlů sousedících s n_{best}
- $g(n_{best})$ – délka cesty z q_{start} do n_{best}
- $h(n_{best})$ – heuristická funkce, odhadovaná nejkratší vzdálenost z n_{best} do q_{goal}
- $f(n_{best}) = h(n_{best}) + g(n_{best})$ – hodnotící funkce, nejkratší odhadovaná délka cesty z q_{start} do q_{goal} přes n_{best}
- *backpointer* – zpětný ukazatel, ukazuje na buňku n_{best}, z které byl x expandován

Algoritmus:

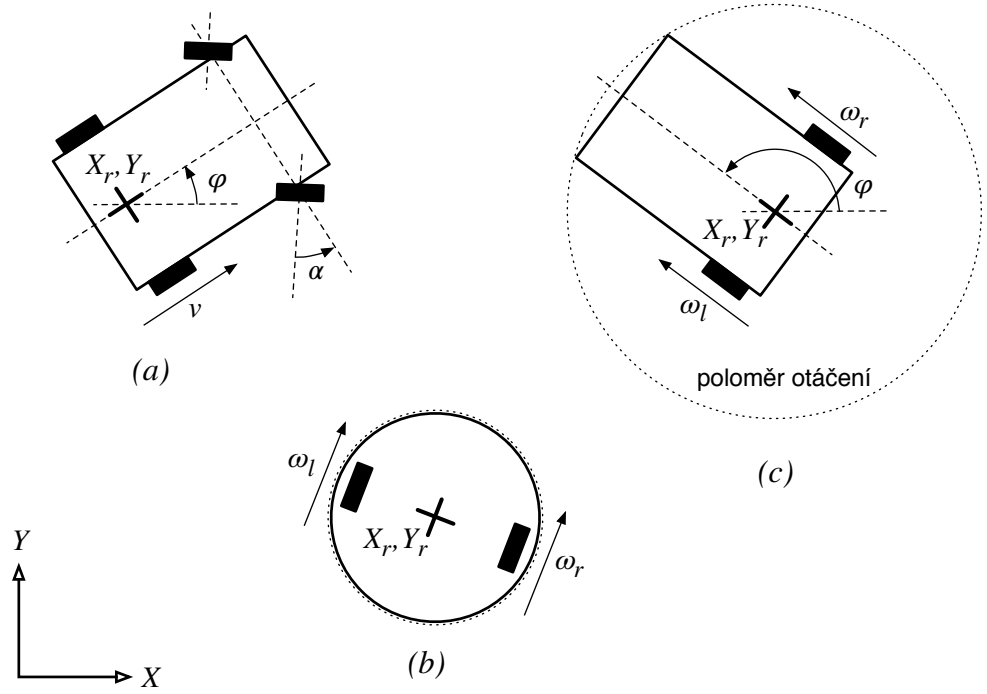
1. repeat
2. Vyber n_{best} z O takové, že $f(n_{best}) \leq f(n), \forall n \in O$
3. Vyjmi n_{best} z O a vlož ho do C
4. Pokud $n_{best} = q_{goal}$, ukonči algoritmus
5. Expanduj n_{best}: pro všechna $x \in N(n_{best})$, která nejsou v C
6. if $x \notin O$ then
7. přidej x do O
8. else if $g(n_{best}) + c(n_{best}, x) < g(x)$
9. aktualizuj *backpointer* uzlu x na n_{best}
10. end if
11. until O není prázdná

Názorný příklad, jak algoritmus postupuje v grafu, je uveden například v [24] nebo [5] .

4.2 Holonomní robot

Robot, který má všechny stupně volnosti řiditelné se nazývá holonomní [3].

Automobil je příklad neholonomního vozidla. Uvažujeme tři stupně volnosti v rovině – souřadnice X, Y v Kartézské soustavě souřadnic, úhel natočení φ. Automobil však má jen dva řiditelné stupně volnosti – dopředná rychlost vozidla v daná rychlostí otáčení hnaných kol a úhel natočení řiditelné nápravy α, obrázek 4.1 (a). V mobilní robotice se často užívá



Obrázek 4.1: Principiální schéma podvozku – automobil (a), jednoduchý kruhový robot (b), soutěžní robot (c)

zjednodušený koncept kolového robota kruhového tvaru s diferenciálním řízením [3], obrázek 4.1 (b). Osa otáčení robota je umístěna ve středu kruhové podstavy robota. Tento přístup má několik výhod. Diferenciální kolový podvozek je snadno řiditelný (rychlost otáčení pravého ω_r a levého ω_l kola). Robot má do všech směrů konstantní rozměr, může se tedy bez kolize otáčet v jednom bodě do libovolného směru. Pokud se předpokládá takovýto robot, který může dosáhnout libovolného úhlu natočení φ pouhým otočením kolem své osy, aniž by se při tomto manévru zvýšila pravděpodobnost kolize, můžeme ho popsat v rovině pouze souřadnicemi X, Y a úhel natočení φ z vnějšího popisu vypustit, protože z tohoto úhlu pohledu se tvar

ani rozměry robota při otáčení nemění. Za těchto podmínek lze nazvat takového to robota holonomním, má dva řiditelné stupně volnosti – rychlost otáčení pravého ω_r a levého ω_l kola, a dva stupně volnosti – souřadnice X, Y v Kartézské soustavě souřadnic.

Výše uvedený princip bohužel nelze aplikovat vždy. V soutěži Eurobot často nastane situace, kdy je potřeba volit jiné konstrukční uspořádání, většinou obdélníkový půdorys robota s osou otáčení blíže k přední nebo zadní hraně robota, obrázek 4.1 (c). Pokud na toto uspořádání aplikujeme princip zjednodušení na holonomního robota, nastanou problémy při plánování trajektorie, viz sekce 4.3.1.

4.3 Algoritmus plánování trajektorie pro soutěž Eurobot

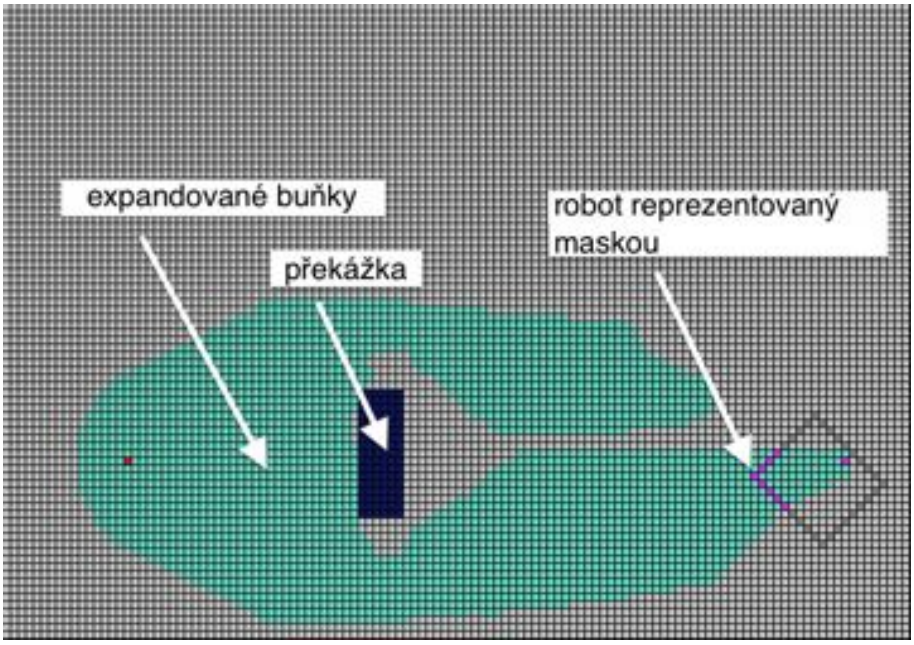
Dosud používaný A* algoritmus pracuje s mapou, která popisuje okolí robota. Robot je v ní reprezentován referenčním bodem (osa otáčení robota). Pokud má být plánovaná trajektorie bezkolizní, je třeba kolem překážek vytvořit bezpečnostní okolí a algoritmus musí zaručit, že trajektorie referenčního bodu robota nebude protínat bezpečnostní okolí žádné překážky. Tento přístup je určen pro holonomní mobilní roboty. Soutěžní robot holonomní není, bezpečností okolí kolem překážek je vzhledem k rozměrům hřiště veliké a v omezeném prostoru na hřišti je pohyb robota velmi obtížný. Je třeba změnit přístup k reprezentaci robota.

Robot tedy bude reprezentován v mapě svým reálným tvarem a rozměry (maskou) a bude se uvažovat i úhel natočení, práce s maskou a úhlem je velmi podobná s [33] a [32], kde se dále používá neuronová síť. Mapa je mřížka s buňkami čtvercového tvaru. Buňky mohou vyjadřovat volný prostor nebo překážku či nést další informace. Přístup v [1] byl testován v obdobném prostředí s obdobným robotem, okolí robota však reprezentuje jako rychlostní profily. V [6] je prostor reprezentován jako vektorové pole, algoritmus pracuje převážně se senzorickými daty ze dvou laserových 2D dálkoměrů (laser range finder) [14]. Tento přístup by byl velmi vhodný, v tento okamžik je jeho implementace velmi složitá, musela by se kompletně změnit práce s mapou, reprezentace překážek v mapě, vyřešit vhodné umístění laserového senzoru, aby naměřené hodnoty byly věrohodné apod., dále tedy bude uvažováno rozšíření základního A* algoritmu.

4.3.1 Vliv tvaru robota na algoritmus plánování trajektorie a jeho reprezentace v mapě

Jak již bylo naznačeno v předešlé části, tvar podstavy a celkové uspořádání robota má velký vliv na algoritmus plánování bezkolizní trajektorie. Problém je více patrný v okamžiku, kdy se velikost volného prostoru pro manévrování blíží rozměrům samotného robota, který se v prostředí pohybuje. Hřiště v soutěži Eurobot 2011 má rozměr $3 \times 2{,}1$ m, samotný robot pak $0{,}27 \times 0{,}3$ m. Na hřišti jsou rozmístěny herní prvky a pohybuje se zde i soupeřův robot, viz kapitola 2. Volný prostor pro pohyb robota je tedy velmi omezený.

