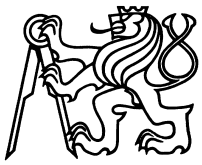
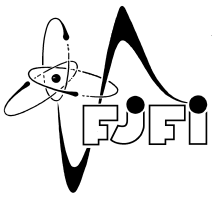


České vysoké učení technické v Praze
Fakulta jaderná a fyzikálně inženýrská



Odhadování pohybu mikrorobotů v magnetickém poli

Estimation of motion of micro-robots in magnetic field

Diplomová práce

Autor: **Bc. Zuzana Sabolová**

Vedoucí práce: **doc. Ing. Václav Šmídl, Ph.D.**

Konzultant: **doc. Ing. Filip Šroubek, Ph.D.**

Akademický rok: 2021/2022

Katedra: matematiky

Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student:	Bc. Zuzana Sabolová
Studijní program:	Aplikace přírodních věd
Studijní obor:	Aplikované matematicko-stochastické metody
Název práce (česky):	Odhadování pohybu mikrorobotů v magnetickém poli
Název práce (anglicky):	Estimation of motion of micro-robots in magnetic field

Pokyny pro vypracování:

1) Seznamte se s problémem sledování rychle se pohybujícího objektu v obrazové sekvenci a modelem generování zaznamenaného obrazu. Proveďte analýzu videa z pohybu mikrorobota v magnetickém poli a diskutujte jeho specifika.
2) Seznamte se s existujícími metodami na odhadování rychle se pohybujících objektů. Zvláštní pozornost věnujte metodám založených na slepé dekonvoluci obrazu.
3) Proveďte rešerši různých přístupů generování trajektorie pohybu robota z odhadnutého konvolučního jádra. Zvláštní pozornost věnujte metodám, které odhadují přímo trajetorii pohybu během iterací dekonvolučního algoritmu.
4) Na dodaných datech z experimentálního mikrorobotického zařízení aplikujte alespoň dvě existující metody a diskutujte jejich výhody a nevýhody. Vytipujte možné předpoklady, které jsou pro tento problém specifické.
5) Pokuste se navrhnout vylepšení existujících metod pro zvolené předpoklady. Odvoďte modifikaci dekonvolučního algoritmu a otestujte její vlastnosti na reálných datech z mikrorobotiky. Diskutujte vhodnost navržených předpokladů.

Doporučená literatura:

1) D. Rozumnyi, J. Kotera, F. Sroubek, L. Novotny, and J. Matas, The world of fast moving objects. In 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition', IEEE, 2017, 5203-5211.

2) M. Jurik, V. Smidl, J. Kuthan and F. Mach, Trade-off Between Resolution and Frame Rate for Visual Tracking of Mini-robots on Planar Surfaces. In 'Proceedings of International Conference on Manipulation, Automation and Robotics at Small Scales (MARSS)', IEEE, 2019, 1-6.

3) J. Kotera, J. Matas and F. Šroubek, Restoration of Fast Moving Objects. IEEE Transactions on Image Processing 29, 2020, 8577-8589.

Jméno a pracoviště vedoucího diplomové práce:

doc. Ing. Václav Šmídl, Ph.D.
Ústav teorie informace a automatizace AV ČR, v.v.i., Pod Vodárenskou věží 4, 182 08 Praha 8

Jméno a pracoviště konzultanta:

doc. Ing. Filip Šroubek, Ph.D
Ústav teorie informace a automatizace AV ČR, v.v.i., Pod Vodárenskou věží 4, 182 08 Praha 8

Datum zadání diplomové práce: 31.10.2020

Datum odevzdání diplomové práce: 3.5.2021

Doba platnosti zadání je dva roky od data zadání.

V Praze dne 26. října 2020

.....
garant oboru

.....
vedoucí katedry

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ
Fakulta jaderná a fyzikálně

.....
děkan

Poděkování:
Chtěla bych zde poděkovat především svému školiteli doc. Ing. Václavovi Šmídlovi, Ph.D. za pečlivost, ochotu, vstřícnost a odborné i lidské zázemí při vedení mého výzkumného úkolu.

Čestné prohlášení
Prohlašuji, že jsem tuto práci vypracovala samostatně a uvedla jsem všechnu použitou literaturu.

V Praze dne 1. května 2022

Bc. Zuzana Sabolová

Název práce:

Odhadování pohybu mikrorobotů v magnetickém poli

Autor: Bc. Zuzana Sabolová

Obor: Aplikované matematicko-stochastické metody

Druh práce: Diplomová práce

Vedoucí práce: doc. Ing. Václav Šmídl, Ph.D., Ústav teorie informace a automatizace AV ČR, v.v.i., Pod Vodárenskou věží 4, 182 08 Praha 8

Konzultant: doc. Ing. Filip Šroubek, Ph.D., Ústav teorie informace a automatizace AV ČR, v.v.i., Pod Vodárenskou věží 4, 182 08 Praha 8

Abstrakt: Cílem této diplomové práce bylo seznámit se s algoritmy používanými k sledování objektů. Hlavními zkoumanými algoritmami byli částicový filtr vrátaně adaptivního vzorkování podle důležitosti a hromadního adaptivního vzorkování podle důležitosti a slepá dekonvoluce. Tyto algoritmy byly poté aplikovány a porovnány v aplikaci na video záznam magneticky poháněného mikrorobota.

Klíčová slova: částicový filtr, dekonvoluce, microrobot

Title:

Estimation of motion of micro-robots in magnetic field

Author: Bc. Zuzana Sabolová

Abstract: The aim of this thesis was to research algorithms used to track moving objects. The main algorithms were particle filter including adaptive sampling and multiple adaptive sampling and blind deconvolution. These algorithms were then applied and compared when used to track magnetically propelled micro-robots.

Key words: deconvolution, micro-robot, particle filter

Obsah

Zoznam použitých symbolov **6**

Úvod **8**

1 Bayesovská filtrácia **9**
1.1 Kalmanov filter . . . 10
1.2 Simulovaný experiment . . . 11
1.3 Nedostatky Kalmonovho filtra a alternatívy . . . 14

2 Časticový filter **16**
2.1 Základný popis fungovania časticového filtra . . . 17
2.2 Prevzorkovanie . . . 18
2.2.1 Multinomické prevzorkovanie . . . 19
2.2.2 Reziduálne prevzorkovanie . . . 20
2.2.3 Systematické (stratifikované) prevzorkovanie . . . 20
2.3 Algoritmus časticového filtra . . . 22
2.4 Časticový filter pre simulovaný experiment . . . 22
2.4.1 Voľba metrík presnosti odhadu . . . 22
2.4.2 Špecifikácia modelov a porovanie algoritmov . . . 24
2.5 Zložitejšie algoritmy vzorkovania . . . 28
2.5.1 Adaptívne vzorkovanie podľa dôležitosti . . . 29
2.5.2 Hromadné adaptívne vzorkovanie podľa dôležitosti . . . 30
2.5.3 Adaptívne a hromadné adaptívne sekvenčné vzorkovanie . . . 31

3 Slepá dekonvolúcia **32**
3.1 Použitý algoritmus slepej dekonvolúcie . . . 33

4 Detekcia polohy robota **37**
4.1 Časticový filter . . . 39
4.1.1 Základný model pozorovania . . . 39
4.1.2 Model pozorovania s rozmazaním . . . 40
4.1.3 Výpočet váh častíc . . . 40
4.1.4 Adaptívne a hromadné adaptívne vzorkovanie podľa dôležitosti . . . 40
4.2 Slepá dekonvolúcia . . . 41
4.2.1 Spresnenie pozadia . . . 41
4.2.2 Normalizácia jadra . . . 42

5 Výsledky a porovnanie algoritmov **43**
5.1 Presnosť určenia polohy 43
5.1.1 Zvolené metriky 43
5.1.2 Porovnanie metrík pre každý uvažovaný algoritmus 44
5.1.3 Porovnanie algoritmov na základe zvolených metrík 47
5.2 Konvergencia časticového filtra 51
5.3 Spresnenie odhadu slepou dekonvolúciou 56
5.3.1 Spresnenie pozadia 56
5.3.2 Normalizácia jadra 57
5.3.3 Výsledky spresnenia 58
5.4 Konvolučné jadro 62

Záver **64**

Zoznam použitých symbolov

α_o, β_o	parametre zobecneného normálneho rozdelenia šumu pozorovaní pre mikrorobota
δ	δ funkcia
λ	parameter hustoty pravdepodobnosti q
ξ	funkcia kladúca požiadavky na parameter λ
ϖ_k	šum procesu
σ^2	rozptyl
ω_k^i	váha i-tej častice v k-tom kroku
Σ	kovariančná matica
ψ	pomocná funkcia v algoritme slepej dekonvolúcie
Γ	gamma funkcia
$P, \gamma, \alpha, \alpha_u, \alpha_h, \alpha_w, \beta_u, \beta_h, \beta_w$	parametre algoritmu slepej dekonvolúcie
a_k	akcelerácia v kroku k pri detekcii polohy robota
$A^{(l)}(\xi_j)$	funkcionál pôsobiaci na funkciu ξ v cykle l v rámci AMIS
B_l^i	pomocná veličina pre časticu i v cykle l v rámci AMIS
C	konštanta
$(c_1, \ldots, c_N)$	kumulatívna suma váh normalizovaných častíc
$\mathscr{D}(f)$	definičný obor funkcie f
$D = [D_x^T, D_y^T]^T$	operátor smerových derivácií
d_x	rozmer stavového vektora
d_z	rozmer pozorovania
$\mathbb{E}$	stredná hodnota náhodnej veličiny
f_k	funkcia určujúca časový vývoj stavového vektora
$\mathrm{supp}(f)$	nosič funkcie f
g	pozorovaný obrázok pri slepej dekonvolúcii
G	oblasť posunutej šablóny robota spolu s okolím veľkosti 10 pixelov
h	konvolučné jadro
$\hat{H}$	matica validnej konvolúcie s h
k	index času a poradové číslo snímku
$L(u, h)$	energia, minimalizovaný výraz pri slepej dekonvolúcii
M	cropping matrix

