Mendelova univerzita v Brně
Provozně ekonomická fakulta

Plánování trajektorie robota v dynamicky měnícím se prostředí s ohledem na splnění specifického cíle

Diplomová práce

Vedoucí práce:
Ing. Jan Kolomazník, Ph.D.

BC. JAN KOTEN

Brno 2017

Rád bych věnoval poděkování vedoucímu práce Ing. Janu Kolomazníkovi, Ph.D. za jeho odbornou pomoc a čas strávený na konzultacích. Dále bych chtěl poděkovat rodině za podporu, kterou mi poskytovala po celou dobu studia.

Čestné prohlášení

Prohlašuji, že jsem tuto práci: **Plánování trajektorie robota v dynamicky měnícím se prostředí s ohledem na splnění specifického cíle**
vypracoval samostatně a veškeré použité prameny a informace jsou uvedeny v seznamu použité literatury. Souhlasím, aby moje práce byla zveřejněna v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů, a v souladu s platnou *Směrnicí o zveřejňování vysokoškolských závěrečných prací.*

Jsem si vědom, že se na moji práci vztahuje zákon č. 121/2000 Sb., autorský zákon, a že Mendelova univerzita v Brně má právo na uzavření licenční smlouvy a užití této práce jako školního díla podle § 60 odst. 1 Autorského zákona.

Dále se zavazuji, že před sepsáním licenční smlouvy o využití díla jinou osobou (subjektem) si vyžádám písemné stanovisko univerzity o tom, že předmětná licenční smlouva není v rozporu s oprávněnými zájmy univerzity, a zavazuji se uhradit případný příspěvek na úhradu nákladů spojených se vznikem díla, a to až do jejich skutečné výše.

V Brně dne 14. května 2017 _____

Abstract

Koten, J. *Plánování trajektorie robota v dynamicky měnícím se prostředí s ohledem na splnění specifického cíle.* Brno: Mendelova univerzita, 2017.

The aim of the thesis is a trajectory optimization of robot in the Ketchup House competition. Environment mapping system is proposed to deal with this issue. As an ideal solution is chosen a method of familiar environment scanning using a camera module and lidar. Camera module and lidar provide a possibility to scan up to nine searched objects, instead of one object allowed by the previous solution. A proximity sensor supplements the whole system, so the efficiency of detection increase. To find a particular position of the object on the playing field, image processing methods are used, namely line detection and thresholding. Moreover, new playing strategy and a new way of representing the playing field are implemented. Robot K4, created by PEF MENDELU AiStorm team, serves for application of the system.

Key words

Ketchup House, robot trajectory planning, computer vision, thresholding, laser detection

Abstrakt

Koten, J. *Plánování trajektorie robota v dynamicky měnícím se prostředí s ohledem na splnění specifického cíle*. Brno: Mendelova univerzita, 2017.

Cílem práce je optimalizovat trajektorii robotu v soutěži Ketchup house. Pro tento problém byl navrhnut systém mapování prostředí. Jako ideální řešení bylo zvoleno skenování známého prostředí pomocí kamerového modulu a laserového zařízení. Pomocí laserového zařízení a kamerového modulu je možné naskenovat devět hledaných objektů namísto jednoho, které poskytovalo předchozí řešení. Celý systém byl doplněn proximitním snímačem pro zvýšení efektivnosti detekce. K nalezení konkrétní pozice objektu na hracím poli byly použity metody segmentace obrazu, a to konkrétně detekce linií a prahování. K navrženému řešení byla implementována nová herní strategie a nový způsob reprezentace hracího pole. Celý systém je aplikován na robotu K4, který byl vytvořen týmem PEF MENDELU AiStorm.

Klíčová slova

Ketchup House, plánování trajektorie robotu, počítačové vidění, prahování, laserový systém detekce

Obsah

1 Úvod a cíl práce 9

1.1 Cíl práce 9

2 Navigace robotu 10

2.1 Globální a lokální navigační systém 10

2.2 Odometrie 10

2.3 Trilaterace a triangulace 10

2.4 Použití BSP pro plánováni trajektorie 11

2.5 Manhattanská metrika 11

3 Senzorové soustavy 12

3.1 Ultrazvukový senzor 12

3.2 Infračervený senzor 12

3.3 Laser 13

3.4 Mechanické senzory 14

4 Snímání obrazu a jeho zpracování 15

4.1 Snímání obrazu 15

4.2 Segmentace obrazu 17

4.3 Prahování 17

4.4 BoofCV 18

4.5 Raspistill 18

5 Hardwarové vybavení pro řízení robota 19

5.1 Raspberry Pi 19

5.2 Arduino 20

6 Metodika práce 22

6.1 Ketchup house 22

6.2 Mobilní robot K4 24

6.3 Výběr mechanizmu pro skenování prostředí 25

6.4 Použití laseru v soutěži Ketchup House ... 26

7 Laserové zařízení a skenování prostředí 27

7.1 Popis zařízení ... 27

7.2 Možné realizace návrhu zařízení ... 28

7.3 Realizace zařízení ... 30

7.4 Ovládání laserového zařízení ... 34

7.5 Odhalení slepých míst ... 37

8 Konfigurace laserového zařízení a kamerového modulu pro detekci konkrétního objektu 39

8.1 Princip identifikace objektů ... 39

8.2 Nastavení kamerového modulu ... 40

9 Návrh a implementace algoritmu pro rozeznávání konkrétních objektů 43

9.1 Návrh algoritmu ... 43

9.2 Detekce objektu ... 43

9.3 Detekce objektu ve snímku ... 44

9.4 Detekce polohy objektu na hracím poli ... 49

9.5 Implementace algoritmu ... 50

10 Plánování trajektorie robota 53

10.1 Reprezentace hracího pole ... 53

10.2 Orientace a pohyb na hracím poli ... 56

10.3 Herní strategie robota ... 57

11 Závěr 59

Literatura 60

Seznam obrázků 62

Seznam tabulek 65

1 Úvod a cíl práce

Práce se zabývá problematikou plánování trajektorie robotu. Robot se pohybuje ve známém prostředí. Zná jeho tvar, velikost a svou polohu v něm. Nicméně v tomto prostředí se nachází jisté předměty, jejichž poloha se mění a robotu není známa. Úkolem robotu je co nejvíce těchto předmětů posbírat a přesunout je na předem určené místo.

Robot disponuje periferiemi, které mu umožnují pohyb v prostředí a sběr předmětů. Robot se řídí algoritmem, který jej vede stále stejnou cestou, a pokud robot narazí na předmět, je schopen ho rozpoznat, sebrat a pokračovat dál po předem dané trajektorii. Na konci trajektorie robot vyloží předměty a pokračuje od začátku.

Trajektorii robotu lze optimalizovat přidáním mechanizmu, který robotu umožní na dálku zmapovat celé prostředí, nebo jeho část. Podle výsledku mapování se robot bude moci podle předem daných pravidel rozhodnout pro další kroky. Tím může být v případě nalezeného předmětu příkaz k jeho sebrání, když mapování nic neukáže, robot může začít mapovat další část prostředí.

K tomuto bude nutné robota vybavit přídavnými senzory, případně kombinací senzorů, které robotu umožní zjistit polohu předmětu dřív, než se bude nacházet v jeho bezprostřední blízkosti.

1.1 Cíl práce

Cílem práce je optimalizace trajektorie robotu s ohledem na dynamicky měnící se prostředí. Pro to je nutné nastudovat nejběžněji používané senzory, je však třeba uvažovat jen o těch senzorech, které je možné reálně umístit na robota K4. Z těchto senzorů vybrat takový senzor nebo kombinaci senzorů, která poskytne robotu co nejlepší informaci o aktuálním stavu na hracím poli.

Po vybrání senzoru nebo skupiny senzoru bude nutné doplnit stávající řešení robotu K4 o implementaci, která bude řešit práci s tímto senzorem nebo soustavou senzorů. Současné řešení nedisponuje žádným mechanizmem, který by dokázal zmapovat situaci v dostatečně velkém okolí před robotem.

Ve stavu, kdy robot bude disponovat mechanizmem pro skenování hracího pole, bude nutné tento mechanizmus aplikovat v herní strategii robota. Jelikož robot disponuje statickou herní strategií, bude nutné tuto strategii zcela odstranit a nahradit strategií novou. Nová strategie musí reagovat na individualitu každé hry.

Poslední úkol, který je nutné vyřešit, je způsob pohybu robota na herním poli. Stávající řešení umožňuje robotu vykonat pohyb na úrovni kroku vpřed, kroku vzad a otočení o daný úhel. Nové řešení musí dovést robota z bodu A do bodu B bez ohledu na to, kde se robot zrovna nachází. Ke splnění tohoto úkolu je nutné navrhnout reprezentaci herního pole.

2 Navigace robotu

Tato kapitola pojednává o základní problematice vedení robotu z jeho výchozí pozice k zadanému cíli. Nejedná se pouze o plánování trajektorie robotu, ale i jeho orientaci v prostoru.

2.1 Globální a lokální navigační systém

Globální navigační systém je takový systém, který má za úkol dovést robota z bodu A do bodu B. Tento systém není nutné chápat jako celosvětový, ale pro náš případ stačí uvést, že se jedná o známé prostředí robota.

Pro zjednodušení je dobré nějakým způsobem identifikovat orientaci robota v tomto prostředí, nejčastější způsob je použití světových stran. Časté je sjednocení s geografickým severem, ale není to podmínkou.

Lokální navigační systém robotu má za úkol jistou korekci k cílovému bodu. Na trajektorii určenou globálním navigačním systémem se mohou vyskytnout jisté překážky, které v některých případech mohou robotu zamezit cestu k cíli. Robot musí být schopen v dostatečném předstihu, pokud je to nutné, na tuto překážku zareagovat a přepočítat trajektorii nebo tuto překážku odstranit. S tím souvisí i výběr vhodné senzorické soustavy, která je schopna tyto překážky odhalit. (Novák, 2007)

2.2 Odometrie

Jedná se o nejpoužívanější metodu spadající do skupiny relativní navigace. Základem relativní navigace je startovní bod robotu. Při pohybu robotu se měří přírůstek od výchozího bodu, nesmíme zde opomenout ani změnu směru. Nevýhoda této metody je stále se zvyšující chyba skutečné polohy robotu od polohy vypočítané.

Tato metody je typická pro roboty disponující systémem kol pro pohyb. Uvnitř robotu je implementován jeho kinematický model, který reaguje na změnu stavu akčních členů, a tímto způsobem dokáže vypočítat aktuální polohu robotu.

V případě kolových robotů zde je velice důležitý styk kola robotu s podlahou. Při prokluzu kola robotu vzniká chyba určení polohy. (Novák, 2007)

2.3 Trilaterace a triangulace

Triangulace a trilaterace jsou metody spadající do skupiny absolutní navigace. Úkolem absolutní navigace je určení polohy robotu vůči vybranému referenčnímu bodu.

K určení polohy robotu od referenčního za pomoci vzdálenosti slouží trilaterace. Triangulace je metoda, která určuje polohu robotu pomocí tří úhlů: $\lambda_{1,2,3}$

a tří referenčních bodů $S_{1,2,3}$. Pomocí těchto tří naměřených úhlů je možné určit polohu robotu.

K těmto dvěma metodám je nutné použít vhodnou senzorickou soustavu. Nejčastěji se používají laserové paprsky, ultrazvuky a IR senzory.

2.4 Použití BSP pro plánováni trajektorie

Pro naplánování trajektorie robotu lze využít algoritmy z teorie grafů. Je nutné si uvědomit, co v prostředí, ve kterém se robot pohybuje, znamenají uzly a co hrany.

Jako příklad uvedu zjednodušenou verzi algoritmu prohledávání do šířky.
V tomto algoritmu navštěvuje uzly v takovém pořadí, v jakém jsou navštíveny od kořene.
Pro zpřehlednění si jako uzly určíme klíčové body, které robot musí navštívit (například křižovatky) a hrany cesty mezi nimi. Tyto hrany můžeme označit hodnotou, která může představovat například vzdálenost mezi křižovatkami. Algoritmus plánování trajektorie by vypadal následovně:

```
1 PlanovaniTrajektorie (uzel);
2 cesta;
3 while uzel != null do
4 |   cesta += uzel;
5 |   uzel = uzel.rodic;
6 end
7 return cesta;
```

Předpoklad pro správnou funkci algoritmu je, že startovací uzel nemá žádného následníka, jedná se tedy o list. (Brackeen, 2004)

2.5 Manhattanská metrika

Tato metrika vychází z organizace ulic v Manhattanu. Ulice spolu svírají pravý úhel a v každém průsečíku ulic se nachází křižovatka. Z ptačí perspektivy si lze tuto část New Yorku představit jako šachovnici. Vzdálenost mezi dvěma křižovatkami (A, B) lze vypočítat pomocí rovnice: $vzdálenost(A, B) = |Ax - Bx| + |Ay - By|$.

V případě, že se robot pohybuje v prostředí šachovnicového charakteru, můžeme tuto metodu použít ve výpočtu nejkratší cesty mezi robotem a průsečíkem os.

3 Senzorové soustavy

Senzor neboli čidlo, též označovaný také jako snímač, je zařízení, které dokáže zpracovávat nějakou veličinu, například fyzikální nebo technickou. Tuto veličinu dále dokáže převádět na signál (digitální nebo analogový). Tato kapitola poskytuje základní přehled o senzorech aplikovatelných na robot pro měření vzdálenosti a rozpoznání překážky před sebou.

Vybrání vhodného senzoru nebo senzorů je pro splnění zadaného úkolu robotem velice důležité. Ze senzorů robot přijímá veškerá data z okolí, se kterými dále pracuje a rozhoduje se podle předem nadefinovaných pravidel, co bude následovat v dalším kroku.

3.1 Ultrazvukový senzor

Jedná se o senzor, který se skládá ze tří částí: vysílače, přijímače a vyhodnocovacího obvodu. Takovýto senzor dokáže měřit vzdálenost od jednotek až po stovky centimetrů.

Princip zjišťování vzdálenosti je podobný tomu, co využívají netopýři. Vysílač ultrazvuku vyšle ultrazvukový signál, ten se v případě, že se před senzorem nachází objekt, od něj odrazí. Odražený signál putuje jako ozvěna nazpátek k senzoru, tam ho zachytí přijímač.

Důležitý je čas, který uběhne od vyslání ultrazvukového signálu k zachycení jeho ozvěny. Z těchto dvou hodnot vyhodnocovací obvod dokáže vypočítat vzdálenost mezi senzorem a detekovaným předmětem. K tomuto výpočtu je nutné znát frekvenci ultrazvukového signálu a rychlost, kterou se šíří. (EVERETT, c1995)

3.2 Infračervený senzor

Infračervený senzor se skládá z vysílače infračerveného světla, přijímače infračerveného světla a pomocného obvodu.

Existují dva základní infračervené senzory: reflexní senzor a senzor využívající principu triangulace.

Vysílač reflexního senzoru vyšle IR signál (posvítí) před sebe, pokud se před senzorem nachází nějaký objekt, infračervené světlo se od něj odrazí a je zachyceno přijímačem. Důležité je, že přijímač reaguje pouze na infračervenou složku světla.

Vzdálenost je určena z intenzity světla odraženého od předmětu před senzorem. Čím je intenzita větší, tím se předmět nachází blíže senzoru, čím je intenzita odraženého světla menší, tím se předmět nachází dále od senzoru.

Pomocný obvod podle intenzity odraženého světla vyšle signál (analogový nebo digitální), podle kterého lze určit vzdálenost předmětu.

Senzor využívající triangulace pracuje obdobně. Také obsahuje vysílač infračerveného světla, který ovšem nesvítí přímo před sebe, ale s určitým sklonem k přijímači. Přijímač je tvořen páskem fotocitlivých rezistorů s vestavěnou čočkou.

Vysílač vyšle signál (posvítí) na objekt před sebou, ten se odrazí a dopadne na jeden z fotocitlivých rezistorů, podle fotocitlivého rezistoru, na který dopadl, zjistíme úhel odrazu. Z dvou úhlů a jedné strany jsme schopni dopočítat vzdálenost mezi předmětem a senzorem. (Novák, 2005)

3.3 Laser

Laser je zařízení, které produkuje úzký svazek světla. Na rozdíl od jiných zdrojů světla (přirozených) laser produkuje světlo, které je monochromatické. Z toho plyne, že laser produkuje světlo pouze v jedné vlnové délce. Díky tomu laser nachází mnohé uplatnění ve zdravotnictví a průmyslu.

Při manipulaci s laserem je nutné dbát zvýšené obratnosti, aby nedošlo k poškození zraku, popálení pokožky nebo zapálení cizích předmětů.

Tab. 1 Bezpečnostní třídy laserů
Zdroj: http://www.carove-lasery.cz/soubor-maximalni-povoleny-vykon-pro-jednotlive-tridy-v-zavislosti-na-vlnove-delce-10-.pdf

Třída	Specifikace – riziko
1	Velmi nízko výkonové lasery nebo zapouzdřené lasery
1M	Velmi nízko výkonové lasery s optickými členy rozšířeným paprskem na velký průměr. Takto mohou obsahovat i zdroje záření vyšší třídy laserového záření.
2	Nízko výkonové lasery ve viditelném spektru
2M	Nízko výkonové lasery s otickými členy rozšířeným paprskem na velký průměr. Takto mohou obsahovat i zdroje záření vyšší třídy laserového záření.
3R	Nižší výkony laserů ve viditelném spektru.
3B	Střední výkony laserů ve viditelném spektru.
4	Výkonové lasery.