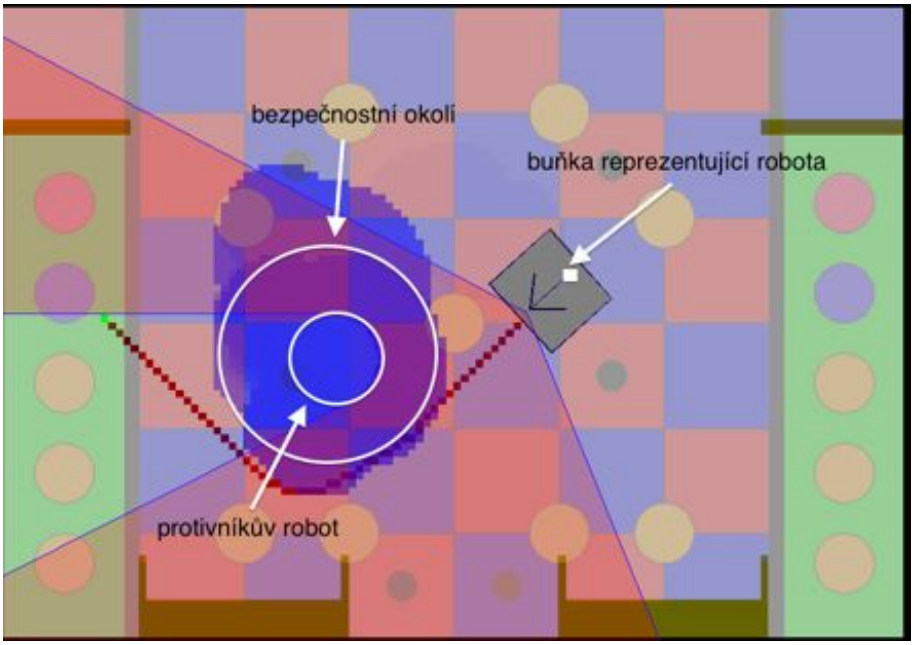
Prostor, ve kterém se robot pohybuje je znám, je vhodné jej tedy popsat mapou. Náhodné prvky, jako je poloha protivníkova robota či poloha herních prvků [27], lze do mapy zanést na základě detekce a určení polohy během zápasu. V našem případě je mapa reprezentována rastrem (mřížkou), obrázek 4.2, o rozměrech 120×84 buněk (rozměry buňky 25×25 mm). Rozměry prvků, které se překreslují do mapy jsou vždy zaokrouhleny na nejbližší vyšší počet buněk.



Obrázek 4.2: Reprezentace robota v mapě pomocí masky

Pokud uvažujeme původně užitý A* algoritmus pracující nad touto mapou, je robot reprezentován jako jedna buňka (bod), obrázek 4.3. Rozměr robota je do mapy zanesen tak, že se kolem každé buňky reprezentující překážku vykreslí bezpečnostní okolí (kruh) o

poloměru větším než poloměr největší kružnice opsané tvaru robota se středem v bodě otáčení robota , obrázek 4.1 (c). Na tomto principu je založen zjednodušený algoritmus plánování bezkolizní trajektorie popsaný v sekci 4.3.2. Čím více je osa otáčení dále od středu robota, tím větší bezpečnostní okolí je. Kruhy v mapě, vyjadřující bezpečnostní okolí, mohou snadno narůst do rozměrů, kdy zásadně nebo zcela omezí pohyb robota po hřišti. Tento problém se



Obrázek 4.3: Reprezentace robota v mapě jedním bodem

snaží řešit přístup zvolený v této práci. Robot je reprezentován v mapě svými skutečnými rozměry pro každý úhel natočení (uvažuje se osmi okolí v rastrové mapě, tedy osm hodnot úhlu natočení) a nevykresluje se bezpečnostní okolí kolem překážek, je tedy uvažován i reálný rozměr překážek. Tvar robota v mapě je reprezentován maskami pro pohyb vpřed pro úhel natočení 0, $\pi/4$, obrázek 4.4 v sekci 4.3.3.1, a pro otočení na místě o úhel $\pi/4$, obrázek 4.5 v sekci 4.3.3.1. Otočení o větší úhel je řešeno postupným otáčením základní masky. Tento přístup je využit v rozšířeném algoritmu pro plánování trajektorie, který je popsán dále v sekci 4.3.3.

4.3.2 Zjednodušený algoritmus plánování bezkolizní trajektorie

Zjednodušený algoritmus plánování bezkolizní trajektorie je v podstatě základní A* algoritmus popsaný výše pracující nad mapou, který považuje robota za buňku v mapě a kolem

překážek vykresluje bezpečnostní okolí, obrázek 4.3. Algoritmus v této formě byl dosud využíván naším týmem v soutěži Eurobot.

Vstupem algoritmu je mapa zakreslená v mřížce, startovní buňka q_{start} a cílová buňka q_{goal}. Pokud se díváme na mřížku jako na graf, jednotlivé buňky mřížky představují uzly grafu a délka hran je rovna Euklidovské vzdálenosti mezi danými buňkami (funkce $c(c_1, c_2)$, vztah 4.1). V mapě se uvažuje osmi okolí každé buňky, množina uzlů $N(n)$ sousedících s n tedy obsahuje 8 uzlů. Funkce $c(c_1, c_2)$ pak vrací hodnotu 1 nebo $\sqrt{2}$ za předpokladu, že jsou buňky volné (představují volný prostor pro pohyb robota), jinak vrací cenu určenou příznakem v dané buňce, např. 1000 pro příznak „překážka“. Uzly c_1 a c_2 jsou vždy uzly sousední. Hodnota funkce $g(n)$, vztah 4.2, je pak součet délek všech hran ze startovního uzlu q_{start} do aktuálního uzlu n, tj. součet všech hodnot funkce $c(c_1, c_2)$ pro všechny buňky cesty.

$$c(c_1, c_2) = \sqrt{(c_{2x} - c_{1x})^2 + (c_{2y} - c_{1y})^2} \tag{4.1}$$

$$g(n) = \sum_{q=q_{start}}^{n} c(q_{i+1}, q_i) \tag{4.2}$$

$$h(n) = \sqrt{(q_{goalx} - n_x)^2 + (q_{goaly} - n_y)^2} \tag{4.3}$$

Heuristická funkce $h(n)$, vztah 4.3, je Eukleidovská vzdálenost mezi aktuální buňkou n a cílovou buňkou q_{goal}. Vztahy 4.1 a 4.3 se rovnají, pokud se uvažují dvě sousedící buňky. Pro ostatní případy není nikdy $g(n) > h(n)$, heuristika je tedy přípustná.

Výstupem algoritmu je nalezená cesta nebo ukazatel na prázdnou cestu, pokud není nalezena. Při rekonstrukci nalezené cesty se postupuje od cíle ke startu. U cílové buňky q_{goal} se určí zpětný ukazatel (*backpointer*) na buňku, z které byla q_{goal} expandována a označí se příznakem „cesta“. Takto se postupuje iterativně až do q_{start}. Buňky q_{start} a q_{goal} se nakonec označí příznakem „start“ a „cíl“.

4.3.3 Rozšířený algoritmus plánování trajektorie

Jak již bylo naznačeno, rozšíření zjednodušeného algoritmu plánování trajektorie přináší především jiný pohled na reprezentaci robota a překážek v mapě. Objekty jsou v mapě reprezentovány svými reálnými fyzickými rozměry. Překážky vycházející z pravidel, známé před startem zápasu, např. součásti hřiště nesoucí herní prvky, jsou v mapě zakresleny hned po spuštění programu. Objekty s neznámými tvary a pozicemi, např. robot soupeře, se do mapy zakreslují v průběhu zápasu na základě informací z laserového 2D dálkoměru (laser range finder), případně jiného senzoru. Kolem překážek se dále nevykresluje bezpečnostní okolí. Algoritmus pracuje s maskou reprezentující reálné rozměry půdorysu robota. V každém

kroku algoritmu, kdy se expanduje další buňka mapy, je maska umístěna do mapy na danou expandovanou buňku a testuje se kolize s překážkou.

Rozšířený algoritmus plánování trajektorie vychází z algoritmu popsaného v sekci 4.3.2.

Definice pojmů:

- $M(x, \varphi)$ – množina buněk masky vztažených k x pro daný úhel natočení φ
- P – množina buněk reprezentující všechny překážky v mapě

Algoritmus:

1. Vytvoř masky a ulož je do paměti
2. ...
3. repeat
4. Vyber n_{best} z O takové, že $f(n_{best}) \leq f(n), \forall n \in O$
5. Vyjmi n_{best} z O a vlož ho do C
6. Pokud $n_{best} = q_{goal}$, ukonči algoritmus
7. Urči úhel natočení φ robota v n_{best}
8. Expanduj n_{best}: všechna $x \in N(n_{best})$, která nejsou v C
9. if $x \notin O$ and $M(x, \varphi) \cap P = \emptyset$ then
10. přidej x do O
11. else if $g(n_{best}) + c(n_{best}, x) < g(x)$
12. aktualizuj zpětný ukazatel uzlu x na n_{best}
13. end if
14. until O není prázdná

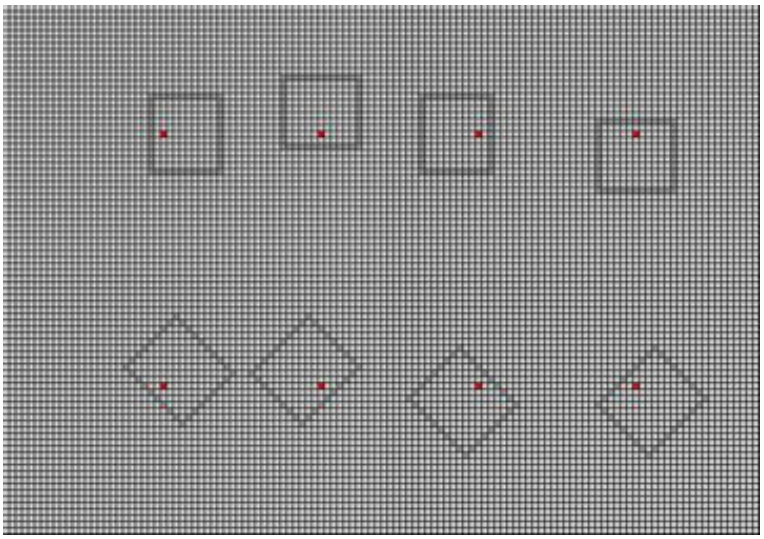
Dále bude v sekci 4.3.3.1 detailně popsán bod 1. a v sekci 4.3.3.2 bod 9. uvedeného algoritmu.

4.3.3.1 Inicializace masek

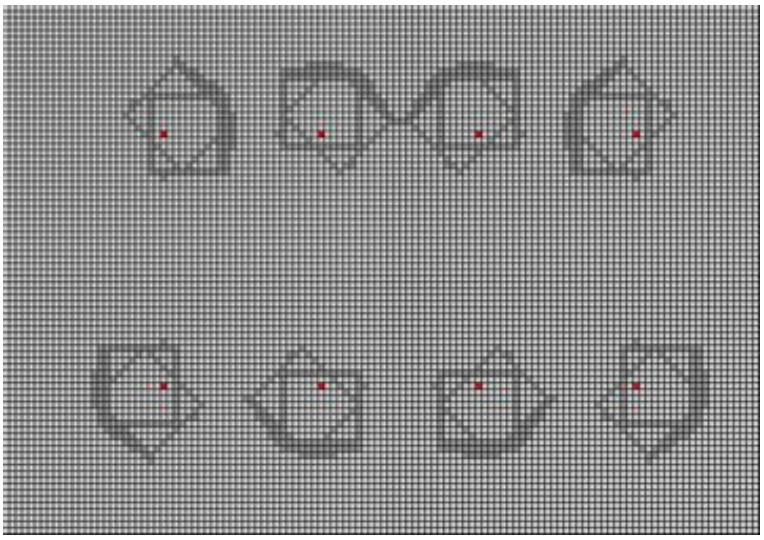
Inicializace masek zahrnuje alokaci paměti a vytvoření tří masek, dvě reprezentují pohyb robota vpřed pod úhlem natočení 0 a $\pi/4$, obrázek 4.4 a jedna reprezentuje rotaci robota o úhel $\pi/4$ a pohyb vpřed zároveň, obrázek 4.5. Maska pro úhel $\pi/4$ se používá ze dvou důvodů – rotace o $\pi/2$ je výpočetně jednodušší (viz prvky rotační matice, rovnice 4.4) a jednak je možné ovlivnit tvar masky v případě, že robot má složitý tvar, který použitou transformací (bod 2. algoritmu popsaného v sekci 4.3.3) není správně interpretován.

