

Bakalářská práce



České  
vysoké  
učení technické  
v Praze

**F3**

Fakulta elektrotechnická  
Katedra kybernetiky

## Následování člověka mobilním robotem

Mykhaylo Zelenskyy

Školitel: Ing. Jan Chudoba  
Březen 2017



## Poděkování

Chtěl bych poděkovat svému vedoucímu, Ing. Janu Chudobovi, za jeho rady a nápady.

## Prohlášení

I declare that this work is all my own work and I have cited all sources I have used in the bibliography.

Prague, March 25, 2017

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

V Praze, 25. března 2017

## Abstrakt

Tento manuál představuje L<sup>A</sup>T<sub>E</sub>Xovou třídu ctuthesis, její použití, požadavky na systém atd.

**Klíčová slova:** manuál, závěrečná práce, L<sup>A</sup>T<sub>E</sub>X

**Školitel:** Ing. Jan Chudoba

## Abstract

This manual shows how to use the ctuthesis L<sup>A</sup>T<sub>E</sub>X class, what are the requirements, etc.

**Keywords:** manual, degree project, L<sup>A</sup>T<sub>E</sub>X

## Obsah

<b>1 Úvod</b>	<b>1</b>
<b>2 Návrh systému</b>	<b>3</b>
2.1 Volba vizuální značky .....	3
2.2 Volba souřadnicového systému...	3
<b>3 Detekce</b>	<b>5</b>
3.1 Zpracování obrazu z kamery ....	5
3.2 Rozpoznávání vzoru .....	5
3.2.1 Korelační algoritmus .....	5
3.2.2 ArUco marker detektor .....	7
3.3 Měření vzdáleností .....	8
3.4 Kalmanův filtr .....	9
<b>4 Řízení</b>	<b>13</b>
4.1 Obstacle avoidance .....	13
4.1.1 Vector Field Histogram .....	13
<b>5 Výsledky simulací</b>	<b>17</b>
<b>6 Závěr</b>	<b>19</b>

## Obrázky

## Tabulky

2.1 Příklad vizuální značky . . . . .	3
2.2 Použitý souřadnicový systém . . . .	4
3.1 Příklad použití adaptivního prahování . . . . .	6
3.2 Hierarchie (topologie) obrazu . . . .	6
3.3 Příklad korelace mezi dvěma veličinami . . . . .	7
3.4 Aproximace křivky pomocí Ramerova Douglasova Peuckerova algoritmu . . . . .	8
3.5 Buňky ArUco markeru . . . . .	8
3.6 Vyhodnocení obrázku pomocí ArUco detektoru . . . . .	9
3.7 Měření veličiny $d$ a $\phi$ . . . . .	10
3.8 Shrnutí algoritmu Kalmanova filtru . . . . .	11
3.9 Odvození hodnot $d$ a $\phi$ ze známé polohy cíle . . . . .	12
4.1 Lokální mapa prostředí o poloměru $r_{map} = 10$ . . . . .	14
4.2 Polární histogram s vyobrazenými prahy $\tau_{low}$ (zeleně) a $\tau_{high}$ (fialově) .	15

# Kapitola 1

## Úvod

Pomocný robot může se zdát mnoha z nás jako futuristický sen. Takový robot, který by místo nás nosil věci, pomáhal při nákupech, asistoval v nemocnicích nebo ošetřoval raněné ve válkách. Podobný robot by měl tolik výhod, že by v budoucnu to bylo trendem vlastnit takového pomocníka.

Bylo provedeno hodně různých výzkumů ohledně návrhu robotu, který by dokázal sledovat člověka. S tímto problémem jsou spojené dvě důležité otázky. První je jaké senzory použít pro lokalizaci člověka, druhá řeší řízení a navigaci robotu, aby udržoval určitou vzdálenost a vyhýbal se případným překážkám.

Nejdříve je třeba definovat, co vlastně je robot následující člověka. Jedná se o mobilního robota, který sleduje určitou osobu a zároveň objíždí překážky a chová se k ostatním lidem, jako k překážkám, tj. nezmění cíl sledování během jízdy.

Takový robot typicky může být vybaven hodně různými senzory, např. laserovým, zvukovým nebo IČ dálkoměrem, aby mohl měřit vzdálenost od překážek, kamerou pro detekci sledovaného objektu, bezdrátovým přenáčem signálu, GPS apod. Tyto senzory by měly fungovat současně, aby robot dokázal všechno, co se od něj očekává.

Cílem této práce je navrhnout systém pro řízení robotu, aby dokázal sledovat člověka a vyhýbat se překážkám. Jelikož cíl pro robota by měl být unikátní, použije se vizuální značka, která by tento problém měla vyřešit. Robot dopředu bude vědět, jaký typ značky musí hledat, tím pádem se snižuje šance, že si splete cíl s jiným člověkem nebo překážkou.