n, n_k	šum pozorovaní, šum pozorovaní v časovom kroku k
N	celkový počet častíc generovaný v časovom kroku k pre BIS a AMIS
$\mathcal{N}$	normálne rozdelenie
N_{eff}	efektívna veľkosť vzorku
N_{thr}	hraničná hodnota efektívnej veľkosti vzorku
O_1	prvý snímok videa, pozadie bez mikrorobota
O_k	k-tý snímok videa
$p_{x,k},\ p_{y,k}$	x a y súradnice mikrorobota v časovom kroku k
q	hustota pravdepodobnosti, z ktorej sú generované častice pri vzorkovaní podľa dôležitosti
$Q(u)$	regularizovaná hustota odpovedajúca zápornému logaritmu apriornej hustoty $p(u)$
$r_k^1,\ r_k^2,\ r_k^3$	obrázky konštruované ako rozdiel $O_k - O_1$ a posunutých šablón
$R_k^1,\ R_k^2,\ R_k^3$	sumy hodnôt pixelov obrázkov $r_1(k),\ r_2(k),\ r_3(k)$
$S(h)$	regularizovaná hustota odpovedajúca zápornému logaritmu apriornej hustoty $p(h)$
s_k	funkcia určujúca časový vývoj stavového pozorovania pre časticové filtre
T	šablóna mikrorobota
$\mathcal{U}$	rovnomerné rozdelenie
u	skutočný obrázok pri slepej dekonvolúcii
$(u_1, \ldots, u_N)$	kumulatívna suma váh, ktorá korešponduje s rovnomerne rozdeleným výberom
$v_{x,k},\ v_{y,k}$	rýchlosť mikrorobota v smere x a y v časovom kroku k
v, v_h, w	pomocné premenné pri algoritme slepej dekonvolúcie
x_k	stavový vektor
X_k	vektor hodnôt stvového vektora do času k
z_k	pozorovanie v čase k
Z_k	vektor pozorovaní do času k

Úvod

Detekcia a sledovanie objektov je problém riešený v mnohých a veľmi rôznorodých oblastiach, preto má zmysel študovať a potenciálne rozvíjať algoritmy používané k riešeniu tohto problému.

Pre sledovanie objektov sú najčastejšie používané Bayesovské odhady. Prvotné metódy boli deterministické ako Kalmanov filter a jeho obdoby. S rovojom výpočetnej techniky ale začalo mať zmysel rozvíjať skôr stochastiké metódy, ktoré sú presnejšie a porovnateľne rýchle. Dnes sa najčastejšie používajú časticové filtre, ktoré vychádzajú z Monte Carlo metód.

Jedna z aktuálne sa rozvíjajúcich oblastí, v ktorej je potrebné riešiť navádzanie a sledovanie objektov je mikrorobotika. Cieľom tejto práce je predstaviť základné algoritmy používané pre sledovanie objektov a vyskúšať ich fungovanie na jednoduchom príklade. Následne budú aplikované na video záznam magneticky poháňaného mikrorobota, úlohou je z videa čo najpresnejšie určiť skutočnú polohu.

Pri riešení tejto úlohy sa budeme venovať algoritmom spadajúcim do dvoch odlišných smerov. Prvým je bayesovská filtrácia, v rámci ktorej implementujeme rozšírený Kalmanov filter a časticový filter s troma rôznymi algoritmami vzorkovania a exploráciou rôznych nastavení parametrov. Druhým smerom je algoritmus slepej dekonvolúcie, v rámci ktorej aplikujeme jej základnú verziu. Následne algoritmy porovnáme a rozoberieme ich klady a zápory pre túto konkrétnu aplikáciu.

Dva uvedené smery sú diametrálne rozličné v prístupe k riešeniu úlohy. Zatiaľčo algoritmy bayesovskej filtrácie pracujú s dynamikou pohybu robota, dekonvolúcia sa na každý snímok díva osobitne a dynamiku vôbec nevyužíva. Zrejmá výhoda dekonvolúcie je, že na rozdiel od filtrácie je schopná pracovať s rozmazaním robota, ktoré je v dátach časté, a výpočetná rýchlosť oproti časticovému filtru. Možnou výhodou aplikácie časticového filtra oproti dekonvolúcii v rámci tejto oblasti odhliadnuc od použitého datasetu je schopnosť časticového filtra pracovať i s iným typom pozorovania ako je video - v detekcii magneticky poháňaného robota to môžu byť merania poskytnuté cievkami.

Prvý smer - algoritmy bayesovskej filtrácie - bol popísaný už v rámci bakalárskej práce autora. Je ale nevyhnutné zhrnutie tohto obsahu zahrnúť i do diplomovej práce. Snahou autora bolo preformulovať ich jasnejšie, detailnejšie a poskytnúť viac kontextu využitím aditívnych zdrojov. Algoritmom filtrácie sa venujú prvé dve kapitoly.

V bakalárskej práce sme ukázali, že zvažované algoritmy bayesovskej filtrácie naozaj úlohu detekcie robota riešia. Hlavným cieľom diplomovej práce bolo zvážiť dekonvolúciu ako ich alternatívu z dôvodu menšej výpočetnej náročnosti. Tomuto prístupu sa venuje tretia kapitola.

Kapitola 1 všeobecne predstavuje princíp bayesovskej filtrovácie, popisuje Kalmanov filtrer, jeho odboby a nedostatky a taktiež definuje jednoduchý simulovaný príklad používaný k ilustrácii a porovnaniu algoritmov. Kapitola 2 je venovaná podrobnému popisu časticového filtra a ukazuje jeho funovanie na simulovenom príklade. V tretej kapitole skúmame slepú dekonvolúciu ako alternatívu časticového filtra. Štvrtá kapitola obsahuje detaily implementácie algoritmov. Posledná 5. kapitola obsahuje porovnanie algoritmov po aplikácii na video mikrorobota.

Kapitola 1

Bayesovská filtrácia

Témy diskutované v kapitolách 1 a 2 boli obsahom bakalárskej práce autora [1]. V diplomovej práci sú uvedené znovu, keďže obsahujú popis použitých algoritmov, avšak rozšírené o širší kontext a detaily po využití aditívnych zdrojov.

Prístup bayesovskej filtrácie je dobrým prístupom pre úlohu detekcie polohy robota, keďže umožňuje uvažovať a modelovať rýchlosť a zotrvačnosť robota.

Bayesovská fitrácia je proces rekurzívneho odhadu skutočného stavu systému pomocou množiny pozorovaní, ktoré sú zašumené. Množinu veličín, ktoré systém charakterizujú a budeme ich chcieť odhadnúť na základe pozorovaní budeme nazývať stavový vektor.

Pre stochastickú filtráciu je potrebné poznať alebo predpokladať vývoj veličín stavového vektora v čase a model šumu pozorovaní.

Jednou z oblastí, v ktorých je bayesovská filtrácia často používaná a do ktorej spadá i konečný cieľ tejto práce je odhad polohy nejakého pohybujúceho sa objektu. V tomto prípade budú najčastejšie súčasťou stavového vektora veličiny ako poloha, rýchlosť alebo zrýchlenie.

Zdrojom pozorovaní môžu byť senzory, ktoré sú súčasťou pohybujúceho sa objektu a zaznamenávajú polohu objektu vzhľadom k nejakému statickému bodu - v tomto prípade hovoríme o probléme navigácie - alebo externé senzory. Do kategórie externých senzorov môže spadať senzor, ktorý je súčasťou plochy, po ktorej sa objekt pohybuje alebo napríklad aj video. V tomto prípade hovoríme o probléme detekcie polohy alebo sledovania objektu. [2]

Nech podľa [3] $x_k \in \mathbb{R}^{d_x}$ je stavový vektor rozmeru d_x, kde k je index času, $k \in \mathbb{N}$. Časový vývoj stavového vektoru je určený nasledovným modelom:

$$x_k = f_{k-1}(x_{k-1}, \varpi_{k-1}), \tag{1.1}$$

kde f_{k-1} je známa funkcia a ϖ_{k-1} je šum procesu. Šum procesu zachytáva nepresnosti a disturbacie. Model šumu pozorovaných veličín:

$$z_k = s_k(x_k, n_k), \tag{1.2}$$

kde s_k je známa funkcia a n_k je postupnosť šumu pozorovaní. Budeme predpokladať, že ϖ_{k-1} a n_k sú nezávislé postupnosti šumov s nulovou strednou hodnotou a konečným rozptylom. Budeme predpokladať že hustoty týchto šumov sú známe a hustota počiatočného stavového vektora $p(x_0)$ je taktiež známa.

V každom čase k je úlohou odhad stavového vektora x_k na základe pozorovaní $\{z_1, \ldots, z_k\}$, túto množinu budeme označovať Z_k. Táto úloha je teda akvivalentná určeniu, resp. odhadu hustoty $p(x_k|Z_k)$. Tento odhad prebieha rekurzívne, predpokladáme že hustota pravdepodobnosti $p(x_0) = p(x_0|z_0)$ na začiatku procesu je známa.

Odhad $p(x_k|Z_k)$ prebieha v dvoch štádiách:

1. predpoklad:
   Odhad na základe pozorovaní $\{z_1, \ldots, z_{k-1}\}$:

$$p(x_k|Z_{k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|Z_{k-1})dx_{k-1}, \tag{1.3}$$

   kde $p(x_k|x_{k-1})$ sa nazýva prechodná hustota. Táto hustota je určená stavovým modelom a šumom procesu. Platnosť rovnosi (1.3) vyplýva z Chapman-Kolmogorovy vety.

2. spresnenie:
   Spesnenie husoty vyplýva z Bayesovského princípu. K $p(x_k|Z_{k-1})$ budeme pristupovať ako k apriornej hustote, ktorá sa pozorovaním z_k v čase k spresní na aposteriornú hustotu $p(x_k|Z_k)$:

$$p(x_k|Z_k) = p(x_k|z_k, Z_{k-1}) = \frac{p(z_k|x_k, Z_{k-1})p(x_k|Z_{k-1})}{p(z_k|Z_{k-1})} = \frac{p(z_k|x_k)p(x_k|Z_{k-1})}{p(z_k|Z_{k-1})} \tag{1.4}$$

$$p(z_k|Z_{k-1}) = \int p(z_k|x_k)p(x_k|Z_{k-1})dx_k, \tag{1.5}$$

   Hustota $p(z_k|x_k)$ je definovaná modelom pozorovaní (1.2) a hustotou šumu pozorovaní. (1.5) vyplýva z Chapman-Kolmogorovy vety ako v predchádzajúcom prípade.