Inicializace masek probíhá v následujících krocích:

1. Vytvoř vektorovou předlohu masky v_m_0PI pro úhel 0π na základě reálných rozměrů robota
2. Vytvoř vektorovou předlohu masky v_m_PI4 rotací v_m_0PI o úhel $\pi/4$
3. Alokuj pamět pro všechny tři masky
4. Převeď oblouky mezi v_m_0PI a v_m_PI4 do buněk masky rotace m_rot
5. Převeď v_m_0PI a v_m_PI4 do buněk masky příměho pohybu m_0PI m_0PI
6. m_rot = m_rot $\cup$ m_0PI $\cup$ m_PI4



Obrázek 4.4: Masky reprezentující pohyb vpřed



Obrázek 4.5: Masky reprezentující otočení a pohyb vpřed

Vektorová předloha masky jsou body v Kartézské soustavě souřadnic definující vrcholy obrysu robota, obrázek 4.6. Osa otáčení robota je umístěna do počátku soustavy souřadnic (0, 0). Maska tedy obsahuje relativní souřadnice vůči referenčnímu bodu robota, absolutní souřadnice na hřišti jsou určeny až během testu kolize robota s překážkou, viz sekce 4.3.3.2.

Oblouky mezi znázorňují dráhu vrcholů obrysu robota při otáčení. Jsou to vždy kratší oblouky kružnice, na které leží korespondující body vektorové předlohy masky.

Převod oblouku nebo úsečky do buněk masky je problém rasterizace vektorů, který se často vyskytuje v např. v počítačové grafice. Převod oblouku do rastru se realizuje v cyklu (krok i) rotací počátečního bodu oblouku A, obrázek 4.6, o úhel $i\alpha$, rovnice 4.4, a následným zaokrouhlením na souřadnice nejbližší buňky.

$$\begin{bmatrix} B_x \\ B_y \end{bmatrix} = \begin{bmatrix} cos(i\alpha) & -sin(i\alpha) \\ sin(i\alpha) & cos(i\alpha) \end{bmatrix} \begin{bmatrix} A_x \\ A_y \end{bmatrix} \tag{4.4}$$

Převod vektorové předlohy do masky probíhá obdobně jako převod oblouku. Úsečka je popsaná směrnicovou rovnicí přímky, rovnice 4.5 (4.6), počátečním a koncovým bodem A, B, obrázek 4.6. V cyklu (krok i, velikost kroku s, s = velikost buňky/2) je pak vzorkována a výsledný bod je zaokrouhlen do nejbližší buňky.

$$y = k(x + is) + q \tag{4.5}$$

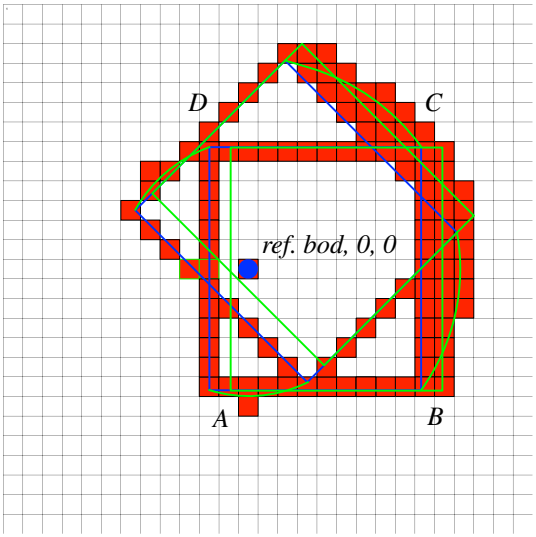
$$y = q + is, x = const. \tag{4.6}$$

4.3.3.2 Expanze buněk mapy a test kolize robota s překážkou

Když je robot reprezentován v mapě jednou buňkou, expanze buněk probíhá shodně pro všechny buňky z množiny N. Testuje se, zda expandovaná buňka neobsahuje příznak „překážka“. Pokud se v algoritmu použije reprezentace robota pomocí masky, test kolize již není shodný pro všechny expandované buňky. Je třeba určit, jaká maska a jak transformovaná bude v testu použita.

Algoritmus expanze buněk a testu kolize, definice pojmů v sekci 4.3.3, příklad postupu expanze ukazuje obrázek 4.7:

1. Urči úhel φ_0, pod kterým je robot natočen v n_{best}
2. Na základě φ_0 vyber masku pro přímý pohyb vpřed m_0PI nebo m_4PI
3. for $i = 0$ to $i \leq 9$ do
4. vyjmi x_i z $N(n_{best})$, pokud již není v C

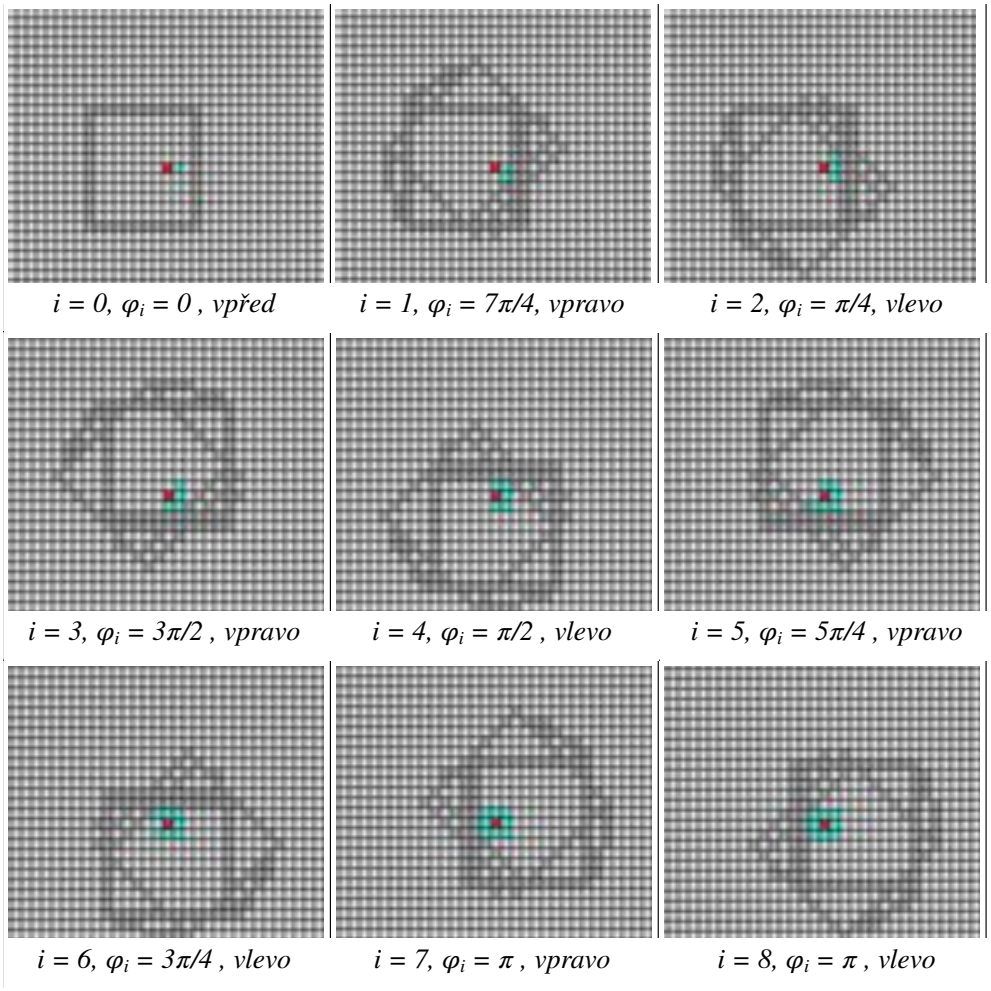


Obrázek 4.6: Převod vektorových masek do buňek, celý manévr otočení a pohyb vpřed, modrá barva – původní masky pohybu vpřed, zelená barva – otočení a pohyb vpřed

5. `if` $x_i \notin O$
6. `Transformuj vybranou masku do požadovaného` φ_i, $i = 0$ `pohyb vpřed,` $(-1)^i < 0$ `otáčení vpravo,` $(-1)^i > 0$ `otáčení vlevo`
7. `if` $M(x_i, \varphi_i) \cap P = \emptyset$ `then`
8. `přidej` x_i `do` O
9. `else`
10. `dále již netestuj otáčení vlevo nebo vpravo v závislosti na` $(-1)^i$
11. `end if`
12. `else if` $g(n_{best}) + c(n_{best}, x_i) < g(x_i)$
13. `aktualizuj zpětný ukazatel uzlu` x_i `na` n_{best}
14. `end if`
15. `Zvol masku rotace m_rot pro další krok cyklu`
16. `end for`

Před samotnou expanzí buněk se určí úhel pod kterým se robot nachází v n_{best}, následně se vybere maska pro pohyb vpřed, `mask_0pi` pro sudé násobky $\pi/4$, `mask_pi4` pro liché násobky $\pi/4$. V prvním kroku cyklu je tedy kontrolována dostupnost buňky z N ve směru pohybu robota.

Maska `mask_0pi` má základní orientaci 0 (`mask_pi4` $\pi/4$). Do požadovaného úhlu jsou tedy relativní souřadnice masky transformovány pomocí rotační matice, ta vyjadřuje rotaci v násobcích $\pi/2$. Prvky rotační matice pak nabývají hodnoty 0, ± 1. Výpočetně je tedy



Obrázek 4.7: Příklad expanze buněk pro $\varphi_0 = 0$

transformace nenáročná. Kombinací dvou masek a rotační matice lze tedy obsáhnout celé osmi okolí expandované buňky.

Maska obsahuje celočíselné souřadnice buněk vztažené k referenčnímu bodu robota (osa otáčení). Při testu jsou určeny absolutní souřadnice masky na hřišti. Referenční buňka je pak x pro `mask_0pi` a `mask_pi4`, n_{best} pro `mask_rot`. Na takto určených souřadnicích je testován příznak „překážka“, pokud je test pozitivní, je ukončen a daná pozice je prohlášená za nedostupnou.

Otáčení robota kolem své osy je reprezentováno maskou rotace `mask_rot`, ta vyjadřuje otočení o úhel $\pi/4$ (z 0 na $pi/4$) a pohyb vpřed po otočení. Symbolizuje tedy celý předpokládaný manévr. V cyklu je střídavě testováno otáčení vpravo ($\langle\varphi, \varphi - \pi\rangle$) a vlevo ($\langle\varphi, \varphi + \pi\rangle$) v závislosti na členu $(-1)^i$. V jednom kroku cyklu se robot otočí o $\pi/4$, tj. do sousední buňky

v osmi okolí. Pokud je některá z testovaných pozic nedostupná, test následujících pozic již neproběhne, tento směr otáčení je pro další hodnoty úhlů prohlášen za nedostupný. Maska `mask_rot` je do požadovaného úhlu transformována rotační maticí stejně jako ostatní masky. Pro liché násobky $\pi/4$ je navíc maska před transformací zrcadlově převrácena kolem osy x. Takto lze dosáhnout s jedinou maskou všechny varianty pro osmi okolí.

