VYUŽITÍ MODULU RASPBERRYPI PRO ŘÍZENÍ MOBILNÍHO ROBOTA

Michal Dolejšek, Jan Mareš

Ústav počítačové a řídicí techniky, VŠCHT Praha

Abstrakt

Modul RaspberryPi je miniaturní plnohodnotné PC s konektivitou odpovídající běžnému osobnímu počítači (rozhraní USB, HDMI, RJ-45, GPIO) a dostatečným výkonem, díky čemuž ho lze použít v nejrůznějších aplikacích (mimo jiné i pro automatizaci a řízení). Specifickými vlastnostmi tohoto modulu jsou minimální rozměry, nízká spotřeba umožňující mobilní využití, univerzálnost použití díky kompatibilitě s platformou Linux a v neposlední řadě i příznivá cena.

Jako konkrétní aplikace byl modul použit pro řízení mobilního robota NXT Mindstorms od společnosti LEGO. Jedná se o robotickou stavebnici umožňující řízení přes rozhraní USB či Bluetooth z řídícího počítače za pomoci mnoha programovacích jazyků, mimo jiné Matlab či Python (použitý v této práci). Tato robotická sada dovoluje připojení krokových servomotorů a senzorů mnoha typů, standardně ultrazvukový senzor vzdálenosti, světelný a dotykový senzor.

1 Úvod

Modul Raspberry Pi byl použit jako řídící modul pro mobilního bezdrátového robota, schopného pohybu vpřed, vzad a otáčení kolem osy. Zkonstruovaný manipulátor umožňuje transport předmětu vhodných rozměrů a hmotnosti.

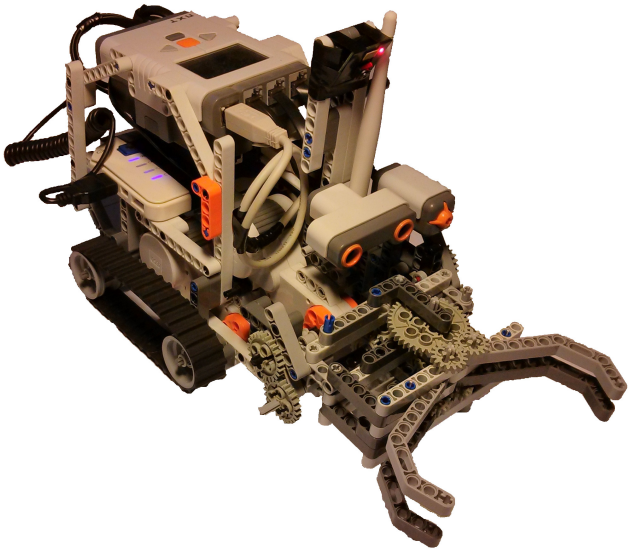
Mimo manuálního ovládání má robot také automatický mód, při kterém vyhledává zabudovanou kamerou za pomoci obrazové analýzy objekt zadané barvy, vypočítá jeho vzdálenost a odchylku od osy robota a iteračně se k objektu přiblíží a manipulátorem uchopí. K výběru jsou tři různé barvy (červená, zelená a modrá) a systém umožňuje jednoduché dodefinování libovolné další syté barvy.

2 Konstrukce robota

Tvorba robota byla složena z řady úkonů, které lze rozdělit do čtyř částí. První část byla konstrukce celé platformy a umístění hardware. Dále to byla instalace operačního systému a základní nastavení řídícího modulu. V další části bylo třeba připravit na modulu prostředí pro běh webové aplikace a pro streamování videa. Poslední, nejrozsáhlejší částí pak bylo vytvoření samotné webové aplikace implementující všechny požadované funkce.

Na obr. 1 je schematicky znázorněno zapojení jednotlivých komponentů hardware.

Obrázek 1: Schema zapojení hardware



Obrázek 2: Platforma robota

2.1 Konstrukce

Platforma robota se skládá z pásových pojezdů, robotického manipulátoru, dále z ovládacího zařízení NXT, řídicího počítače RaspberryPi s nainstalovanou kamerou a s napájecího zdroje pro řídicí modul. Celé zapojení je znázorněno na obr. 2

2.2 Instalace Raspberry Pi

Jako operační systém byl zvolen Rasbian, upravená varianta Linuxové distribuce Debian. Dále byl nainstalován webový server Apache a balíčky potřebné pro běh frameworku Django.

2.2.1 Video stream

Pro přenos videa z kamery modulu Raspberry do klienta bylo třeba nastavit video stream.

Pro zprovoznění video streamu byly třeba dvě zásadní části:

1. aplikace pro zachytávání obrazu z kamery Raspberry - v tomto případě integrovaná aplikace raspstill (instalovaná spolu se systémem).
2. nástroj mjpg-streamer určený ke streamování tohoto záznamu po síti. Tento stream je pak možné přehrávat v kompatibilních video přehrávačích či moderních prohlížečích. v tomto nastavení streamer streamuje video na dané IP adrese na portu 9000.