Odhad stavového vektora získavame z hustoty $p(x_k|Z_k)$ najčastejšie ako strednú hodnotu alebo argument maxima.

Na tomto mieste je dôležité poznamenať, že výrazy 1.3 až 1.5 je možné analyticky vyjadriť len v prípade že systém je lineárny a Gaussovský, teda že funkcie f_{k-1} a s_k sú lineárne a šum procesu i šum pozorvaní majú gaussovské rozdelenie. V tomto prípade postupom Bayesovskej filtrácie vzniká algoritmus Kalmanovho filtra diskutovaný v nasledujúcej časti. Predpoklad lineárneho a Gaussovského systému môže byť obmedzujúci, ak neplatí, výrazy 1.3 až 1.5 nie je možné analyticky vyjadriť a teda ich aproximujeme. Aproximácia môže byť v tomto prípade deterministická, predstaviteľom je napríklad rozšírený Kalmanov filter a tento prípad je obsahom je obsahom tejto kapiroly, alebo stochastická - predstaviteľom je častícový filter, ktorému sa venuje kapitola 2.

1.1 Kalmanov filter

Kalmanov filter je prvá a najjednoduchšia z použitých metód, uvažovaná hlavne pre porovnanie a nepoužiteľná pre zložitejšie problémy. Tento algoritmus je deterministický a detailne popísaný napríklad v [3]. V svojej základnej podobe predpokladá linearitu funkcií $f_{k-1}(x_{k-1}, v_{k-1})$, $s_k(x_k, n_k)$ a normalitu rozdelení šumov. Platí, že ak $p(x_{k-1}|Z_{k-1})$ je normálne rozdelená, potom aj $p(x_k|Z_k)$ je normálne rozdelená.

Predpoklad linearity funkcií $f_{k-1}(x_{k-1}, v_{k-1})$, $s_k(x_k, n_k)$ je veľmi obmedzujúci ale vďaka nemu je možné ich nahradiť maticami a stavový model a model pozorovaní prejde do tvaru

$$x_k = F_{k-1}x_{k-1} + v_{k-1} \tag{1.6}$$

$$z_k = H_k x_k + w_k, \tag{1.7}$$

kde $F_{k-1} \in \mathbb{R}^{n_x \times n_x}$ a $H_k \in \mathbb{R}^{n_z \times n_x}$. Naďalej predpokladáme, že v_{k-1} a w_k sú rozdelené normálne, stredná hodnota je rovná nule. Kovariančné matice príslušných normálnych rozdelení budeme označovať ako Q_{k-1} a R_k. Predpokladáme, že Q_{k-1} a R_k sú vzájomne nezávislé. Podľa predpokladov Kalmanovho filtra $p(x_{k-1}|Z_{k-1}) = \mathcal{N}(x_{k-1}; \hat{x}_{k-1|k-1}, P_{k-1|k-1})$. Obecný postup filtrácie popísaný v 1 má v podaní Kalmanovho filtra naslednovný tvar uvedený v [3]:

- predpoklad:

$$p(x_k|Z_{k-1}) = \mathcal{N}(x_k; \hat{x}_{k|k-1}, P_{k|k-1}) \tag{1.8}$$

$$\hat{x}_{k|k-1} = F_{k-1}\hat{x}_{k-1|k-1} \tag{1.9}$$

$$P_{k|k-1} = Q_{k-1} + F_{k-1}P_{k-1|k-1}F_{k-1}^T \tag{1.10}$$

- spresnenie:

$$p(x_k|Z_k) = \mathcal{N}(x_k; \hat{x}_{k|k}, P_{k|k}) \tag{1.11}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(z_k - H_k\hat{x}_{k|k-1}) \tag{1.12}$$

$$P_{k|k} = P_{k|k-1} - K_k S_k K_k^T \tag{1.13}$$

$$S_k = H_k P_{k|k-1} H_k^T + R_k \tag{1.14}$$

$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{1.15}$$

Kalmanov filter je pre lineárny problém s normálne rozdelenými chybami optimálnym riešením. Tieto prísne predpoklady však nesplní veľké množstvo problémov reálneho sveta.

Rozvoľnenie týchto prísnych predpokladov ponúka zobecněný (alebo rozšírený) Kalmanov filter, ktorý miesto liearity funkcií $f_{k-1}(x_{k-1}, v_{k-1})$, $s_k(x_k, n_k)$ požaduje len ich spojitosť a následne ich aproximuje Taylorovým rozvojom. Namiesto matíc F_{k-1} a H_k sú použité ich jakobiány:

$$\hat{F}_{k-1} = [\Delta_{x_{k-1}} f_{k-1}^T(x_{k-1})]^T\big|_{x_k=\hat{x}_{k-1|k-1}} \tag{1.16}$$

$$\hat{H}_k = [\Delta_{x_k} h_k^T(x_k)]^T\big|_{x_k=\hat{x}_{k|k-1}}, \tag{1.17}$$

kde $\Delta_{x_k} = \left(\frac{\partial}{\partial x_k(1)}, \ldots, \frac{\partial}{\partial x_k(n_x)}\right)$, $x_k(i)$, $i \in \hat{nx}$ je i-tá zložka vektora x_k.

V ostatných ohľadoch je algoritmus zobecneného Kalmanovho filtra analogický základnému algoritmu Kalmanovho filtra a podrobne popísaný v rovnakej publikácii [3].

Vďaka rozvoľneniu požiadavku na linearitu funkcií $f_{k-1}(x_{k-1}, v_{k-1})$, $s_k(x_k, n_k)$ je možné zobecnený Kalmanov filter použiť na širšiu triedu problémov, s variabilnou mierou presnosti. Keďže v podstate zobecnenia leží lokálna aproximácia Taylorovým rozvojom, tento algoritmus je možné úspešne použiť za predpokladu, že táto lokálna aproximácia funkcií $f_{k-1}(x_{k-1}, v_{k-1})$, $s_k(x_k, n_k)$ dostatočne popisuje ich priebeh v tomto bode.

1.2 Simulovaný experiment

Hlavným cieľom práce je odhad polohy mikrorobota pomocou niekoľkých algoritmov, ich porovnanie a vyjadrenie ich úspešnosti. Prvotne však chceme algoritmy filtrácie aplikovať na jednoduchý umelý príklad sínusovej vlny, parametre ktorej sa v rámci procesu prudko zmenia. Tento príklad nám umožňuje vlastnosti algoritov pochopiť a demonštrovať názornejšie. Prudká zmena parametrov sa odkazuje na nízku zotrvačnosť, ktorá je typická pre magneticky poháňané objekty.

Sínusovú vlnu definujeme nasledovne:

$$y_k = a_k sin(\omega k + \varphi_k), \tag{1.18}$$

kde frekvencia bude počas celého procesu konštantná: $\omega = 2\pi f$, $f = 50$ Hz, avšak amplitúda a fáza sa zmenia v polovici procesu.

V reči bayesovského filtrovania je teda $x_k = (a_k, \varphi_k)^T$ stavový vektor, predpokladáme že v čase k máme k dispozícii pozorovania vlny v diskrétnych časoch $\{z_1, \ldots, z_k\}$. Časový vývoj stavového vektora modelujeme ako náhodnú prechádzku, šum procesu $(\varepsilon_a, \varepsilon_\varphi)^T$ má štandardné normálne rozdelenie, šum meraní $\varepsilon \sim \mathcal{N}(0,\ 0,\ 0,04)$. Rozdelenie oboch šumov je v čase koštantné. Model vývoja stavového vektora a model pozorovaní majú teda nasledujúci tvar:

$$\begin{pmatrix} a_k \\ \varphi_k \end{pmatrix} = \begin{pmatrix} a_{k-1} \\ \varphi_{k-1} \end{pmatrix} + \begin{pmatrix} \varepsilon_a \\ \varepsilon_\varphi \end{pmatrix} \tag{1.19}$$

$$z_k = a_k sin\left(\omega k + \varphi_k\right) + \varepsilon \tag{1.20}$$

V rovnici (1.16) je $f_{k-1}(x_{k-1})$ identita $\forall k \in \mathbb{N}$ a teda lineárna, jej jakobián splýva s maticou identity, ktorou by sme ju reprezentovali v prípade klasického Kalmanovho filtra. Funkcia $h_k(x_k) = a_k \sin(\omega k + \varphi_k)$. Sínus je nelineárna, ale spojitá funkcia, ktorú jej lokálna linearizácia aproximuje veľmi dobre.

$$\begin{aligned}
\hat{H}_k &= \left[\begin{pmatrix} \frac{\partial}{\partial a_k} \\ \frac{\partial}{\partial \varphi_k} \end{pmatrix} a_k sin(\omega k + \varphi_k) \right]^T \Bigg|_{\binom{a_k}{\varphi_k} = \binom{\hat{a}_{k|k-1}}{\hat{\varphi}_{k|k-1}}} = \\
&= \left(\frac{\partial a_k sin(\omega k + \varphi_k)}{\partial a_k},\ \frac{\partial a_k sin(\omega k + \varphi_k)}{\partial \varphi_k} \right) \Bigg|_{\binom{a_k}{\varphi_k} = \binom{\hat{a}_{k|k-1}}{\hat{\varphi}_{k|k-1}}} = \\
&= \left(sin(\omega k + \varphi_k),\ a_k cos(\omega k + \varphi_k) \right) \Bigg|_{\binom{a_k}{\varphi_k} = \binom{\hat{a}_{k|k-1}}{\hat{\varphi}_{k|k-1}}}
\end{aligned} \tag{1.21}$$

Tento príklad je simulovaný, pozorovania generujeme z normálneho rozdelenia s kovariančnou maticou (rozptylom) $R = 0,04$. Predpokladáme, že šum procesu má taktiež normálne rozdelenie.