Tyto bezpečnostní třídy jsou v souladu s normou IEC 60825-1. (Bezpečnost laserových zařízení, 2015) Pro používání laserů v prostředí, kde je možné, aby se oko střetlo s laserovým paprskem, je nutné, aby laser nepřekročil třídu 3R. Pro větší bezpečnost je možné na předmět nesvítit přímo, ale odrazem. Dále je vhodné neponechávat paprsek zaměřený na jedno místo příliš dlouhou dobu, abychom eliminovali riziko vznícení zaměřených předmětů.

Laser lze použít k měření vzdálenosti. K měření vzdálenosti lze opět využít princip triangulace. Opět vycházíme z toho, že paprsek dopadá na měřený předmět a od něj se odráží v konstantním úhlu. Odražený paprsek dopadá na snímač, který vyhodnotí vzdálenost předmětu podle toho, na jakou část snímače paprsek dopadl. Z toho je jasné, že měření vzdálenosti funguje na měření vzdálenosti od-do.

Druhá metoda měření vzdálenosti je podobná ultrazvukovému senzoru. Měříme čas, který uběhne od vyslání paprsku vysílačem (laserem) až do jeho

přijmutí přijímačem (senzorem). Pokud známe rychlost, kterou se paprsek šíří, jsme schopni dopočítat vzdálenost. (Vojáček, 2015)

Měření vzdálenosti není jediný problém, ke kterému lze laser využít. Výběrem vhodné optiky a jejím umístěním před paprsek lze tento paprsek upravit, například na čáru, kříž atd. S použitím čárového filtru je možné na předmět vysvítit souvislou světelnou linku, která se skládá pouze z jedné vlnové délky (barvy). S použitím vhodného hardwaru pro zachycení linky a algoritmu pro její analýzu lze například zjistit tvar osvíceného předmětu. Zde záleží na materiálu předmětu a jeho barvě. Paprsek laseru nemusí být na některých barvách příliš čitelný.

3.4 Mechanické senzory

Jako mechanický senzor může sloužit například jednoduché tlačítko. Toto řešení vyniká hlavně svojí spolehlivostí a jednoduchostí. Pro konkrétní úlohu je nutné senzor modifikovat tak, aby ji plnil co nejlépe.

Málokdy se stane, že již zmiňované tlačítko bude mít ideální parametry pro zadaný problém, v takovémto případě je nutné navrhnout vlastní přídavnou konstrukci, případně vytvořit i vlastní pomocný obvod. Tlačítko se se dá využít jako indikátor dorazu při kalibraci nebo senzor dotyku, takovýto mechanizmus využívá i náš robot.

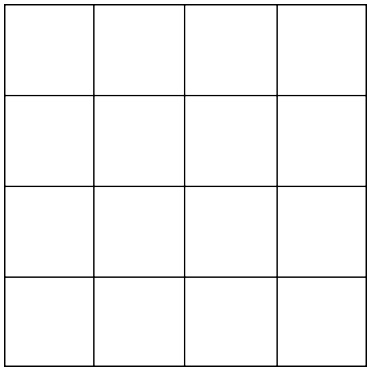
4 Snímání obrazu a jeho zpracování

4.1 Snímání obrazu

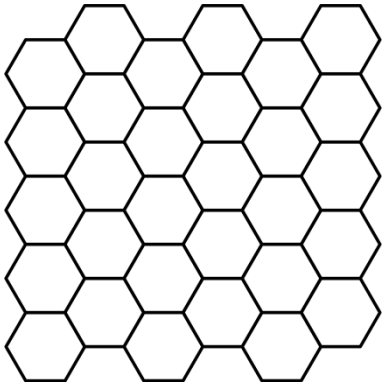
Hlavní úlohou snímání a digitalizace je získání obrazu z reálného světa a jeho převedení. Kroky a nastavené parametry v zpracování se mnohdy liší, především záleží na dané úloze, tedy na tom, jak chceme s obrazem dále naložit.

Sejmutí je proces převedení optické veličiny na analogový signál. Pro sejmutí a digitalizaci obrazu je vhodné znát okolní parametry a zařízení nastavit tak, abychom dosáhli co nejlepší kvality obrazu. Mezi tyto okolní vlivy může patřit například okolní světlo.

Digitalizace je dalším krokem. Jedná se o převod spojitého analogového signálu na signál digitální. Tento krok převádí obraz do podoby, která je vhodná pro zpracování počítačem. Obraz je nevzorkován do matice M x N. Vzorkování do této matice se řídí Shannonovým teorémem. Při pořizování fotografie musíme vědět předem, k čemu fotografii použijeme, kolik času nám může pořízení fotografie zabrat a podle toho nastavit její rozlišení. Pořízení fotografií s vysokým rozlišením nebo s dlouho dobou expozice zpravidla zabere velké množství výpočetního času. Ne vždy je možné takové množství času poskytnout. V takových případech musíme snížit naše nároky na kvalitu, ne vždy je to však překážkou. Jednotka pro rozlišení obrazu je DPI (Dots Per Inch), tato jednotka říká, kolik pixelů obsahuje jeden palec. Dále je pro digitalizaci obrazu důležitá volba vzorkovací mřížky. Nejčastěji se můžeme setkat s mřížkou čtvercovou obrázek 1 a hexagonální obrázek 2.



Obr. 1 Čtvercová mřížka
Zdroj: https://www.quora.com/How-many-ways-can-you-traverse-a-four-by-four-square-grid-without-doubling-back-or-going-in-the-wrong-direction

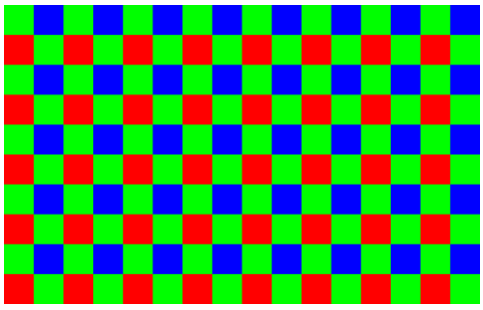


Obr. 2 Hexagonální mřížka
Zdroj: https://commons.wikimedia.org/wiki/File:Hexagonal_tiling.svg

Každá z těchto mřížek má své výhody i nevýhody, důležité je si ujasnit, k čemu bude pořízená fotografie sloužit.

Pro reprezentaci **Barvy** v obraze existují různé barevné modely. Výběr modelu ovlivňuje barevné vzezření obrazu. Barevný model RGB (Red, Blue, Green) se skládá ze tří barev. Tyto barvy jsou vlastně barevné složky, které jsou umístěny velmi blízko sebe. Nastavením hodnoty složek, a následně jejich součtem získáme výslednou barvu. Pro hodnoty R: 0, G: 0, B: 0 získáme výslednou barvu černou a pro R: 255, G: 255, B: 255 barvu bílou. Modely CMY a CMYK jsou používány při tisku fotografií.

S barvou také souvisí typ filtru. Nejpoužívanější je Bayerův filtr obrázek 3, jedná se o filtr, který filtruje dopadající světlo na čip snímače. Tento filtr se skládá ze čtyř filtrů, z nich každý propouští světlo pouze jedné vlnové délky. Světla propuštěná filtrem jsou: červená, zelená, modrá. Tyto filtry jsou uspořádané v mřížce 2 x 2, přičemž se zde zelený filtr nachází dvakrát. Zelený filtr je zde dvakrát kvůli vlastnostem lidského oka, to na zelenou barvu reaguje nejcitlivěji. Nevýhodou filtru je, že každý pixel obsahuje informaci pouze o jedné barvě. Další dvě barvy je nutné dopočítat.
(Fiřt, Holota, 2002)



Obr. 3 Bayerův filtr
Zdroj: https://commons.wikimedia.org/wiki/File:Bayer_matrix.svg

4.2 Segmentace obrazu

Segmentace obrazu je metoda, která se snaží rozdělit obraz na smysluplné celky. Takovými celky můžeme rozumět například oddělení předmětu od pozadí. Jedná se o proces přetváření digitálního obrazu do segmentů. Takto můžeme v obraze hledat různé objekty, přímky, křivky atd. Rozdělením obrazu do jednotlivých segmentů získáme jednu či více skupin pixelů, které spojuje nějaká společná vlastnost, to může být například barva, textura a vlnová délka. Takovouto skupinu pixelu nazýváme superpixel. Obraz upravený takovýmto způsobem se stává čitelnější pro strojové vidění. Segmentace obrazu též nachází velké uplatnění v lékařství.

4.3 Prahování

Prahování paří mezi nejjednodušší metody segmentace obrazu. Jedná se o převádění obrazu do binární podoby, tudíž každý pixel v obrázku může nabývat pouze jedné ze dvou hodnot. U této metody je důležité stanovení prahu, podle tohoto prahu se stanovuje hodnota každého pixelu nacházejícího se v upravovaném obrázku.

Hodnota pixelu je upravena podle prahu. Pokud je hodnota pixelu vyšší než prahová, je jeho hodnota přenastavena na bílou barvu. Pokud je hodnota testovaného pixelu nižší než prahová hodnota, je jeho hodnota přenastavena na barvu černou. Tato metoda je často používána na převod černobílých obrázků na binární.

Problém u standardního prahování je ve stanovení hodnoty prahu. Algoritmus upravuje hodnotu každého pixelu, ale nebere ohled na ucelené celky pixelů. Problém nastává u přechodů objektů. Pokud se jedná o hledaný objekt, může se stát, že některé jeho hraniční pixely nebudou zahrnuty nebo naopak. Problémem pro tento algoritmus jsou obrázky, ve kterých je šum, či obrázky, které jsou nerovnoměrně osvětleny. Výhodou je ovšem nízká výpočtová náročnost.

Pro tento problém existují různé metody, které tento algoritmus optimalizují a snaží se dosáhnout lepšího výsledku (Vlach, 2011).

Adaptivní prahování

Adaptivní prahování je jednou z optimalizací algoritmu. Tato metoda poskytuje lepší výsledek, a to například u obrázků, které nejsou rovnoměrně osvětlené.

Metoda na rozdíl od standardního prahování nepoužívá globální práh, který platí pro všechny pixely, ale lokální práh. Obrázek je rozdělen do více částí a pro každou část je stanoven lokální práh. U rozděleného obrázku jsou důležité velikosti částí. Ideální je, aby části nebyly příliš veliké a aby na nich bylo stejné osvětlení. U této metody je nevýhodou větší výpočetní náročnost. (Ijsc, 2015)

4.4 BoofCV

BoofCV je open-source knihovna vytvořená pro programovací jazyk Java. Její kód je volně otevřený, tudíž si ho každý vývojář v případě nutnosti může upravit. Tato knihovna je určena především pro počítačové vidění a aplikace ze zabývající se robotikou. Knihovna pracuje v reálném čase. Knihovna je napsána tak, aby její využitý bylo co nejjednodušší a zároveň nám poskytla co nejvyšší výkon. Problémy, které knihovna řeší, jsou rozdělené do balíčků. Jsou to balíčky: Image processing, Features, Geometric vision, Calibration, Recognition, Visualize a IO. Níže stručně popíši, co balíčky poskytují.

- Balíček *Image processing* se specializuje na běžnou práci s obrázky, a to na úrovni běžné práce s pixely.
- Balíček *Features* obsahuje implementované algoritmy řešící konkrétní problémy. Tento balíček je vhodný pro použití na vyšších úrovních řízení.
- Balíček *Calibration* napomáhá najít ideální nastavení fotoaparátu nebo kamery pro dosažení co nejkvalitnějšího snímku.
- Balíček *Recognition* řeší problematiku rozeznávání a sledování komplexních objektů vyskytujících se ve videu nebo na fotografii.
- Balíček *Geometric vision* slouží pro extrakci prvků z obrazu za pomocí 2D a 3D geometrie.
- *Visualize* je balíček poskytující řešení pro renderování.
- Poslední balíček *IO* poskytuje vstup a výstup pro různé datové struktury.

Každý z balíčků se obsahuje řešení pro specifický problém a třídy, které jsou k vyřešení konkrétního problému nezbytné.

Na stránkách http://boofcv.org je k nalezení mnoho příkladů použití na řešení konkrétních problémů. Například v balíčku Binary images lze najít nástroje pro řešení problému prahování. V balíčku Features lze najít mnoho postupů, které řeší problémy vyhledávání objektů v obraze, sledování bodů a další.

4.5 Raspistill

Raspistill je aplikace, která slouží k pořízení fotografie z jednodeskového počítače Raspberry Pi. K tomu slouží kamerový modul, který je připojen přímo k jednodeskovému počítači speciálním patnáctipinovým konektorem. Raspistill je velice podobná aplikaci raspistillyuv. K pořízení videa slouží aplikace raspivid.

Aplikaci je možné spouštět z příkazové řádky. Kamera využívá čtyři OpenMAX komponenty, jsou to: camera, preview, encoder a null_sink. Komponenta preview je volitelná. Náhled může být nastaven na celou obrazovku, nebo může být umístěn do libovolné její části, dále může být velikost náhledu nastavena. Rámeček náhled musí být vždy zobrazen, dokonce, i když to není uživatelským nastavením vyžadováno. Rámeček preview je zobrazován vždy, protože v okamžiku jeho zobrazení jsou vypočítány veličiny jako délka expozice a vyvážení bílé barvy.

Za klíčovým slovem raspistill následují parametry, pomocí kterých lze nastavit výsledný vzhled fotografie. Seznam těchto parametrů se nachází v dokumentaci.

5 Hardwarové vybavení pro řízení robota

Robot K4 se skládá ze dvou hlavních zařízení, která ovládají krokové motory, servomotory a další periferie.

Toto rozdělení je zvoleno tak, že jedna skupina ovládá periferie na nejnižší úrovni a druhá skupina posílá těmto zařízením příkazy. Popisu těchto zařízení se věnuji v následujících kapitolách.

5.1 Raspberry Pi

Raspberry je malý jednodeskový počítač. Jeho velikost je srovnatelná s velikostí platební karty, proto není složité ho umístit do robota. Tento počítač byl vyvinut s cílem podpořit školy, tomu také odpovídá jeho nízká pořizovací cena. Pro tento počítač vychází oficiální operační systém Raspbian, který je volně ke stažení.

V našem robotu K4 se nachází jeden takovýto malý počítač, proto se omezím na popis pouze modelu Raspberry Pi B+ 3, který pro svůj chod využívá robot K4.

Tento jednodeskový počítač je osazen čtyřjádrový procesorem ARM Cortex-A53 1,2GHz 64 - bitů. K dispozici je operační paměť 1 GB, která je sdílená s GPU. Dále se na desce nachází integrovaná síťová karta 10/100 Mbit/s Ethernet + WiFi 802.11n a Bluetooth 4.1. Počítač je napájen pomocí micro USB konektoru. Vstupní napětí je 5 V a doporučený proud zdroje jsou 2 A. Pro připojení k monitoru lze využít HDMI vstup.

Dále se na těle počítače nachází 40 pinů. Některé z pinů jsou určeny k napájení, Raspberry poskytuje napětí 3.3 V a 5 V, na osmi pinech nalezneme zem. Dva piny jsou pro UART, další dva pro I2C a šest jich lze využít pro SPI. Ostatní piny jsou GPIO, to znamená, že představují vývody, které lze programovat pomocí software. Z takovýchto pinů lze číst elektrický signál nebo naopak jej z něj vysílat. Je zřejmé, že Raspberry Pi nedisponuje žádnými piny pro práci s analogovým signálem.

Velkou výhodou počítače je možnost spuštění aplikací napsaných v programovacích jazycích Python, Java atd. Díky tomuto můžeme využívat nepřeberné množství knihoven, které jsou určeny například pro analýzu a práci s obrazem. V tomto případě musíme brát v potaz výpočetní výkon Raspberry. S přesunutím vyšší řídicí logiky do Raspberry získáváme více možností, co se týká řízení, výpočtů, komunikace, analýzy prostředí atd.

5.2 Arduino

Ardurduino je levný vývojový set, který je určen zejména pro studenty a nadšence, kteří dlouho dobu hledali alternativu za drahou desku BASIC stamp. Pomocí tohoto setu je velice jednoduché realizovat velké množství projektů, a to zejména v robotice.

Pro tvorbu softwaru je bezplatně poskytováno vývojové prostředí, které podporuje jazyky C a C++. Arduino se skládá z komponent, které lze běžně zakoupit, takže jsme schopni vytvořit vlastní set, nicméně existují spousty výrobců, které vyrábějí takzvané klony, které se mohou lišit různými specifikacemi a cenou.

I v našem případě najde Arduino využití. Pomocí Arduina můžeme obsluhovat periferie robota, které není vhodné napojit přímo na hlavní řídicí počítač. Důvody jsou různé, například použité senzory. Mnohé senzory poskytují měřenou veličinu ve formě analogového signálu, který nedokáže jednodeskový počítač Raspberry Pi zpracovat, proto využijeme Arduina. Další výhodou začlenění setu do robota je dekompozice na více částí, tím je myšleno, že Arduino se může starat o řízení na nejnižší úrovni. Tímto řízením myslíme například počet vykonaných kroků na motoru, zapnutí pomocného obvodu, přečtení hodnoty senzoru atd. Takovéto řešení je pro vývojáře mnohem snazší. Vývojář je schopen testovat nebo ladit jednotlivé části robota zvlášť, a tím pádem odhalit případné chyby nebo nedostatky mnohem pohodlněji a rychleji.