Instalace potřebných balíčků:

```
$ sudo apt-get install libjpeg8-dev
$ sudo apt-get install imagemagick
$ tar xvzf mjpg-streamer-r63.tar.gz
$ cd mjpg-streamer-r63
$ make
```

Aby byl stream iniciován automaticky po startu, byly příkazy pro spuštění streamu umístěny do souboru */etc/init.d/camera_init* a zařazen do automatického spuštění po startu systému.

```
#! /bin/sh
# /etc/init.d/camera_init

case "$1" in
  start)
    echo "Initializing camera"
    mkdir /tmp/stream
    cd /home/pi/mjpg/
    raspistill --nopreview -w 640 -h 480 -q 90 -o /tmp/stream/pic.jpg -tl 500 -t
     9999999 -th 0:0:0 &
    LD_LIBRARY_PATH=./ ./mjpg_streamer -i "input_file.so -f /tmp/stream -n pic.
    jpg"  -o "output_http.so -p 9000 -w ./www" &
    ;;
esac

exit 0
```

Více viz [12] [13]

2.3 Programování

2.3.1 Frontend

Html Jako značkovací jazyk byl použit XHTML 1.0 Transitional.

Javascript S frameworkem jQuery a společně s technologií AJAX zajišťoval dynamičnost rozhraní bez nutnosti reloadu stránek.

Video stream Napojení na video stream bylo implementováno za pomoci Javascriptové funkce, která byla součástí demo stránky projektu $mjpg-streamer$. Funkce načítá postupně jednotlivé snímky a překrývá původní umístěné na stránce.

2.3.2 Backend

Napojení na NXT Pro spojení s NXT s použitím knihovny *python − nxt* je třeba nastavit v konfiguračním souboru */.nxt − python* parametry daného NXT zařízení (především MAC adresu a typ přípojení).

```
[Brick]
host = 00:16:53:1B:7D:2B
strict = 0
method = usb=True, bluetooth=False, fantomusb=False
```

Model robota, metody Ovládací prvky robota byly implementovány v hlavní třídě v souboru */project/main/models.py*. Třída obsahuje inicializaci komunikace s NXT rozhraní, nastavení konstant a jednotlivé metody pro práci s robotem.

Pohledy Další významnou částí kódu jsou pohledy realizující jednotlivé povely. Pohled zpravidla vytvoří instanci pro práci s robotem a zavolá danou metodu.

3 Módy práce

3.1 Manuální mód

V manuálním módu lze ovládat robota přes webové rozhraní jednotlivými povely. Jednotlivé ovládací prvky lze ovládat myší a základní lze ovládat za pomoci klávesnice. Komunikace se serverem je realizována za pomoci systému AJAX (Asynchronous JavaScript and XML), který dovoluje komunikaci se serverem bez nutnosti reloadu stránky. Volání jsou zpravidla dvojího typu, jednofázové a dvoufázové. Jednofázové pro funkce pro jednorázový příkaz, např. příkaz "úchop manipulátorem". Dvoufázové pak pro úkony, které mají začátek a konec, tj. například "pohyb vpřed"
a následně "zastavit". Zde je každá fáze realizována odděleným HTTP requestem.

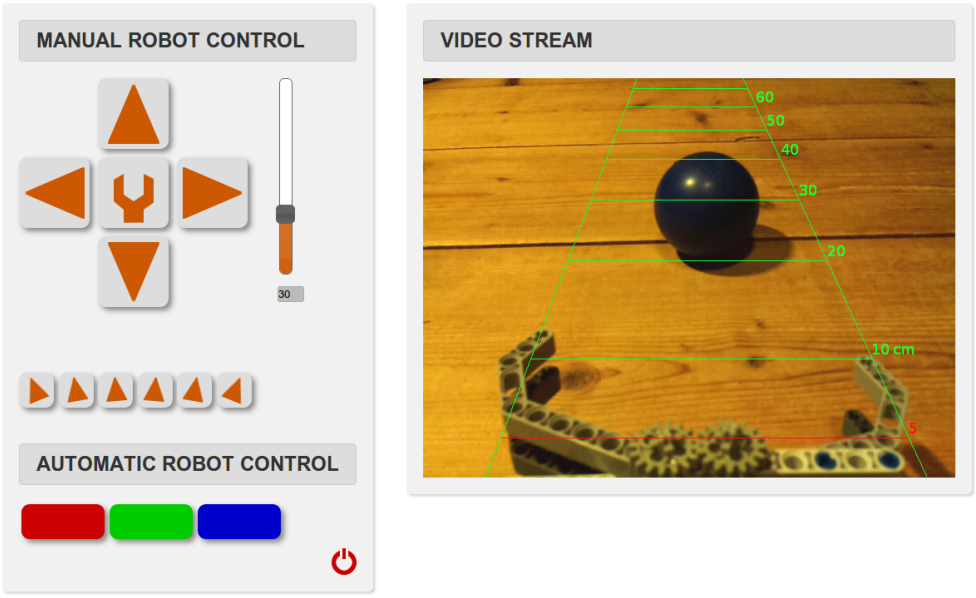
Do uživatelského rozhraní je dále přenášen video stream z kamery snímající prostor před robotem. Úhel naklopení kamery je 30° a snímá prostor od hranice manipulátoru až cca 70cm před robotem. Video stream má rozlišení 640x480px a při kvalitním signálu zpoždění do 1 sekundy při frekvenci okolo 2fps. V přenášeném obraze je také znázorněna mřížka pro lepší orientaci vzdálenosti a šířky robota. Příklad video streamu viz obr. **??**