Kovariančnú maticu šumu procesu Q nepoznáme, jej voľba veľmi výrazne ovplyvňuje fungovanie filtra. Predpokladáme, že amplitúda a fáza sú nezávislé náhodné veličiny a teda diagonálne prvky kovariančnej matice budú nulové. Rozptyly oboch veličín zvolíme rovnaké a označíme σ^2.

Hypotéza: Filter bude poskytovať presné odhady v častiach procesu, kde sú hodnoty amplitúdy a fázy konštantné.

Konkrétne pre $(a_k, \varphi_k)^T = (1, 0)^T,\ k \in \{1, \ldots, 50\},\ (a_k, \varphi_k)^T = (5, -\frac{\pi}{2})^T,\ k > 50$:

Presnosť filtrov pre rôzne voľby rozptylu σ^2 je vyhodnotená na základe priemerov absolútnych a kvadratických odchýlok pre $S = 1000$ generovaných setov pozorovaní:

- Priemer stredných absolútnych a kvadratických odchýlok pre celý proces

$$\bar{\Delta} = \frac{1}{S} \sum_{j=1}^{S} \left(\frac{1}{100} \sum_{k=1}^{100} |\widehat{y_k}^j - y_k| \right),\ \ \bar{\delta} = \frac{1}{S} \sum_{j=1}^{S} \left(\frac{1}{100} \sum_{k=1}^{100} (\widehat{y_k}^j - y_k)^2 \right)$$

- Priemer stredných absolútnych a kvadratických odchýlok pre časť procesu pred skokom

$$\bar{\Delta}_{1\to 50} = \frac{1}{S} \sum_{j=1}^{S} \left(\frac{1}{50} \sum_{k=1}^{50} |\widehat{y_k}^j - y_k| \right),$$

$$\bar{\delta}_{1\to 50} = \frac{1}{S} \sum_{j=1}^{S} \left(\frac{1}{50} \sum_{k=1}^{50} (\widehat{y_k}^j - y_k)^2 \right)$$

- Priemer stredných absolútnych a kvadratických odchýlok pre časť procesu po skoku

$$\bar{\Delta}_{51\to100} = \frac{1}{S}\sum_{j=1}^{S}\left(\frac{1}{50}\sum_{k=51}^{100}|\widehat{y_k}^{\,j} - y_k|\right),$$

$$\bar{\delta}_{51\to100} = \frac{1}{S}\sum_{j=1}^{S}\left(\frac{1}{50}\sum_{k=51}^{100}(\widehat{y_k}^{\,j} - y_k)^2\right)$$

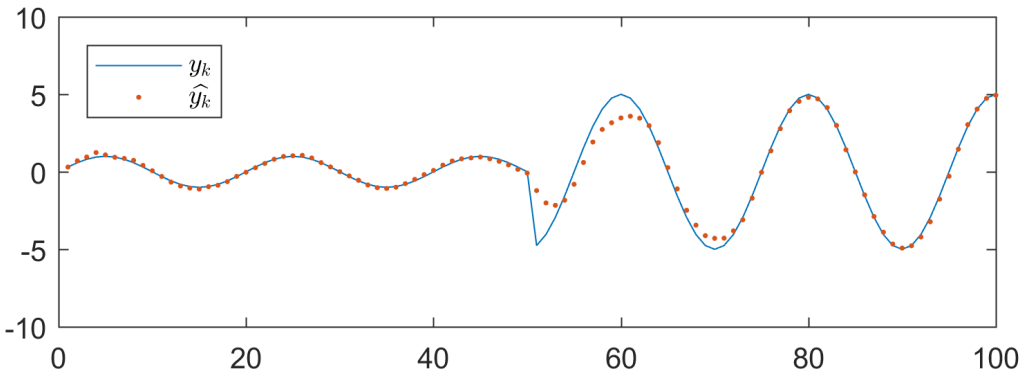
- Priemer absolútnych a kvadratických odchýlok v skoku

$$\bar{\Delta}_{51} = \frac{1}{S}\sum_{j=1}^{S}|\widehat{y_{51}}^{\,j} - y_{51}|, \quad \bar{\delta}_{51} = \frac{1}{S}\sum_{j=1}^{S}(\widehat{y_{51}}^{\,j} - y_{51})^2$$

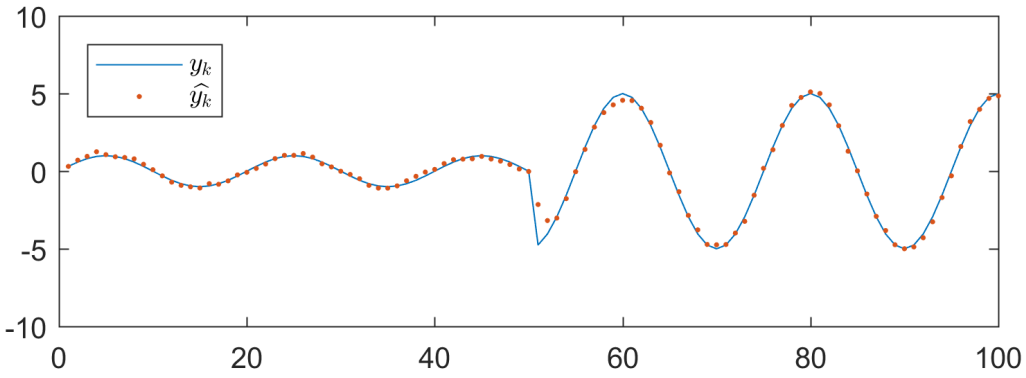
σ^2	$\bar{\Delta}$	$\bar{\Delta}_{1\to50}$	$\bar{\Delta}_{51\to100}$	$\bar{\Delta}_{51}$	$\bar{\delta}$	$\bar{\delta}_{1\to50}$	$\bar{\delta}_{51\to100}$	$\bar{\delta}_{51}$
0,001	0,2642	0,0715	0,4569	3,5037	0,3167	0,0131	0,6203	12,2850
0,01	0,1563	0,0950	0,2175	2,6480	0,1067	0,0151	0,1984	7,0869
0,1	0,2073	0,1291	0,2855	5,0893	0,3812	0,0271	0,7353	30,6036
1	0,2420	0,1529	0,3311	5,8478	0,5897	0,0415	1,1379	45,9559
10	0,2530	0,1597	0,3463	5,9314	0,6908	0,0503	1,3313	51,8290
100	0,2565	0,1615	0,3514	5,9630	0,7769	0,0600	1,4937	55,2433
1000	0,2565	0,1614	0,3515	5,9883	0,7658	0,0581	1,4734	56,1184

Tabuľka 1.1: Porovnanie chýb rozšíreného Kalmanovho filtra pre rôzne voľby rozptylu σ^2

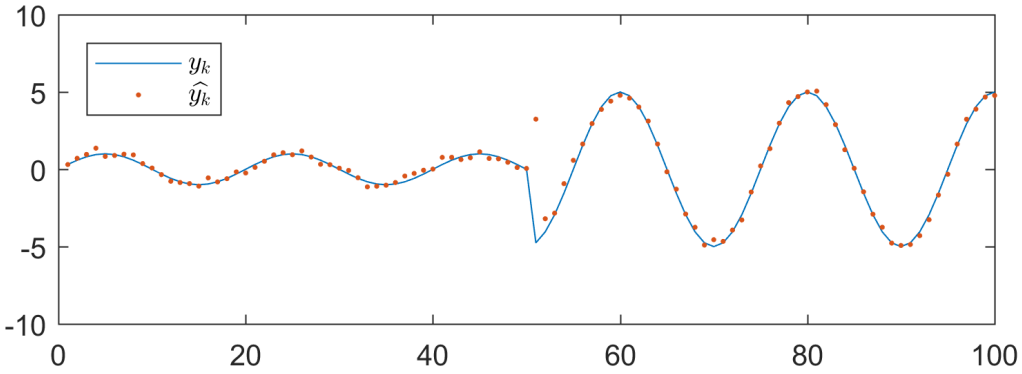
Najlepšie odhady filter poskytuje pri voľbe rozptylu $\sigma^2 = 0,01$. Tabuľka 1.1 ilustruje ako priemer stredných odchýlok v časti pred skokom monotónne klesá so zmenšujúcim sa roptylom, ale pre príliš malý rozptyl ($\sigma^2 = 0,001$) filter nefunguje dobre v bode skoku a novým hodnotám sa prispôsobuje pomaly. Nasledujúce grafy ukazujú odhady pre rovnaký set pozorovaní pre rôzne rozptyly.



Obr. 1.1: Rozšírený Kalmanov filter pre $\sigma^2 = 0,001$



Obr. 1.2: Rozšírený Kalmanov filter pre $\sigma^2 = 0,01$



Obr. 1.3: Rozšírený Kalmanov filter pre $\sigma^2 = 1$

Všeobecne funguje rozšírený Kalmanov filter dobre pre časti procesu, kde sú amplitúda a fáza konštantné. V čase, keď sa amplitúda a fáza náhle zmenia, model vývoja stavového vektora (1.19) neodpovedá jeho skutočnému vývoju a tak odhad stavového vektora môže byť v závislosti na posledných niekoľkých pozorovaniach veľmi rôzny a málokedy odpovedá skutočnosti. Po niekokých krokoch s novými hodnotami amplitúdy a fázy začne filter opäť poskytovať presné odhady.

1.3 Nedostatky Kalmonovho filtra a alternatívy

Ako bolo spomenuté, základná verzia Kalmanovho filtra je použiteľná len pre lineárny Gaussovský systém.

Zobecnený Kalmanov filter nelineárny systém linearizuje. Môže poskytovať dobré výsledky pre nie výrazne nelineárne systémy a ak je skutočná aposteriorná hustota unimodálna a symetrická. Pre výrazne nelineárne problémy nie je použiteľný [2].

Nasledujúcim krokom vo vývoji algoritmov Kalmovho filtra je Unscented Kalman filter, sktrátene UKF, tejto téme sa detailne venuje napríklad publikácia [4].