Při návrhu tohoto systému je třeba uvědomit si několik věcí:



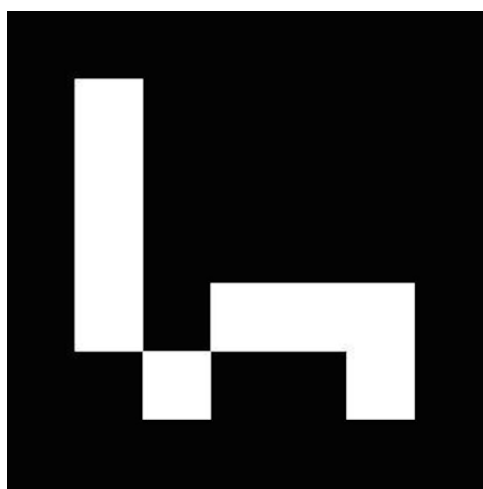


## Kapitola 2

### Návrh systému

#### 2.1 Volba vizuální značky

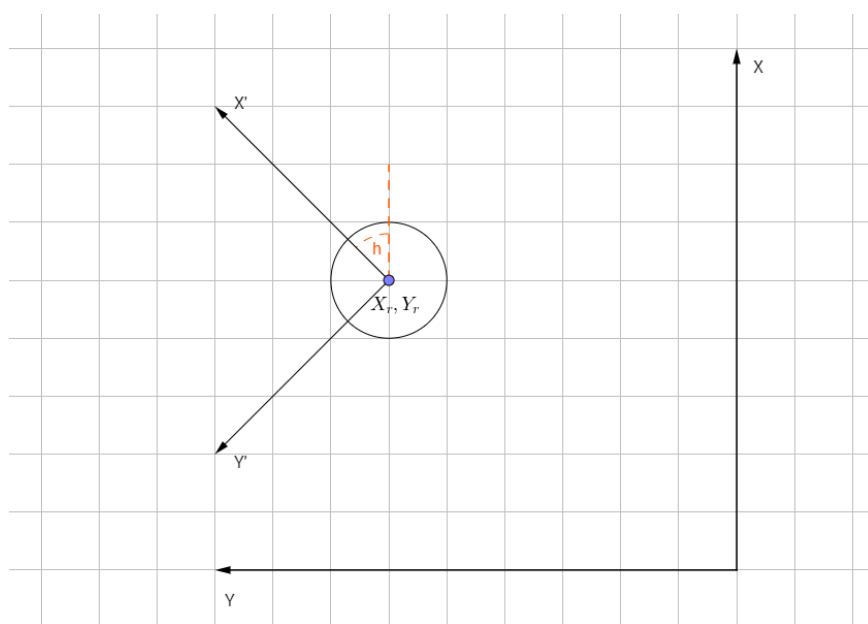
Jak již bylo zmíněno, mobilní robot bude následovat člověka, který má na sobě umístěnou předem známou vizuální značku. Pro tyto účely se v praxi běžně používají markery pro rozšířenou realitu. Jejich detekce je většinou jednoduchá a můžou v sobě uchovávat užitečnou informaci, např. identifikační číslo, podle kterého robot pozná, koho sleduje. Jednu z možných vizuálních značek, které jsou použity v práci, lze vidět na obrázku 2.1.



Obrázek 2.1: Příklad vizuální značky

#### 2.2 Volba souřadnicového systému

Pro řízení robotu je třeba zvolit souřadnicový systém, ve kterém se bude pohybovat. Z obrázku 2.2 je patrné, že je pohyb ve zvoleném souřadnicovém systému prováděn ve směru osy  $X'$



**Obrázek 2.2:** Použitý souřadnicový systém

## Kapitola 3

### Detekce

#### 3.1 Zpracování obrazu z kamery

Než bude možné použít rozpoznávací algoritmy, výstup z kamery musí být náležitě zpracován, aby odpovídal formátu, se kterým algoritmy pracují.

Načtený snímek je převeden do černobílé podoby, je tedy použito prahování. Jelikož se předpokládá, že se robot může pohybovat v prostředí s nerovnoměrným osvětlením, pro binarizaci obrazu je vhodné použít adaptivní prahování 3.1. Tato metoda počítá práh pro malé části obrazu místo globálního nastavení prahu pro celý snímek.

Dále je třeba najít kontury. Algoritmus pro nalezení kontur je implementován v OpenCV knihovně. Tato funkce navíc zajišťuje zachování hierarchii kontur, tedy topologii obrázku, jako je vidět na 3.2. Tato informace je užitečná pro následující rozpoznávání, protože pak kontury na nejnižší nebo nejvyšší úrovni hierarchie můžou být ignorovány, pokud se předpokládá, že vizuální značka bude umístěna v ochranné zóně a bude obsahovat další pod-vzory.

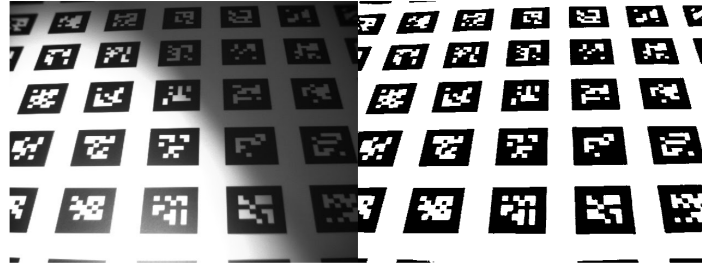
#### 3.2 Rozpoznávání vzoru

Pro rozpoznávání vizuální značky byly zvoleny dva algoritmy: jeden založený na korelaci známých vzorů s nalezeným, druhý přímo z OpenCV knihovny – ArUco marker detektor.