3.2 Automatický mód

V automatickém režimu robot automaticky vyhledává v obraze snímaném kamerou objekty na základě barvy. Celý proces hledání probíhá iteračně a každá iterace se skládá ze tří základních úkonů:

- Analýza obrazu - nalezení souřadnic objektu ve snímku
- Výpočet úhlu a vzdálenosti od objektu
- Otočení a přesun robota

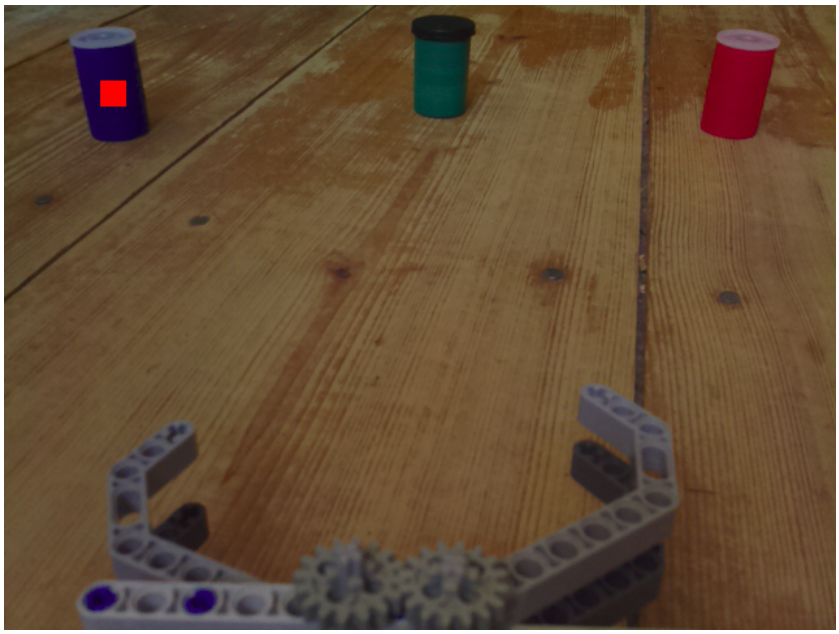
Iterace probíhají dokud robot nedorazí k objektu. Následně je iniciováno uchopení manipulátoru.



Obrázek 3: Ukázka webového rozhraní

3.2.1 Analýza obrazu

Pozice objektu je vypočítávána ze snímku snímaného kamerou, který je načtený ke zpracování ze streamu sloužícím zároveň k přenosu obrazu do ovládacího rozhraní. Ke zpracování je využívána knihovna *SciPy* programovacího jazyka *Python*, která se sadou funkcí podobá programovacímu prostředí *Matlab*.

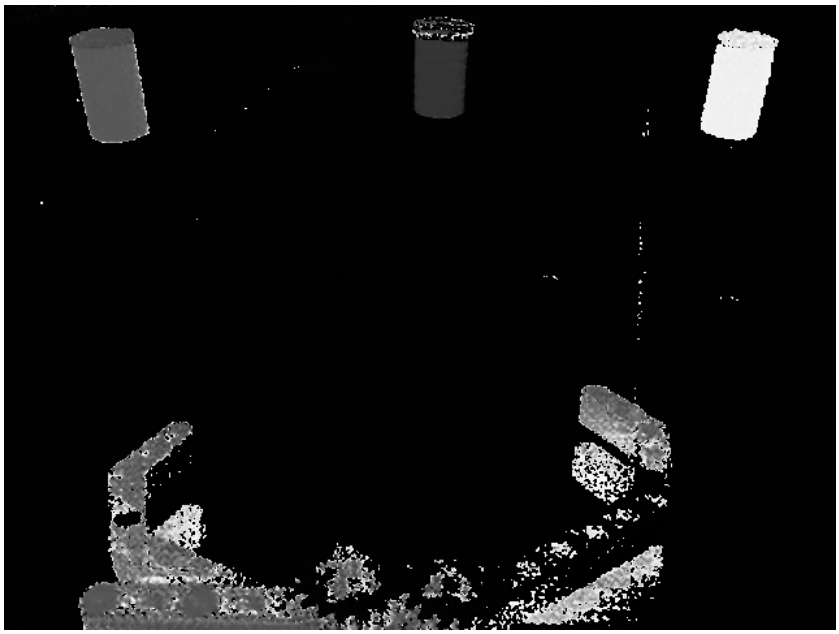


Obrázek 4: Původní snímek se znázorněním vypočtené pozice hledaného objektu

Pro demonstraci postupu byly vytvořeny snímky jednotlivých kroků analýzy. V originálním snímku (obr. 4), na kterém se nacházejí tři objekty různé barvy, byl hledán modrý objekt (v programu jsou předdefinovány 3 barvy - červená, modrá a zelená; lze jednoduše dodefinovat libovolnou sytou barvu, podmínkou je pouze dostatečná odlišnost od ostatních umístěných na snímku). Ve snímku jsou zároveň červeným čtverečkem označeny souřadnice objektu spočtené algoritmem. **Postup analýzy:**