UKF splňuje očakávaný postup odhadu v rámci nelineárnej filtrácie - dva kroky a to predpoklad a spresnenie. Pred týmito krokmi je ale tzv. Unscented transformace, skrátene UT. Hlavná myšlieka tejto transormácie je reprezentácia hustoty pravdepodobnosti deterministicky vybraným malým množstvom vzorkov. Informácia o strednej hodnote a kovariancii tejto hustoty je zachytená pomocou týchto bodov, ktoré sa nazývajú sigma body. Na tieto body je možné následne priamo aplikovať nelineárne rovnice na rozdiel od zobecneného Kalmanovho filtra, kde bola použitá len liearizácia nelineárnej rovnice pomocou Taylorovho rozvoja. Ďalšou výhodou implementácie UKF je, že nie je potrebné zostrojiť a pracovať s jakobiánom. Najčastejšie je generovaných $2d_x+1$ sigma bodov, kde d_x je rozmer stavového vektora. UKF podobne ako rozšírený Kalmanov filter poskytuje dobré výsledky ak je skutočná aposteriorná hustota unimodálna.

Ďalších obmedzemí je zbavený point-mass filter, skrátene PMF. PMF je možné použiť na nelineárny negaussovský systém a pri akejkoľvek aposteriornej hustote. PMF počíta hodnoty aposteriornej hustoty len na diskrétnej mriežke. Potenciálnym obmedzením je výpočetná zložitosť spôsobená prekliatím dimenzionality. Point-mass filter má k časticovému filtru principiálne najbližšie, keďže oba tieto algoritmy aproximujú aposteriornú hustotu diskrétnym spôsobom. [2]

Algoritmy UKF a PMF nebudeme diskutovať detailnejšie, keďže vrámci práce neboli implementované, ale sú zaujímavým premostením k časticovému filtru, ktorý je úplne zbavený obmedzení Kalmanovho filtra a je predmetom nasledujúcej kapitoly.

Kapitola 2

Časticový filter

Ako bolo spomenuté v rámci diskusie o jednoduchom algoritme Kalmanovho filtra, obecný princíp bayesovskej filtrácie popísaný v kapitole 1 veľmi často nie je možné vyjadriť analyticky v prípade aplikácií reálneho sveta - časticový filter je riešením.

Táto metóda je na rozdiel od Kalmanovho filtra stochastická, nie deterministická. Široké využitie vyplýva z výrazne miernejších predpokladov, ktoré sú vyžadované. Časticové filtre sa stali alternatívou pri riešení problémov v reálnom čase, kde býval používaný Kalmanov filter, respektíve zobecnený Kalmonov filter. Čím menej lineány a gaussovský je uvažovaný problém, tým výraznejší potenciál časticové filtre predstavujú voči Kalmanovmu filtru.

Prvé zmienky časticového filtra boli publikované v roku 1954 v [5], intenzívny výskum a rozvoj v tento oblasti však začal až koncom 20. storočia. Prispeli k tomu dva faktory a to uvedenie prevzorkovania, ktoré bude podrobne diskutované v rámci tejto kapitoly, a je kľúčové v úspešnosti časticového filtra. Druhým faktorom bol vývoj výpočetnej techniky, ktorý umožnil implementáciu výpočetne náročnejších algoritmov.

Ďalším dôvodom, pre ktorý je časticový filter pokladaný za atraktívne rišenie, je, že pre dostatočný počet vzorkov odhad, ktorý filter poskytuje, konverguje k skutočnej hodnote [7].

Aj časticové filtre však majú svoje obmedzenia - aposteriorná hustota je reprezentovaná vzorkami, ktorých musí byť dostatok pri očakávaní dobrého výsledku, i pri použití časticovýh filtrov môžme naraziť na kliatbu dimenzionality, napríklad v aplikácii pri pohybe v troch rozmeroch [2].

V tomto prípade je niekedy možné použiť Rao Blackwell časticový filter. Tento algoritmus sa používa pri problémoch s vysokou dimenzionalitou, v ktorých ale existuje len malý počat stavov, ktoré sú výrazne nelineárne. V týchto prípadoch sa naozaj použije časticový filter, v ostatných stavoch, ktoré sú blízko gaussovských a lineárnych postačuje rozšírený Kalmanov filter. [2]

Pre pripomenutie v rámci stochastickej filtrácie je cieľom odhad hustoty pravdepodobnosti $p(x_k|Z_k)$, ktorá vyjadruje rozdelenie pravdepodobnosti stavového vektora v čase k v závistosti na pozorovaniach získaných do času k, predpokladáme problém kde čas budeme pokladať za diskrétnu veličinu. Časticový filter ponúka alternatívu k analytickému vyjadreniu v podobe náhodného generovania vzorkov, tzv. častíc, ktoré túto hustotu aproximujú. Časticiam sú taktiež priradené váhy, ktorých úlohou je niesť informáciu o tom nakoľko dobre daná častica reprezentuje neznámu hustotu $p(x_k|Z_k)$. Táto myšlienka náhodného generovania častíc vychádza z Monte Carlo metód.

Náhodné generovanie častíc sa v rámci časticového filtra nazýva vzorkovanie podľa dôležitosti (importance sampling). V prípade že generujeme po sebe nasledujúce súbory častíc, ktoré majú reprezentovať systém vyvíjajúci sa v čase, hovoríme o sekvenčnom vzorkovaní podľa dôležitosti (sequential importance sampling). Tieto metódy sú detailnejšie popísané v [3].

2.1 Základný popis fungovania časticového filtra

Pre jednoduchosť sa na okamih odpútame od obecného postupu a modelov stochastickej filtrácie.

Nech je cieľom aproximácia hustoty pravdepodovnosti $p(x)$. V reálnych aplikáciách býva táto hustota neznáma a teda budeme generovať častice z inej hustoty pravdepodobnosti $q(x)$, ktorá by mala byť $p(x)$ podobná. Následne generovaným časticiam priradzujeme váhy. Základným požiadavkom na hustotu $q(x)$ je rovnaký nosič ako má hustota $p(x)$. Nosič funkcie $f : \mathscr{D}(f) \to \mathbb{R}$ je definovaný ako $\text{supp}(f) = \{x \in \mathscr{D}(f) | f(x) \neq 0\}$, kde $\mathscr{D}(f)$ značí definičný obor funkcie f.

Nech $x \sim p(x)$. Potom

$$x \sim p(x) \sim p(x)\frac{q(x)}{q(x)} \sim q(x)\frac{p(x)}{q(x)}, \tag{2.1}$$

teda z hustoty $q(x)$ budeme nezávisle generovať N častíc a priradíme im váhy:

$$w^i = \frac{p(x^i)}{q(x^i)}, \tag{2.2}$$

kde $w^i \in \hat{N}$ značí váhu a určuje relevantnosť i-tej vygenerovej častice pri reprezentácii neznámej hustoty $p(x)$.

Toto je základný princíp generovania častíc. V prípade sekvenčného vzorkovania sa toto generovanie opakuje a rekurzívne sa realizujú Bayesovské odhady.

Nech teda $X_k = (x_0, \ldots, x_k)$ je trajektória systému a $\{X_k^i, w_k^i\}_{i=1}^N$ je postupnosť k súborov N náhodne generovaných častíc a ich normalizovaných váh w_k^i. Tieto súbory diskrétne aproximujú aposteriornú hustotu nasledovne:

$$p(X_k|Z_k) \approx \sum_{i=1}^{N} w_k^i \delta(X_k - X_k^i). \tag{2.3}$$

Funkčná hodnota δ funkcie pôsobiacej na vektor je 1 ak sú všetky prvky vektora nulové, inak je funkčná hodnota δ funkcie 0. Častice sú generované z hustoty pravdepodobnosti $q(X_k|Z_k)$. Váhy častíc sú počítané analogicky k nesekvenčnému vzorkovaniu:

$$w_k^i = \frac{p(X_k^i|Z_k)}{q(X_k^i|Z_k)}. \tag{2.4}$$

Za určitých dodatočných predpokladov kladených na hustotu $q(X_k|Z_k)$ je možné výpočet váh výrazne zjednodušiť.

Tieto predpoklady sú:

1. $q(X_k|Z_k)$ je možné faktorizovať nasledovne:

$$q(X_k|Z_k) = q(x_k|X_{k-1}, Z_k)q(X_{k-1}|Z_{k-1}) \tag{2.5}$$

2. $q(X_k|Z_k)$ nezávisí na celej trajektórii procesu:

$$q(x_k|X_{k-1}, Z_k) = q(x_k|x_{k-1}, z_k) \tag{2.6}$$

Pri splnení týchto predpokladov platí:

$$w_k^i \propto w_{k-1}^i \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, z_k)} \tag{2.7}$$

$$p(x_k|Z_k) \approx \sum_{i=1}^{N} w_k^i \delta(x_k - x_k^i) \tag{2.8}$$

Je možné ukázať, že pre N blížiace sa nekonečnu takto konštruovaná aproximácia konverguje k skutočnej hustote $p(X_k|Z_k)$ [7].

Posledným krokom je konštrukcia samotného odhadu stavového vektoru, najčastejšou voľbou býva vážený priemer častíc.

Najdôležitejšou voľbou pri aplikácii časticového filtra je práve hustota $q(X_k|Z_k)$, resp. $q(x_k|x_{k-1}^i, z_k)$, z ktorej sú častice generované. Táto voľba má najväčší vplyv na konečnú úspešnosť a presnosť odhadov poskytovaných časticovým filtrom.

Za optimálnu považujeme hustotu

$$q(x_k|x_{k-1}^i, z_k) = \frac{p(z_k|x_k, x_{k-1}^i)p(x_k|x_{k-1}^i)}{p(z_k|x_{k-1}^i)}, \tag{2.9}$$

avšak tú veľmi často nie je možné použiť. Jednoduchou a často používanou alternatívou je

$$q(x_k|x_{k-1}^i, z_k) = p(x_k|x_{k-1}^i), \tag{2.10}$$

kde $p(x_k|x_{k-1}^i)$ je prechodná hustota pravdepodobnosti.