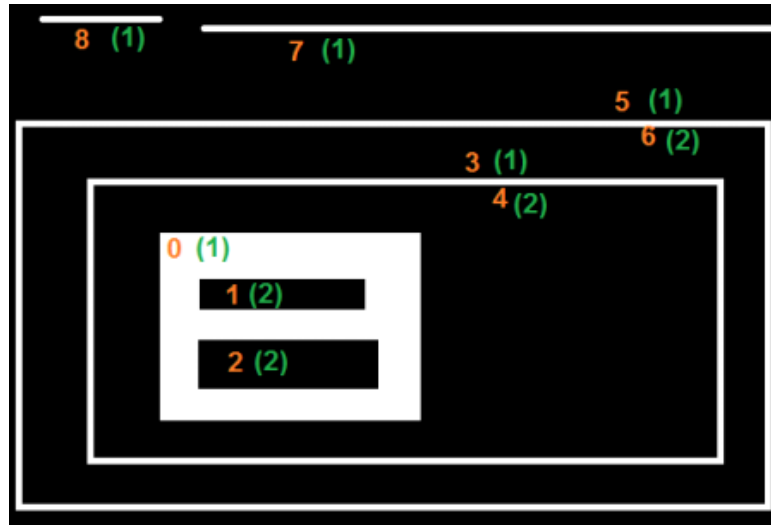
##### 3.2.1 Korelační algoritmus

Tento algoritmus využívá principu korelaci mezi dvěma veličinami, tedy mezi známým vzorem a nalezenou oblastí zájmu (Region of Interest, ROI). Korelace uvádí, jak jsou na sobě tyto veličiny závislé, a může nabývat hodnot od -1 do +1. Hodnota korelačního koeficientu blížící se -1 značí nepřímou závislost veličin, koeficient blížící +1 naopak značí přímou závislost, což je zobrazeno na 3.3.

Hledaná ROI v zjednodušeném případě je konvexní čtyřúhelník, proto stačí s použitím informace o topologii obrazu spočítat polygony nalezených kontur a vybrat pouze ty, co obsahují čtyři strany a nejsou konkávní. Pro aproximaci



Obrázek 3.1: Příklad použití adaptivního prahování



Obrázek 3.2: Hierarchie (topologie) obrazu

křivky se používá Ramerův Douglasův Peuckerův algoritmus (viz 3.4), který je naimplementován v OpenCV knihovně.

Následně je spočítána transformační matice mezi nalezeným polygonem a známým vzorem. Pomocí této matice se polygon převede do takové podoby, aby byl porovnatelný se vzorem.

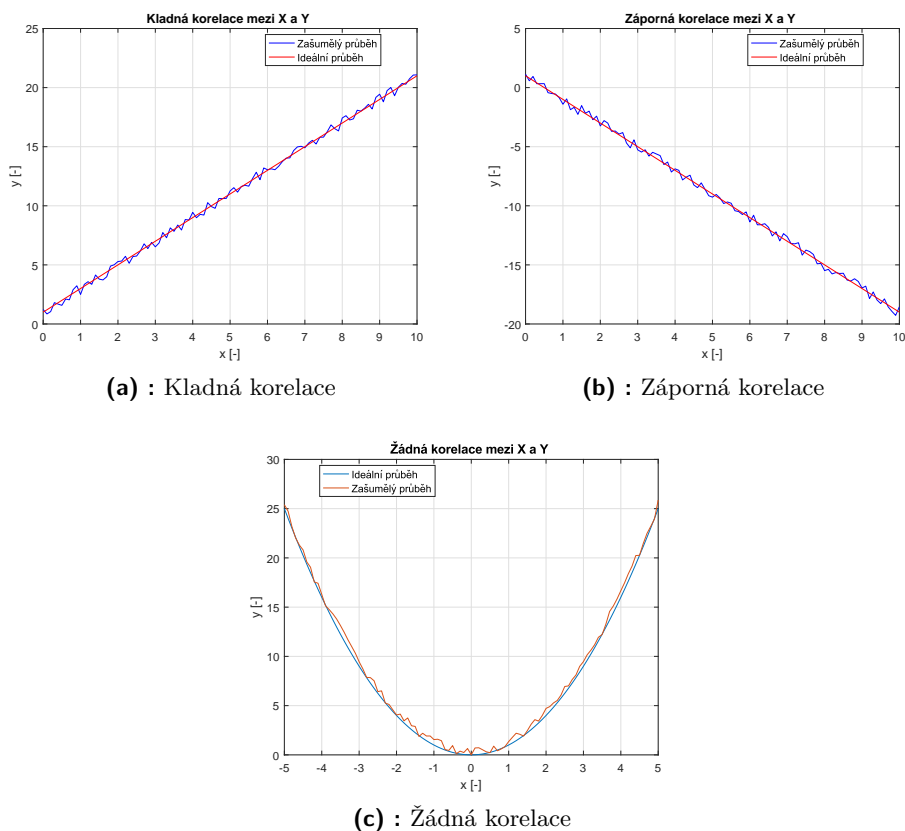
Dále se jenom vyhodnotí, jak nalezená ROI a známý vzor jsou na sobě závislé. Pokud je vypočtený korelační koeficient větší, než zadaný práh, ROI se vyhodnotí jako správně označena a algoritmus vrátí souřadnice jejích rohů pro následující zpracování.