1. **Zmenšení velikosti snímku.** Zmenšení velikosti snímku má za následek zrychlení následujících operací. Velikost originálního načteného snímku je 640 x 480px, zmenšen je pak na 20% původní velikosti. Vzhledem k velikosti objektu na snímku nemá tato ztráta rozlišení vliv na kvalitu analýzy.

   V uvedeném příkladě byla ponechána původní velikost snímku pro lepší názornost. Doba běhu algoritmu na nezmenšeném snímku vzrostla na čtyřicetinásobek.



Obrázek 5: Barevný tón snímku

2. **Převod snímku do barevného modelu HSV.** V tomto barevném modelu je možné snáze identifikovat barevné objekty.

   Ve vrstvě barevného tónu (obr. 5) lze snadno odlišit jednotlivé barevné tóny objektů.

   V kombinaci s vrstvou sytosti (obr. 6) lze identifikovat pouze ty barvy, které jsou dostatečně syté a odfiltrovat například světlé plochy, které můžou mít barevný tón shodný s hledaným. Vrstva jasu (obr. 7) je nejvíce závislá na světelných podmínkách a není pro identifikaci barev použitelná (v uvedeném příkladě je vidět, že jeden z objektů zcela splývá s okolím). Tato vrstva nebyla tedy při analýze nijak využita.

3. **Analýza barevného tónu.** Ve složce barevného tónu snímku je nejprve provedeno prahování podle definovaného rozsahu analyzované barvy. Hodnoty jednotlivých pixelů jsou pak přepočteny podle vzorce

   $$y = 1 - |x - b| \tag{1}$$

   kde b je hodnota hledané barvy a x je hodnota konkrétního pixelu. Výsledná matice tedy obsahuje hodnoty reflektující podobnost s hledaným barevným tónem, tj. čím vyšší hodnota, tím podobnější barevný tón. Pixely mimo zadaný práh jsou pak nulové.

   V příkladě (obr. 8) lze vidět, že kromě hledaného objektu má podobný barevný tón i například část manipulátoru.

4. **Analýza sytosti.** Za účelem identifikace barev s dostatečnou sytostí (např. světlá plocha může mít v závislosti na osvětlení různý barevný tón, ale kvůli vysokému podílu ostatních barev má malou sytost) je složka sytosti prahována hodnotou 0.6 - všechny nižší hodnoty jsou změněny na 0.

   V příkladě (obr. 9) lze vidět, že všechny barevné objekty, včetně hledaného, mají velkou barevnou sytost. Naopak např. manipulátor má sytost nízkou a byl prahováním odfiltrován.

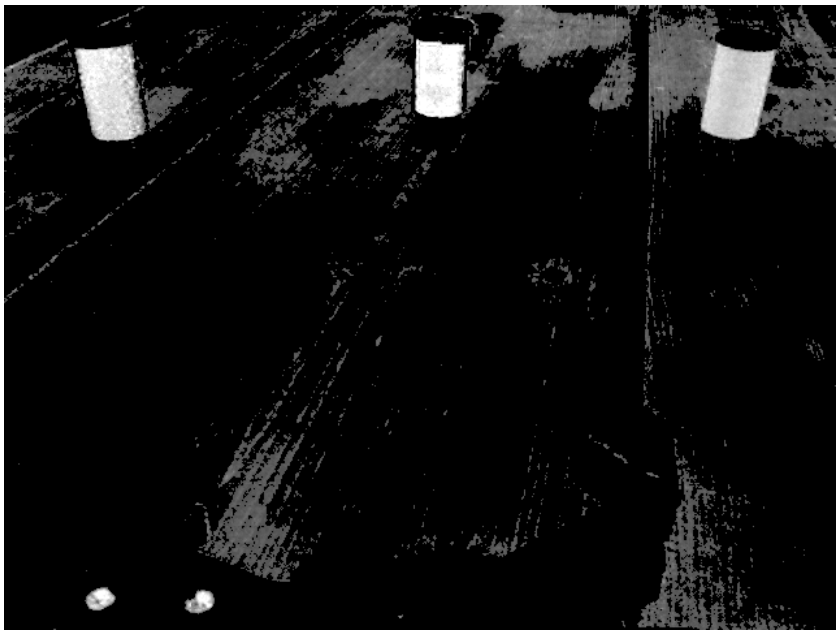
5. **Součin prahovaných vrstev barevného tónu a sytosti.** Tento součin vybírá pixely s dostatečnou sytostí a v rozsahu hledaného barevného tónu, přičemž jednotlivé hodnoty udávají podobnost s hledanou barvou.