Po vygenerovaní súboru časíc sú ich váhy normalizované.

Či už pri simulovanom prípade sínusovej vlny alebo pri plánovanej implementácii algoritmu pri odhade polohy mikrorobota je práve náhla zmena statového vektora práve bodom, v ktorom bude dochádzať k výraznej degenerácii. Tento jav je podrobne vysvetlený v nasledujúcej časti a budeme sa ho snažiť zmierniť aplikáciou logaritmickej tranformácie váh pred prípadným prevzorkovaním.

$$w_k^i \leftarrow \ln(w_k^i) - \max\{\ln(w_k^i)|i \in \hat{N}\} \tag{2.11}$$

$$w_k^i \leftarrow \frac{\exp(w_k^i)}{\sum_{j=1}^{N} \exp(w_k^j)} \tag{2.12}$$

Takto popísaný algoritmus generovania častíc budeme nazývať Basic Importance Sampling, skrátene BIS. V nasledujúcej kapitole budeme totiž diskutovať algoritmy vzorkovania ktoré vychádzajú z rovnakého princípu avšak obsahujú ďalšie úpravy, ktoré v niektorých prípadoch môžu priniesť výrazné spresnenie odhadov stavového vektora bez nutnosti navýšenia počtu častíc.

2.2 Prevzorkovanie

Samostatnú podkapitolu si zaslúži prevzokovanie, keďže je to dôležitý krok, ktorý môže nasledovať po vygenerovaní nového súboru v rámci sekvenčného vzorkovania. Tento krok napomáha tomu, aby schopnnosť častíc efektívne reprezentovať hustotu $p(X_k|Z_k)$ časom neupadala.

Pri použití sekvenčného generovania častíc vždy nasledujúca generácia častíc vychádza z tej predchádzajúcej. V prípade že nejaká častica aproximuje hustotu veľmi zle a nie je vôbec reprezentatívna by dôsledkom mohlo byť, že aj častica v nasledujúcom generovanom súbore nebude reprezentatívna

vzhľadom k $p(X_k|Z_k)$. Agoritmus časticového filtra by ponútal set nezávislých trajektórií, ktoré by pravdepodobne časom divergovali. Po výpočte normalizovaných váh by došlo k tomu, že len jediná častica/trajektória v danom súbore má váhu 1 a ostatné častice majú váhu 0. Tento jav sa v literatúre nazýva degenerácia vzorku alebo ochudobnenie vzorku [2].

Mieru degenerácie sme schopní kvantifikovať pomocou efektívnej veľkosti vzorku, ktorá je definovaná ako

$$N_{eff} = \frac{N}{1 + \frac{Var(w^i_{k|k})}{(E(w^i_{k|k})^2)}} = \frac{N}{1 + N^2 Var(w^i_{k|k})}, \tag{2.13}$$

kde N je počet generovaných častíc v súbore, ako E označujeme strednú hodnotu, ako Var rozptyl. N_{eff} teda bude nadobúdať hodnoty medzi celkovým počtom častíc N v prípade, ak sú váhy všetkých častíc rovné $\frac{1}{N}$, a 1 v prípade, že $w^i_{k|k} = 0$ s pravdepodobnosťou $\frac{N-1}{N}$ a $w^i_{k|k} = 1$ s pravdepodobnosťou $\frac{1}{N}$ [2]. Efektívny počet časíc aproximujeme ako:

$$\hat{N_{eff}} = \frac{1}{\sum_{i=1}^{N} (w^i)^2} \tag{2.14}$$

Aproximácia zdieľa horný aj dolný limit so skutočnou hodnotou efektíveho počtu častíc, nadobúda hodnoty N ak majú všetky častice rovnakú váhu a 1 ak má jedna častica váhu 1 a ostatné 0.

Nízka hodnota odhadu efektívnej veľkosti vzorku oproti reálnemu počtu generovaných častíc N teda značí vysokú mieru degenerácie súboru.

Odpoveďou na degeneráciu je práve už spomínané prevzorkovanie. Krok prevzorkovania je do algoritmu najčastejšie zaradené vprípade, že ak hodnota efektívnej veľkosti vzorku je menšia ako nejaký vopred zvolený prah, ktorý budeme označovať N_{thr}. Tento prah je možné zvoliť rôzne, napríklad jednoducho ako polovicu počtu častíc v súbore [3].

Algoritmov prevzorkovania je pomerne veľké množstvo, tie najviac používané je ale možné popísať nasledovne. V rámci prevzorkovania sa častice s nízkymi váhami zahodia, keďže práve tie najhoršie reprezentujú neznámu husotu $p(X_k|Z_k)$. Miesto týchto častíc sú do plného počtu N do súboru doplnené častice s vysokými váhami. Následne sa takto zkonštruovanému súboru častíc priradia rovnaké váhy.

Prevzorkovanie je možné implementovať prostedníctvom rôznych algoritmov ako napríklad reziduálne alebo systematické prevzorkovanie, ktoré sú podrobne popísané napríklad v [3] alebo [8]. Pri popise algoritmov budeme predpokladať že vyžadujeme rovnaký počet častíc na vstupe i výstupe každého algoritmu prevzorkovania.

2.2.1 Multinomické prevzorkovanie

Multinomické prevzorkovanie je najjednochším diskutovaným algoritmom. Pozostáva z dvoch krokov. V prvom generujeme N nezávislých realizácií $(u_1, \ldots, u_N)$ uniformného rozdelenia $\mathcal{U}(0, 1]$. V druhom kroku použijeme tieto realizácie k výberu prevzorkovaných častíc. Častica X^i_k je vybratá ak $\sum_{j=1}^{k-1} w^i_j < u_k \leqq \sum_{j=1}^{k} w^i_j$. Pri použití tochto algoritmu je možné že daná častica bude vybraná minimálne 0-krát a maximálne N-krát. Multinomické vzorkovanie nie je efektívny algoritmus a teda v praxi nie často používaný [2].

Pseudoalgoritmus multinomického prevzorkovania:

if $\frac{1}{\sum_{i=1}^{N} (w^i_k)^2} < N_{thr}$ **then**
  for $i = 1 : N$ **do**
    $c_i = \sum_{j=1}^{i} w^j_k$ ▷ kumulatívna suma váh súboru $(c_1, \ldots, c_N)$

end for
$i = 0$
while $n < N$ **do**
$u \sim \mathcal{U}(0, 1]$
$i = 1$
while $c_i < u$ **do**
$i = i + 1$
end while
$n = n + 1$
$x_k^j = x_k^i$
end while
end if

2.2.2 Reziduálne prevzorkovanie

Reziduálne prevzorkovanie taktiež prebieha v dvoch krokoch. V prvom sú identifikované a zreplikované všetky častice, ktorých váha je vyššia ako $\frac{1}{N}$. Druhý krok znamená vzorkovanie použitím zbytku častíc, napríklad použitím multinomického vzorkovania. Pravdepobnosť výberu častice v tomto kroku je úmerná jej reziduálnej váhe.

Ak je v druhom kroku miesto multinomického prevzorkovania použité systematické prevzorkovanie hovoríme o reziduálnom systematickom prevzorkovaní [2].

Pseudoalgoritmus reziduálneho prevzorkovania:

if $\frac{1}{\sum_{i=1}^{N}(w_k^i)^2} < N_{thr}$ **then**
for $i = 1 : N$ **do**
$N_k^i = \lfloor (N\ w_k^i)$
$u = u + \frac{N_k^i}{N - w_k^i}$
end for
$j = 0$ ▷ Deterministická replikácia častíc
for $i = 1 : N$ **do**
for $h = 1 : N_k^i$ **do**
$j = j + 1$
$x_k^j = x_k^i$
end for
end for
$N = n$
for $i = 1 : N$ **do** ▷ Vzorkovanie zbytku častíc
$w_k^i = w_k^i\ \frac{N}{N - N_k^i}$
end for
Multinomické/systematické prevzorkovanie
end if

2.2.3 Systematické (stratifikované) prevzorkovanie

Systematické prevzorkovanie spočíva v porovnávaní členov kumulatívnej sumy normalizovaných váh častíc, ktoré majú byť prevzorkované $(c_1, \ldots, c_N)$ a kumulatívnej sumy, kde u_1 je generované z uni-

formného rozdelenia $\mathcal{U}(0, \frac{1}{N})$ a rozdiely medzi ďalšími po sebe nasledujúcimi členmi sú $\frac{1}{N}$. Takto sme vytvorilli náhodný výber uniformné rozdelenie na intervale $(0, 1)$. Pre každý člen u_j sumy $(u_1, \ldots, u_N)$ je nájdený posledný člen $(c_1, \ldots, c_N)$ menší alebo rovný - c_i. J-ta častica prevzorkovaného súboru je i-ta častica pôvodného. Dôsledkom tohto postupu je, že pravdepodobnosť výskytu j-tej častice v súbore po prevzorkovaní je rovná jej váhe w_k^j [3]. V práci bude implementované systematické prevzorkovanie.

Algoritmus systematického prevzorkovania:

if $\frac{1}{\sum_{i=1}^{N}(w_k^i)^2} < N_{thr}$ **then**
  for $i = 1 : N$ **do**
    $c_i = \sum_{j=1}^{i} w_k^j$ ▷ kumulatívna suma váh súboru $(c_1, \ldots, c_N)$
  end for
  $i = 1$
  $u_1 \sim \mathcal{U}(0, \frac{1}{N})$ ▷ kumulatívna suma $(u_1, \ldots, u_N)$
  for $j = 1 : N$ **do**
    $u_j = u_1 + \frac{j-1}{N}$
  end for
  while $u_j > c_i$ **do**
    $i = i + 1$
  end while
  $x_k^j = x_k^i$ ▷ j-ta častica prevzorkovaného súboru a jej váha
  $w_k^j = \frac{1}{N}$
end if

Ako bolo zdôvodnené, prevzorkovanie je veľmi užitočný a široko implementovaný krok v rámci sekvenčného vzorkovania. Jedniným potenciálne negatívnym dopadom na budúce generácie častíc je možné ochudobnenie súboru. Nahradením častíc s nízkou váhou klesá celkový počet unikátnych častíc v súbore a tým i neurčitosť a teda i informácia, ktorú daný súbor častíc môže poskytnúť. Táto skutočnoť je hlavný dôvod prečo neprevzorkujeme automaticky v každom kroku ale len vtedy ak je to skutočne potrebné [2].