4.4 Implementace

Původní zdrojové kódy algoritmu plánování, které vytvořil Jose Maria Martin Laguna (jmmartin@etud.insa-toulouse.fr) při své stáži na FEL ČVUT, byly dále upraveny a rozšířeny. Přepsána byla hlavní funkce algoritmu `aAlgorithm`, doplněny pomocné funkce, napsána knihovna *map_2_png* a nový kód algoritmu byl začleněn do software robota.

4.4.1 Modul aalgorithm

Softwarový modul *aalgoritm* je součástí knihovny *pathplan*. Reprezentují ho soubory `aalgorithm.h` a `aalgorithm.c` v adresáři `/src/pathplan`. Modul využívá knihovnu *map*, která definuje sdílenou mapu hřiště a funkce pro práci s ní, a případně knihovnu vhodnou pro ladění a dokumentaci *map_2_png*, která vizualizuje stav sdílené mapy do souboru .PNG. Knihovna *map_2_png* dále využívá open source knihovnu *libpng* [16].

4.4.1.1 aalgorithm.h

Hlavičkový soubor `aalgorithm.h` v adresáři `/src/pathplan` deklaruje rozhraní vůči modulu *aalgoritm*. Algoritmus si pro své specifické informace alokuje vlastní mapu, která je odlišná od sdílené mapy. Jedna buňka této mapy je reprezentována strukturou `GraphMapCell`. Význam jednotlivých položek struktury je vysvětlen v komentářích kódu.

```
typedef struct _GraphMapCell {
        // hodnota heuristické funkce z aktuální do cílové buňky
        float h;
        // odhadovaná nejkratší cesta mezi startovní a cílovou buňkou
        float f;
        // reálná vzdálenost ze startovní do aktuální buňky
        float g;
        // ukazatel na buňku, z které byla aktuální buňka expandována
```

```
        struct _GraphMapCell *backpointer;
        // příznak, že byla buňka již expandována
        bool processed;
        // příznak, že je buňka v prioritní frontě
        bool in_queue;
        // ukazatel na další buňku v prioritní frontě
        struct _GraphMapCell *next;
} GraphMapCell;
```

Modul *aalgorithm* obsahuje funkci `aAlgorithm`. Deklarace funkce je popsána v hlavičkovém souboru. Definice funkce a definice jejích pomocných funkcí lze nalézt v souboru `/src/pathplan/aalgorithm.c`.

```
int aAlgorithm(
        // spojité souřadnice startovního bodu na hřišti
        double xstart_real, double ystart_real,
        // spojité souřadnice cílového bodu na hřišti
        double xgoal_real, double ygoal_real,
        // úhel natočení robota ve startovním a cílovém bodě
        double start_angle, double goal_angle,
        // výstupní parametr, ukazatel na ukazatel na první buňku nalezené cesty
        GraphMapCell **original_path);
```

Spojité souřadnice startovního a cílového bodu plánování trajektorie (`x*_real`, `y*_real`) vyjadřují v metrech pozici referenčního bodu robota (osa otáčení) na hřišti ve spojitém uzavřeném intervalu $\langle$ `0, MAP_PLAYGROUND_WIDTH_M` $\rangle$ pro souřadnici x a $\langle$ `0, MAP_PLAYGROUND_HEIGHT_M` $\rangle$ pro souřadnici y. Rozměry a popis hřiště včetně orientace os X, Y pro potřeby A* algoritmu jsou uvedeny v hlavičkovém souboru knihovny *map* `/src/pathplan/map.h`. Rozměry robota a hřiště pro obecné užití jsou popsány v souboru `/src/robodim/robodim.h`.

Funkce `aAlgorithm` využívá masek, které jsou v modulu *aAlgorithm* globálními proměnnými. Před prvním voláním funkce `aAlgorithm` je potřeba masky inicializovat, funkce `init_aalgorithm`, před ukončením programu pak uvolnit alokovanou paměť, funkce `freemem_aalgorithm`.

```
int init_aalgorithm(void)
int freemem_aalgorithm(void)
```

4.4.1.2 aalgorithm.c

Soubor `/src/pathplan/algorithm.c` obsahuje definice funkcí `aAlgorithm`, `init_aalgoritm` a `freemem_aalgorithm`, dále pak pomocné funkce, datové typy a globální proměnné nutné pro realizaci algoritmu. Zdrojové kódy jsou komentované, dále bude popsána jen implementace datové struktury reprezentující masky.

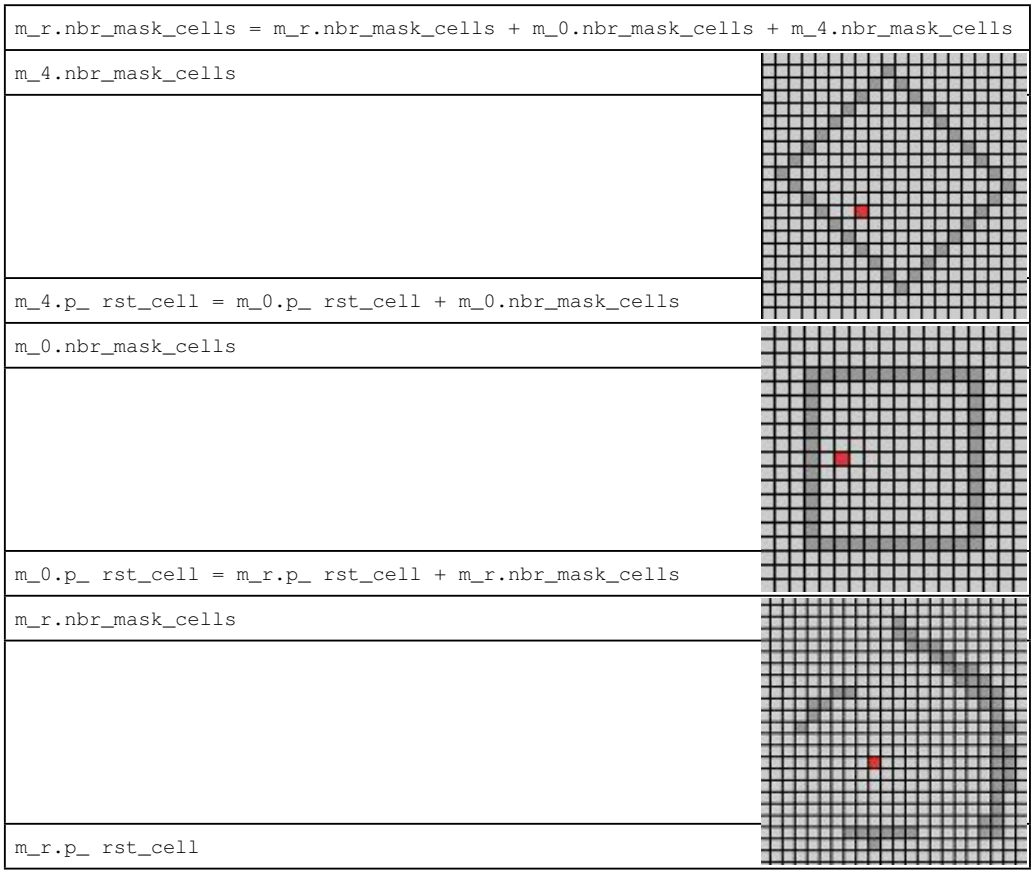
Globální proměnné `mask_0pi`, `mask_pi4`, `mask_rot` jsou struktury typu `mask_t` a reprezentují masky využívané v algoritmu. Samotné souřadnice buněk (typ `cell_coordinates_t`) jsou uloženy v paměti na haldě, dynamicky alokované funkcí `calloc`. Ta má dva argumenty, počet prvků `init_nbr_of_cells` a velikost jednoho prvku. Před voláním funkce `calloc` není zcela znám počet prvků `init_nbr_of_cells`, je tedy nadsazen a po naplnění paměti je paměť zmenšena na skutečnou velikost (funkce `realloc`).

```
typedef struct {
        int x;
        int y;
} cell_coordinates_t;

(cell_coordinates_t *) calloc(
        (size_t) init_nbr_of_cells,
        sizeof(cell_coordinates_t))

typedef struct {
        // ukazatel na souřadnice první buňky
        cell_coordinates_t *p_first_cell;
        // počet inicializovaných buněk
        int nbr_mask_cells;
        // typ masky, DIRECT_MASK 0, ROTATE_MASK 1
        int type_of_mask;
} mask_t
```

Buňky všech tří masek jsou uloženy v jednom bloku paměti, obrázek 4.8. Maska rotace `mask_rot` používá buňky ostatních masek. K jednotlivým buňkám se přistupuje pomocí pointrové aritmetiky. Typ `mask_t` je tedy jakási hlavička, která udržuje informace o inicializované paměti (ukazatel a velikost). Paměť je alokována ve funkci `init_aalgorithm`, `aAlgorithm` z paměti pouze čte a `freemem_aalgorithm` ji uvolňuje.



Obrázek 4.8: Uložení masek v paměti (`mask_rot` – `m_r`, `mask_0pi` – `m_0`, `mask_pi4` – `m_4`)

4.4.2 Knihovna map_2_png

Knihovna *map_2_png* slouží k vizualizaci sdílené mapy. Vizualizace je vhodná při tvorbě dokumentace nebo při ladění algoritmů. Knihovna obsahuje funkci `map_2_png`, která má dva parametry – ukazatel na sdílenou mapu `map` a název výsledného souboru `file_name`. Výstupem je obrázek typu .PNG, ve kterém barva jednotlivých políček symbolizuje příznaky uložené v mapě a koresponduje s barvami použitými v simulačním programu Robomon. Souřadnicový systém a orientace výsledného obrázku odpovídá mapě.

```
int map_2_png(struct map *map, char* file_name)
```

Barvy použité v knihovně *map_2_png*:

- **černá** – oddělující čára mezi políčky v obrázku

- **světle šedá** – volný prostor bez překážek
- **tmavě šedá** – maska použitá v algoritmu (`MAP_FLAG_PLAN_MASK`)
- **zelená** – cílová buňka (`MAP_FLAG_GOAL`)
- **červená** – startovní buňka (`MAP_FLAG_START`)
- **tmavě červená** – cesta nalezená algoritmem (`MAP_FLAG_PATH`)
- **modrá** – překážka (`MAP_FLAG_WALL`, `MAP_FLAG_DET_OBST`)
- **tyrkysová** – buňky expandované algoritmem (tmavě tyrkysová v Robomon, `MAP_FLAG_PLAN_EXPANDED_CELLS`)
- **oranžová** – bezpečnostní okolí překážky (`MAP_FLAG_PLAN_MARGIN`)
- **purpurová** (magenta) – více jak jeden výše uvedený příznak

Soubor `/src/pathplan/map_2_png.h` popisuje rozhraní knihovny. Obsahuje deklaraci funkce `map_2_png` a konstant `CELL_DIMENSION` – velikost jedné buňky v obrázku v pixelech, `LINE_THICK` – síla dělící čáry v pixelech. Těmi je možné nastavit vzhled a rozlišení výsledného souboru.