   V příkladě (obr. 10) je vidět, že v součinu se nachází pouze hledaný objekt, velmi malé fragmenty (v řádech několika pixelů) a dva kruhovité objekty, které jsou součástí manipulátoru (s velmi podobnou barvou jako hledaný objekt).

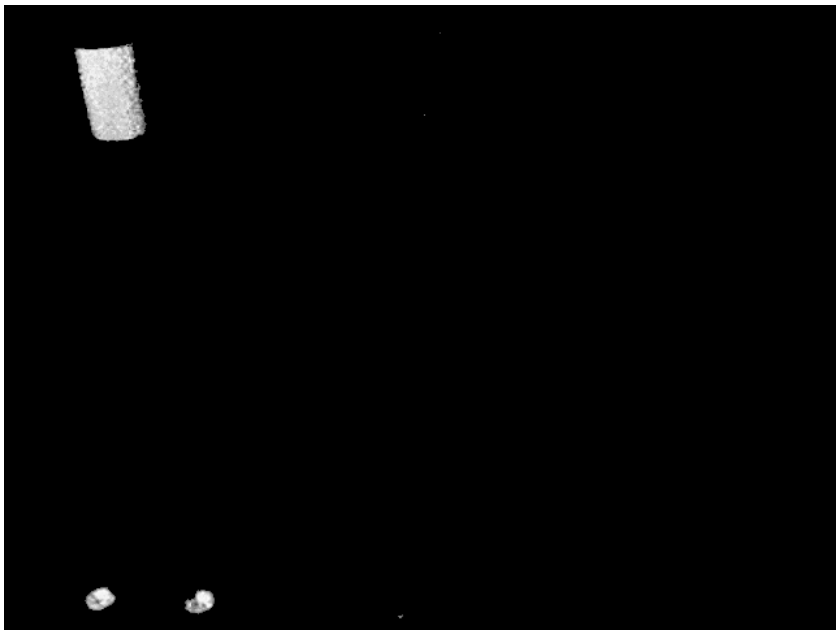
Obrázek 6: Sytost barev snímku

Obrázek 7: Jasová složka snímku

Obrázek 8: Barevný tón snímku po prahování



Obrázek 9: Sytost snímku po prahování



Obrázek 10: Součin prahovaných vrstev barevného tónu a sytosti

6. **Oříznutí obrazu.** Na spodní části snímku se nalézá část manipulátoru, který může působit jako rušivý element snímku a proto je tato část snímku přepsána nulovými hodnotami.

   V příkladě (obr. 11) jsou tyto větší rušivé elementy odstraněny.

7. **Morfologické úpravy snímku.** Za účelem odstranění nežádoucích malých objektů ve snímku a ucelení hledaného objektu jsou na snímek aplikovány operace uzavření a otevření strukturním elementem 3x3px (při zmenšení na 20%).

   V příkladě (obr. 12) je vidět již pouze hledaný objekt, všechny ostatní elementy byly odstraněny.

8. **Vypočtení těžiště snímku.** Z analyzovaného snímku je vypočteno těžiště snímku, které odpovídá poloze hledaného objektu. Pokud by ve snímku i po analýze zbyly některé jiné menší fragmenty, než hledaný objekt, vypočtené souřadnice by se díky výpočtu těžiště přibližovaly blíže hledanému objektu.

   Vypočtené souřadnice v příkladu jsou znázorněné v úvodním originálním snímku (obr. 4).

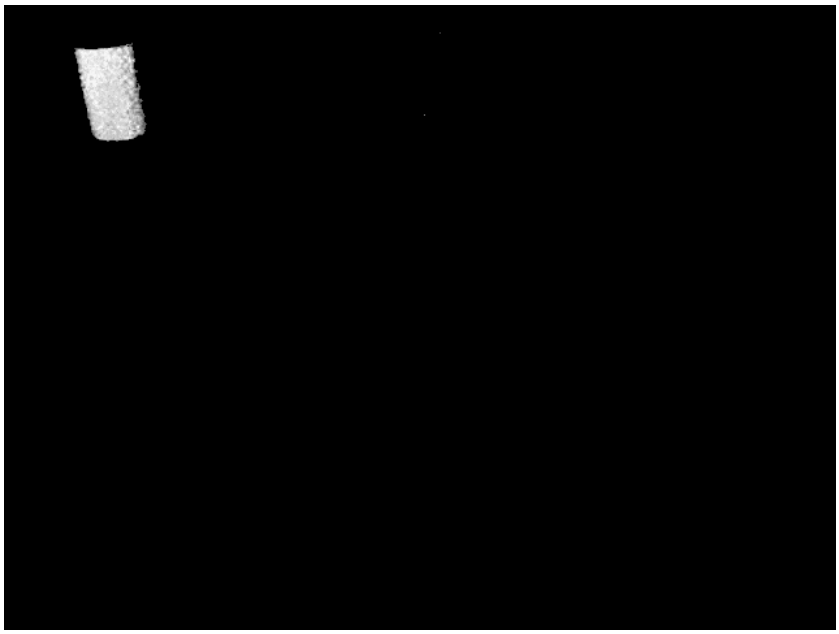
Analýza předpokládá umístění pouze jednoho objektu dané barvy ve snímku. v případě dvou nebo více objektů bude spočtený vektor pro pohyb směřovat do jejich společného těžiště.