Arduino se skládá z více částí, ty nejzákladnější jsou:

- resetovací tlačítko, slouží pro reset setu,
- USB konektor (konektor může chybět a na jeho místě se nacházet piny pro napojení RS232), pomocí něhož lze do vývojového setu nahrát program a také je možné jím celý set napájet,
- samostatný napájecí konektor, ten se u některých verzí nemusí nacházet, a napájení je řešeno prostřednictvím pinů nebo USB konektoru,
- USB sériový převodník, ten slouží jako prostředním mezi počítačem a hlavním čipem Arduina,
- indikační diody, diody Rx, Tx signalizují přijímání a odesílání dat přes sériovou linku, dioda L je připojena k pinu 13 a signalizuje práci na něm, lze využít samostatně pro blikání, vybavení Arduina diodami se liší, záleží na konkrétním modelu,
- hlavní čip, záleží na tom, jaký model používáme, čipy mohou být integrované na desce nebo výměnné (zasazené do patice),
- digitální piny, ty jsou vyvedené a připravené pro komunikaci s okolními periferiemi, přes piny lze data odesílat i přijímat, piny označené vlnovkou podporují PWM
- analogové piny, slouží k práci s analogovými zařízeními.

Komunikace Arduina s okolními zařízeními

Abychom mohli využít v robotu Arduino naplno, musíme zajistit, aby bylo schopné komunikovat se svým okolím. Uvedu základní možnosti, jak Arduino může tuto komunikaci realizovat.

Jak už je zřejmé ze seznamu částí, Arduino disponuje s*ériovou linkou (RS232).* U některých modelů je provoz na této lince indikován dvěma diodami, a to při přijímání dat nebo při odesílání dat. Zapojení je jednoduché, přijímač Arduina propojíme s vysílačem zařízení, se kterým chceme komunikovat a naopak, při tomto zapojení je důležité, aby zařízení měla společnou zem. Pro práci se sériovou linkou slouží funkce Serial.

Další možností pro komunikaci s okolím je možné využít *I2C* sběrnici. Tato sběrnice umožňuje propojit až 128 zařízení, z čehož každé zařízení má přiřazenou 7 - bitovou adresu. Pro použití I2C sběrnice musíme využít dvou pinů označených SCL (clock line) a SDA (data line). Tyto piny propojíme se zařízením, se kterým potřebujeme komunikovat. U komunikace tohoto typu jsou zařízení rozdělena tak, že vždy jedno zařízení je řídicí (označeno master) a okolní připojená zařízení jsou řízená (označena slave). U tohoto typu komunikace je také pro správnou funkčnost potřeba zajistit společnou zem všech zařízení. Pro práci s touto komunikací je předpřipravena knihovna Wire, která v sobě obsahuje všechny důležité funkce.

Arduino Leonardo a Arduino Micro

Arduino Leonardo vychází svou konstrukcí z modelu Uno. Je osazeno čipem ATmega32u4. Na jeho desce se nachází 20 pinů, z čehož 7 podporuje PWM. Analogových vstupů se na desce nachází celkem 12. Napájet desku můžeme buď 5 V z USB konektoru nebo použít napájecí piny, které akceptují napětí v rozsahu 5-12V. Rozhodně není bezpečné tuto hranici překročit.

Arduino micro je oproti Leonardu podstatně menší, proto je vhodné ho použít v projektech, kde potřebujeme ušetřit místo. Základem desky je čip ATmega328. O napájení a nahrání programu je set osazen mini USB konektorem. Napájení je možné z USB nebo přes napájecí piny, které akceptují napětí od 5–12 V. Při překocení této hranice může dojít k poškození zařízení, v případě její nedosažení není zaručen bezproblémový provoz. Na desce se nachází celkem 20 pinů, ze kterých je 14 digitálních, a z toho 6 podporuje PWM a 6 analogových.

Je třeba zmínit proudové omezení modelů. Některé senzory, motory a jiné periferie pro svůj chod vyžadují vyšší proud, než je Arduino schopno dodat, takovéto případy jsou řešeny pomocí přídavných obvodů s tranzistory nebo relé a externího zdroje s vyšší proudovou kapacitou. Maximální proud I/O je 40 mA. Maximální proud integrovaného zdroje je 200 mA (Voda, 2014).

6 Metodika práce

Tato kapitola popisuje soutěž, pro kterou byl robot zkonstruován, konstrukci robotu, herní strategii a uvažované možnosti vylepšení.

V závěru kapitoly je popsán vybraný způsob, který podléhá kritériím, která byla zjištěna od organizátorů „Robotického dne“.

6.1 Ketchup house

Ketchup house je jednou z disciplín, které je možné se zúčastnit v rámci „Robotického dne “, který je pořádán v Praze. V rámci České republiky je možné se této soutěže zúčastnit jednou ročně.

Podmínkou pro účast v soutěži je dodat minimálně dvě fotografie robota a jeho popis, který bude mít délku minimálně dva textové odstavce. Tyto podmínky je nutné splnit ještě před příjezdem na „Robotický den“.

Soutěž začíná tak, že se roboti postaví proti sobě na hrací plochu. Na tuto plochu jsou umístěny dvě plechovky s rajským protlakem, tyto dvě plechovky mají známou pozici, ostatních pět plechovek je náhodně rozmístěno na hracím poli. Na hracím poli je tedy celkem sedm plechovek, se kterými mohou roboti manipulovat.

Na pokyn rozhodčího odstartují oba dva roboti. Úkolem robotů je nashromáždit co největší množství plechovek, počítají se pouze ty plechovky, které robot umístí na domovskou startovací čáru. Další podmínkou je, že se plechovka musí nacházet mimo konvexní obal robotu. Robot může manipulovat i s plechovkami, které již soupeř umístil na svou domovskou čáru.

Požadavky na robota

Robot musí být zcela autonomní a nesmí ohrožovat sebe ani své okolí, dále robot nesmí být nadmíru obtěžující.

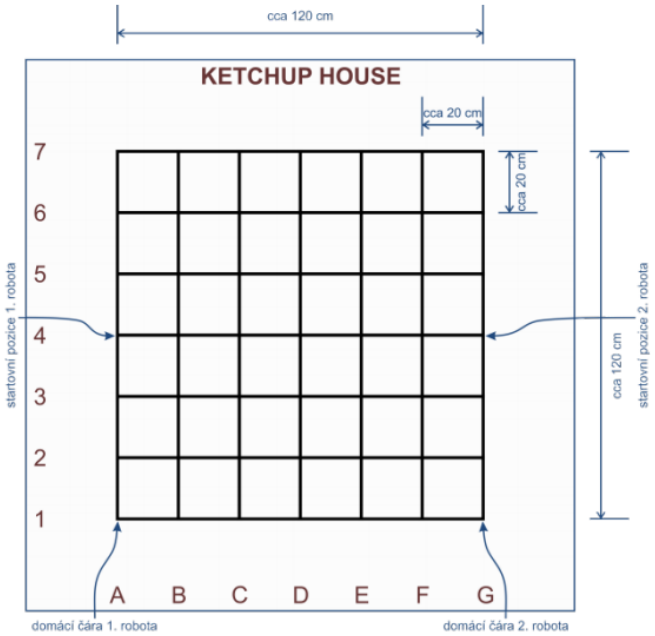
Od odstartování soutěže nesmí být do chodu robota externě zasahováno. Kromě odstartování se na robota nesmí sahat. Externí zásah je možný pouze se svolením rozhodčího. Je povinné, aby byl v horní části robota umístěn vypínač, který je vůči robotovi dostatečně velký a výrazný, aby bylo možné robota v případě potřeby nouzově vypnout. Dále musí mít robot v horní části volnou plochu o rozměrech 10 x 7 cm. Na toto místo bude umístěna nálepka, která slouží k označení robota. Maximální šířka robota je 30 cm, maximální délka je také 30 cm, maximální výška není specifikována.

Hrací pole

Hrací pole tvoří čtvercové pole. Toto pole je rozděleno na 6 x 6 čtverců. Každý z těchto 36 čtverců má rozměry 20 x 20 cm. Dělicí čáry jsou cca 1,5 cm široké. Okolo hracího hřiště se nachází bílá plocha, která má z každé strany šířku 30 cm. Svislé čáry jsou označeny čísly 1 až 7, vodorovné jsou označené písmeny A až G.

Před zápasem je do hracího pole umístěno 7 plechovek s rajčatovým protlakem. Známé plechovky se nachází na pozicích D3 a D5. Pokud jeden z robotů sebere

některou z náhodně umístěných plechovek a vzdálí se s ní o více než jedno pole, a soupeř je vzdálen od místa, kde se plechovka nacházela, také na více než jedno pole, je na místo umístěna nová plechovka. Maximální počet plechovek ve hře je 12. Oba roboti startují z pozice A4. Během hry povoleno oběma robotům se libovolně pohybovat po celé hrací ploše. Hrací pole je vidět na obrázku 4.



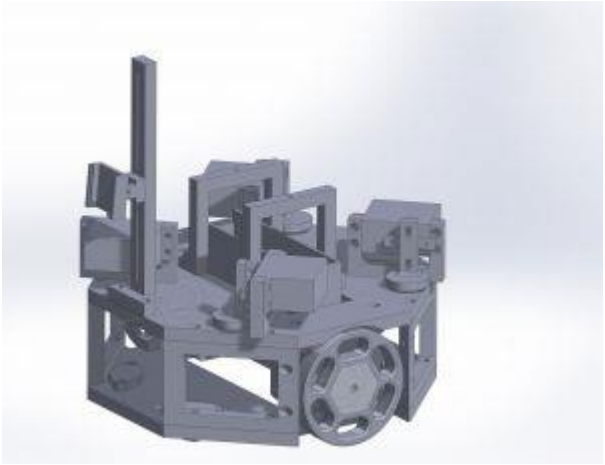
Obr. 4 Hrací pole
Zdroj: http://roboticday.org/2017/rules/2017-Ketchup_House-CZv1.pdf

Povinnosti hráčů

Účastníci hry Ketchup House jsou zodpovědní za veškeré škody způsobené jimi samými, jejich roboty nebo vybavením, kterým robot disponuje.

6.2 Mobilní robot K4

Jedná se indoorového robota, navrženého pro soutěž Ketchup House. Tento robot byl od začátku navržen tak, aby všechny jeho součásti bylo možné vytisknout na průmyslové 3D tiskárně. Jeho návrh je vidět na obrázku 5 (AiStorm, 2017).

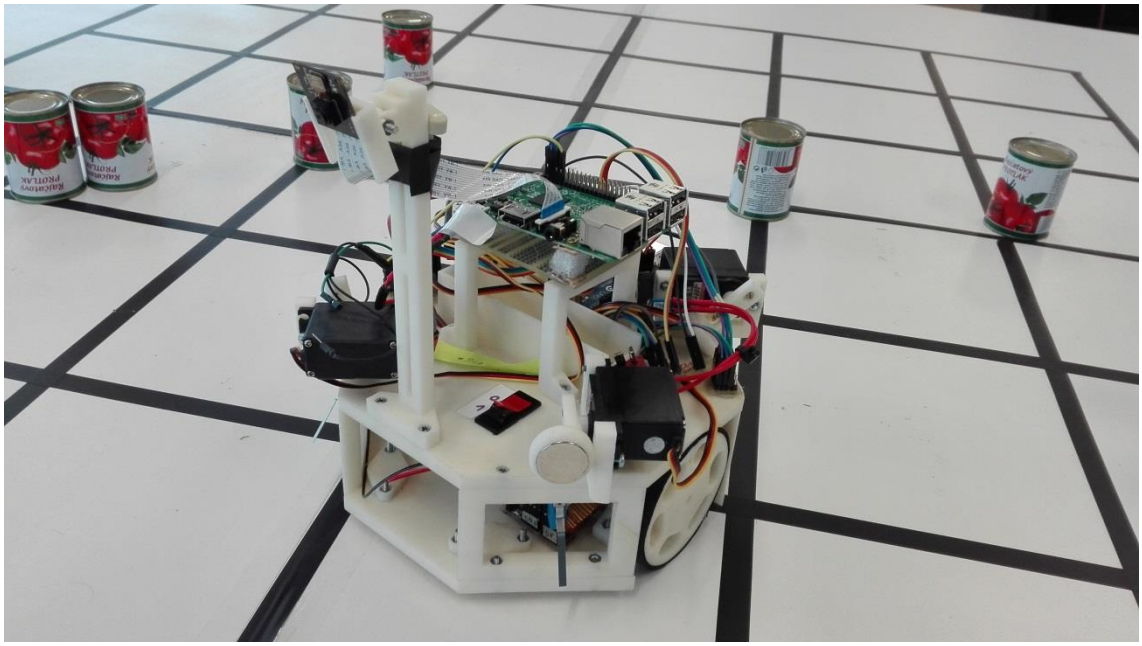


Obr. 5 3D návrh mobilního robota K4
Zdroj: https://aistorm.mendelu.cz/cz/projekty/k4

Robota tvoří dvě hlavní desky z ABS materiálu. Na spodní desce jsou v hlavní ose umístěna kola s krokovými motory a jejich řídící elektronikou, kterou tvoří dva drivery DRV8825 a Arduino Nano.

Na druhé desce jsou umístěny čtyři servomotory, na kterých jsou umístěny ramena s magnety, tento mechanizmus je uzpůsoben pro sběr plechovek s rajčatovým protlakem. Pod každým servomotorem je umístěn tlačítkový senzor, který indikuje přítomnost plechovky. Tento mechanizmus je ovládán Arduinem Uno.

V prostředku mezi servomotory je umístěn stojan, na kterém se nachází jednodeskový počítač Raspberry Pi B+. Tento počítač se stará o řízení celého robota a pomocí I2C sběrnice komunikuje s oběma Arduiny. Na tento počítač je pomocí patnácti pinového konektoru připojena kamera. Pod stojanem se nachází baterie. Reálný vzhled robota lze robota vidět na obrázku 6.



Obr. 6 Mobilní robot K4

6.3 Výběr mechanizmu pro skenování prostředí

Nejběžnější způsoby, které roboty na soutěží Ketchup House používají, je hledání předmětů pomocí ultrazvukových nebo infračervených senzorů. Při volbě ultrazvukového senzoru je možné jej namířit na místo, kde si myslíme, že by mohl být umístěn předmět a vyslat pulz, podle kterého se dozvíme, jestli se na tomto místě nějaký předmět nachází.

Úskalí této metody se nachází v tom, že robot pouze ví, že na předpokládaném místě se něco nachází, bohužel nejsme schopni zjistit tvar ani velikost. Jediná informace, kterou jsme schopni získat, je vzdálenost robota od předmětu. Obdobné informace jsme schopni získat použitím infračerveného senzoru.

Poslední nejjednodušší možností je přiblížit se k náhodnému místu a podle sepnutí mechanického senzoru (tlačítka) zjistit, jestli se před robotem nachází předmět.

Po prozkoumání všech možností byla vybrána možnost mapovat prostředí pomocí laserového paprsku. Dráha tohoto paprsku bude zachycena a zpracována jednodeskovým počítačem s kamerovým modulem, který poskytne výstup v podobě obrázku ve formátu JPG nebo jiné. Tento obrázek bude zanalyzován tak, že budeme schopni zjistit, kde se objekt nachází, dále jeho velikost, a vyhodnotit, jestli se jedná o hledaný předmět.

6.4 Použití laseru v soutěži Ketchup House

Soutěž má jasně daná pravidla, která zakazují osazení robota jakýmikoliv nebezpečnými zařízeními.

Laserové moduly mohou díky jejich výkonu snadno spadnout do této kategorie nebezpečných zařízení. Proto po konzultaci s organizátory soutěže mi bylo doporučeno nepřekračovat výkon 5mW při vlnové délce 532nm. Dále mi bylo doporučeno nesvítit na jedno místo delší dobu pro eliminaci rizika zapálení předmětu a poškození zraku. A posledním opatřením pro větší bezpečnost je nesvítit na skenovaný předmět přímo, ale odrazem od předmětu, který dostatečně odráží světlo, například běžné zrcadlo nebo zrcadlový hranol, který se požívá v laserových tiskárnách.

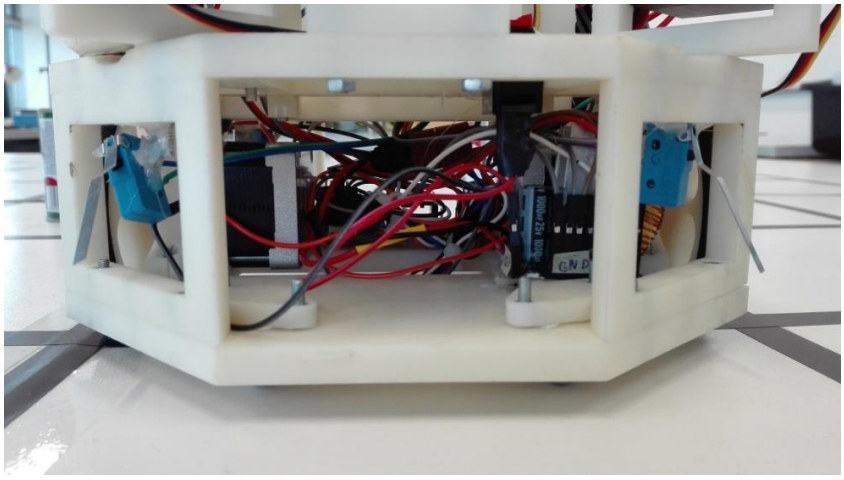
7 Laserové zařízení a skenování prostředí

V době tvorby této práce jsem nenarazil na žádný senzor (zařízení), který by splňoval všechny kritéria, která jsem obdržel od organizátorů soutěže. Proto jsem se rozhodl pro vlastní konstrukci.

V následujících kapitolách je tato vlastní konstrukce popsána, je popsán princip funkce laserového zařízení a uveden způsob jeho ovládání.