Knihovna *map_2_png* využívá open source knihovnu *libpng* [16], která implementuje samotný zápis dat do souboru ve správném formátu. Aktuálně je použita a otestována verze 1.2.44 knihovny *libpng dev* (*libpng12-dev*) z repozitáře linuxové distribuce Ubuntu [22]. Knihovna se používá pouze při běhu algoritmů na osobním počítači (*host*), nikoliv na řídícím počítači robota (*PPC*).

4.5 Testování, ověření funkce a výsledky

Navržený algoritmus popsaný v sekci 4.3.3 byl implementován, jak je popsáno v sekci 4.4, laděn a testován. Pro základní ladění a testování implementace byly sestaveny testovací programy popsané v sekci 4.5.2. Po otestování samotného algoritmu, popsáno v sekci 4.5.3, byl algoritmus začleněn do softwarové výbavy robota a dále testován s robotem jako celek, popsáno v sekci 4.5.4.

4.5.1 Procesorové platformy

Ve spojitosti se software robota se používají dvě procesorové platformy. První je počítač vývojáře software kategorie běžného PC, na kterém se software robota vyvíjí, kompiluje (pro všechny platformy, crosscompiler [25]), simuluje a ladí základní funkčnost algoritmů. Dále slouží k práci s robotem a pro vizualizaci stavů robota. Označuje se jako *host*. Popis platformy *host* použité v této práci:

- Virtualizovaný operační systém Lubuntu 11.04 32bit
  (verze jádra Linux 2.6.38-13-generic (i686))
- Virtualizační nástroj VirtualBox 4.1.6
- Virtuální stroj – jedno jádro procesoru, 2048 MB operační paměti
- Fyzický stroj – notebook Apple Macbook 6,1
- Dvoujádrový procesor Intel Core 2 Duo 2,26 GHz
- Operační pamět 4 GB, 1067 MHz, DDR3
- Operační systém Mac OS X 10.7.2 Lion 64bit

Druhá platforma je samotný řídící počítač robota kategorie embedded PC (počítač pro nasazení v průmyslu), který při soutěži zajišťuje veškerý výpočetní výkon pro řídící aplikace a algoritmy. Označuje se jako *PPC*. Popis platformy:

- Modul průmyslového počítače MIDAM Shark [18]
- Základní deska RYU_edu, autor Ing. Jiří Kubias [7]
- Procesor Motorola PowerPC MPC5200, jádro MPC603e 400 MHz
- Paměť RAM 128 MB, 133MHz, SDRAM, FLASH 64 MB
- Síťové rozhraní Ethernet 10/100 Mbit
- 2 × rozhraní CAN 2.0 A/B kompatibilní
- USB 1.1 host kontrolér
- Operační systém GNU/Linux, verze jádra 2.6 (ppc)

Software robota se tedy kompiluje pro obě platformy, v repozitáři odpovídají platformám adresáře pro kompilaci `/build/host` a `/build/ppc`, výsledné spustitelné soubory se nalézají v adresářích `/build/host/_compiled` a `/build/ppc/_compiled`. Pro *host* se navíc kompilují další programy a utility, např. `Robomon` 4.5.2.4.

V současné době se pracuje na nasazení nové platformy jako řídícího počítače. Jedná se o procesorový modul Gumstix [15] kategorie embedded PC. V soutěžním robotu je nyní

použit s modulem displeje pro zobrazení stavů robota. V nasazení jako řídícího počítače brání především absence vhodné periférie sběrnice CAN. Dále je třeba otestovat vývojové nástroje a software robota konkrétně na této platformě. Popis platformy:

- Modul průmyslového počítače Gumstix Fire COM
- Základní deska Chestnut43 s dotykovým displejem 4,3"
- Procesor TI OMAP 35300, jádro ARM Cortex-A8 720 MHz
- Paměť RAM 512 MB, FLASH 512 MB
- Síťové rozhraní Ethernet 10/100 Mbit
- Bezdrátové síťové rozhraní WiFi 802.11b/g
- USB 2.0 host a USB OTG kontrolér
- Operační systém GNU/Linux

4.5.2 Testovací programy

Testovací programy pro ověření funkce a testování knihovny *pathplan* jsou umístěny v adresáři `/src/pathplan/test`.

4.5.2.1 testmask

Program *testmasks* slouží k ověření funkce `init_aalgorithm`, zda se správně inicializovaly a vytvořily masky `mask_0pi`, `mask_pi4`, `mask_rot`. Výstupem jsou soubory `masks_0pi_pi4.png` a `masks_rot.png`, které obsahují masky ve všech požadovaných orientacích. Výstupní obrázky 4.4, 4.5 jsou použity v sekci 4.3.3.1.

4.5.2.2 testastar

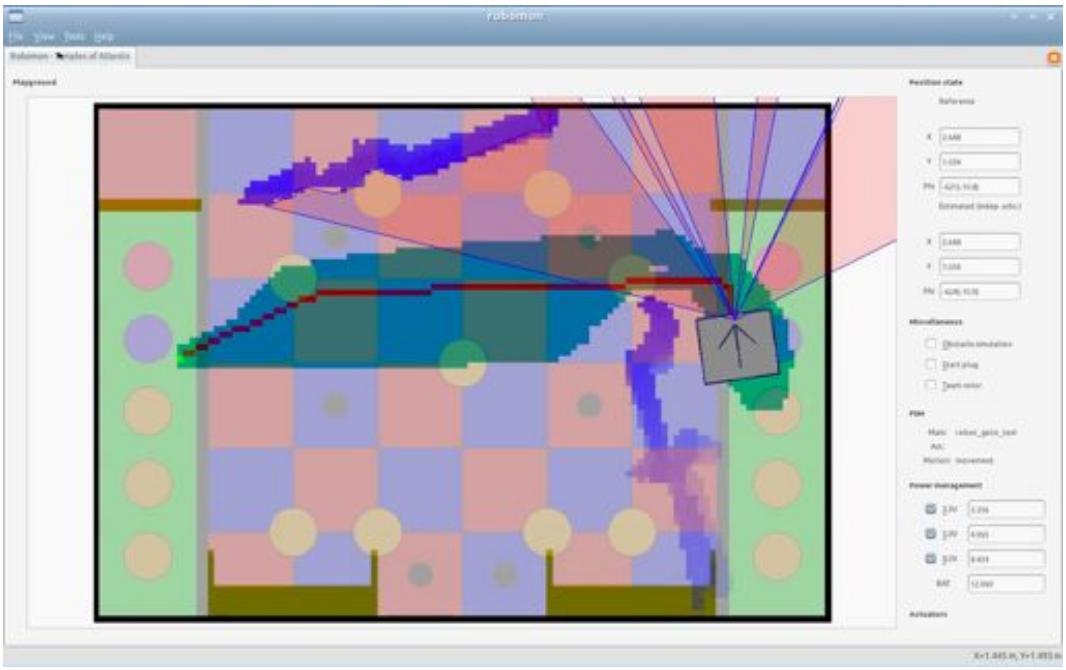
Program `testastar` testuje samotnou funkci `aAlgorithm`. Do mapy se zakreslí pevné nebo náhodně vygenerované překážky, počet překážek určuje konstanta `OBST_COUNT`. Algoritmus pak plánuje trasu mezi náhodně vygenerovaným startovním a cílovým bodem v mapě. Výstupem programu je soubor `testastar___TIME__.png` (makro `__TIME__` vloží čas testu), který zobrazuje stav mapy po nalezení trajektorie. Mapu lze též zobrazit ve formě výpisu terminálu. Počet opakování testu se vždy náhodně vygenerovanými souřadnicemi startovního a cílového bodu lze nastavit konstantou `NBR_OF_TESTS`. Výstup programu, obrázky 4.10 a 4.11, jsou použity v sekci 4.5.3.

4.5.2.3 testpathplan

Časovou náročnost algoritmu lze zjistit programem `testpathplan`. Překážky, startovní a cílový bod v mapě se určují stejně jako v `testastar`. Program pak opakovaně volá funkci `path_planner` a počítá čas, jak dlouho algoritmus plánoval trajektorii. Výstupem programu je histogram, který zobrazuje vertikálně časové intervaly a horizontálně počet průchodu algoritmem, které svou časovou náročností spadají do daného časového intervalu. Výstupy tohoto programu, histogramy, jsou použity v sekci 4.5.5.

4.5.2.4 Robomon

`Robomon` je velmi užitečný nástroj, který zobrazuje hlavní stavy robota v uživatelsky přívětivé formě. V reálném čase zobrazuje sdílenou mapu (mapa je uložená ve sdílené paměti, její obsah může v reálném čase monitorovat jiný proces na stejném počítači), stavy stavových automatů pohybu *fsm_move*, aktuátorů *fsm_act* a hlavního automatu *fsm_main*, který reprezentuje celkové chování robota (testovaní, zápas, demo ...). Dále pak pozici robota na hřišti – referenční (kde by robot měl být dle *fsm_move*) a odhadovou z nezávislé odometrie. Položka menu programu *View* umožňuje zobrazení dalších dodatečných informací. Okno programu `Robomon` ukazuje obrázek 4.9.



Obrázek 4.9: Obrazovka programu `Robomon`

`Robomon` získává informace o mapě ze sdílené paměti, pro zobrazení mapy je tedy nutné, aby veškeré programové vybavení pracující se sdílenou mapou (`Robomon`, *fsm_move*, *map_handling*) běželo současně na jednom fyzickém stroji. Ostatní informace `Robomon` získává pomocí middleware *ORTE* („komunikační protokol“) [19]. *ORTE* komunikuje síťovým protokolem *UDP*, lze tedy informace šířit lokálně nebo vzdáleně přes síť. Toto se využívá při práci s robotem, kdy je robot spojen s počítačem přes bezdrátovou síť Wi-Fi a na obou strojích běží program `ortemanager`, který se stará o navázání komunikace mezi aplikacemi.

Barvy buněk v mapě programu `Robomon` korespondují s knihovnou *map_2_png* a jsou popsány v 4.4.2.

Simulace chování robota Když se vyvíjí algoritmus plánování nebo třeba herní strategie, je praktické, pokud lze ověřit chování robota na hřišti i bez běhu programu na reálném robotu. Tuto simulaci lze provést a její průběh sledovat v okně programu `Robomon`. K simulaci na osobním počítači (označujeme *host*) je potřeba zkompilovat programy `Robomon`, `ortemanager` a program realizující chování robota, např. testovací program `rectangle`, kdy robot plánuje a projíždí trajektorii tvaru čtverce o délce strany 1 m. Postup spuštění simulace je následující:

1. Spustit `ortemanager` `build/host/_compiled/bin/ortemanager -e`
2. Spustit `Robomon` v dalším terminálovém okně,
   `build/host/_compiled/bin/robomon`
3. Spustit `rectangle` v dalším terminálovém okně
   `build/host/_compiled/bin-tests/rectangle`

Před opakovaným spuštěním aplikace (`rectangle`) je třeba znovu spustit i `Robomon`, ten neumí detekovat ukončení aplikace a nezobrazuje pak aktuální stav sdílené mapy. Všechny cesty k programům jsou relativní vůči umístění kořenového adresáře repozitáře `eurobot`. Stav mapy v programu `Robomon` při simulaci ukazuje obrázek 4.12 použitý v sekci 4.5.4.