Pokud se ve snímku nenachází ani jeden dané barvy, proces automatického přiblížení skončí.

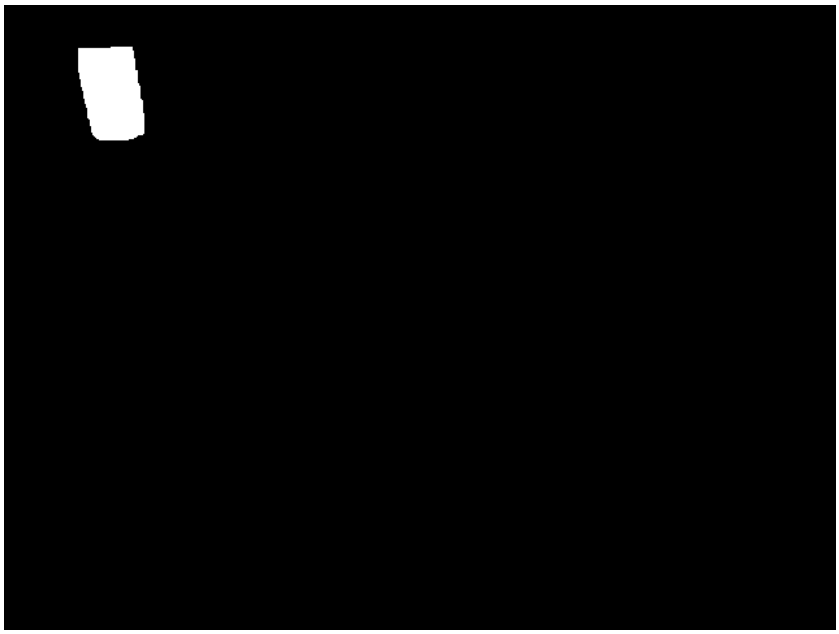
3.2.2 Výpočet úhlu a vzdálenosti od objektu

Na základě souřadnic objektu vypočtených analýzou obrazu je počítán úhel a vzdálenost objektu od robota. Očekává se, že objekt je umístěn ve stejné rovině, po které se robot pohybuje.

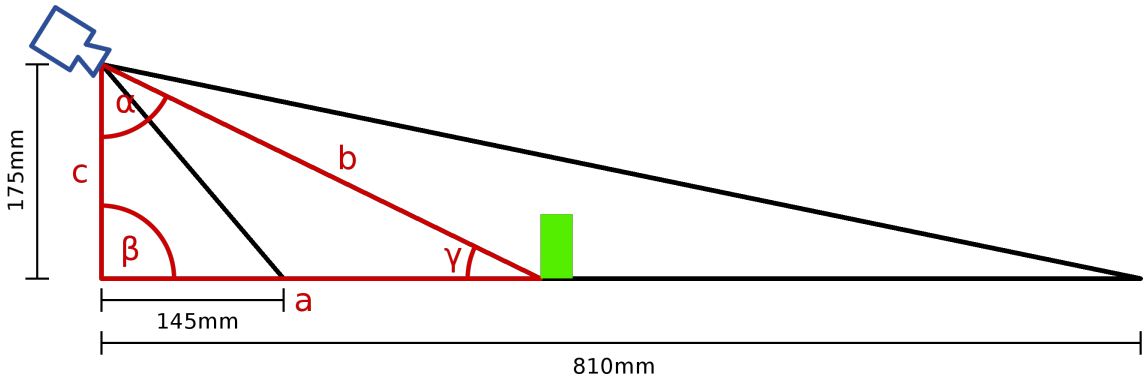
Díky znalosti úhlu pod kterým je kamera nasměřována k zemi, tzn. i vzdálenosti nejbližšího a nejvzdálenějšího snímaného bodu a výšky kamery nad zemí lze spočítat jednak vzdálenost objektu od robota po ploše (vzdálenost a) a také přímou vzdálenost objektu od kamery (vzdálenost b), viz obr. 13. Úhel α je spočten podle pozice hledaného bodu na snímku; pokud je umístěn zcela dole, úhel odpovídá minimální hodnotě zorného pole kamery, pokud je zcela nahoře, odpovídá maximu zorného pole.



Obrázek 11: Součin prahovaných vrstev barevného tónu a sytosti po oříznutí



Obrázek 12: Analyzovaný snímek po aplikaci mat. morfologie



Obrázek 13: Výpočet pozice objektu - řez

Vzdálenost a lze spočítat za použití sinové věty, podle které platí:

$$\frac{a}{sin(\alpha)} = \frac{c}{sin(\gamma)} \tag{2}$$

Po vyjádření proměnné a, dosazení $\gamma = 180 - \beta - \alpha$ a dosazení konkrétních číselných hodnot dostáváme:

$$a = \frac{17.5 * sin(\alpha)}{sin(90 - \alpha)} \tag{3}$$

Vzdálenost b pak lze jednoduše dopočítat za pomoci Pythagorovy věty.