7.1 Popis zařízení

Návrh vychází z rozměrů robota a pravidel soutěže Ketchup House. Zařízení se musí vejít do spodní části robota, které je vidět na obrázku 7. Tento prostor nám umožňuje vložit senzor o maximální velkosti cca: délce 65 mm, hloubce 40 mm a výšce 35 mm. Software pro ovládání zařízení bude v případě nutnosti nahrán do Arduina Nano umístěného ve spodní části robota. Toto Arduino bude obsluhovat jak krokové motory, tak laserové zařízení.



Obr. 7 Pohled přední část robota

Dále je pro zvýšení bezpečnosti vhodné, aby na cíl nebylo svíceno přímo, ale odrazem. Tato podmínka eliminuje možnost umístit do spodní části robota laser s čárovým filtrem.

Další možnost, jak vytvořit čáru, je přejet po skenovaném předmětu vícekrát laserovým paprskem. Nastavíme-li na kamerovém modulu dostatečně dlouho dobu expozice, aby ve výsledném obrázku se promítla celá dráha laserového paprsku, získáme potřebný výstup i bez požití čárového filtru. Navíc tímto řešením vyhovíme hned dvěma omezením soutěže. Bohužel jsme limitování dobou expozice. Když bude nastavena na příliš dlouhou dobu, fotografie bude přesvětlena a nebude z ní nic čitelné.

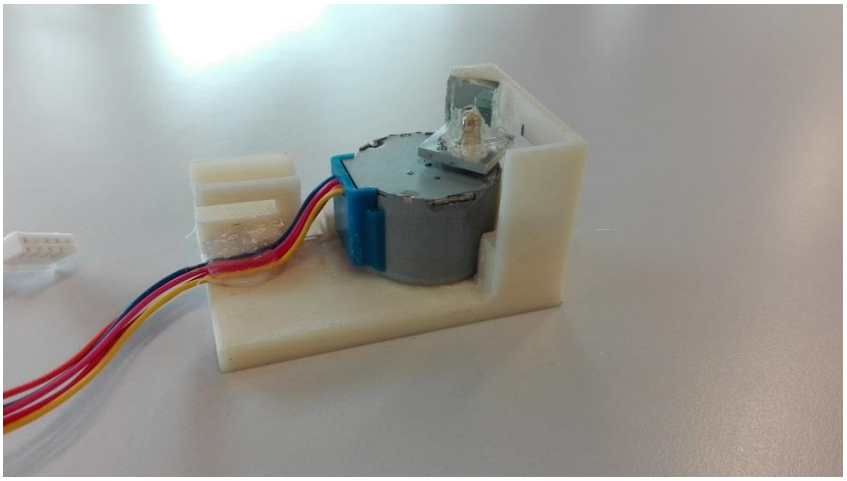
Prvním omezením je použití zeleného laseru, jehož výkon není větší 5mW při 532nM. Takovýto laser s certifikovanými parametry s takto malým výkonem je

velmi těžko k dostání. S použitím daného řešení si vystačíme s laserovým modulem produkujícím pouze tečku. Další doporučení je nesvít příliš dlouho na jedno místo, přejížděním laserovým paprskem po skenovaném předmětu nikdy paprsek nespočine na jednom místě příliš dlouho dobu. Poslední problém, který musíme v konstrukci vyřešit, je nesvítit na předmět přímo, ale odrazem.

7.2 Možné realizace návrhu zařízení

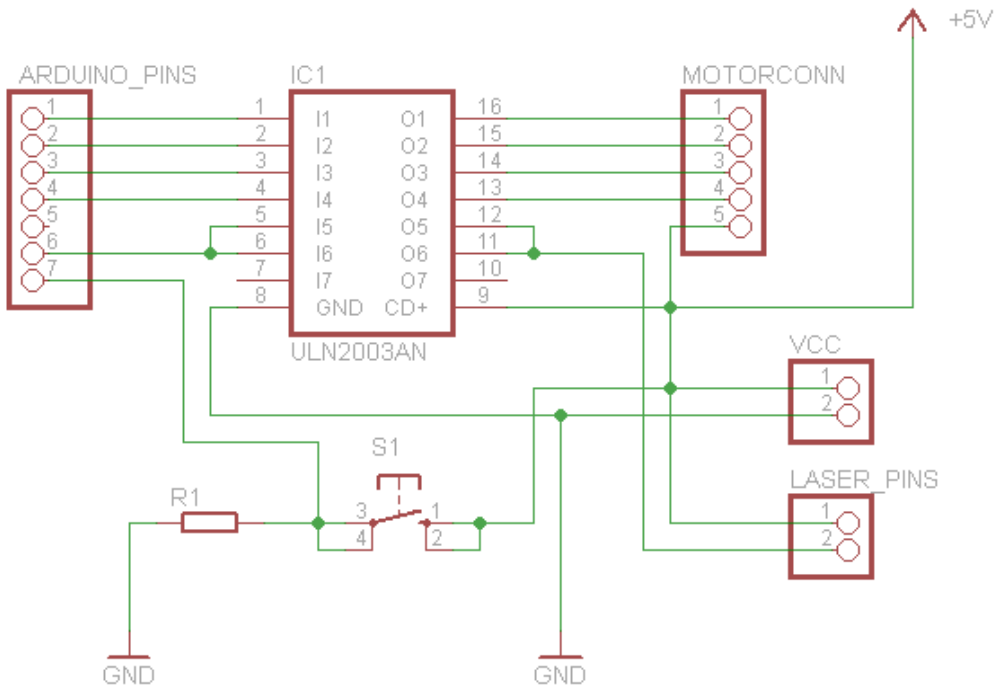
Prvním krokem bylo vybrat součástky pro zařízení. První model zařízení se skládal z lineárního motoru, zrcadlového hranolu z laserové tiskárny, laserového modulu a řídícího obvodu. Obvod obsahoval 2x tranzistor TIP120, první pro zapnutí laserového modulu a druhý pro regulaci otáček motoru. Problém tohoto návrhu byl ve vysoké rychlosti motorku, kvůli které docházelo k vyosení zrcadlového hranolu, a tak se místo jedné silné čáry na plechovce objevily čtyři tenké čáry.

Druhý návrh vypadal velmi podobně, ale lineární motor byl vyměněn za krokový, jak je vidět na obrázku 8.



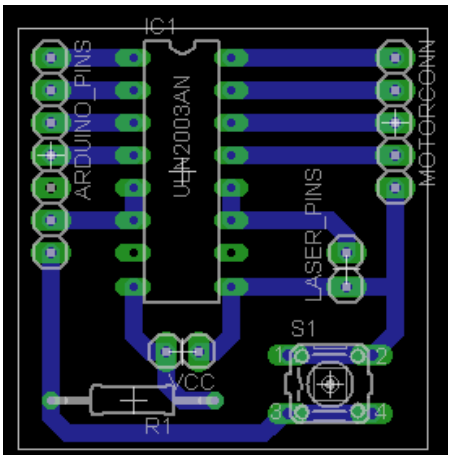
Obr. 8 Laserové zařízení s krokovým motorem

Toto zařízení již bylo schopné vykreslit na objektu jednu souvislou čáru. Pro řízení toho zařízení byl navrhnut obvod, který se skládal z Darlingtonova pole, pinů pro připojení Arduina, pinů pro připojení motoru a tlačítka pro libovolnou funkci, například zapnutí nebo vypnutí laseru. Jeho schéma je vidět na obrázku 9. (Bezděk, 2002)



Obr. 9 Schéma obvodu pro řízení krokového motoru a laserového modulu

Ke schématu byl navrhnut plošný spoj obrázek 10 (Juránek, Hrabovský, 2005), který měl být umístěn na spodní desku robota společně s motory a jejich řídicím obvodem.



Obr. 10 Návrh DPS pro řízení krokového motoru a laserového modulu

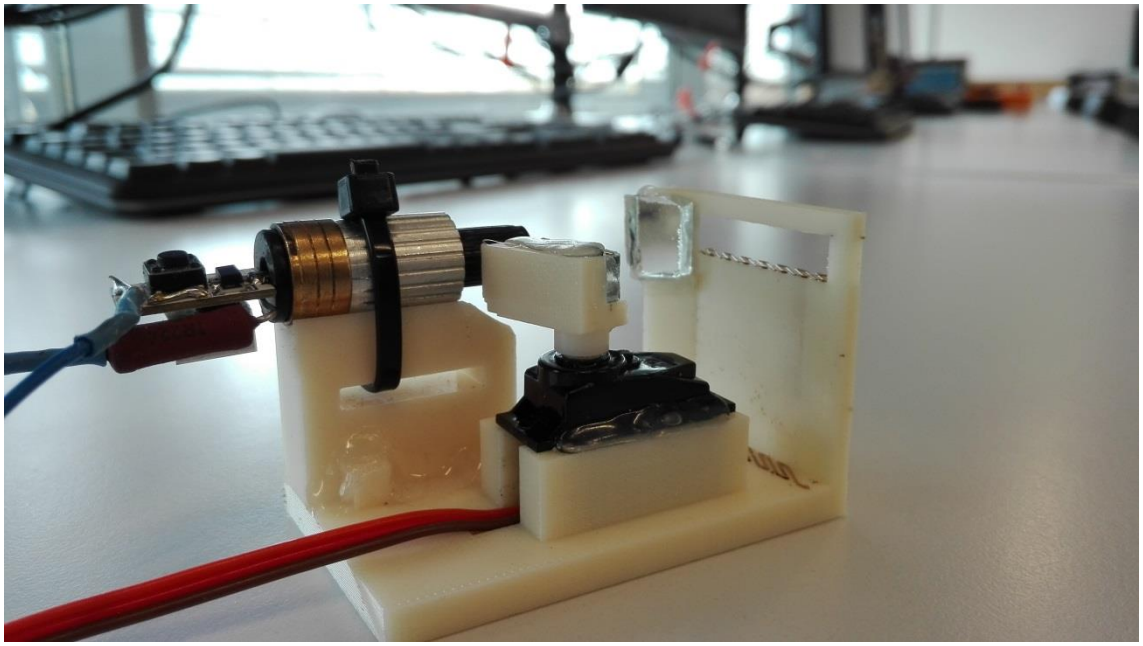
Nicméně slabinou tohoto řešení byla rychlost krokového motorku, která nebyla dostačující. Při pořízení fotografie linka vykreslená na objektu nebyla příliš viditelná.

Proto jako poslední řešení bylo zvoleno krokový motor vyměnit za servomotor a předělat celý řídicí obvod. Servomotor disponuje vyšší rychlostí a jeho ovládání za pomocí Arduina je jednodušší.

7.3 Realizace zařízení

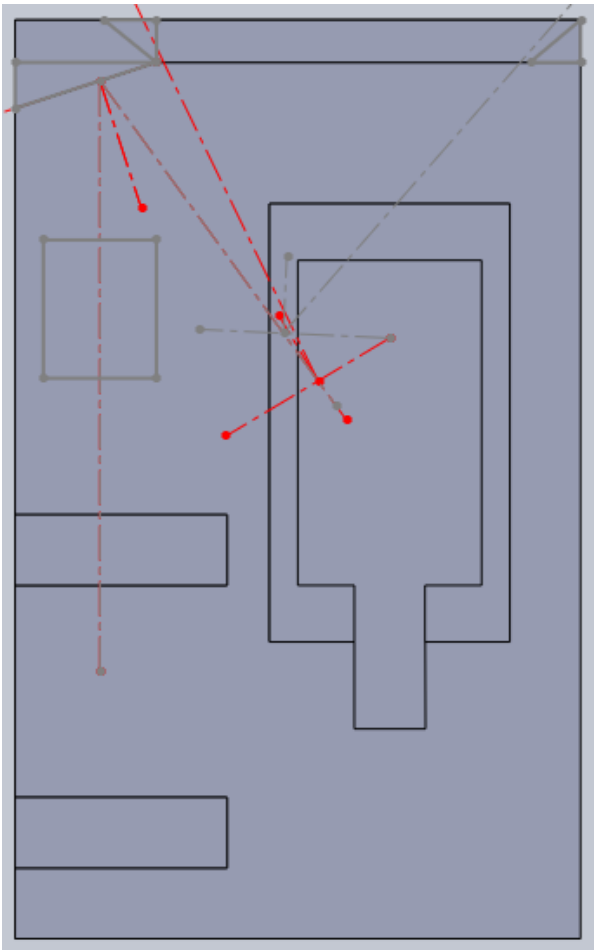
Konečná verze zařízení se skládá z pomocného obvodu, který popíšu později v této kapitole, servomotoru, dvou odrazových ploch a základny, na které jsou umístěny všechny komponenty.

První odrazová plocha je pevně umístěna na základně a odráží laserový paprsek v konstantním úhlu. Druhá odrazová plocha je umístěna na otočné části servomotoru, změnou polohy servomotoru je možné pozici paprsku změnit. Zařízení je na obrázku 11.



Obr. 11 Laserové zařízení se servomotorem

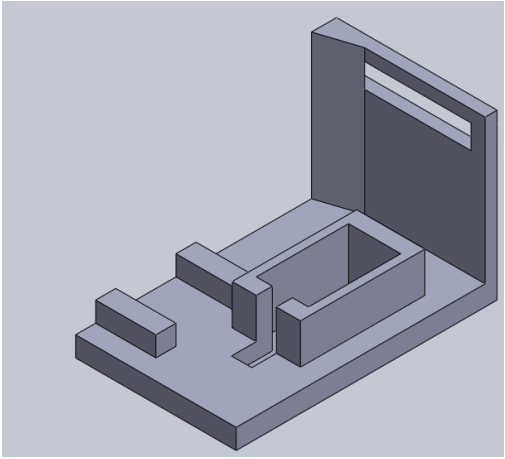
Na obrázku 12 je vyobrazen pohled z vrchu. Na tomto obrázku můžeme vidět, jak se laserové paprsky (znázorněny čerchovanou čárou) odráží v závislosti na natočení variabilní odrazové plochy.



Obr. 12 Skica znázorňující odraz laserového paprsku

Laserový modul je umístěn v levé části. Pokud se odrazová plocha nachází v nejspodnější poloze (červená čerchovaná čára), a tedy laserový paprsek směřuje co nejvíce k levé straně, pak svírá s kolmicí na odrazovou plochu úhel cca 5°. Pokud se odrazová plocha nachází v nejhornější poloze (šedá čerchovaná čára), laserový paprsek s kolmicí svírá úhel cca 39°. Úhel svírající paprsky v obou nejkrajnějších polohách je úhel cca 67°. Jedná se o měření provedená v programu SolidWorks. Při zkonstruování zařízení nebylo možné umístit všechny komponenty na milimetr přesně, jak je v návrhu, proto se mohou některé hodnoty úhlů lišit, ne však natolik, aby významně ovlivnily funkčnost a efektivnost zařízení.

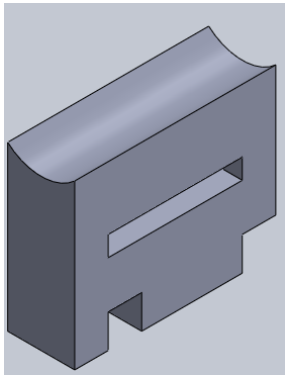
Celé zařízení bylo navrhnuto v programu SolidWorks (Pagáč, 2017) a vytisknuto na 3D tiskárně. Konstrukce zařízení se skládá ze tří částí. První částí je základna, viz obrázek 13, která tvoří základ, na kterém jsou umístěny všechny ostatní komponenty.



Obr. 13 Základna laserové zařízení

V levé části se nachází dvě kolejnice, na kterých je umístěn držák pro laserový modul. V pravé části se nachází prostor pro servomotor, tento prostor je ohraničen pro snadnější umístění, uchycení a vystředění servomotoru. V přední části základny se nachází stínítko s průzorem pro laserový paprsek. Stínítko se zde nachází proto, aby dalo možnost průchodu pouze laserovému paprsku a zachytilo případné nežádoucí odrazy. Levá část stínítka je upravena tak, aby odrážela paprsek v konstantním úhlu 18°.

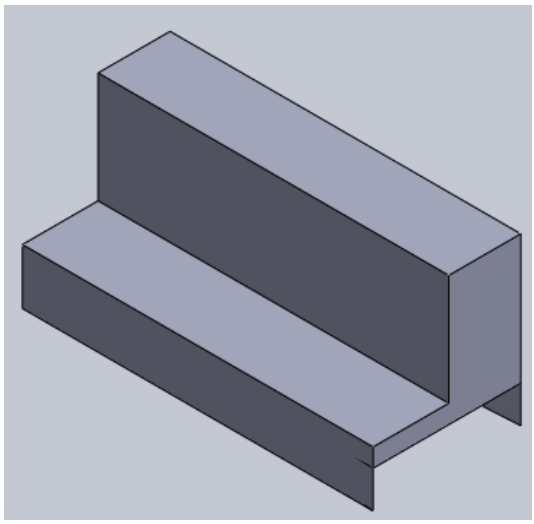
Druhá část je již zmíněný uchycení laserové modulu obrázek 14.



Obr. 14 Uchycení laserového modulu

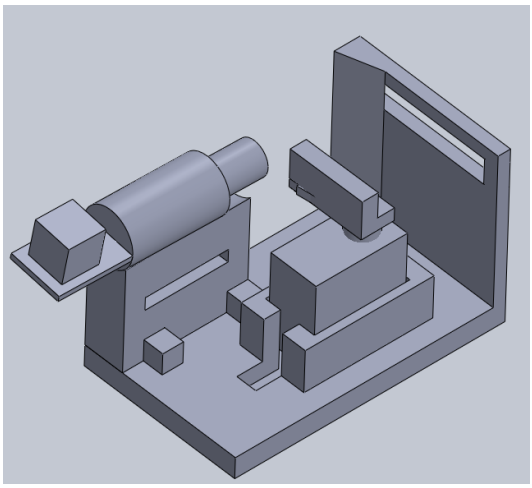
Horní část je zaoblena tak, aby do ní laserový modul co nejlépe zapadl, obdélníkový otvor uprostřed slouží k uchycení modulu za pomocí stahovací pásky. Spodní část je vytvarována tak, aby zapadla do kolejnic umístěných na základně. Horizontálním posunem uchycení po kolejnicích lze ovlivnit místo dopadu laserové paprsku.