Spolupráce robota a Robomon Při testování na reálném robotu lze opět využít `Robomon`. Postup je shodný jako při simulaci, navíc potřebujeme připraveného robota na hřišti a bezdrátové spojení Wi-Fi s robotem. Spuštění `ortemanager` provedeme příkazem:

```
build/host/_compiled/bin/ortemanager -p 10.1.1.1 -e
```

Parametr `-p` vyjadřuje IP adresu řídícího počítače robota. Tímto způsobem běží algoritmy na počítači vývojáře a robot pouze vykonává nízkoúrovňové povely zasílané přes ORTE a odesílá naměřená data zpět do počítače vývojáře, kde jsou vizualizovány. `Robomon` zobrazuje data z *ORTE* i ze sdílené mapy. Tato možnost je využita v sekci 4.5.4, kde jsou prezentovány obrázky mapy z programu `Robomon`.

Detailní testovaní a samotná soutěž vyžaduje běh veškerých algoritmů na samotném robotovi. V tomto případě je třeba zkompilovat všechny potřebné aplikace pro platformu řídícího počítače robota *PPC* a výsledné binární soubory robotu předat. To lze buď připojením souborového systému *host* do *PPC* přes NFS (viz skript `devel-utils/mount-robot`) nebo přímým nakopírováním např. pomocí příkazu scp. Druhý způsob se doporučuje pouze pro finální verze programů a pro samotnou soutěž, souborový systém *PPC* je uložen v paměti typu FLASH s omezeným počtem zápisů. Spouštění aplikací na robotu pro účel testování se pak provádí přes ssh připojení k robotu. Když se v této situaci spustí `ortemnanager` a `Robomon`, viz výše, jsou zobrazeny již pouze informace šířené přes *ORTE*. Sdílenou mapu nelze zobrazit, neboť je alokována na jiném stroji než běží `Robomon`.

4.5.3 Testování algoritmu

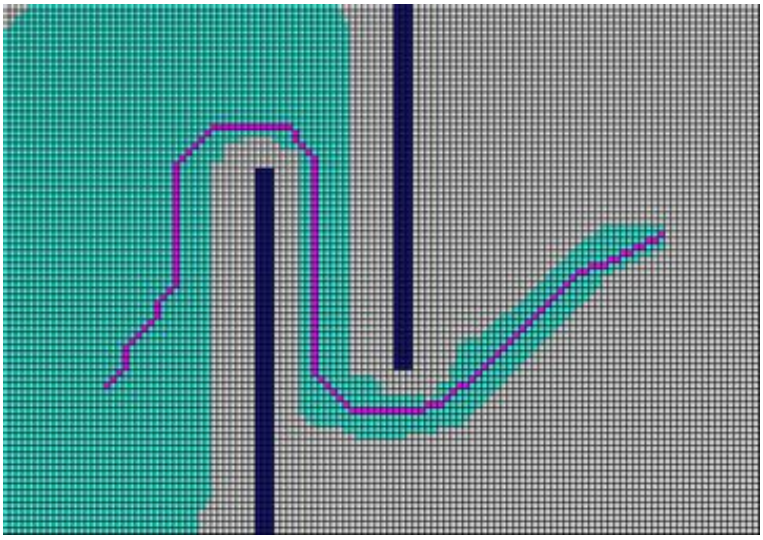
Samotný algoritmus byl laděn testovacími programy popsanými v sekci 4.5.2. Správnou funkci `init_algorithm` ověřil program `testmasks`. Masky se generují v požadované velikosti, orientaci a tvaru. Následovalo ověření, zda se při expanzi buněk a ověření dostupnosti pozice masky přiřazují správně vzhledem k aktuálnímu úhlu natočení, obrázek 4.7 v sekci 4.3.3.2. Posledním krokem bylo nechat algoritmus naleznout trasu ze startovního do cílového bodu, obrázek 4.10. Obrázek 4.11 zobrazuje stejnou situaci řešenou původním zjednodušeným algoritmem. Cesta nebyla nalezena, bezpečnostní okolí nedovolí algoritmu dosáhnout cílového bodu, i když fyzický robot je schopen trasu projet.

Během testování byla využívána knihovna *map_2_png*. Výstupem testovacích programů tak byly soubory vizualizující stav sdílené mapy. Program `testastar` generoval soubor v každém kroku (expanze) algoritmu. Animace z takto získaných obrázků jsou na přiloženém CD, příloha B.

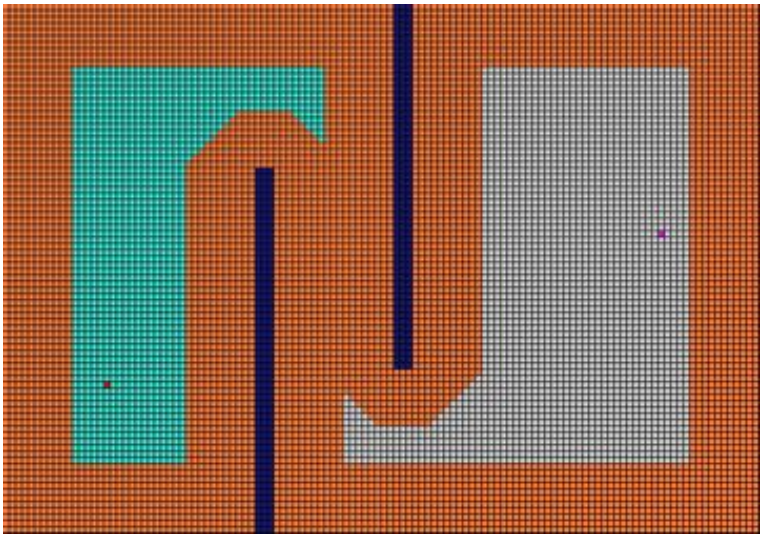
Podmínky testu:

- Rozměr prostoru 3 × 2,1 m, rozměr soutěžního hřiště
- Rozměr mapy 120 × 84 buněk, rozměry buňky 25 × 25 mm
- Rozměr masky robota 275 × 300 mm

- Testovací platforma *host*
- Počet cyklů algoritmu 30867



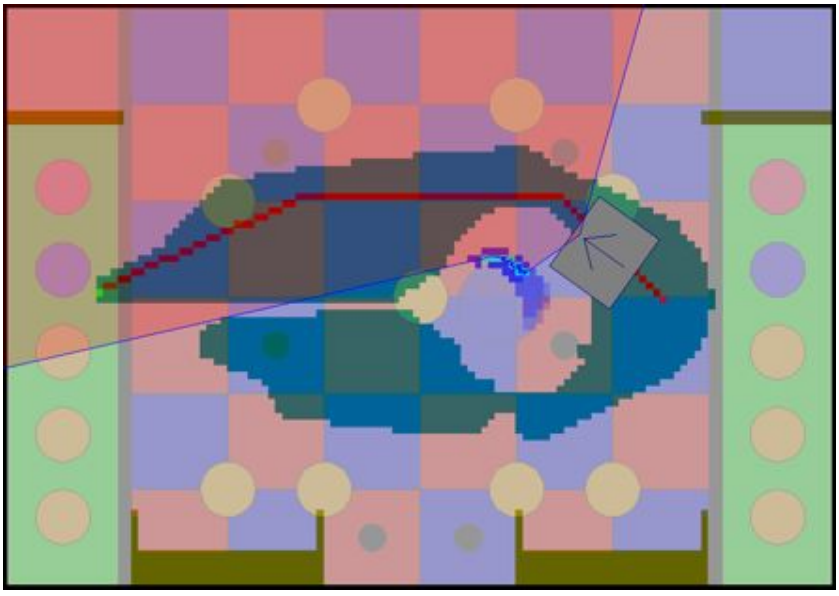
Obrázek 4.10: Cesta nalezená rozšířeným algoritmem, výstup programu `testatar`, obrázek vygenerován knihovnou *map_2_png*



Obrázek 4.11: Nenalezená cesta zjednodušeným algoritmem (oranžová barva – bezpečnostní okolí), výstup programu `testatar`, obrázek vygenerován knihovnou *map_2_png*

4.5.4 Testování robota

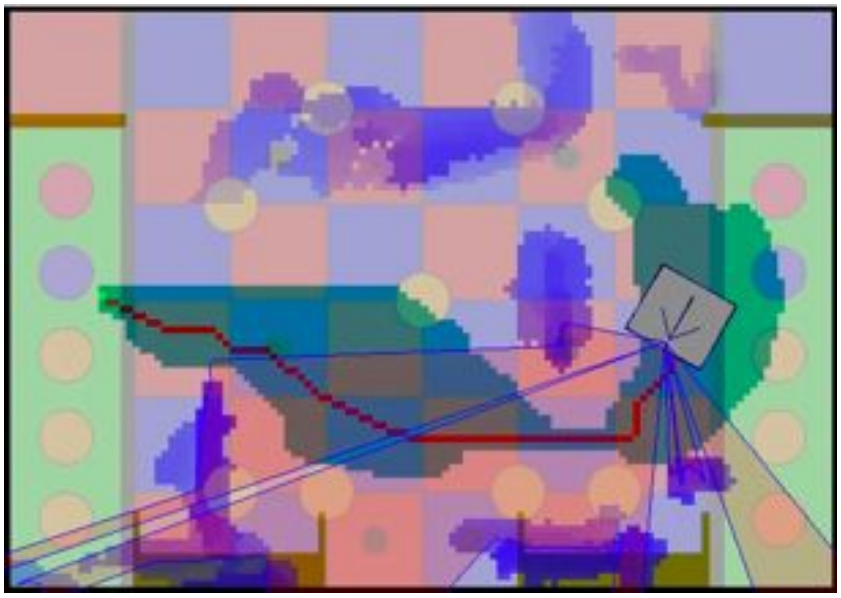
Po samostatném testování byl algoritmus zahrnut do software robota a dále testován jako celek. Práce algoritmu byla ověřena simulací na *host*, obrázek 4.12, a na reálném robotu, kdy algoritmus běžel na *host* (zobrazení mapy, popsáno v 4.5.2.4). Robot se pohyboval v prostředí kanceláře, reagoval na předložené překážky a trajektorii pohybu vždy přeplánoval při detekci překážky. Reakce robota a algoritmu ukazují obrázky 4.13, 4.14 a 4.15. Algoritmus naplánoval trajektorii na základě aktuálních informací v mapě, obrázek 4.13. Při projíždění trajektorie byla detekována překážka v cestě, obrázek 4.14, a původní trajektorie přeplánována, obrázek 4.15. Modré jsou buňky, kde byly překážky detekovány v minulosti (stáří překážky je znázorněné různou sytostí). Tyrkysově jsou značené aktuálně detekované překážky. Tmavě červené buňky značí nalezenou cestu, zelené políčko je cílový bod s červené políčko bod startovní. Videozáznamy z průběhů testu jsou na přiloženém CD, příloha B.