Jedným z riešení poklesu rôznorodosti vzorku pri prevzorkovaní je medzikrok nazývaný "jittering". Spočíva v tom, že šum modelu alebo pozorovaní je umelo navýšený, čo spôsobí to, že pravdepodobnosť prevzorkovania narastá pomalšie s tým ako "nevhodná"je daná častica pre reprezentáciu aposteriornej hustoty [2].

Spomenuté boli len najzákladnejšie algoritmy vzorkovania. Pri implementácii časticového filtra je treba myslieť na výpočetnú náročnosť - okamžite zrejmé potenciálne nároky sú veľký počet generovaných častíc a v niektorých aplikáciách potreba odhadu v reálnom čase. V tomto smere je teda najnáročnejšie samotné generovanie častíc, výpočet váh a následné prevzorkovovanie. Zatiaľčo prvé dva body sú paralelizovateľné, pre uvedené algoritmy prevzorkovania to neplatí. V posledných rokoch vznikajú nové algoritmy, ktoré už paralelizáciu umožňujú a teda znovu zlepšujú použiteľnosť časticových filtrov [8].

Pokročilejšie algoritmy taktiež nemusia pristupovať pristupovať k prevzorkovaniu každej indiduálnej častice rovnako a častice po prevzorkovaní nemusia mať rovnaké váhy ako to platilo pre doteraz spomínené metódy prevzorkovania. Napríklad metódy zlúčeného prevzorkovania (compound sampling) je založené na rozdelení celého súboru častíc do disjunktých skupín, najčastejšie pomocou rozdelenia ich váh podľa prahov podľa veľkosti. Následne sú častice v jednej skupine - s váhami podobnej veľkosti - prevzorkované určitým spôsobom[8]. Práve tento rozvoj zvyšuje naďalej rozširuje možnosť využitia časticových filtrov [2].

2.3 Algoritmus časticového filtra

Popísaný algoritmus časticového filtra odpovedá nasledujúcemu algoritmu. Algoritmus bol v tejto podobe súčasťou bakalárskej práce [1] a jeho pôvodným zdrojom je [3].

Pseudoalgoritmus časticového filtra:

for $k = 1 : K$ **do**
  for $i = 1 : N$ **do** ▷ sekvenčné vzorkovanie
    $x_k^i \sim q(x_k|x_{k-1}^i, z_k)$ ▷ generovanie i-tej častice a výpočet jej váhy
    $w_k^i = w_{k-1}^i \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, z_k)}$
  end for
  for $i = 1 : N$ **do**
    $w_k^i = \ln(w_k^i) - \max\{\ln(w_k^i)|i \in \hat{N}\}$ ▷ logaritmická transformácia váh
  end for
  for $i = 1 : N$ **do**
    $w_k^i = \frac{\exp(w_k^i)}{\sum_{j=1}^N \exp(w_k^j)}$ ▷ normalizácia váh
  end for
  Prevzorkovanie
end for

2.4 Časticový filter pre simulovaný experiment

V tejto časti aplikujeme predstavený časticový filter na simulovaný príklad sínusovej vlny definovaný v časti 1.2, ukážeme jeho nadradenosť voči jednoduchšiemu Kalmanovmu filtru a jeho potenciál pre implementáciu pre reálnu aplikáciu detekcie polohy mikrorobota. Kalmanov i časticový filter sú často používané pre úlohu detekcie polohy objektov, napríklad v [12], kde sú simultánne využité výhody oboch algoritmov.

Pre tento príklad nie je možné použiť optimálnu hustotu, teda namiesto nej budeme častice generovať z prechodnej hustoty.

2.4.1 Voľba metrík presnosti odhadu

Pre takto definovaný príklad aplikujeme časticový filter pre dva rôzne modely podľa voľby hustôt p a q. Presnosť bude vyhodnotená rovnako ako pre Kalmanov filter v časti 1.2 pre 1000 setov pozorovaní.

Presnosť jednotlivých filtrov je vyhodnotená na základe absolútnych a kvadratických odchýlok, y_k je skutočná hodnota sínusovej vlny z (1.18), jej odhad je značený $\widehat{y_k}$. Vyhodnotíme aj priemer stredných absolútnych odchýlok jednotlivých zložiek stavového vektora - a_k, φ_k.

Odhady sínusovej vlny v čase $k \in \hat{K}$ sú konštruované ako vážené priemery funkčných hodnôt sínusovej vlny v generovaných časticiach $x_k^i = (a_k^i, \varphi_k^i)$. Odhady amplitúdy a fázy boli konštruované analogicky:

$$\widehat{y_k} = \sum_{i=1}^{N} a_k^i \, \sin(\omega k + \varphi_k^i) \, w_k^i \tag{2.15}$$

$$\widehat{a_k} = \sum_{i=1}^{N} a_k^i \, w_k^i \tag{2.16}$$

$$\widehat{\varphi_k} = \sum_{i=1}^{N} \varphi_k^i \, w_k^i \tag{2.17}$$

pred prípadným prevzorkovaním.

- Priemer stredných absolútnych odchýlok sínusovej vlny pre n spustení filtra pre daný set pozorovaní
$$\bar{\Delta} = \frac{1}{n} \sum_{j=1}^{n} \Delta^j = \frac{1}{n} \sum_{j=1}^{n} \left(\frac{1}{100} \sum_{k=1}^{100} |\widehat{y_k}^j - y_k| \right)$$

- Priemer stredných absolútnych odchýlok amplitúdy pre n spustení filtra pre daný set pozorovaní
$$\bar{\Delta}_a = \frac{1}{n} \sum_{j=1}^{n} \Delta^j = \frac{1}{n} \sum_{j=1}^{n} \left(\frac{1}{100} \sum_{k=1}^{100} |\widehat{a_k}^j - a_k| \right)$$

- Priemer stredných absolútnych odchýlok fázy pre n spustení filtra pre daný set pozorovaní
$$\bar{\Delta}_\varphi = \frac{1}{n} \sum_{j=1}^{n} \Delta^j = \frac{1}{n} \sum_{j=1}^{n} \left(\frac{1}{100} \sum_{k=1}^{100} |\widehat{\varphi_k}^j - \varphi_k| \right)$$

- Priemer absolútnych odchýlok sínusovej vlny v bode skoku pre n spustení filtra pre daný set pozorovaní
$$\bar{\Delta}_{51} = \frac{1}{n} \sum_{j=1}^{n} \Delta_{51}^j = \frac{1}{n} \sum_{j=1}^{n} |\widehat{y_{51}}^j - y_{51}|$$

- Priemer stredných kvadratických odchýlok sínusovej vlny pre n spustení filtra pre daný set pozorovaní
$$\bar{\delta} = \frac{1}{n} \sum_{j=1}^{n} \delta^j = \frac{1}{n} \sum_{j=1}^{n} \left(\frac{1}{100} \sum_{k=1}^{100} (\widehat{y_k}^j - y_k)^2 \right)$$

- Priemer kvadratických odchýlok sínusovej vlny v bode skoku pre n spustení filtra pre daný set pozorovaní
$$\bar{\delta}_{51} = \frac{1}{n} \sum_{j=1}^{n} \delta_{51}^j = \frac{1}{n} \sum_{j=1}^{n} (\widehat{y_{51}}^j - y_{51})^2$$

2.4.2 Špecifikácia modelov a porovanie algoritmov

Budeme uvažovať nasledujúce dva modely:

1. Gaussovský model

   Budeme predpokladať, že prechodná hustota $p(x_k|x_{k-1}^i)$ je normálna. Hustotu $q(x_k|x_{k-1}^i, z_k)$ zvolíme ako $p(x_k|x_{k-1}^i)$. Teda:

$$q(x_k|x_{k-1}^i, z_k) = p(x_k|x_{k-1}^i) = \mathcal{N}(x_k; x_{k-1}^i, Q)$$
$$w_k^i \propto w_{k-1}^i \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, z_k)} = w_{k-1}^i p(z_k|x_k^i) =$$
$$= w_{k-1}^i \mathcal{N}(z_k; a_k^i sin(\omega k + \varphi_k^i), R) = w_{k-1}^i \mathcal{N}(z_k; a_k^i sin(\omega k + \varphi_k^i), 0.04)$$

   Analogicky ku Kalmonovmu filtru budeme skúmať vplyv hodnoty rozptylu na presnoť odhadu.

$$Q = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix}$$

   Hypotéza: Optimálna voľba σ^2 bude väčšia ako pre Kalmanov filter.

σ^2	$\bar{\Delta}$	$\bar{\Delta}_{1\to50}$	$\bar{\Delta}_{51\to100}$	$\bar{\Delta}_{51}$	$\bar{\delta}$	$\bar{\delta}_{1\to50}$	$\bar{\delta}_{51\to100}$	$\bar{\delta}_{51}$
0,001	1,1694	0,0776	2,2611	4,1882	3,2774	0,0113	6,5434	17,5501
0,01	0,4585	0,0990	0,8181	3,8977	0,8296	0,0158	1,6435	15,2319
0,1	0,1990	0,1359	0,2621	3,2872	0,1763	0,0290	0,3236	11,0229
1	0,1893	0,1589	0,2197	2,3318	0,1053	0,0397	0,1708	6,0085
10	0,1999	0,1689	0,2309	0,4244	0,0742	0,0450	0,1034	0,3267
100	0,2506	0,1917	0,3095	0,3024	0,1748	0,0657	0,2840	0,1643
1000	0,4471	0,3301	0,5641	0,5649	1,7778	0,7222	2,8334	1,7732

Tabuľka 2.1: Porovnanie priemerov chýb pre Gaussovský model, 50 častíc a rôzne voľby σ^2

   Optimálny rozptyl je $\sigma^2 = 10$, väčší ako $0,01$ pre Kalmanov filter.