Posledním a zároveň nejmenším dílem je nosič odrazové plochy obrázek 15. Ten společně se zrcadlem tvoří odrazku. Nosič je umístěn na servomotoru a jeho spodní část je vytvarována tak, aby nosič bylo možné umístit do packy dodávané se servomotorem.



Obr. 15 Nosič odrazové plochy

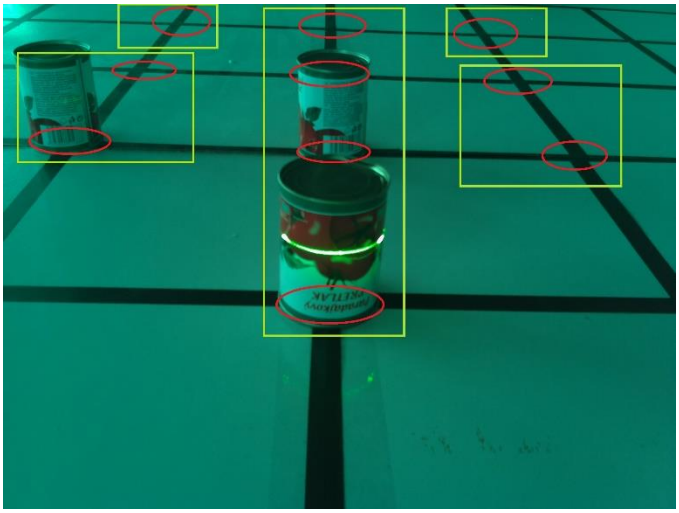
Na obrázku 16 lze vidět, jak jednotlivé komponenty do sebe zapadají, pro lepší názornost byl vymodelován do sestavy přidán laserový modul i servomotor.



Obr. 16 Model laserového zařízení

7.4 Ovládání laserového zařízení

Laserové zařízení je ovládáno pomocí Arduina Uno, které také ovládá motory. Do původního programu byly doplněny metody pro skenování jednotlivých částí hracího pole, jak je vidět na obrázku 17. Hrací pole obsahuje devět bodů, které je možné laserovým zařízením naskenovat. Pro tyto body je implementováno pět metod.



Obr. 17 Skenované pozici na hracím poli

Červenými elipsami jsou na obrázku zvýrazněny ty pozice, na kterých se robot snaží najít předmět. Žluté obdélníky zobrazují skupiny pozic, které je možné naskenovat jednou z pěti metod. Poslední dvě pozice jsou samostatné, a to kvůli úspoře času a pořízením lepšího snímku. Této problematice se věnuji v následujících kapitolách.

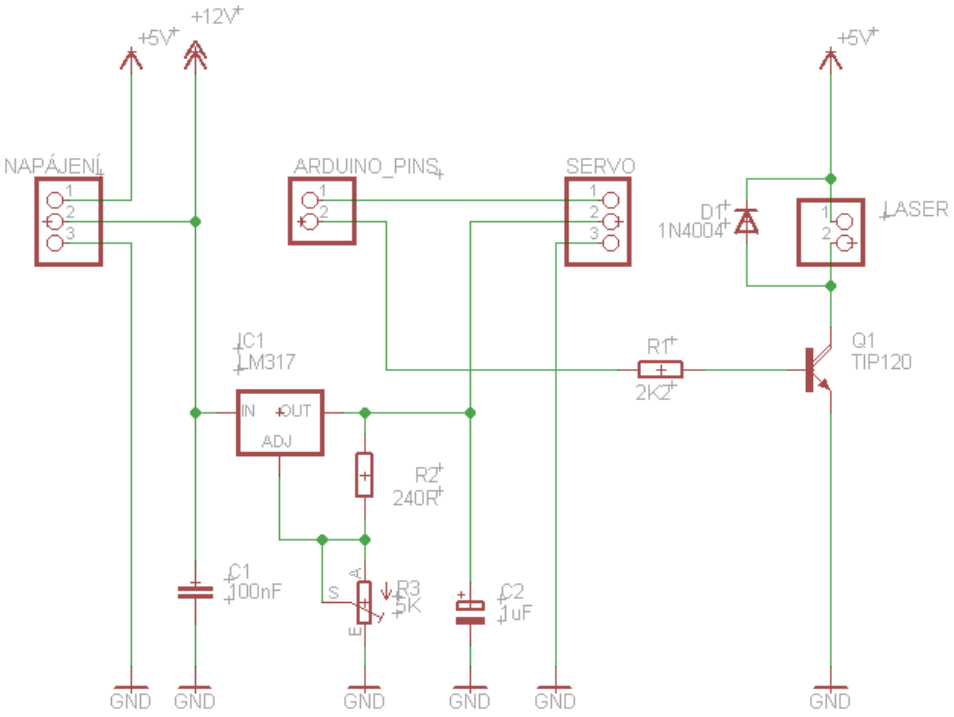
Ke komunikaci mezi zařízeními je využita I2C sběrnice. Pro použití těchto metod byl rozšířen komunikační protokol. (Liang, 2013) Jedná se o přidání nové hodnoty prvního odesílaného bitu, ta programu říká, že další bity budou určeny pro spouštění laserového zařízení. Druhý bit spustí jednu z pěti metod pro skenování zařízení. Jedná se o metody pro naskenování levé strany, prostředku hracího pole, pravé strany levé strany ve čtvrté řadě a levé strany ve čtvrté řadě.

Pro zařízení byl navrhnut vlastní elektronický obvod, ten slouží k jeho řízení a zaručuje jeho správnou funkci. Seznam použitých součástek je uveden v tabulce 2.

Tab. 2 Seznam součástek pro obvod laserového zařízení

Název součástky	Hodnota / Označení
Kolíková lišta	DS1021
Rezistor	2K2, 240R
NPN tranzistor	TIP120
Dioda	1N4004
Napěťový stabilizátor	LM317
Trimr 5 K	PT10MVK010
Keramický kondenzátor	100nF
Elektrolytický kondenzátor	1uF

Zařízení je sice ovládáno pomocí Arduina, ale pro zaručení správného chodu je nutné mezi Arduino a zařízení vložit mezičlánek, který představuje obvod na obrázku 18 (Bezděk, 2002).



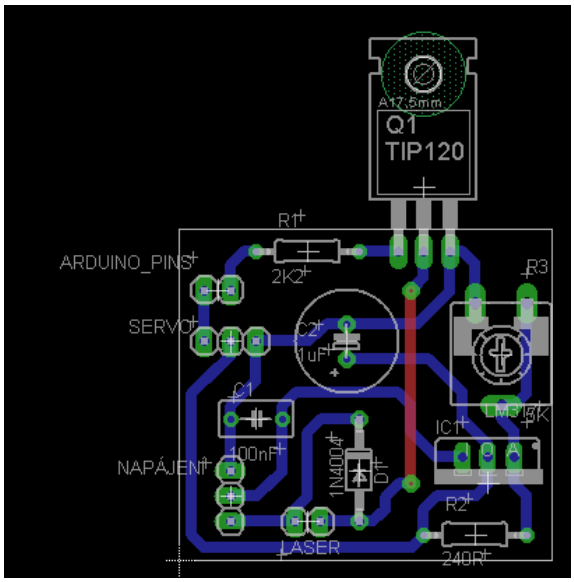
Obr. 18 Schéma zapojení elektronického obvodu pro řízení servomotoru

Hlavní důvody pro sestrojení obvodu jsou dva komponenty, a to laserový modul a servomotor.

Laserový modul je napájen 3 V a potřebný proud dosahuje hranice 300 mA, takovýto proud nejsme schopni dodat z GPIO ani ze stabilizovaného zdroje integrovaného na desce Arduino. Taktéž nemůže být laserový modul zapnut po celou dobu ze dvou důvodů. Prvním důvodem je již zmiňovaná bezpečnost a druhým důvodem je konstrukce modulu, která není na dlouhodobý provoz uzpůsobena. Proto obvod obsahuje tranzistor TIP120. Mezi kolektor tranzistoru a 5 V ze stabilizovaného zdroje K4 jsou zapojeny dva piny určené pro laser (plus a mínus). Báze tranzistoru je připojena přes rezistor k digitálnímu portu Arduina. Přivedením napětí na bázi je sepnut přechod kolektor emitor a laserovým modulem začne protékat dostatečně velký proud. Z důvodu napojení báze na digitální port je nutné před laserový modul umístit rezistor 4R7 s dostatečně velkým výkonem, tím zaručíme, že na modulu se neobjeví napětí 5 V, ale potřebné 3 V.

Druhá část obvodu je navržena pro servomotor. Servomotor naopak pro svou maximální rychlost a maximální točivý moment potřebuje napětí 6,2 V. Proto se na obvodu nachází napěťový stabilizátor LM317. Tento stabilizátor je na základě velikosti rezistoru připojeného mezi zem a piny out a adj schopen regulovat napětí v rozmezí 1,25 – 37 V. Tento rezistor byl nahrazen odporovým trimerem pro plynulou regulaci napětí.

Na obrázku 19 je vidět navržená deska plošného spoje a zároveň osazovací plán (Juránek, Hrabovský, 2005).

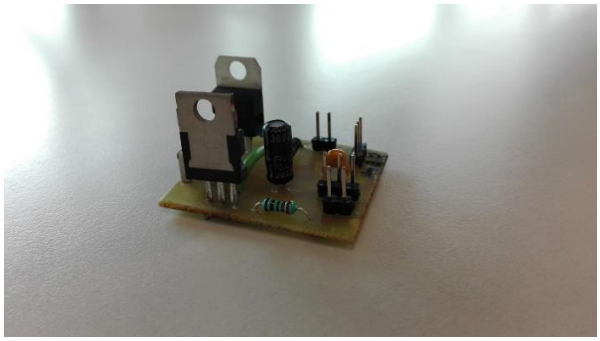


Obr. 19 Osazovací plán laserového zařízení se servomotorem

V levé části se nachází tři skupiny pinů. První vodorovná dvojice slouží k ovládání zařízení skrze Arduino. První pin z této dvojice slouží k ovládání servomotoru a druhý k ovládání tranzistoru, tedy zapnutí nebo vypnutí laserové modulu.
Druhá trojice pinů slouží k připojení servomotoru. První pin je signál, druhý pin je napájecí, velikost napětí na tomto pinu je určeno hodnotou trimeru a třetí pin je zem.

Třetí svislá trojice slouží k napájení desky ze stabilizovaného zdroje robota K4. První pin je společná zem, na druhý pin je přivedeno 12 V a na třetí pin 5 V.

Ve spodní části zhruba uprostřed se nachází poslední dvojice pinů. Tato dvojice je určena pro připojení laserové modulu. Na levý pin je připojeno plus z modulu a na pravý mínus. Konečný vzhled obvodu je na obrázku 20.



Obr. 20 Obvod pro ovládání laserového zařízení se servomotorem (skutečný vzhled)

7.5 Odhalení slepých míst

Obrázek 21 demonstruje příklad slepých míst na snímku, která není možné naskenovat laserovým zařízením.

První takové slepé místo se nachází přímo před robotem, tedy horizontální souřadnice robota plus jedna. Toto místo je dáno natočením kamery.

Další slepá místa vznikají, pokud jsou umístěny dvě plechovky za sebou, robot osvítí první plechovku, zaznamená její polohu, druhá plechovka, která se však nachází za ní, už není vidět, protože ji zastínila první.

Nyní uvedu příklad, který demonstruje problém slepých míst. Budu vycházet z obrázku 21.

Pro upřesnění si snímek rozdělíme na souřadnice ve tvaru (x; y). Souřadnice x představuje vertikální osy a souřadnice y horizontální osy. Souřadnice x jsou číslovány od první viditelné černé horizontální osy ze spodní strany snímku (x = 1) a souřadnice y od druhé viditelné vertikální osy z pravé strany snímku (y = 1).

Obr. 21 Příklad zastínění jedné plechovky druhou

Robot provede skenování prostředí, odhalí plechovku v levé části, která se nachází na souřadnicích (2; 1), druhý sken odhalí plechovku, která se nachází na souřadnici (1; 2). Zde vzniká problém, tato plechovka zakrývá plechovku na souřadnici (2; 2) a posledním skenem odhalí plechovku na souřadnici (2; 3).

Poté je naplánována trajektorie. Jako první je nabrána plechovka na souřadnici (1; 2), poté robot pokračuje k souřadnici (2; 1) přes souřadnici (2; 2). Při této cestě by robot narazil do plechovky na souřadnici (2; 2) nenabral by ji, zbytečně přišel o potencionální bod, a také by při otáčení o ni mohl zavadit a ztratit již sebrané plechovky.

Jako řešení jsem zvolil umístění proximitního senzoru QRD 1114 do přední části robota. Při každém kroku vpřed je změřena vzdálenost mezi hracím polem a senzorem, pokud se pod senzorem nic nenachází (hodnota větší jak 600), robot pokračuje vřed, pokud je vzdálenost menší než prahová hodnota (menší jak 70), robot vyhodnotí situaci jako plechovku, která se nachází těsně před ním, nabere ji a pokračuje v naplánované trajektorii. (Warren, 2011)

8 Konfigurace laserového zařízení a kamerového modulu pro detekci konkrétního objektu

K samotné detekci je nutné sladit laserové zařízení vlastní výroby s kamerovým modulem umístěným na Raspberry Pi.

Pro pořízení ideálního snímku bylo nutné pořídit testovací snímky, podle kterých bylo nalezeno ideální řešení. Postup je popsán v následujících kapitolách.

8.1 Princip identifikace objektů

Pro identifikaci objektu je použit laserový paprsek vyřazovaný laserovým zamířením popsaným v kapitole 6.

Laserovým zařízením nejsme schopni vykreslit souvislou čáru, ale jsem schopni přejíždět laserovým paprskem po skenovaném zleva doprava a naopak. Z toho vyplývá, že je třeba vyřešit problém sledování dráhy laserového paprsku.

S běžným nastavením fotoaparátu je tento problém jen velice obtížně řešitelný. Naštěstí aplikace Raspistill poskytuje uživateli široké možnosti nastavení. Pro pořízení fotografie, na které bude co nejlépe viditelná celá dráha laserového paprsku, je třeba prodloužit dobu expozice, tedy dobu osvitu čipu kamerového modulu. Pro nastavení doby expozice Raspistill poskytuje parametr shutter speed.

Takto osvícený předmět bude na sobě obsahovat vysvícenou souvislou linku jako na obrázku 22 (vysvícenou paprskem, který obsahuje pouze jednu vlnovou délku), tato linka se stane později klíčovým prvkem při použití metod analýzy obrazu.

Obr. 22 Předmět osvícený laserovým zařízením

8.2 Nastavení kamerového modulu

S prodloužením doby expozice roste množství světla dopadajícího na čip kamerového modulu. Tento problém je velice znatelný při denním světle. Takto pořízený snímek je velice přesvětlený a stává se těžko čitelný jak pro lidské oko, tak i pro strojové zpracování.

Takovýto problém lze vyřešit použitím vhodného filtru, při prvním testováním byl zvolen neutrální filtr obrázek 23.

Obr. 23 Neutrální filtr ND 16

Jedná se o filtr s označením ND 16. Toto označení znamená, že filtr propouští 6.25% světla dopadajícího na filtr. Z toho plyne, že filtr omezuje všechny vlnové délky, což nám sice poskytne mnohem lepší výsledek než při pořízení snímku bez filtru, ale zároveň také utlumí vykreslenou dráhu laseru, což působí problémy při pozdějším zpracování obrazu. Snímek pořízený s filtrem ND 16 je vidět na obrázku 24.

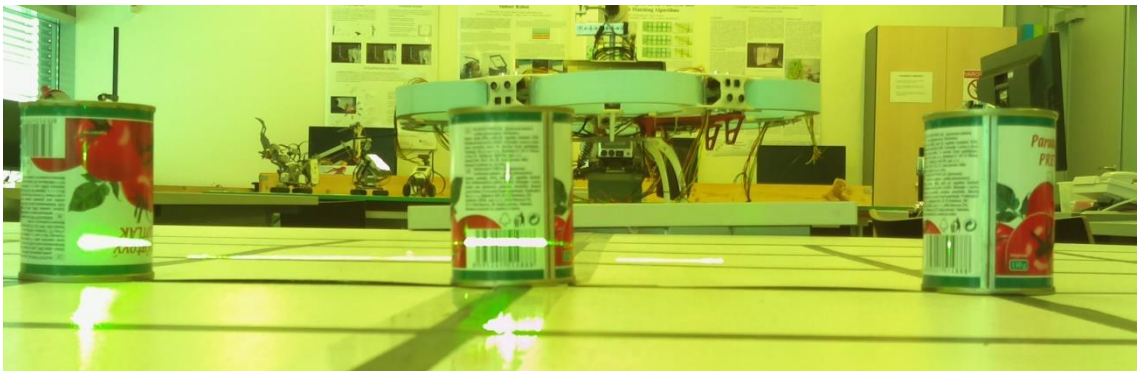
Obr. 24 Snímek objektů s použitím filtru ND16 a dloubou expozice 700000uS

Obrázek 23 je složen ze dvou snímků. Na první polovině snímku je vidět, jak je dráha laseru špatně čitelná na červeném podkladu. Na druhé polovině obrázku 23 je snímána dráha paprsku na bílém podkladu. Z fotografie je zřejmé, že bílý podklad je mnohem vhodnější.