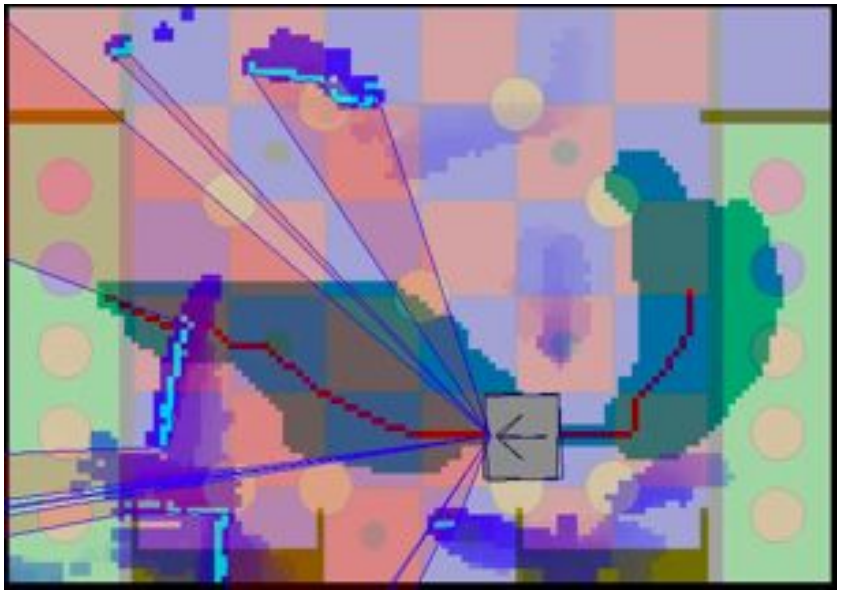
Obrázek 4.12: Simulace pohybu robota na *host*, program `Robomon`

Podmínky testu:

- Rozměr prostoru cca 5 × 2 m, využit jen rozměr soutěžního hřiště
- Rozměr mapy 120 × 84 buněk, rozměr buňky 25 × 25 mm
- Rozměr masky robota 325 × 350 mm
- Testovací platforma *host*
- Překážky definují tyrkysové a modré buňky v mapě, rozměr překážek je zakreslen na základě informací z 2D laserového dálkoměru



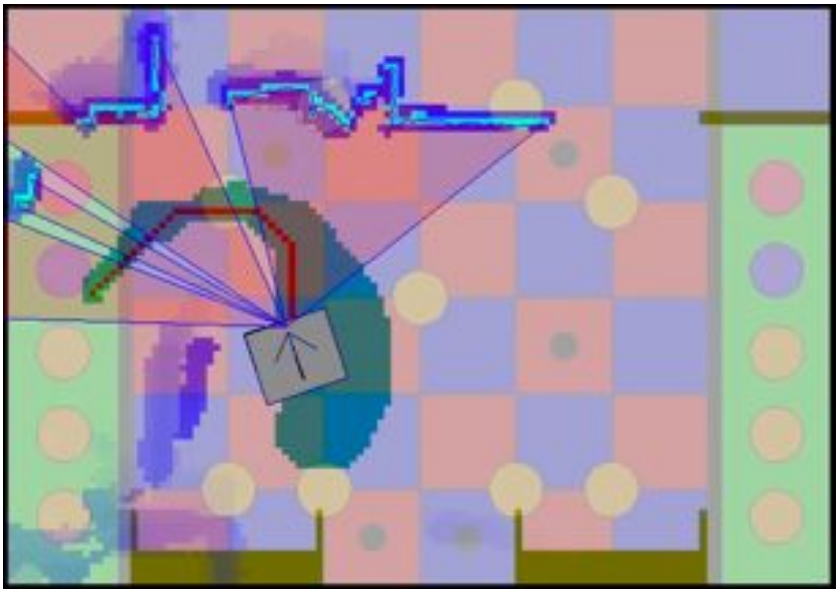
Obrázek 4.13: Naplánovaná trase ze startu do cíle (program `Robomon`)



Obrázek 4.14: Detekce překážky při průjezdu trajektorie

4.5.5 Časová náročnost algoritmu

Výstupem programu `testpathplan` je histogram vyjadřující časovou náročnost algoritmu. Testy proběhly na obou platformách, *PPC* i *host*. Testován byl rozšířený algoritmus v porovnání s dříve užívaným algoritmem. Z výsledků vyplývá, že rozšířený algoritmus je



Obrázek 4.15: Nalezení nové cesty do cílového bodu

pro na platformě *PPC* pro delší trajektorie časově náročnější. Histogramy zobrazují obrázky – rozšířený algoritmus, obrázek 4.16 a 4.17 a zjednodušený algoritmus, obrázky 4.18 a 4.19.

Podmínky testu:

- Rozměr prostoru 3 × 2,1 m, rozměr soutěžního hřiště
- Rozměr mapy 120 × 84 buněk, rozměr buňky 25 × 25 mm
- Rozměr masky robota 275 × 300 mm
- Testovací platforma *host*, *PPC*
- Počet opakování algoritmu 10000
- Náhodně generované souřadnice startovního a cílového bodu
- Náhodně generované rozměry a souřadnice překážek
- Počet překážek 4
- Test několikrát opakován, výsledky obdobné, vybrána jedna série výsledků

```
Results:min=0.000000000 mean=0.006983916        max=0.049326000
  min=   0 ns
  mean=  7 ms
  max=  49 ms
  0 ns -   1 us   0.01%      1
  1 us -   2 us   0.01%      1
  2 us -   3 us   0.03%      3 =
  3 us -   6 us   0.02%      2 =
  6 us -  10 us   0.04%      4 =
 10 us -  18 us   0.24%     24 =
 18 us -  32 us   0.19%     19 =
 32 us -  56 us   0.07%      7 =
 56 us - 100 us   0.37%     37 =
100 us - 178 us  12.62%   1262 =======
178 us - 316 us  15.26%   1526 ========
316 us - 562 us   4.26%    426 ===
562 us -   1 ms   7.31%    731 ====
  1 ms -   2 ms   9.95%    995 =====
  2 ms -   3 ms  10.45%   1045 ======
  3 ms -   6 ms   9.46%    946 =====
  6 ms -  10 ms   7.06%    706 ====
 10 ms -  18 ms   4.52%    452 ===
 18 ms -  32 ms  14.38%   1438 ========
 32 ms -  56 ms   3.74%    374 ==
 56 ms - 100 ms   0.00%      0
100 ms - 178 ms   0.00%      0
178 ms - 316 ms   0.00%      0
316 ms - 562 ms   0.00%      0
562 ms -   1  s   0.00%      0
  1  s - inf  s   0.00%      0
```

Obrázek 4.16: Histogram, rozšířený algoritmus, *host*

```
Results:min=0.012077000 mean=0.159652701        max=0.978851000
  min=  12 ms
  mean=160 ms
  max= 979 ms
  0 ns -   1 us   0.00%      0
  1 us -   2 us   0.00%      0
  2 us -   3 us   0.00%      0
  3 us -   6 us   0.00%      0
  6 us -  10 us   0.00%      0
 10 us -  18 us   0.00%      0
 18 us -  32 us   0.00%      0
 32 us -  56 us   0.00%      0
 56 us - 100 us   0.00%      0
100 us - 178 us   0.00%      0
178 us - 316 us   0.00%      0
316 us - 562 us   0.00%      0
562 us -   1 ms   0.00%      0
  1 ms -   2 ms   0.00%      0
  2 ms -   3 ms   0.00%      0
  3 ms -   6 ms   0.00%      0
  6 ms -  10 ms   0.00%      0
 10 ms -  18 ms  28.67%   2867 ==============
 18 ms -  32 ms  11.79%   1179 ======
 32 ms -  56 ms  12.79%   1279 =======
 56 ms - 100 ms  12.16%   1216 =======
100 ms - 178 ms   9.27%    927 =====
178 ms - 316 ms   5.97%    597 ===
316 ms - 562 ms   9.33%    933 =====
562 ms -   1  s  10.01%   1001 =====
  1  s - inf  s   0.00%      0
```

Obrázek 4.17: Histogram, rozšířený algoritmus, *PPC*

```
Results:min=0.000000000 mean=0.000811780       max=0.027425000
  min=   0 ns
  mean=812 us
  max=  27 ms
  0 ns -   1 us   0.01%      1
  1 us -   2 us   0.01%      1
  2 us -   3 us   0.01%      1
  3 us -   6 us   0.00%      0
  6 us -  10 us   0.04%      4 =
 10 us -  18 us   0.27%     27 =
 18 us -  32 us   0.05%      5 =
 32 us -  56 us   0.11%     11 =
 56 us - 100 us   0.24%     24 =
100 us - 178 us   0.71%     71 =
178 us - 316 us   1.64%    164 =
316 us - 562 us  31.95%   3195 ================
562 us -   1 ms  45.92%   4592 =======================
  1 ms -   2 ms  14.34%   1434 ========
  2 ms -   3 ms   3.68%    368 ==
  3 ms -   6 ms   0.89%     89 =
  6 ms -  10 ms   0.08%      8 =
 10 ms -  18 ms   0.02%      2 =
 18 ms -  32 ms   0.02%      2 =
 32 ms -  56 ms   0.00%      0
 56 ms - 100 ms   0.00%      0
100 ms - 178 ms   0.00%      0
178 ms - 316 ms   0.00%      0
316 ms - 562 ms   0.00%      0
562 ms -   1  s   0.00%      0
  1  s - inf  s   0.00%      0
```

Obrázek 4.18: Histogram, zjednodušený algoritmus, *host*

```
Results:min=0.033950000 mean=0.045977917       max=0.223126000
  min=  34 ms
  mean= 46 ms
  max= 223 ms
  0 ns -   1 us   0.00%      0
  1 us -   2 us   0.00%      0
  2 us -   3 us   0.00%      0
  3 us -   6 us   0.00%      0
  6 us -  10 us   0.00%      0
 10 us -  18 us   0.00%      0
 18 us -  32 us   0.00%      0
 32 us -  56 us   0.00%      0
 56 us - 100 us   0.00%      0
100 us - 178 us   0.00%      0
178 us - 316 us   0.00%      0
316 us - 562 us   0.00%      0
562 us -   1 ms   0.00%      0
  1 ms -   2 ms   0.00%      0
  2 ms -   3 ms   0.00%      0
  3 ms -   6 ms   0.00%      0
  6 ms -  10 ms   0.00%      0
 10 ms -  18 ms   0.00%      0
 18 ms -  32 ms   0.00%      0
 32 ms -  56 ms  89.13%   8912 ==================================
 56 ms - 100 ms  10.40%   1040 ======
100 ms - 178 ms   0.45%     45 =
178 ms - 316 ms   0.02%      2 =
316 ms - 562 ms   0.00%      0
562 ms -   1  s   0.00%      0
  1  s - inf  s   0.00%      0
```

Obrázek 4.19: Histogram, zjednodušený algoritmus, *PPC*

4.5.6 Výsledky a zhodnocení

Simulované testy ukázaly spolehlivou funkci algoritmu. Praktické testy na pohybujícím se robotu odhalily některé detaily, které je pro bezchybnou práci algoritmu třeba odstranit. Hlavně je to systematická chyba určení pozice a úhlu natočení robota z nezávislé odometrie, kdy je chyba (po ujetí rovné trajektorie délky 2 m a otočení o úhel π) 8 cm v obou osách x a y a úhlu $\pi/8$ a dále se integruje. Tuto chybu registrujeme již delší dobu a zabránila lepšímu umístění týmu Flamingos v soutěži Eurobot 2011. Dále je třeba vyřešit jiné zakreslovaní překážek do mapy a mnoho detailů v software robota přizpůsobených pro původní algoritmus. Nalezené chyby se snažíme postupně odstranit, to však vyžaduje hluboké znalosti složitého software soutěžního robota a dlouhodobé testovaní, na odstranění všech nalezených chyb nebyl již v této práci prostor.