Korelaci mezi dvěma obrázky  $I_1$  a  $I_2$  s  $N$  pixely spočítáme následujícím způsobem:

$$\mu_{1,2} = \frac{\sum_{i,j} I_{i,j}}{N}, \quad (3.1)$$

$$\sigma_{1,2} = \sqrt{\left( \frac{\sum_{i,j} (I_{i,j} - \mu_{1,2})^2}{N} \right)^2}, \quad (3.2)$$

$$\text{covar}(I_1, I_2) = \frac{(I_1 - \mu_1) \cdot (I_2 - \mu_2)}{N}, \quad (3.3)$$



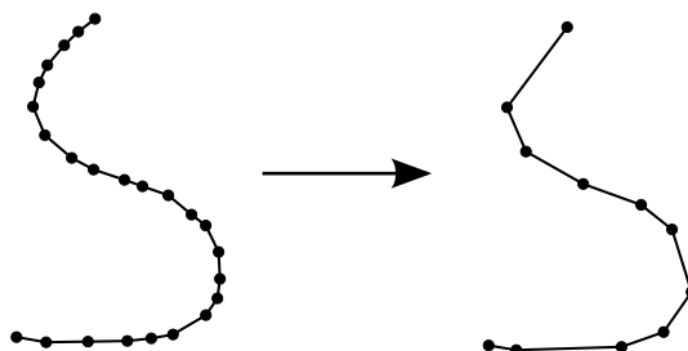
Obrázek 3.3: Příklad korelace mezi dvěma veličinami

$$\text{cor}(I_1, I_2) = \frac{\text{covar}(I_1, I_2)}{\sigma_1 \sigma_2}. \quad (3.4)$$

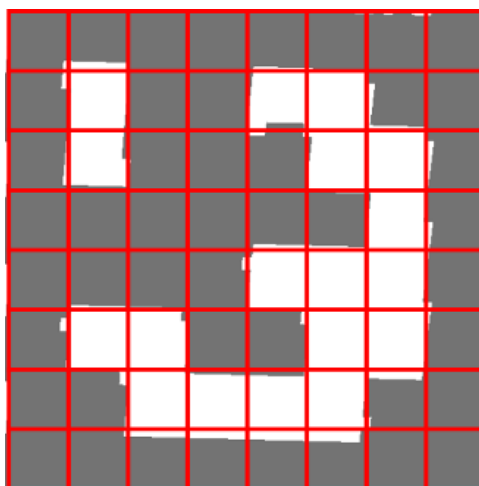
### 3.2.2 ArUco marker detektor

Tento algoritmus je naimplementován v OpenCV knihovně a umožňuje rozpoznávání ArUco markerů nebo podobných vizuálních značek. Podobně jako výše uvedený korelační algoritmus, hledá konvexní čtyřúhelník. Rozdíl pak spočívá v odlišném zpracování nalezené ROI, kde se místo korelace využívá binární mapa kandidáta.

ROI je rozdělena na mřížku s počtem buněk rovným počtu bitů hledaného vzoru (viz 3.5). Následně se spočítá, kolik bílých a černých pixelů obsahuje každá buňka, na základě čehož se vyhodnotí, jestli buňka je černá nebo bílá (viz 3.6). Pokud detektor ve svém slovníku obsahuje vzor se stejnou binární maskou, vrátí souřadnice rohů nalezené ROI.



**Obrázek 3.4:** Aproximace křivky pomocí Ramerova Douglasova Peuckerova algoritmu



**Obrázek 3.5:** Buňky ArUco markeru

### 3.3 Měření vzdáleností

Algoritmy popsané výše naleznou polohu vizuální značky v obraze, z čehož lze spočítat její umístění vůči kameře, je-li známa velikost této značky. Pro výpočty lze použít model ideální (dírkové) kamery, kde vzdálenost mezi kamerou a značkou je vyjádřena jako

$$d = \frac{fX}{x}, \quad (3.5)$$

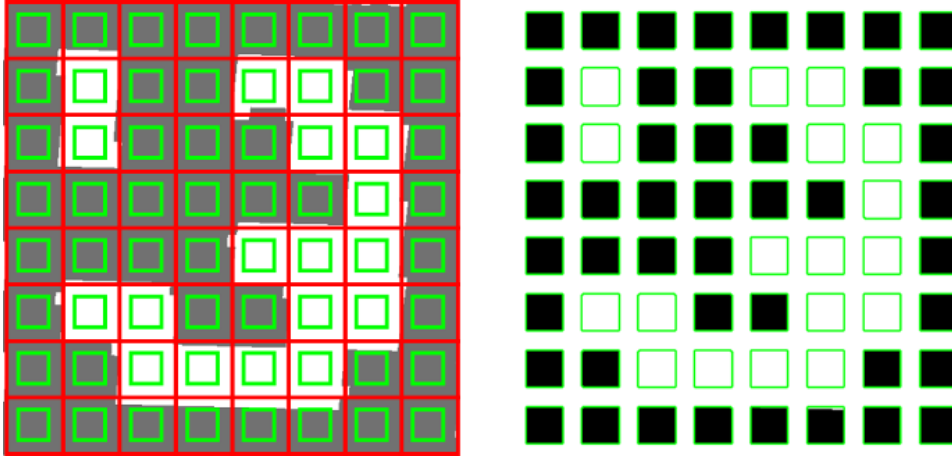
kde

$x$ : Nejkratší vzdálenost mezi dvěma rohy nalezené značky.

$f$ : Ohnisková vzdálenost.

$X$ : Znamá délka hrany značky.