Při dalším pokusu byl použit zelený filtr obrázek 25. Tento filtr má stejné rozměry jako neutrální filtr.

Obr. 25 Zelený filtr

S použitím tohoto filtru bylo dosaženo mnohem lepších výsledků než s použitím zeleného nebo neutrálního filtru. Výsledek můžeme vidět na obrázku č 26. Na prostředním objektu (plechovce) lze vidět velmi dobře zachycená dráha laserového paprsku. Na levém objektu je stopa také znatelná. Výsledek je mnohem lepší než při použití neutrálního filtru.



Obr. 26 Objekty osvícené laserovým paprskem s použitím zeleného filtru

Nicméně, jak je na snímku vidět, doba expozice pohybující se kolem jedné vteřiny je velmi krátká na osvícení všech objektů laserovým paprskem. Při testování nebylo výjimkou, že se nepodařilo zachytit dráhu paprsku na žádném předmětu. Naskytuje se řešení, a to dobu expozice prodloužit, a tím získat nějaký čas navíc na přejetí objektů paprskem. Nicméně s použitím tohoto řešení se snímek stává přesvětleným a nepoužitelným pro další zpracování. Další možností je zvýšení rychlosti pohybu paprsku. Zvýšení rychlosti paprsku bohužel není možné, jsme limitováni technickými parametry servomotoru, které jsou již nastaveny na maximum s ohledem na životnost.

Po mnoha pokusech a měřeních byl zjištěn nejlepší možný způsob pro identifikaci objektů. Tento způsob spočívá ve skenování hledaných objektů postupně. Postupné skenování, respektive osvit, si můžeme dovolit, protože se robot pohybuje ve známém prostředí, tudíž jsou známy potencionální pozice hledaných objektů. Příklad postupného skenování můžeme vidět na obrázku 27.

Zde laserový parsek přejíždí pouze po objektu na levé straně, samozřejmostí je, že dráha paprsku nekončí s hranou objektu, ale pokračuje dále na pravou i levou stranu. Je nutné počítat s tolerancí umístění předmětu plus, mínus 2,5 cm na pravou i levou stranu. Mezi maximálním vychýlením paprsku na pravou a levou stranu je svírán úhel cca 11°.

Obr. 27 Hledání objektu v levé části hracího pole

Na závěr kapitoly je přiložen obrázek 28, na kterém je vidět porovnání snímku ze všech testovaných variant. Všechny snímky byly pořízeny za stejných světelných podmínek, ISO parametrem nastaveným na 100 a dobou expozice nastavenou na 700000uS. V levé části se nachází snímek bez použití jakéhokoliv filtru, uprostřed byl použit zelený filtr a na pravé straně snímku je použit neutrální filtr ND 16.

Obr. 28 Trojce snímků. Bez použití filtru, zelený filtr, neutrální filtr ND16.

9 Návrh a implementace algoritmu pro rozeznávání konkrétních objektů

Pro analýzu snímku, s cílem nalezení konkrétního objetu, bylo nutno vyvinout algoritmus, který bude řešit tuto konkrétní problematiku.

Návrh a princip algoritmu je popsán v následující kapitole, kde jsou řešeny problémy spojené se segmentací obrazu a její využití k řešení konkrétní problematiky.

9.1 Návrh algoritmu

Jak již bylo zmíněno v kapitole 6, stěžejní bod pro rozeznávání objektů je zachycená dráha laserového paprsku, která je dostatečně čitelná pro identifikaci a měla by být v ideálním případě nejjasnější skupinou pixelů na snímku.

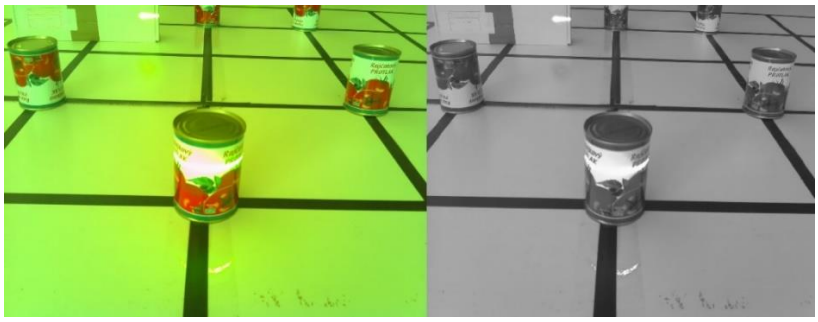
První krokem algoritmu je tedy upravit snímek tak, aby bylo potlačeno veškeré okolí dráhy laserového paprsku. Poté je nutné identifikovat umístění této dráhy na snímku a z tohoto umístění zjistit polohu objektu. Pro identifikaci dráhy je nutné provést testování na minimálně 100 snímcích pro vyšší přesnost algoritmu.

9.2 Detekce objektu

V prvním kroku je nutné je nutné potlačit veškeré okolí dráhy laserového paprsku. S použitím zeleného filtru se stává zelený paprsek nejjasnějším místem na snímku.

Pro potlačení veškerého okolí byl zvolen algoritmus prahování (thresholding). Tuto funkcionalitu poskytuje aplikace ImageJ, které poskytuje knihovnu pro Javu.

V prvním kroku, jak je vidět na obrázku 29 (na levé straně je pořízený snímek, na pravé je jeho osmibitová verze), musel být snímek převeden do osmibitového formátu.



Obr. 29 Pořízený snímek, osmibitová verze snímku

Po převodu byl použit algoritmus prahování, kde je klíčové určit hodnotu prahu. (Corke, 2011) Pro určení hodnoty je nutné vzít v potaz použitý filtr a množství světla dopadajícího na snímek. Ideální je pořídit několik testovacích snímků a podle těchto snímků určit hodnotu prahu. V tomto případě není nijak složité hodnotu prahu stanovit.

Na následujícím obrázku 30 je vidět, jak stanovená hodnota prahu ovlivňuje konečný výsledek.

Obr. 30 Příklad nastavení hodnoty prahu

Na levé straně obrázku 29 je hodnota prahu nastavena tak, že jsou zřetelné okolní odrazy světla, nicméně takovýto výsledek už je relativně vhodný pro další zpracování. V pravé části obrázku je nastavena vyšší hodnota prahu. V této části jasně dominuje zachycená dráha laserového paprsku. V okolí této fotky se vyskytují další skupiny pixelů se stejnou hodnotou jako barva paprsku. Tyto skupiny jsou způsobeny přesvětlenými místy na scéně nebo odrazy laserové paprsku. Takovéto skupiny není vhodné se snažit odstranit za pomocí prahování.

9.3 Detekce objektu ve snímku

K určení polohy objektu byl použit algoritmus vyhledávání jednoduchých linií (Fisher, Perkins, Walker, Wolfrt, 2017). Řešení tohoto problému poskytuje knihovna BoofCV, ve které jsou uvedeny dva příklady zabírající se touto problematikou.

Byl zvolen způsob detekce segmentů. S použitím tohoto algoritmu jsme schopni určit linii přechodu mezi černým podkladem a dráhou laserového paprsku.

Třída poskytující tuto metodu se jmenuje FactoryDetectLineAlgs a metoda implementující tuto funkcionalitu je lineResac. Tato metoda má šest vstupních parametrů, a to:

- regionSize – velikost oblasti, také by se dalo říci, že se jedná o délku linie,
- thresholdEdge – hodnota hrany, která určí, zda pixel patří k hraně či nikoliv,
- thresholdAngle – tolerance úhlu, která určuje, do jaké míry mohou být hrany spárovány,
- connectLines – optimalizuje a spojuje hrany,
- imageType – formát vstupního snímku,
- derivType – odvozený typ.

Návratový typ metody lineResac je objekt typu DetectLineSegmentsGridRansac, tento objekt obsahuje metodu detect, jejíž vstupní parametr je objekt typu BufferedImage, tedy analyzovaný snímek. Metoda detect vrací kolekci objektů typu LineSegment2D_F32.

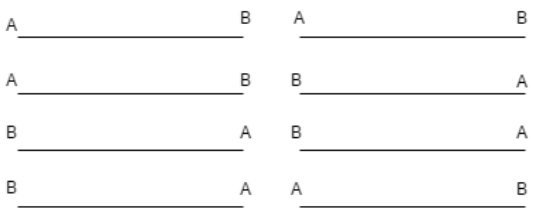
Na obrázku 31 je demonstrována důležitost parametru regionSize na výsledek. V první polovině obrázku je hodnota nastavena na 50. Je vidět, že linie dokonale nekopírují okraj přechodu černého podkladu a bílého objektu. V pravé části je hodnota regionSize nastavena na 20, takovéto nastavení poskytuje o mnoho lepší výsledek. (Laganiére, 2011)

Obr. 31 Ukázka nastavení parametru regionSize

Nastavení, které bylo použito v pravé části, je mnohem vhodnější pro další zpracování, důležitou výhodou je více bodů, které jsou obsaženy i v nejkrajnějších polohách objektu. Nevýhodou nastavení v levé části je o něco větší výpočetní náročnost.

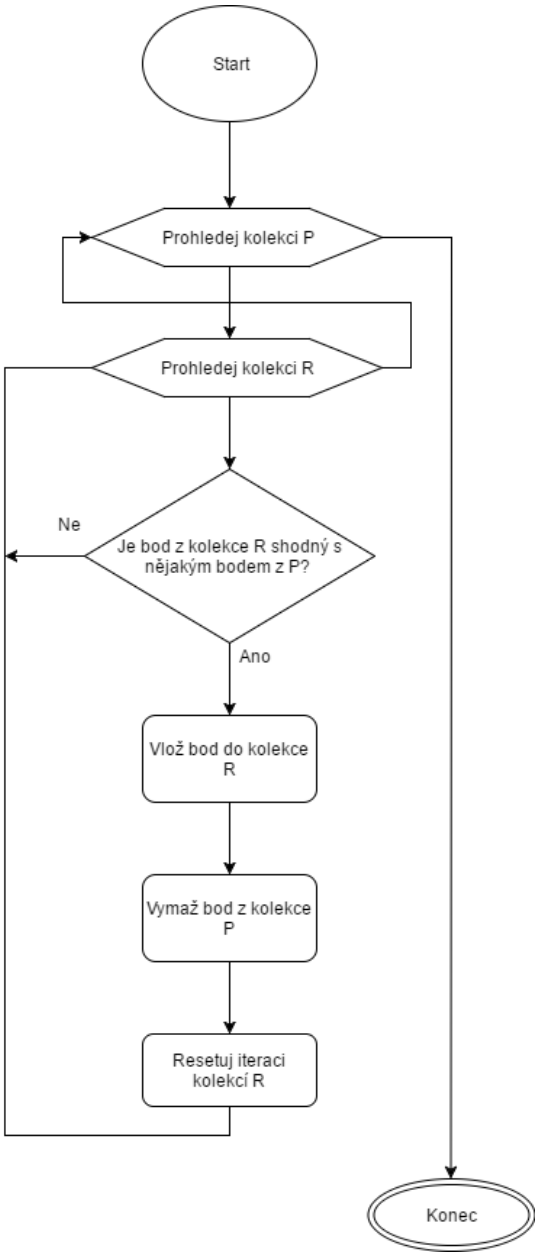
Ačkoli se na první pohled zdá, že jsou červené linie vzájemně propojeny, tak tomu není. Linie jsou tvořeny objektem LineSegment2D_F32. Tento objekt uchovává odkaz na dva objekty (bod A, bod B) typu Point2D_F32, který reprezentuje souřadnici x a y. Při výpisu těchto souřadnic bylo zjištěno, že vzdálenost bodů sousedních linií, které se zdají na první pohled propojené, se pohybuje mezi jedním až třemi pixely.

Pro další zpracování je nutné, aby sousední body linií měly stejné souřadnice, respektive, aby spojené linie obsahovaly jeden společný bod. Pro spojení linií byla implementována vlastní metoda, která spojí společným bodem ty linie, u kterých je vzdálenost krajních bodů menší než pět pixelů. Vždy neplatí, že bod s menší souřadnicí x je vždy A bod s větší souřadnicí x je vždy B. Pořadí bodů je ve všech kombinacích, jak je znázorněno na obrázku 32.



Obr. 32 Příklady rozmístění linií

Pro spojení byla implementována vlastní metoda, která pracuje s kolekcí objektů LineSegment2D_F32. Algoritmus vyhledání vzájemně spojených využívá dvě kolekce, a to kolekci P kde se na začátku nachází všechny linie, a kolekci R, kde se nachází vzájemně propojené linie. Algoritmus je znázorněn na obrázku 33.



Obr. 33 Diagram algoritmu hledání vzájemně propojených bodů

Tento algoritmus se puštěn pro každou linii v kolekci R. Po dokončení algoritmu v kolekci R zbydou pouze ty linie, které s žádnou další linií nesdílí jeden z krajních bodů. Při tvorbě tohoto algoritmu jsem se inspiroval z algoritmů řešících problém hledání kostry grafu.

Z předchozího popisu vyplývá, že nás pouze zajímají linie, které jsou „vodorovné“. Bohužel ne všechny snímky nejsou ideální jako ten, na kterém byl demonstrován algoritmus. Příklad nepříliš vhodného snímku je na obrázku 34. Proto bylo nutné odstranit vodorovné linie. Zda je linie vodorovná nebo svislá, je dáno výpočtem absolutní hodnoty rozdílu souřadnice y bodu A a též souřadnice bodu B. Pokud je tento výsledek větší než zvolná hodnota, je tato linie označena jako „svislá“, je odstraněna a není s ní dále pracováno.

Obr. 34 Nevhodně nasvícený snímek

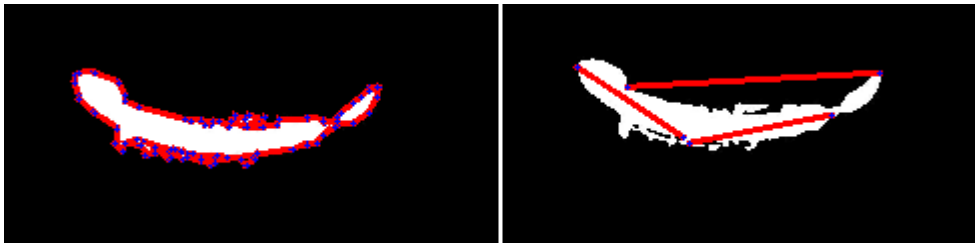
Ve střední části snímku je příliš osvíceno hrací pole. Toto osvětlení zamezuje potlačení veškerého okolí dráhy laserového paprsku. Jako řešení by se nabízelo zvednutí hodnoty prahu, nicméně se jedná o velmi jednoduché řešení, které by mohlo zásadně snížit efektivnost algoritmu u více (ale rovnoměrně) nasvícených snímků.

Dalším problémem, který lze na tomto obrázku demonstrovat, jsou již zmíněné „svislé“ linie. Díky velkému množství světla ve střední části snímku je viditelná část hracího pole, která obsahuje několik svislých linií. Detektor vyhledávající linie ve snímku rovněž identifikuje tyto linie, a jelikož je známo, že se paprsek pohybuje pouze „vodorovně“, tedy zprava doleva a naopak, jsou tyto linie vymazány.

Nyní po odfiltrování nežádoucích linií jsme ze snímku získali skupiny vzájemně propojených linií. Takováto informace obsahuje již nějakou vypovídající hodnotu, ale stále je to málo k detekci objektu.

Proto je nutné tuto informaci dále zpracovat a podle ní určit, zdali se jedná o hledaný objekt či nikoliv. Byl zvolen následující postup: ze vzájemně propojených linií je vytvořena jedna linie, která je mnohonásobně delší, než je průměrná délka vkládaných linií. Mohlo by se zdát, že postup je zbytečně složitý a že by bylo možné stejného výsledku dosáhnout úpravou hodnot vstupních parametrů metody pro vyhledávání linií (lineResac (...)). Po sérii testů se ukázalo, že tento způsob nefunguje a výsledek není příliš použitelný.

Pro nahrazení skupiny vzájemně propojených linií jedinou linií byl zvolen následující postup. Jako vstupní data přijmeme kolekci vzájemně propojených linií (postup na obr 32). Z této kolekce vybereme bod s nejmenší souřadnicí x a bod z největší souřadnicí x. Z těchto souřadnic je vytvořena linie a jí jsou nahrazena všechna vstupní data. Výsledek je vyobrazen na obrázku 35.



Obr. 35 V levé části se nachází výřez snímku před transformací, v části pravé snímek po transformaci

Jestli se jedná či nejedná o hledaný snímek, zjistíme z délky nejdelší linie. Pokud se objekt nachází v přední části snímku, tedy jeho souřadnice y (např. 1200px) má vysokou hodnotu, je jasné, že nemůže být porovnáván podle stejných kritérií jako objekt v horní části snímku, tedy objekt s nízkou hodnotou y (např. 170px).

Proto musely být stanoveny spodní a horní hranice pro skupiny objektů umístěných na horizontálních čarách (1, 2, 3, 4, 5, 6, 7) hracího pole. Kritéria, aby byla linie považována za objekt, jsou uvedena v tabulce 3.