Dalším krokem je výpočet vzdálenosti h znázorněné na obr. 14. Na obrázku je znázorněn pohled na situaci shora, kde znázorněný řez neleží v ploše po které se robot pohybuje, ale prochází kamerou robota. Úsečka b odpovídá stejnojmenné úsečce zobrazené na obr. 13. Při znalosti délky této úsečky a úhlu β, spočteném obdobně jako v předchozím odstavci úhel $alfa$, lze taktéž obdobným způsobem za použití sinové věty z rovnice 2 spočíst proměnnou h určující vzdálenost objektu od osy robota.

Po vyjádření proměnné h tedy dostáváme:

$$h = \frac{b * sin(\beta)}{sin(90 - \beta)} \tag{4}$$

Předchozími dvěma výpočty byla spočtena vzdálenost a, tj. vzdálenost kamery robota ke kolmici na osu robota procházející objektem a vzdálenost h, tj. vzdálenost objektu po této kolmici od osy robota. Dále je třeba dopočítat dvě hodnoty, které budou použity pro pohyb robota: úhel otočení a vzdálenost, o kterou se má robot posunout vpřed. Při výpočtu je třeba vzít do úvahy dva fakty. V prvé řadě osa otáčení robota se nenachází na místě, kde je připevněna kamera, ale 7 cm za ním (v obr. 15 je tato vzdálenost označena jako rco a rco'). Dále vzdálenost od objektu není počítána ke kameře, ale pouze k manipulátoru, který naopak vyčnívá před kameru o 12 cm (v obr. 15 jako mo).

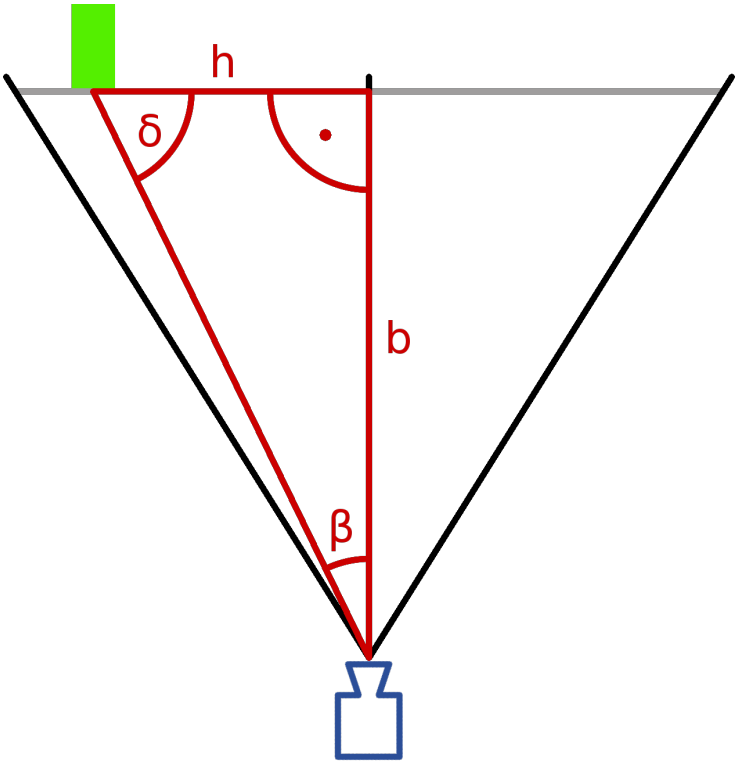
Prvně je třeba spočítat vzdálenost, kterou má robot urazit. Z obr. 15 je jasně vidět, že při znalosti vzdálenosti h (tedy C) a vzdálenosti a a rco (tedy A) lze jednoduše Pythagorovou větou spočítat vzdálenost B, ze které lze dopočítat po odečtení mo a rco vzdálenost $dist$, která je skutečnou vzdáleností, kterou je třeba po otočení urazit.

Dále pro výpočet úhlu otočení γ můžeme použít kosinovu větu:

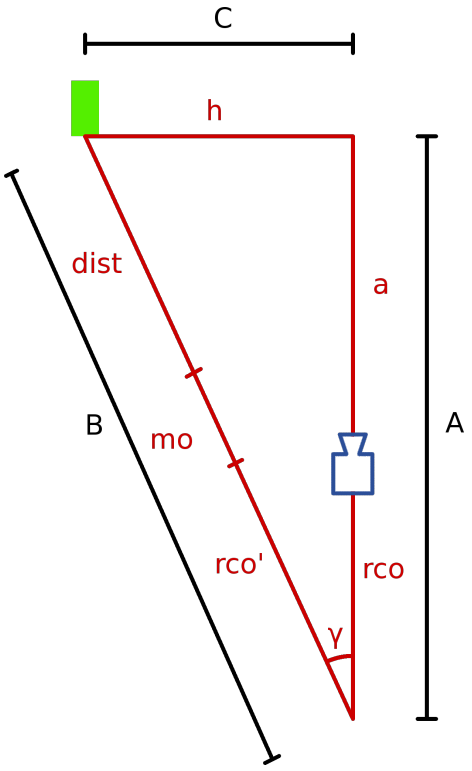
$$c^2 = b^2 + a^2 - 2ab * cos(\gamma) \tag{5}$$