Pro nalezení posunutí značky vůči kameře při odklonění od její osy, musí být znám zorný úhel kamery. Ten lze spočítat dle vztahu



**Obrazek 3.6:** Vyhodnocení obrázku pomocí ArUco detektoru

$$\alpha = 2 \arctan \left( \frac{w}{2f} \right), \quad (3.6)$$

kde

$w$ : Šířka výstupního obrazu z kamery.

Potom úhel, který pokrývá jeden pixel snímku, je vyjádřen vztahem

$$APP = \frac{\alpha}{w}. \quad (3.7)$$

Následně posunutí objektu vůči středu kamery je

$$\phi = (x_c - x_t)APP, \quad (3.8)$$

kde

$x_c$ : Pixel vyjadřující střed kamery v horizontálním směru.

$x_t$ : Pixel vyjadřující střed nalezené značky v horizontálním směru.

### 3.4 Kalmanův filtr

V případě, že poloha značky vůči kameře není detekovatelná, robot by neměl zastavovat. Z tohoto důvodu je třeba predikovat polohu cíle z předchozích měření. Pro tyto účely v práci je použit Kalmanův filtr.

Model lze popsat pomocí následujícího stavového vektoru

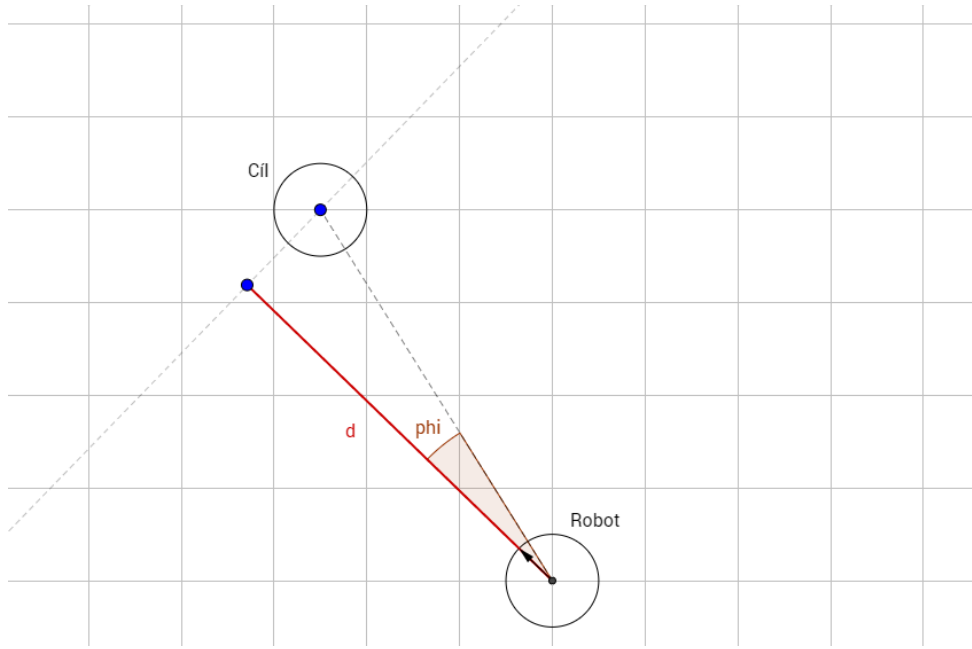
$$\mathbf{x} = \begin{bmatrix} x & y & v_x & v_y \end{bmatrix}^T, \quad (3.9)$$

kde  $v_{lin}$ , resp.  $v_{ang}$ , značí rychlost ve směru osy X, resp. Y.

Stavový model je pak vyjádřen jako

$$\mathbf{x}_{t+1} = \mathbf{F}\mathbf{x}_t, \quad (3.10)$$

kde  $\mathbf{F}$  je matice přechodu stavů  $\mathbf{x}$  z času  $t$  do času  $t + 1$  a platí, že

Obrázek 3.7: Měřené veličiny  $d$  a  $\phi$ 

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & dt & 0 \\ 0 & 0 & 0 & dt \end{bmatrix}. \quad (3.11)$$

Pro sledování cíle lze Kalmanův filtr rozdělit do tří kroků:

**1. Inicializace** ( $t = 0$ ). Během tohoto kroku se nastavuje počáteční pozice cíle  $\mathbf{x}_0$  a výchozí hodnota pro kovarianční matici  $\mathbf{P}_0$ . Jelikož počáteční pozice cíle nemusí být známa, je vhodné zvolit  $\mathbf{x}_0$  tak, aby v případě, že kamera nedetekuje značku po první iteraci, robot zůstal na místě.