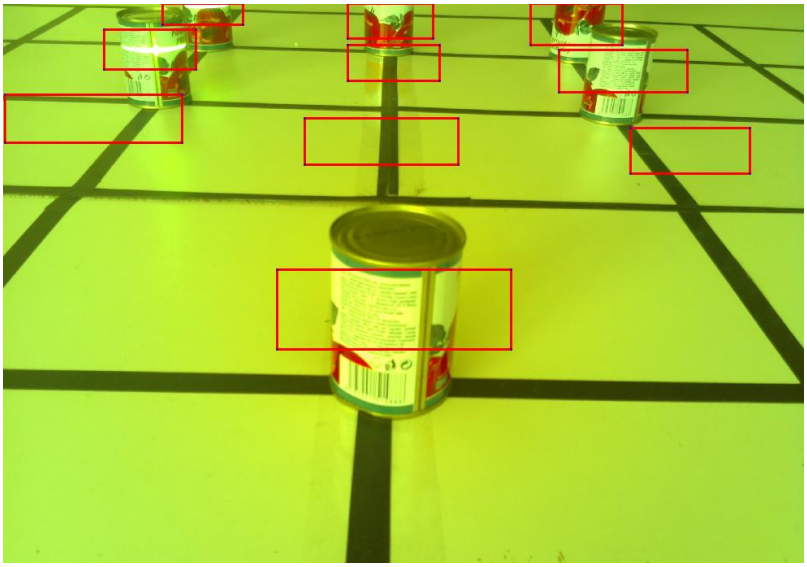
Tab. 3 Kritéria pro identifikaci objektu

Horizontální čára	**Délka (px)**	
	Minimální	**Maximální**
První	252	420
Druhá	180	300
Třetí	125	215
Čtvrtá	102	170

9.4 Detekce polohy objektu na hracím poli

V kapitole 7 jsem popsal, jakým způsobem jsou skenovány objekty. Pro oživení zmíním, že objekty jsou skenovány postupně. Je možné naskenovat levou, prostřední nebo pravou část, z toho je jasné, že dráha paprsku se může objevit vždy jen v určité části snímku. Z toho jsem vycházel i při detekci polohy na hracím poli.

Robot se sice pohybuje po hracím poli, ale pořízené snímky díky čtvercovému rozdělení hracího pole vypadají vždy velice podobně. Ve snímku bylo vybráno devět oblastí (region of interest), které jsou pro robota významné. Tyto oblasti mají tvar čtverce a jsou určené jedním bodem, výškou a délkou čtverce. Rozdělení bylo provedeno podle série testovacích snímků a je vyobrazeno na obrázku 36.



Obr. 36 Příklad point of interests na hracím poli

Každá oblast na obrázku 35 má jiný rozměr, i když se nacházejí v jedné řadě. Tato nerovnoměrnost je dána pozicí robota a kamerového modulu. Robot vždy nesvírá pravý úhel s deskou hracího pole, ani kamera nemůže zabírat scénu ideálně.

Oblasti jsou několikanásobně větší než pozice, na kterých se má vyskytovat dráha laserového paprsku. Toto bylo zavedeno kvůli určité toleranci pozice objektu. Ve hře se plechovky na hrací pole rozmisťují rychle, to zapříčiní, že plechovka není vždy umístěna přesně na kříž, další důvod je kalibrace robota, robot při jízdě může občas ztratit ideální stopu, to zapříčiní i lehký posun celé scény, tento problém řeší již zmíněná tolerance.

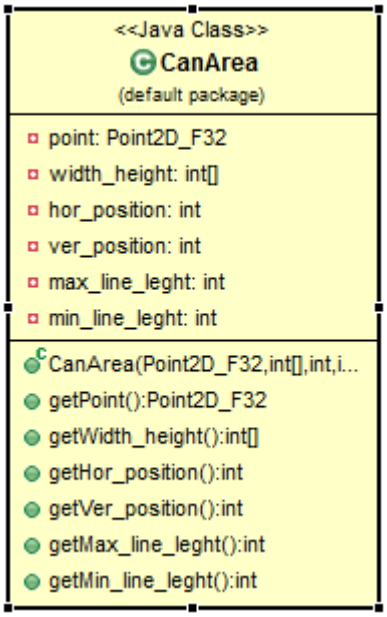
Nicméně v případě, že se robot začne pohybovat po jiném než domovském poli (pole v Q11), je nutné zkontrolovat, zda všechny point of interest odpovídají realitě.

9.5 Implementace algoritmu

Celý algoritmus byl implementován v jazyce Java. V době, kdy započala práce na novém řešení, již měl robot zavedené API pro komunikaci prostřednictvím I2C sběrnice, pořizování snímků a jejich předání k dalšímu zpracování, ovládání servomotorů a motorů pro pohyb robota. Toto řešení bylo použito i v mé práci.

Robot také disponoval systémem mapování pozice, detekce objektů a vlastní herní strategií. Tyto poslední tři již zavedené body byly nahrazeny vlastním řešením.

Jako první bude popsána třída CanArea, která řeší částečně problematiku herní strategie a její class diagram je na obrázku 37. Tato třída reprezentuje pozici objektu na hracím poli:

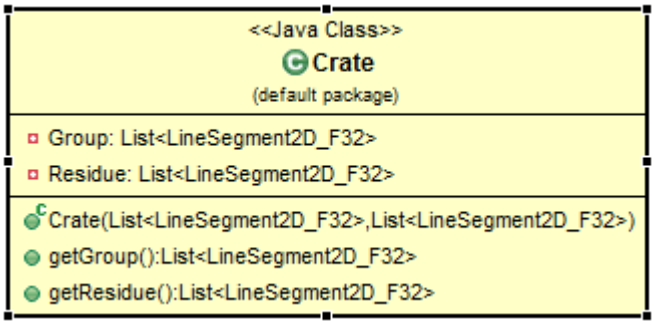


Obr. 37 Class diagram třídy CanArea

- proměnná point je pravý horní bod oblasti předpokládané dráhy laserového paprsku,
- proměnná whidth_height je dvouprvkové pole, prvek indexem jedna je délka, prvek s indexem dva je šířka,
- proměnné hor_position a ver_positions jsou souřadnice oblasti na hracím poli (např. 4 B),
- proměnné max_line_leght a min_line_leght jsou omezení pro detekci objektu.

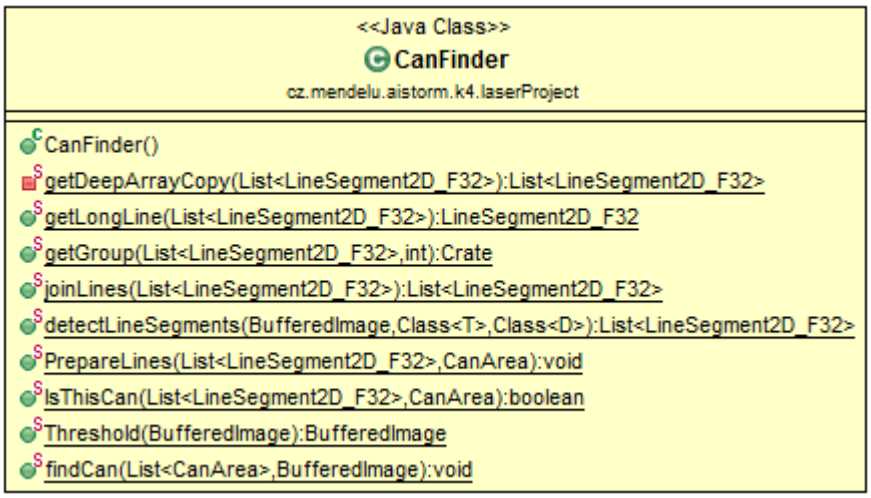
Tato třída byla implementována kvůli větší přehlednosti kódu. Navíc je praktické držet informace o oblasti a kritéria pro kvalifikaci objektu (plechovky s rajským protlakem) na jednom místě.

Třída Crate obrázek 38 byla vytvořena kvůli vyšší efektivnosti a čitelnosti metody getGroup (obrázek 38). V proměnné Group je uložena skupina vzájemně propojených linií a v proměnné Residue zbytek linií, u kterých nebylo ještě testováno vzájemné propojení.



Obr. 38 Class diagram třídy Crate

Třída CanFinder slouží pro identifikaci objektu. Její class diagram je na obrázku 39.



Obr. 39 Class diagram třídy CanFinder

Metody:

- metoda getArrayCopy(…) jako vstupní parametr přijímá kolekci linií, ze které vytvoří hlubokou kopii, tato metoda je používána v metodě (getGroup(…))
- metoda getLongLine(…) přijímá jako vstupní parametr kolekci linií, z této kolekce vybere bod (Point2D_F32) s nejmenší a největší souřadnicí x, z těchto dvou bodů vytvoří instanci objektu LineSegment2D_F32, kterou následně vrátí,

- metoda getGroup(…) vyhledá první skupinu vzájemně propojených objektů, následně vytvoří instanci třídy Crate, která uchovává již zmíněnou skupinu a zbylé linie
- metoda joinLines(…) spojí linie, jejíž vzdálenost je menší než 5px, jedná se o vzdálenost jak souřadnic x, tak souřadnic y, princip je vyobrazen na Algoritmu 1
- metoda detectLineSegments(…) vyhledává linie na vstupním snímku, parametr typu CanArea je zde proto, aby metoda vyhledávala linie jen v předpokládaném prostoru
- metoda IsThisCan(…) vrací true v případě, že se jedná o hledaný objekt a false v případě, že se o hledaný objekt nejedná, jako vstupní parametr přijímá kolekci linií a objekt typu CanArea, který obsahuje všechna důležitá kritéria
- metoda Threshold(…) řeší problém prahování, upraví snímek podle hodnoty zvoleného prahu.

10 Plánování trajektorie robota

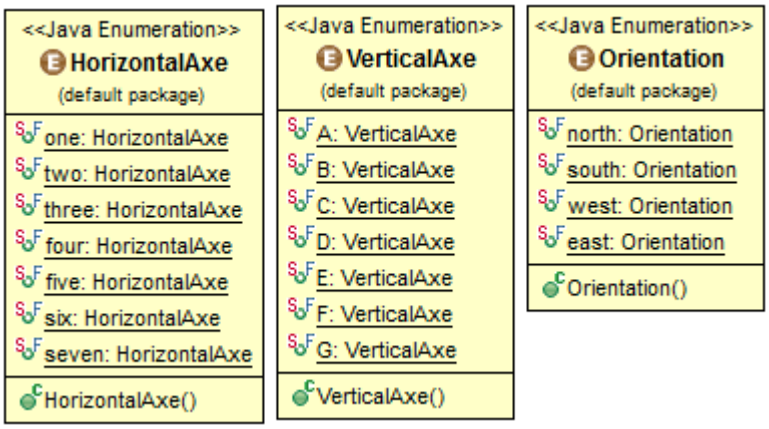
Naplánování trajektorie je poslední část, kterou je nutno vyřešit pro použití robota. V předchozích kapitolách bylo popsáno, jakým způsobem robot mapuje herní pole.

Tato kapitola obsahuje možný způsob, jak naložit s pořízenými informacemi. Tento způsob naložení s informacemi můžeme nazvat herní strategií, ta může být díky již implementovaným metodám jednoduše změněna.

Do strategie může být například přidána větev pro krádež soupeřových plechovek, i takovéto techny jsou na soutěži k vidění. V rámci zachování fair-play jsem se rozhodl tento mechanizmus neimplementovat.

10.1 Reprezentace hracího pole

Hrací pole má čtvercový tvar, kamera robota bohužel není schopná kvůli technickým specifikacím tento celý tvar zachytit, je schopna zachytit pouze výřez. K popisu hracího pole slouží dva výčtové tipy (HorizontalAxe, VerticalAxe), které jsou na obrázku 40. (Bloch, 2002)



Obr. 40 Class diagram enumerátorů: HorizontalAxe, VerticalAxe, Orientation

Výčtové typy HorizintalAxe a VerticalAxe slouží k označení os výřezu, který je zachycen kamerovým modelem, připevněným na robotu.

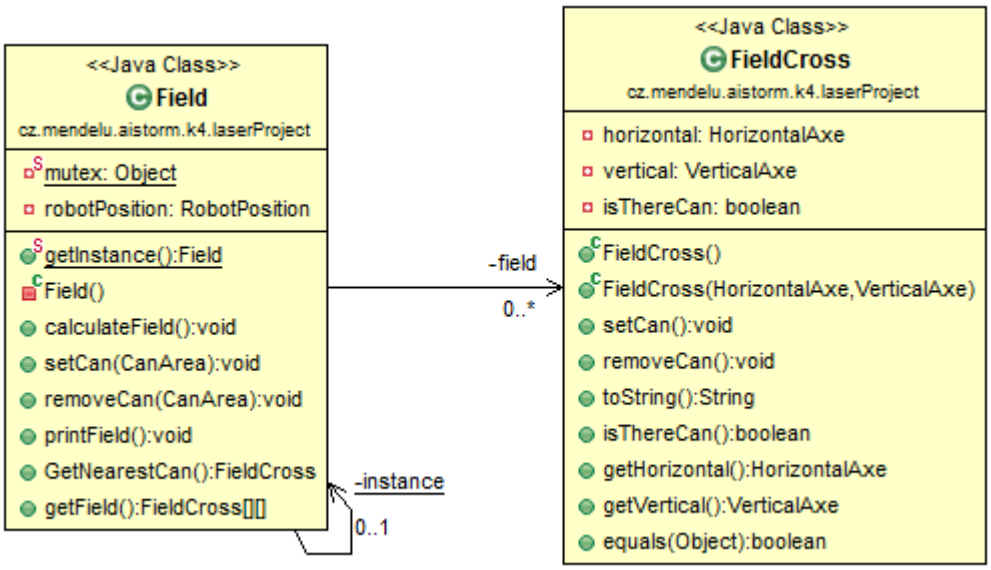
Typ HorizontalAxe označuje horizontální čáry na hracím poli, které jsou označeny číselně. Pokud se při pohybu robota hodnota této osy nemění, můžeme říci, že se robot pohybuje po hracím poli zprava doleva nebo naopak.

Typ VerticalAxe označuje vertikální čáry na hracím poli, které jsou označeny prvními sedmi písmeny abecedy. Pokud se při jízdě robota hodnota této osy nemění, můžeme říci, že se robot pohybuje po hracím poli zdola nahoru nebo naopak.

Třetí výčtový typ Orientation určuje natočení robota na hracím poli, a tím pádem odhodnocení os. Tento výčtový typ je nutné chápat tak, jak to je na zeměpisných mapách. Jedná se pouze o natočení robota vůči hracímu poli, světové

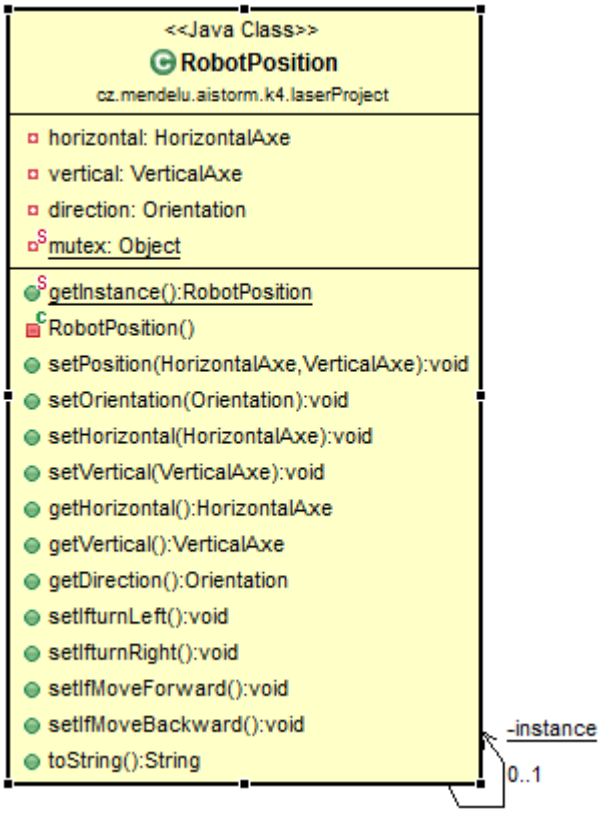
strany nemusí odpovídat realitě. Natočení North (sever) je vždy, pokud je robot natočen kamerovým modulem k soupeřově straně.

Zmíněný výřez je reprezentován třídou Field, jejíž Class diagram je v levé části obrázku 41.



Obr. 41 Class diagram tříd Field a FieldCross

Proměnná field je dvojrozměrná matice, jejíž každý prvek je tvořen instancí objektu FieldCross, jejíž Class diagram je v pravé části obrázku 40. Proměnné horizontal a vertical používají dříve popsané výčtové typy pro identifikaci jejich polohy. Proměnná isThereCan říká, jestli se zde nachází plechovka. Metoda setCan nastaví hodnotu proměnné isThereCan na true a metoda removeCan na false. Proměnná robotPosition obsahují odkaz na třídu, která shromažďuje informace o pozici robota na hracím poli. Class diagram třídy RobotPosition je na obrázku 42. Metoda
GetNearestCan vrací pozici plechovky (FieldCross), která je nejblíže robotu, k tomu je použita Manhattanská metrika. Poslední metoda ve třídě FieldCross je metoda equals, jedná se o překrytou metodu, pokud se horizontální a vertikální osy shodují, metoda vrací true, pokud ne, pak vrací false.



Obr. 42 Class diagram třídy RobotPosition

Proměnné horizontal a vertical určují průsečík os, na kterém se nachází robot, proměnná direction určuje natočení robota. Toto natočení je určeno světovými stranami. Na tuto třídu byl použit návrhový vzor singleton.

Metody začínající slovem set, slouží k nastavení hodnot privátních proměnných. Metody začínající slovem get, vrací hodnoty privátních proměnných.

Metoda setIfTurnLeft nastaví pozici robota, pokud se otáčí proti směru hodinových ručiček s ohledem na aktuální pozici. Metoda setIfTurnRight provádí to samé, ale s tím rozdílem, že se robot točí po směru hodinových ručiček, tedy na pravou stranu.