Dle testu `testpathplan` je rozšířený algoritmus na platformě *PPC* časově náročnější. Tento nedostatek by měl odstranit přechod na novou platformu řídícího počítače robota, modul Gumstix.

Kapitola 5

Závěr

Obě části diplomové práce byly úspěšně splněny. V první části je popsán mechanický koncept robota. Dle něho byl robot vyroben a sestaven. Studentský tým Flamingos se s ním úspěšně účastnil národního kola soutěže Eurobot v Česku (8. místo) a Německu (5. místo). Poznámky a návrhy na další vylepšení mechanické konstrukce jsou uvedeny v sekci 3.6, jde především o zajištění perfektního kontaktu hnaných kol s povrchem hřiště.

Druhá část práce se zabývá plánováním bezkolizní trajektorie. Původní A* algoritmus byl rozšířen o reprezentaci robota jeho reálným tvarem a rozměry. Popsaný přístup byl implementován a testován simulací a na reálném robotu. Testy potvrzují spolehlivou funkci navrženého algoritmu, robot se nyní může přesně pohybovat v omezeném prostoru. Budoucí práce by měla dále testovat integraci algoritmu v software robota a dokončit přechod na novou platformu řídícího počítače robota Gumstix. Výpočetní náročnost algoritmu dále může snížit optimalizace algoritmu a použití některých technik z [9] či [10].

Dle mého názoru je projekt robotické soutěže Eurobot úžasný, přináší studentům rozhled v mnoha odvětvích a racionální pohled na komplexní úlohy. Jsem velmi rád, ze jsem se mohl projektu účastnit, obohatil mě o cenné vědomosti a zkušenosti.

Literatura

[1] K. O. Arras, J. Persson, N. Tomatis, and R. Siegwart. Real-time obstacle avoidance for polygonal robots with a reduced dynamic window. In *Proceedings of the 2002 IEEE International Conference on Robotics Automation, Washington, DC*, pages 3050–3055, 2007.

[2] J. Benda. *CANOpen komunikace pro mobilního robota*. 2008. Diplomoá práce, ČVUT.

[3] O. K. Bruno Siciliano, editor. *Handbook of Robotics*. Springer, 2008.

[4] V. Burian. *Využití programovatelného pole pro řízení bezkartáčových motorů*. 2011. Bakalářská práce, ČVUT.

[5] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. The MIT Press, 2005.

[6] Y. K. Hiroaki Seki and M. Hikizu. Real-time obstacle avoidance using potencial field for a nonholonomic vehicle. In J. Silvestre-Blanes, editor, *Factory Automation*, pages 523–542, 2010.

[7] J. Kubias. *Hardware robota pro soutěž Eurobot*. 2010. Diplomoá práce, ČVUT.

[8] P. Novák. *Mobilní roboty – pohony, senzory, řízení*. 2005.

[9] W. Y. Sunglok Choi, Jae-Yeong Lee. Fast any-angle path planning on grid maps with non-collision pruning. In *Robotics and Biomimetics (ROBIO), 2010 IEEE International Conference*, pages 1051 – 1056, 2010.

[10] C. Warren. Fast path planning using modified A* method. In *Robotics and Automation, IEEE International Conference*, volume 2, pages 662 – 667, 1993.

[11] Robotický den – hlavní stránka.
http://www.eurobot.cz/, stav z 20. 3. 2011.

[12] Eurobot Association – hlavní stránka.
http://www.eurobot.org/, stav z 20. 3. 2011.

[13] Český překlad doplnění pravidel soutěže Eurobot – FAQ.
http://www.eurobot.cz/2011/E2011_FAQ1-CZ.pdf, stav ze 29. 12. 2011.

[14] HOKUYO AUTOMATIC CO., LTD. – hlavní stránka.
http://www.hokuyo-aut.jp/, stav z 29. 12. 2011.

[15] Gumstix small open source hardware.
http://gumstix.com/, stav ze 29. 12. 2011.

[16] libpng Home Page.
http://www.libpng.org/pub/png/libpng.html, stav ze 29. 12. 2011.

[17] maxon motor ag – hlavní stránka.
www.maxonmotor.com, stav ze 29. 12. 2011.

[18] MIDAM Control system – Produkty.
http://www.midam.cz/Produkty.html, stav ze 29. 12. 2011.

[19] OCERA – ORTE.
http://www.ocera.org/download/components/WP7/orte-0.3.1.html, stav ze 29. 12. 2011.

[20] Řetězy olomouc – katalog kuželových soukolí.
http://www.retezyolomouc.cz/data/produkty/files/202.pdf, stav ze 29. 12. 2011.

[21] Robot Klub Rychnov – Inkrementální enkodér.
http://www.vosrk.cz/robotika/guide/sensors/decode/cs, stav ze 29. 12. 2011.

[22] Ubuntu Packages Search.
http://packages.ubuntu.com/, stav ze 29. 12. 2011.

[23] Wiki týmu Flamingos.
http://rtime.felk.cvut.cz/robot, stav ze 29. 12. 2011.

[24] Wikipedia – A* search algorithm.
http://en.wikipedia.org/wiki/A*_search_algorithm, stav ze 29. 12. 2011.

[25] Wikipedia – Cross compiler.
`http://en.wikipedia.org/wiki/Cross_compiler`, stav ze 29. 12. 2011.

[26] Wikipedia – Euclidean distance.
`http://en.wikipedia.org/wiki/Euclidean_distance`, stav ze 29. 12. 2011.

[27] Český překlad pravidel soutěže Eurobot.
`http://www.eurobot.cz/2011/E2011_Rules-CZ-v1.pdf`, stav ze 29. 12. 2011.

[28] Alwayse Engineering Limited – hlavní stránka.
`http://www.alwayse.com`, stav ze 29. 12. 2011.

[29] Google SketchUp – hlavní stránka.
`http://sketchup.google.com`, stav ze 29. 12. 2011.

[30] Omni-Wheel Robot.
`http://www.societyofrobots.com/robot_omni_wheel.shtml`, stav ze 29. 12. 2011.

[31] Robotický tým FELaaCZech – hlavní stránka.
`http://www.felaaczech.eu/`, stav ze 29. 12. 2011.

[32] C. H. Xuefu Xiang, Jiye Zhang. An efficient system for nonholonomic mobile robot-path planning. In *Proceedings on Intelligent Systems and Knowledge Engineering ISKE2007*, 2007.

[33] S. X. Yang and M. Q.-H. Meng. Real-time collision-free motion planning of mobile robot using a neural dynamics-based approach. *IEEE Transactions On Neural Networks*, 14(6):1541–1552, 2003.

Příloha A

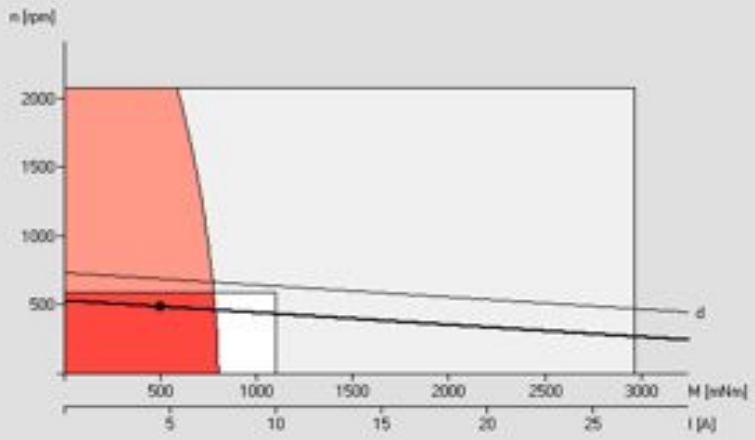
Protokol z programu maxon Selection Program

maxon motor
driven by precision

maxon selection program 1 / 4

maxon drive unit			
Gear	370692	GP 22 HP	Gear reduction ratio 29:1, Ball bearings
Motor	386676	EC 22	100W, brushless, 2 Shaft ends, Ball bearings
Control			

Key data of drive		
(Needed) motor voltage (= Umot)	8,7	V
No-load speed (at Umot)	525	rpm
Load stall torque (theoretic, at Umot)	6180	mNm
Starting current (theoretic, at Umot)	56,2	A
Max. load current	4,86	A
Motor load	67	%
Diameter	22	mm
Length	91,9	mm
Ambient temperature (=Tamb)	25	°C
Average winding temperature	58	°C
Housing temperature	54	°C



[ ] Scale motor	[ ] Output power (e)
[x] Working points	[ ] Efficiency (f)
[ ] Max. contin. torque motor (a)	[ ] Winding temperature (g)
[ ] Contin. current (b)	[ ] Max. contin. torque gear (h)
[ ] Max. current (c)	[ ] Max. torque gear (i)
[x] Max. voltage (d)	[ ] Max. speed gear (k)

maxon motor
driven by precision

maxon selection program 2 / 4

Load input		
Max. load speed	480	rpm
Load torque (RMS)	500	mNm
Max. load torque	500	mNm
Duration of max. load	86400	s

Power supply input		
Min. supply voltage	0	V
Max. supply voltage	12	V
Max. continuous current	10	A
Max. current (peak)	10	A
Max. duration of peak current	1	s

Drive parameter			
Belt drive	Diameter d1	15	mm
	Diameter d2	15	mm
	Efficiency	100	%
	Mass inertia J1	0	gcm²
	Mass inertia J2	0	gcm²
	Belt mass m	0	kg

maxon motor
driven by precision

maxon selection program 2 / 4

Load input			
Max. load speed		480	rpm
Load torque (RMS)		500	mNm
Max. load torque		500	mNm
Duration of max. load		86400	s

Power supply input			
Min. supply voltage		0	V
Max. supply voltage		12	V
Max. continuous current		10	A
Max. current (peak)		10	A
Max. duration of peak current		1	s

Drive parameter			
Belt drive	Diameter d1	15	mm
	Diameter d2	15	mm
	Efficiency	100	%
	Mass inertia J1	0	gcm²
	Mass inertia J2	0	gcm²
	Belt mass m	0	kg

maxon motor
driven by precision

maxon selection program 2 / 4

Load input			
Max. load speed		480	rpm
Load torque (RMS)		500	mNm
Max. load torque		500	mNm
Duration of max. load		86400	s

Power supply input			
Min. supply voltage		0	V
Max. supply voltage		12	V
Max. continuous current		10	A
Max. current (peak)		10	A
Max. duration of peak current		1	s

Drive parameter			
Belt drive	Diameter d1	15	mm
	Diameter d2	15	mm
	Efficiency	100	%
	Mass inertia J1	0	gcm²
	Mass inertia J2	0	gcm²
	Belt mass m	0	kg

Příloha B

Obsah přiloženého CD

`/pathplan` – zdrojové kódy, složka `/src/pathplan` repozitáře týmu Flamingos (.h, .c)

`/mechanika` – výkresy, modely a dokumentace týkající se návrhu mechaniky robota (.pdf, .skp, .dxf, .art)

`/videa` – videa a animace z testů algoritmu a robota (.mov)

`/dp` – diplomová práce v elektronické podobě, verze pro tisk a pro elektronické publikování (redukovaná velikost souboru) (.pdf)