**2. Predikce** ( $t > 0$ ). V tomto kroku se provádí predikce polohy cíle v čase  $t + 1$ , tj.  $\mathbf{x}_{t+1}$ , dle 3.10. Také se počítá nová kovarianční matice dle následujícího vztahu

$$\mathbf{P}_{t+1} = \mathbf{F}\mathbf{P}_t\mathbf{F}^T + \mathbf{Q}_{t+1}, \quad (3.12)$$

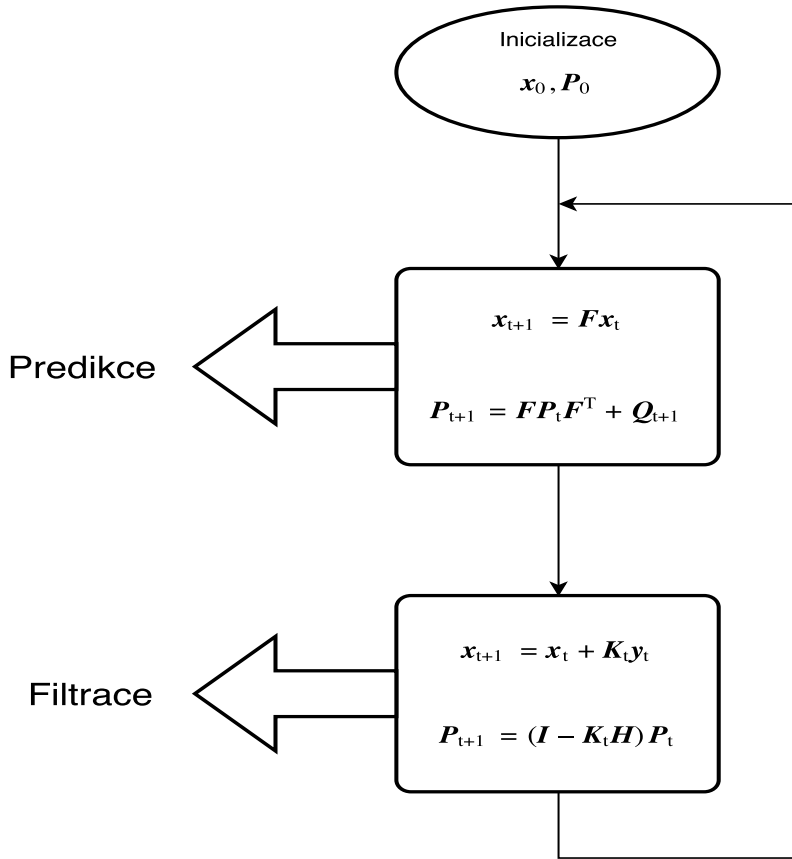
kde  $\mathbf{Q}$  značí matici kovariancí šumů.

**3. Filtrace** ( $t > 0$ ). Během tohoto kroku poloha cíle je upřesněna na základě provedeného měření. Nejdříve se spočte rozdíl mezi reálnou polohou a predikovanou v kroku 2.:

$$\mathbf{y}_t = \begin{bmatrix} x_{m_t} \\ y_{m_t} \end{bmatrix} - \mathbf{H}\mathbf{x}_t, \quad (3.13)$$

kde





Obrázek 3.8: Shrnutí algoritmu Kalmanova filtru

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}. \quad (3.14)$$

Dále se vypočítá Kalmanovo zesílení, pro nějž platí, že

$$\mathbf{K}_t = \mathbf{P}_t \mathbf{H}^T (\mathbf{H} \mathbf{P}_t \mathbf{H}^T + \mathbf{R})^{-1}, \quad (3.15)$$

kde  $\mathbf{R}$  je matice kovariancí šumů.

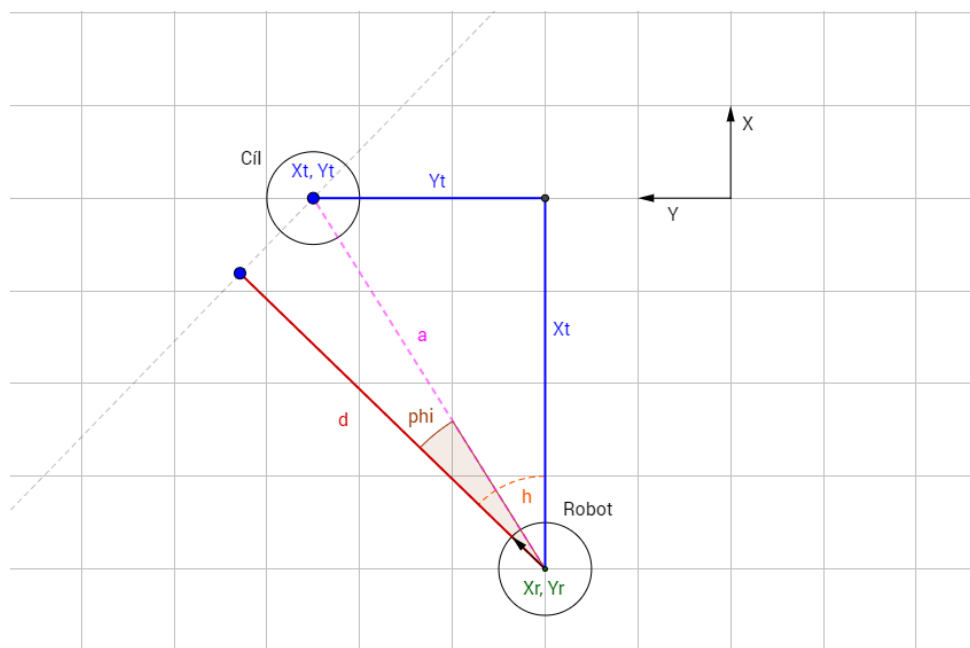
Následně se provede zpřesnění polohy cíle a aktualizuje se kovarianční matice:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{K}_t \mathbf{y}_t, \quad (3.16)$$

$$\mathbf{P}_{t+1} = (\mathbf{I} - \mathbf{K}_t \mathbf{H}) \mathbf{P}_t. \quad (3.17)$$