Po vyjádření úhlu γ dostáváme při použití proměnných A, B, C z obr. 15:

$$\gamma = \arccos\left(-\frac{C^2 - B^2 - A^2}{2AB}\right) \tag{6}$$



Obrázek 14: Výpočet pozice objektu - pohled shora



Obrázek 15: Výpočet pozice objektu - řez

Tím je spočtený úhel pro otočení γ a vzdálenost $dist$.

3.2.3 Otočení a přesun robota

Na základě spočteného úhlu a vzdálenosti od objektu je spuštěna metoda $turn\_left$ nebo $turn\_right$ s úhlem o který se má robot otočit jako vstupním parametrem. Následuje metoda $move$ se vzdáleností v cm jako vstupním parametrem.

Aby byly co možná nejvíce eliminovány nepřesnosti (při analýze, při otáčení, při pohybu vpřed), robot neurazí na začátku celou vzdálenost k objektu, ale pouze 40%. Poté následuje nová iterace s novou analýzou, která má za účel další zpřesnění směru a vzdálenosti. Až po dosažení definované vzdálenosti od objektu robot provede poslední přiblížení, při kterém urazí celkovou zbývající vzdálenost.

Vzhledem k zornému poli robota se může hledaný objekt nacházet maximálně 70 cm daleko. Tomu odpovídají 3-4 přiblížení. Pokud se objekt nachází blíže, může jich být úměrně méně, v extrémním případě pouze jedno, finální přiblížení.

4 ZÁVĚR

Modul Raspberry Pi byl použit jako řídící modul pro mobilního bezdrátového robota, schopného pohybu vpřed, vzad a otáčení kolem osy. Zkonstruovaný manipulátor umožňuje transport předmětu vhodných rozměrů a hmotnosti.

Naprogramovaná webová aplikace umožňuje ovládání robota přes bezdrátovou technologii WiFi z libovolného zařízení se standardním webovým prohlížečem. Toto zařízení musí být umístěné ve stejné lokální síti jako řídící modul nebo musí mít jiným způsobem zařízený přístup k síťovému rozhraní modulu. Podmínkou komfortního použití je také dostatečně kvalitní spojení. Součástí vytvořené aplikace je jednoduše použitelné webové rozhraní umožňující ovládání všech funkcí robota (viz kapitola 3.1).

Mimo manuálního ovládání má robot také automatický mód, při kterém vyhledává zabudovanou kamerou za pomoci obrazové analýzy objekt zadané barvy, vypočítá jeho vzdálenost a odchylku od osy robota a iteračně se k objektu přiblíží a manipulátorem uchopí. K výběru jsou tři různé barvy (červená, zelená a modrá) a systém umožňuje jednoduché dodefinování libovolné další syté barvy. Více viz kapitola 3.2.

Reference

[1] Microsoft .NET framework [online]. [cit. 2014-05-08]. Dostupné z: http://www.netmf.com/what-is-the-net-micro-framework.aspx

[2] Arduino [online]. [cit. 2014-05-25]. Dostupné z: http://www.arduino.cc/

[3] Beagle Bone [online]. [cit. 2014-05-20]. Dostupné z: http://beagleboard.org/Products/BeagleBone+Blac

[4] Raspberry Pi Dokumentace [online]. [cit. 2014-04-08]. Dostupné z: http://www.raspberrypi.org/documentation/

[5] Django documentation [online]. [cit. 2014-04-11]. Dostupné z: http://goo.gl/4o683Q

[6] JQuery framework [online]. [cit. 2014-05-12]. Dostupné z: http://jquery.com/

[7] Matematická morfologie - M.Mudrová [online]. [cit. 2014-05-28]. Dostupné z: http://uprt.vscht.cz/mudrova/zob/prednasky/10-MORFOLOGIE/morfologie.pdf

[8] An-Chyau,H.,Ming-Chih,Ch.,Adaptive Control of Robot Manipulators, World Scientific, New Jersey, 2010

[9] instalace rasbian [online]. [cit. 2014-03-01]. Dostupné z: http://goo.gl/Q6GHr9

[10] HTTPD - Apache2 Web Server [online]. [cit. 2014-03-05]. Dostupné z: http://goo.gl/XWZSC9

[11] How to enable mod_rewrite in Apache2 on Debian or Ubuntu [online]. [cit. 2014-03-10]. Dostupné z: http://goo.gl/TjJwM5

[12] Raspberry Pi camera board video streaming [online]. [cit. 2014-03-20]. Dostupné z: http://goo.gl/9lBGbI

[13] How to automatically start a program on boot in Debian [online]. [cit. 2014-03-25]. Dostupné z: http://goo.gl/6yOWlF