Metoda setIfMoveForward změní pozici robota při jízdě dopředu. Pokud je robot natočen na „north nebo south“, mění se pouze vertikální osa. Pokud je robot natočen na „west nebo east“, mění se pouze horizontální osa.

Pozice robota je dána výčty, které obsahují atributy v pořadí, v jakém se robot pohybuje po hracím poli. Proto při pohybu robota stačí k proměnné vertical nebo horizontal „přičíst jedničku“ (posunout se ve výčtovém typu o atribut dopředu) v případě pohybu dopředu nebo „odečíst jedničku“ (posunout se ve výčtovém typu o atribut dozadu) v případě pohybu dopředu.

V neposlední řadě je nutná kontrola při inkrementaci a dekrementaci. V případě dekrementace nesmí být hodnota menší než nula a v případě inkrementace nesmí být hodnota větší než sedm.

10.2 Orientace a pohyb na hracím poli

Robot musí mít implementován pružný systém pohybu po hracím poli tak, aby mohl naplánovat trajektorii, která je specifická pro co největší množství situací, které mohou ve hře nastat.

K potřebám soutěže byla naimplementována metoda GetNearestCan ve třídě Field, která vrací instanci objektu typu FieldCross, tato instance představuje konzervu, která se nachází nejblíže k robotu.

Pokud je známa cílová pozice, je nutné se k ní dostat, pro tento problém byla implementována metoda MoveOnPosition ve třídě KetchupHouse. První vstupní parametr této metody je objekt FieldCross, který reprezentuje cílovou pozici na hracím poli.

Metoda obsahuje tři základní větve pro pohyb na hracím poli. První větev je určena pro sběr konzerv, pokud se robot pohybuje primárně vpřed. Algoritmus je demonstrován na obrázku 43.

Algorithm 1: Jizda robotu kupredu

```
1  Jizda vpred (cil, sberPlechovek);
2  if HorizontalniOsaRobota > HorizontalniOsaCile then
3      while Dokud horizontalni pozice robota je ruzna od horizontalni
       pozice cile do
4          pojed o jedno pole vpred;
5          if Je pred robotem konzerva AND sberPlechovek then
6              naber plechovku;
7          end
8      end
9      if Horizontalni osa cile a robota jsou shodne, ale vertikalni se lisi
       then
10         if Je plechovka nalevo then
11             otoc robota vlevo;
12         else
13             otoc robota vpravo;
14         end
15         while Dokud vertikalni pozice robota je ruzna od vertikalni pozice
           cile do
16             pojed o jedno pole vpred;
17             if Je pred robotem konzerva AND sberPlechovek then
18                 naber plechovku;
19             end
20         end
21     end
22 end
```

Obr. 43 Pseudo-algoritmus jízdy robotu k cíli, který se nachází před robotem.

Pokud je vertikální osa, na které stojí robot shodná s cílovou vertikální osou a liší se pouze horizontální, robot se otočí na sever (pokuj již není natočen), popojede dopředu a nabere předmět.

V případě že se liší horizontální i vertikální osa pozice robota od pozice cíle, je nutné trajektorii lehce upravit. V tomto případě ideální trajektorie robota je

podobná písmenu „l“ (pohyb je podobný tahu koněm v šachu). První krok je stejný jako v předchozím případě, robot dojede na horizontální osu, na které se nachází cílová pozice.

Jakmile se nachází na této ose, rozhodne se, jestli se natočí na západ nebo na východ, natočení vypočítá opět ze své a cílové pozice. Pokud je ordinální číslo cílové pozice menší než ordinální číslo pozice robota, robot se natočí na západ a poté pokračuje rovně k cílové pozici, kde nabere předmět. Pokud je ordinální číslo cílové pozice větší, robot se natočí na východ a pokračuje k cílové pozici, kde nabere předmět. (Kubík, 2004) Algoritmus je znázorněn pomocí pseudokódu na obrázku.

Obdobným způsobem je implementována jízda robotu na zpět (k domovské ose). Toto je nutné k vyložení nákladu.

Poslední situace, která není pokryta, se sestává ze dvou případů, kdy je robot natočen na západ nebo východ a konzerva se nachází na stejné horizontální ose. V takovémto případě robot zjistí, jestli se konzerva nachází vůči jeho pozice na západě nebo na východě, a vyrazí k ní.

Poslední možnost ovlivnění pohybu robota je dána druhým vstupním parametrem. Jedná se o parametr typu boolean, pokud tento parametr nabývá hodnoty true, robot po cestě sebere i takovou konzervu, která nebyla zmapována (po každém kroku vpřed se dotazuje na informaci z proximitního snímače). Pokud je tento parametr false, robot ignoruje veškeré plechovky, na které po cestě narazí.

10.3 Herní strategie robota

Tato kapitola popisuje herní strategii robota. Tato strategie byla zvolena tak, aby se robot pohyboval co nejčastěji kolem středu hracího pole, kde se vyskytuje co nejvíce hledaných konzerv.

Robot se pohybuje po svislých a vertikálních osách. Protože hledané konzervy se vyskytují vždy na průsečíku vertikální o horizontální osy (pokud jí robot neposune), byl zvolen tento způsob pohybu, který kopíruje Manhattanskou metriku.

Robot začíná na startovním místě D1. Po odstartování soutěže robot spustí skenování všech tří částí hracího pole. Jakmile je sken hotov, robot se vydá posbírat všechny konzervy, které pomocí skenu objevil. Při jízdě vždy kontroluje, jestli nenarazí na konzervu, kterou se mu nepodařilo objevit pomocí analýzy obrazu. Pokud na takovouto konzervu narazí, nabere ji.

Jakmile je nákladový prostor robota plný nebo sesbírá všechny objevené konzervy, vydá se na souřadnici, na které provede složení nákladu.

Tento postup je nutný, protože může nastat situace, kdy robot nenavštíví všechny plánované pozice, a už má plný nákladový prostor. Jakmile je nákladový prostor robota plný, vyrazí na první místo složení nákladu, které se nachází na souřadnicích B1 a C1. Jakmile vyloží tyto plechovky, vydá se na souřadnici G4 přes souřadnici C3.

Tato cesta je zvolena úmyslně, aby robot po cestě mohl nasbírat další konzervy (bez skenování). Jakmile robot dorazí na souřadnici G4, provede natočení na west a spustí další sken. A jako v předchozím případě se vydá pro plechovky, které odhalil

a po cestě sbírá konzervy, které se mu nepodařilo odhalit. Zde také platí pravidlo složení konzerv na předem určené místo v případě plného obsazení nákladového prostoru. Jeho trasa je dána implementací algoritmu, který vede robota k cíli. Druhé místo výkladové místo se nachází na souřadnicích E1 a F1.

Z předchozího textu vyplývá, že robot má implementováno pět statických operací, které provede každou hru. Jedná se následující kroky:

- sken z domovské souřadnice,
- první složení nákladu,
- přesun robota na druhou pozici pro sken,
- druhý sken ze souřadnice G4,
- druhé složení nákladu.

Celá strategie je napsána přehledně a jednoduše, proto není problém ji lehce upravit. První úprava, která se nabízí je po nasbírání tří konzerv, je poslat robota na místo složení a předpokládat, že po cestě ještě jednu konzervu nabere.

Po těchto dvou krocích program robota končí.

11 Závěr

Na začátku práce byl prozkoumán stávající stav robotu. Byla zmapována senzorická soustava robotu, ovládání periferií a herní strategie.

V prvním kroku práce byly prozkoumány nabízené senzory, které by bylo reálné použít na robotu K4. Z těchto senzorů byl jako nejvhodnější vybrán laser. V souvislosti s použitím laseru bylo nutné nastudovat problematiku spojenou s jeho bezpečným použitím. Z důvodu bezpečnosti proběhl dialog s pořadateli, kteří jasně stanovili podmínky pro použití laserem.

V souladu s pravidly soutěže Ketchup house bylo zkonstruováno laserové zařízení, ke kterému byl implementován ovládací software.

Kamerový modul robotu K4 byl synchronizován s laserovým zařízením, a pro co nejlepší snímek byl použit zelený filtr.

Pro rozpoznání hledaných předmětů na hracím poli byl napsán algoritmus, který je založen na segmentaci obrazu, pro detekci využívá dvě disciplíny, a to prahování a hledání linií v obraze. Tento algoritmus byl podroben sérií testů pro ověření jeho funkčnosti. Díky této implementaci je nyní robotu známa část aktuální situace na hracím poli, a je schopen se na základě jednoduchých pravidel zvolit následující krok.

Při realizaci tohoto řešení došlo ke vzniku slepých míst, to jsou místa, která jsou pro robota důležitá, ale robot je nedokáže efektivně zmapovat. Pro eliminaci tohoto problému byl do přední části robotu umístěn proximitní senzor.

Dále byl vytvořen nový způsob reprezentující známe prostředí robotu. Nyní robot hrací pole vnímá jako množinu křižovatek, z čehož křižovatka může obsahovat informaci o objektu nacházejícím se na ní.

V předposledním kroku byl navrhnut a implementován algoritmus, který dokáže co nejrychleji přemístit robota na libovolnou cílovou křižovatku na hracím poli z libovolné křižovatky na hracím poli.

Posledním krokem bylo navrhnout a implementovat novou herní strategii. Tato strategii se skládá ze dvou hlavních bodů: skenování hracího prostředí a vyložení nákladu na domovskou osu. Tato strategie má pevně daných pět kroků, co se děje mezi nimi záleží na aktuální konfiguraci hry.

Literatura

AiStorm, *Mobilní robot K4* [online]. 2017 Dostupné z: https://aistorm.mendelu.cz/cz/projekty/k4/.

Bezděk, Miloslav. *Elektronika*. České Budějovice: Kopp, 2002. ISBN 8072321714.

Bezpečnost laserových zařízení. *ČSN EN 60825-1*. 3rd ed. 2015.

Bloch, Joshua. *Java efektivně: 57 zásad softwarového experta*. Praha: Grada, 2002. Moderní programování. ISBN 8024704161.

Brackeen, David, Bret BARKER a Laurence VANHELSUWÉ. *Vývoj her v jazyku Java*. Praha: Grada, 2004. Moderní programování. ISBN 8024708744.

Corke, Peter I. *Robotics, vision and control: fundamental algorithms in MATLAB.* Berlin: Springer, 2011. Springer tracts in advanced robotics, v. 73. ISBN 9783642201431.

Everett, H. R. *Sensors for mobile robots: theory and application.* Wellesley, Mass.: A.K. Peters, c1995. ISBN 978-1568810485.

Fisher, Robert, Simon PERKINS, Ashley WALKER a Erik WOLFART. *Line Detection*. Http://homepages.inf.ed.ac.uk [online]. [cit. 2017-05-13]. Dostupné z: http://homepages.inf.ed.ac.uk/rbf/HIPR2/linedet.htm

Fiřt, Jaroslav a Radek HOLOTA. *Digitalizace a zpracování obrazu* [online]. Plzeň, 2002[cit.2017-05-03].
Dostupné z: http://home.zcu.cz/~holota5/publ/DigZprO.pdf

IJSC, *A New Local Adaptive Thresholding Technique in Binarization.* International Journal of Computer Science Issues [online]. 2015 [cit. 2017-05-03]. ISSN 1694-0814. Dostupné z: https://arxiv.org/ftp/arxiv/papers
/1201/1201.5227.pdf

Juránek, Antonín a Miroslav HRABOVSKÝ. *EAGLE: návrhový systém plošných spojů pro začátečníky: uživatelská a referenční příručka.* Praha: BEN – technická literatura, 2005. ISBN 8073001772.

Kubík, Aleš. *Inteligentní agenty*. Brno: Computer Press, 2004. ISBN 8025103234.

Laganiére, *Robert. OpenCV 2 computer vision application programming cookbook: over 50 recepies to master this library of programming functions for real-time computer vision.* Brimmingham: Packt Publishing, 2011. Quick Answers to Common Problems. ISBN 978-1849513241

Liang, O. *Raspberry Pi and Arduino Connected Using I2C*. [online]. 2013. URL: http://blog.oscarliang.net/raspberry-pi-arduino-connected-i2c/.

Novák, Petr. *Mobilní roboty: pohony, senzory, řízení*. Praha: BEN – technická literatura, 2005. ISBN 8073001411.

Pagáč, Marek. *Učebnice SolidWorks*. V Brně: Vydavatelství Nová média, 2017. ISBN 978-80-270-0918-3.

Prata, Stephen. *Mistrovství v C++*. Praha: Computer Press, 2001. Všechny cesty k informacím. ISBN 8072263390.

Upton, Eben. *Learning computer architecture with raspberry PI.* 1rd. Canada: Wiley, 2016. ISBN 978-1-119-18393-8.

Vlach, Jaroslav. *Hledání úseček a kružnic s využitím Houghovy transformace při zpracování obrazu v LabView.* Automa [online]. 2011 [cit. 2017-05-03]. Dostupné z: http://download.ni.com/pub/branches/ee/article_archive/cz/2011/ni_in_automa_in_feb_2011.pdf

Voda, Zbyšek. *Průvodce světem ARDUINA* [online]. 2014 [cit. 2017-05-03]. Dostupné z: https://drive.google.com/file/d/0B5k9VLyI1vUoTWVndVhub3J6ZUU/edit

Vojáček, *Antonín. Princip laserových snímačů vzdálenosti s triangulačním principem měření* [online]. 2015 [cit. 2017-05-03]. Dostupné z: http://automatizace.hw.cz/mereni-a-regulace/princip-funkce-laserovych-snimacu-vzdalenosti-s-triangulacnim-principem-mereni.html

Warren, John-David., Josh. ADAMS a Harald MOLLE. *Arduino robotics*. New York, NY: Apress, c2011. Technology in action series. ISBN 978-1-4302-3183-7.

Seznam obrázků

Obr. 1 Čtvercová mřížka Zdroj: https://www.quora.com/How-many-ways-can-you-traverse-a-four-by-four-square-grid-without-doubling-back-or-going-in-the-wrong-direction 15

Obr. 2 Hexagonální mřížka Zdroj: https://commons.wikimedia.org/wiki/File:Hexagonal_tiling.svg 16

Obr. 3 Bayerův filtr Zdroj: https://commons.wikimedia.org/wiki/File:Bayer_matrix.svg 16

Obr. 4 Hrací hřiště Zdroj: http://roboticday.org/2017/rules/2017-Ketchup_House-CZv1.pdf 23

Obr. 5 3D návrh mobilního robota K4 Zdroj: https://aistorm.mendelu.cz/cz/projekty/k4 24

Obr. 6 Mobilní robot K4 25

Obr. 7 Pohled přední část robota 27

Obr. 8 Laserové zařízení s krokovým motorem 28

Obr. 9 Schéma obvodu pro řízení krokového motoru a laserového modulu 29

Obr. 10 Návrh DPS pro řízení krokového motoru a laserového modulu 29

Obr. 11 Laserové zařízení se servomotorem 30

Obr. 12 Skica znázorňující odraz laserového paprsku 31

Obr. 13 Základna laserové zařízení 32

Obr. 14 Uchycení laserového modulu 32

Obr. 15 Nosič odrazové plochy 33

Obr. 16 Model laserového zařízení 33

Obr. 17 Skenované pozici na hracím poli 34

Obr. 18 Schéma zapojení elektronického obvodu pro řízení servomotoru 35

Obr. 19 Osazovací plán laserového zařízení se servomotorem 36

Obr. 20 Obvod pro ovládání laserového zařízení se servomotorem (skutečný vzhled) 37

Obr. 21 Příklad zastínění jedné plechovky druhou 38

Obr. 22 Předmět osvícený laserovým zařízením 39

Obr. 23 Neutrální filtr ND 16 40

Obr. 24 Snímek objektů s použitím filtru ND16 a dloubou expozice 700000uS 40

Obr. 25 Zelený filtr 41

Obr. 26 Objekty osvícené laserovým paprskem s použitím zeleného filtru 41

Obr. 27 Hledání objektu v levé části hracího pole 42

Obr. 28 Trojce snímků. Bez použití filtru, zelený filtr, neutrální filtr ND16. 42

Obr. 29 Pořízený snímek, osmibitová verze snímku 43

Obr. 30 Příklad nastavení hodnoty prahu 44

Obr. 31 Ukázka nastavení parametru regionSize 45

Obr. 32 Příklady rozmístění linií 45

Obr. 33 Diagram algoritmu hledání vzájemně propojených bodů 46

Obr. 34 Nevhodně nasvícený snímek 47

Obr. 35 V levé části se nachází výřez snímku před transformací, v části pravé snímek po transformaci 48

Obr. 36 Příklad point of interests na hracím poli 49

Obr. 37 Class diagram třídy CanArea 50

Obr. 38 Class diagram třídy Crate 51

Obr. 39 Class diagram třídy CanFinder 51

Obr. 40 Class diagram enumerátorů: HorizontalAxe, VerticalAxe, Orientation 53

Obr. 41 Class diagram tříd Field a FieldCross 54

Obr. 42 Class diagram třídy RobotPosition 55

Obr. 43 Pseudo-algoritmus jízdy robotu k cíli, který se nachází před robotem. 56

Seznam tabulek

Tab. 1 Bezpečnostní třídy laserů Zdroj: http://www.carove-lasery.cz/soubor-maximalni-povoleny-vykon-pro-jednotlive-tridy-v-zavislosti-na-vlnove-delce-10-.pdf 13

Tab. 2 Seznam součástek pro obvod laserového zařízení 35

Tab. 3 Kritéria pro identifikaci objektu 48

Přílohy

A Program pro robot K4

Na přiloženém CD se nachází kompletní projekt pro řízení robota K4 v soutěži Ketchup House.