Z nalezených hodnot  $x$  a  $y$  lze jednoduše odvodit hodnoty  $d$  a  $\phi$ , což je také vidět na 3.9. Jelikož se předpokládá, že poloha robotu je známa, lze spočítat  $x_t$  a  $y_t$ , což je poloha cíle vůči robotu, jako

$$x_t = x - x_r, \quad (3.18)$$



**Obrázek 3.9:** Odvození hodnot  $d$  a  $\phi$  ze známé polohy cíle

$$y_t = y - y_r. \quad (3.19)$$

Dále pak  $\phi$  a  $d$  jsou nalezeny dle vztahů

$$\phi = h - \text{atan2}(y_t, x_t), \quad (3.20)$$

$$d = \sqrt{x_t^2 + y_t^2} \cos(\phi). \quad (3.21)$$

## Kapitola 4

### Řízení

#### 4.1 Obstacle avoidance

Pro řízení autonomního robotu je důležité, aby se dokázal vyhýbat překážkám, když sleduje svůj cíl.

##### 4.1.1 Vector Field Histogram

Princip tohoto algoritmu spočívá v tom, že se na základě naměřených dat z dálkoměru vytvoří lokální mřížková mapa prostředí o poloměru  $r_{map}$ . Každá buňka této mapy tak obsahuje hodnotu obsazenosti odpovídající oblasti v reálném světě (viz 4.1). Z této mapy se následně spočítá polární histogram, podle kterého se určí směr jízdy robotu.

Celý proces tímto způsobem můžeme rozdělit na tři kroky: vytvoření primárního polárního histogramu, prahování primárního histogramu a vytvoření z něj binárního a výběr kandidátů pro směr pohybu.

##### Primární polární histogram

Polární histogram je rozdělen na sektory tak, aby každý z nich odpovídal úhlu  $\alpha$ , který se volí takovým způsobem že  $\frac{360}{\alpha}$  je celý číslo. Tedy například pro  $\alpha = 15^\circ$  histogram obsahuje 24 sektorů (viz 4.2).

Pro každou buňku aktivního regionu, tedy lokální mapy vytvořené kolem robotu, se spočte směr a její význam (magnitude).

Směr spočteme dle následujícího vztahu

$$\beta_{i,j} = \text{atan2}(y_o - y_i, x_o - x_i), \quad (4.1)$$

kde

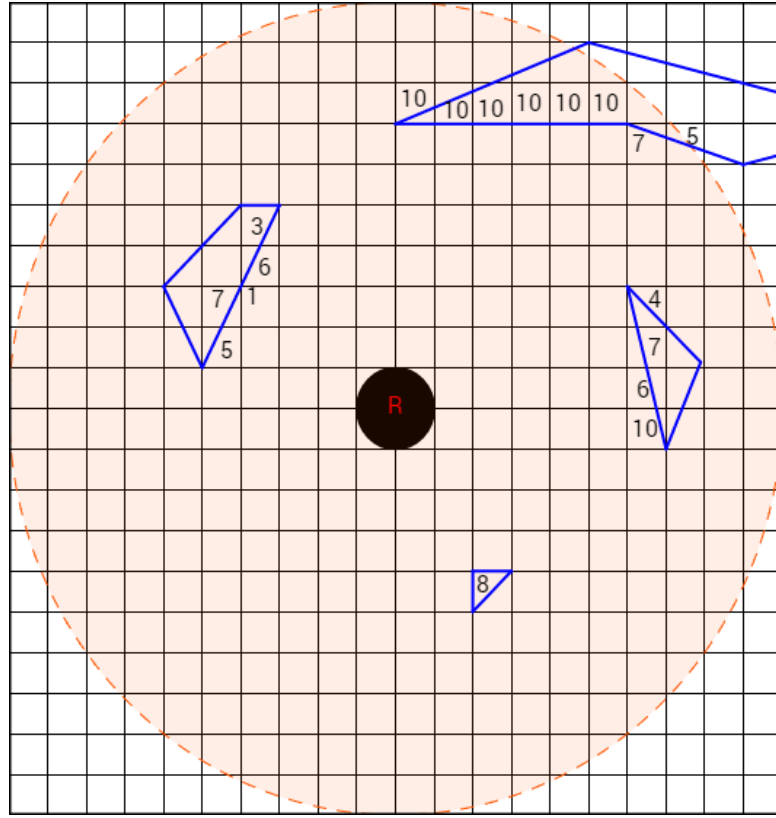
$x_o, y_o$ : Souřadnice robotu v mapě (centrum mapy).

$x_i, y_i$ : Souřadnice buňky.

Dále pak magnitude

$$m_{i,j} = c_{i,j}^2(a - bd_{i,j}^2), \quad (4.2)$$

kde



**Obrázek 4.1:** Lokální mapa prostředí o poloměru  $r_{map} = 10$

$c_{i,j}$ : Obsazenost buňky.

$d_{i,j}$ : Vzdálenost buňky od pozice robotu.

Parametry  $a$  a  $b$  se volí dle vztahu:

$$a - b \left( \frac{r_{map} - 1}{2} \right) = 1. \quad (4.3)$$

Aby robot nejel blízko okrajům překážek, zavádí se kompenzace jeho velikosti pomocí poloměru robotu  $r_{rob}$  a minimální povolené vzdálenosti mezi robotem a překážkou  $d_{safety}$ :

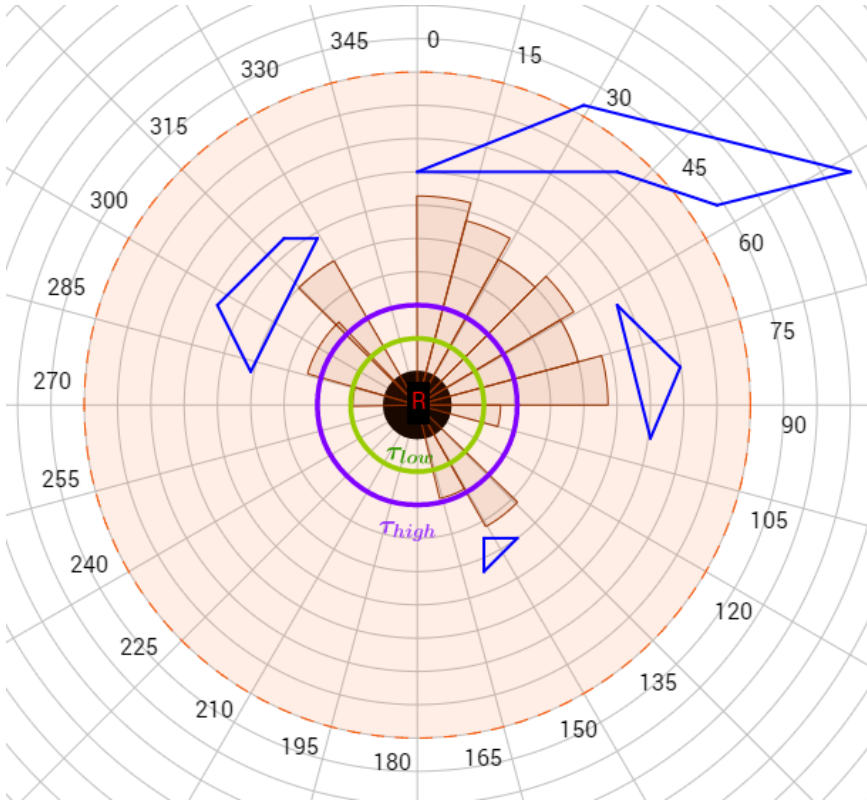
$$\gamma_{i,j} = \arcsin \left( \frac{r_{rob} + d_{safety}}{d_{i,j}} \right)$$

Histogram se potom vytvoří dle vztahu

$$H_k^p = \sum_{i,j \in C_\alpha} m_{i,j} h_{i,j}, \quad (4.4)$$

kde

$$h_{i,j} = \begin{cases} 1 & \text{jestli } k\alpha \in [\beta_{i,j} - \gamma_{i,j}; \beta_{i,j} + \gamma_{i,j}], \\ 0 & \text{jinak.} \end{cases}$$



**Obrázek 4.2:** Polární histogram s vyobrazenými prahy  $\tau_{low}$  (zeleně) a  $\tau_{high}$  (fialově)

### ■ Binární polární histogram

Aby bylo možné určit kandidáty pro další směr pohybu, primární histogram musí být převeden do binární podoby.

$$H_{k,i}^b = \begin{cases} 1 & \text{jestli } H_{k,i}^p > \tau_{high}, \\ 0 & \text{jestli } H_{k,i}^p < \tau_{low}, \\ H_{k,i-1}^b & \text{jinak.} \end{cases}$$

### ■ Výběr kandidátů

Kandidáty pro směr pohybu se volí podle toho, do jaké kategorie je zařazeno volné místo v binárním histogramu. Ty průjezdy, jež mají velikost (tedy vzdálenost mezi pravým okrajem  $k_r$  a levým  $k_l$ ) menší, než  $s_{max}$ , se nazývají úzké, jiné jsou naopak nazývány široké.

Pro úzké průjezdy lze zvolit pouze jednoho kandidáta, a to

$$c_n = \frac{k_r + k_l}{2} \quad \text{centrální sektor}$$

Široké průjezdy mají tři možné kandidáty:

$$\begin{aligned}
c_r &= k_r + \frac{s_{max}}{2} && \text{pravý sektor,} \\
c_l &= k_l - \frac{s_{max}}{2} && \text{levý sektor,} \\
c_t &= k_t && \text{pokud } k_t \in [c_r; c_l]
\end{aligned}$$

Nový směr pohybu se zvolí dle minimální ceny spočítané pro kandidáta  $c_i$  pomocí vztahu

$$g(c_i) = \mu_1 \Delta(c_i, k_t) + \mu_2 \Delta\left(c_i, \frac{\theta_n}{\alpha}\right) + \mu_3 \Delta(c_i, k_{d,n-1}) \quad (4.5)$$

přičemž

$$\Delta(c_1, c_2) = \min \left\{ |c_1 - c_2|, \left| c_1 - c_2 - \frac{360^\circ}{\alpha} \right|, \left| c_1 - c_2 + \frac{360^\circ}{\alpha} \right| \right\}.$$

$\theta_n$ : Heading direction.

$k_{d,n-1}$ : Minule zvolený kandidát.

$\mu_1, \mu_2, \mu_3$ : Konstanty, zvolené dle vztahu  $\mu_1 > \mu_2 + \mu_3$ .



## Kapitola 5

### Výsledky simulací







## Kapitola 6

### Závěr