2. Zmiešaný model

Ako alternatívu použijeme model kde použijeme zmiešanú prechodnú hustotu. Je možné že táto voľba umožní lepšie prispôsobenie sa chrakteru problému - konštatnosť stavového vektora na dlhých úsekoch i náhla zmena - v čase $k = 51$ je zmena $|x_k - x_{k-1}|$ veľká.

Úlohou $q(x_k|x_{k-1}^i, z_k)$ je prostredníctvom generovaia častíc preskúmavať priestor možných hodnôt stavového vektora v nasledujúcom časovom kroku. Táto hustota teda musí umožnosť veľkú zmenu.

$$q(x_k|x_{k-1}^i, z_k) = \alpha\, \mathcal{N}(x_k; x_{k-1}^i, Q_1) + (1-\alpha)\, \mathcal{N}(x_k; x_{k-1}^i, Q_2),$$

kde $\alpha > 0,5$ a hodnota rozptylu Q_2 je relatívne veľká.

Prechodná hustota $p(x_k|x_{k-1}^i)$ je taktiež zvolená ako zmiešaná normálna

$$p(x_k|x_{k-1}^i) = \beta\, \mathcal{N}(x_k; x_{k-1}^i, Q_3) + (1-\beta)\, \mathcal{N}(x_k; x_{k-1}^i, Q_4),$$

β je blízko 1 a rozptyl Q_3 je veľmi malý.

S touto voľbou by mali byť generované aj častice s veľkým rozptylom, ktoré by mali mať schopnosť zachytit zmenu v čase 51, mimo tohto času dostanú malú váhu. Ak bude táto voľba hustoty efektívnejšia, pre dosiahnutie danej presnosti by mohol byť postačujúci menší počet generovaných časíc.

Zmiešané hustoty boli zvolené nasledovne:

$$q(x_k|x_{k-1}^i, z_k) = 0,6\, \mathcal{N}(x_k; x_{k-1}, 10) + 0,4\, \mathcal{N}(x_k; x_{k-1}, 100) \tag{2.18}$$

$$p(x_k|x_{k-1}^i) = 0,99\, \mathcal{N}(x_k; x_{k-1}, 10^{-4}) + 0,01\, \mathcal{N}(x_k; x_{k-1}, 100) \tag{2.19}$$

Pre $n = 10$:

filter	počet častíc	$\bar{\Delta}$	$\bar{\Delta}_a$	$\bar{\Delta}_\varphi$	$\bar{\Delta}_{51}$	$\bar{\delta}$	$\bar{\delta}_{51}$
rozšírený Kalmanov pre $\sigma^2 = 0,01$		0,1625	0,2624	1,6314	2,9756	0,1271	8,8542
častícový pre Gaussovký model a $\sigma^2 = 10$	50	0,2160	3,9138	2,4679	0,4325	0,0792	0,2814
	100	0,1898	3,3491	2,5582	0,3408	0,0568	0,1398
	500	0,1792	3,0354	2,5032	0,3017	0,0514	0,0945
častícový pre Gaussovký model a $\sigma^2 = 0,1$	50	0,2115	0,4098	1,6986	3,3199	0,1825	11,1521
	100	0,2010	0,8262	1,6925	3,1138	0,1601	9,8771
	500	0,1942	0,3496	1,6815	2,7826	0,1292	7,8693
častícový pre zmiešaný model	50	0,2639	5,6883	2,4876	0,4767	0,1689	0,3008
	100	0,2105	4,9536	2,5463	0,3905	0,0735	0,1659
	500	0,1832	3,5550	2,5592	0,3080	0,0530	0,0954

Tabuľka 2.2: Porovnanie priemerov chýb

Keďže častícový filter je na rozdiel od Kalmanovho stochastická metóda, je potrebné kvalifikovať neurčitosť, kterej zdrojom sú rôzne realizácie častíc. Pozorujeme teda najväčšie a najmenšie chyby v n kolách.

Pre $n = 10,\ j \in \hat{n}$:

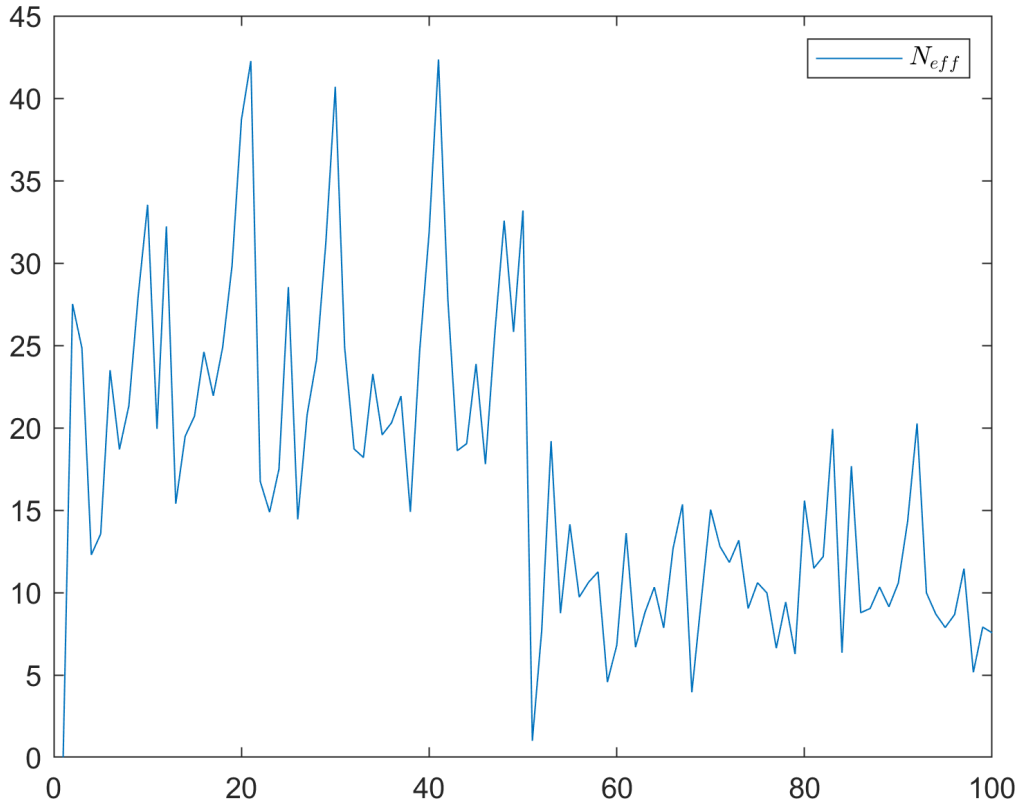
filter	počet častíc	$\max\{\Delta^j\}$	$\max\{\Delta^j_{51}\}$	$\max\{\delta^j\}$	$\max\{\delta^j_{51}\}$
rozšírený Kalmanov pre $\sigma^2 = 0,01$		0,1625	2,9756	0,1271	8,8542
častícový pre Gaussovký model a $\sigma^2 = 10$	50	0,2357	0,9066	0,0921	0,8219
	100	0,1988	0,6990	0,0619	0,4886
	500	0,1842	0,4080	0,0530	0,1664
častícový pre Gaussovký model a $\sigma^2 = 0,1$	50	0,2270	3,8584	0,2358	14,8869
	100	0,2141	3,6982	0,2065	13,6766
	500	0,2044	3,2457	0,1641	10,5344
častícový pre zmiešaný model	50	0,4805	1,0198	0,7417	1,0400
	100	0,2298	0,5859	0,1005	0,3433
	500	0,1896	0,3726	0,0610	0,1388

Tabuľka 2.3: Porovnanie najväčších chýb

filter	počet častíc	$\min\{\Delta^j\}$	$\min\{\Delta^j_{51}\}$	$\min\{\delta^j\}$	$\min\{\delta^j_{51}\}$
rozšírený Kalmanov pre $\sigma^2 = 0,01$		0,1625	2,9756	0,1271	8,8542
častícový pre Gaussovký model a $\sigma^2 = 10$	50	0,2037	0,0130	0,0680	0,0002
	100	0,1845	0,1757	0,0527	0,0309
	500	0,1755	0,2365	0,0492	0,0559
častícový pre Gaussovký model a $\sigma^2 = 0,1$	50	0,1995	2,8734	0,1344	8,2561
	100	0,1850	2,4248	0,1013	5,8795
	500	0,1811	2,0080	0,0814	4,0320
častícový pre zmiešaný model	50	0,2036	0,0540	0,0755	0,0029
	100	0,1892	0,1980	0,0577	0,0392
	500	0,1785	0,2832	0,0494	0,0802

Tabuľka 2.4: Porovnanie najmenších chýb

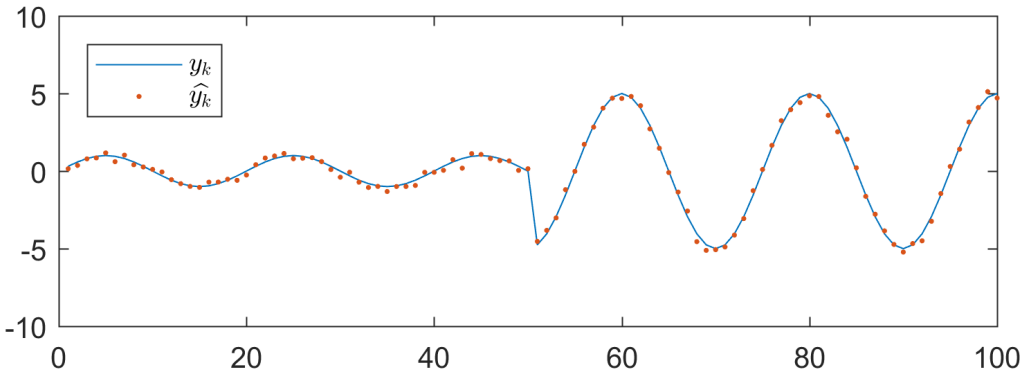
Na tomto mieste pre ilustráciu vykreslíme i priebeh efektívneho počtu častíc pri aplikácii častícového filtra pri použití základného modelu.



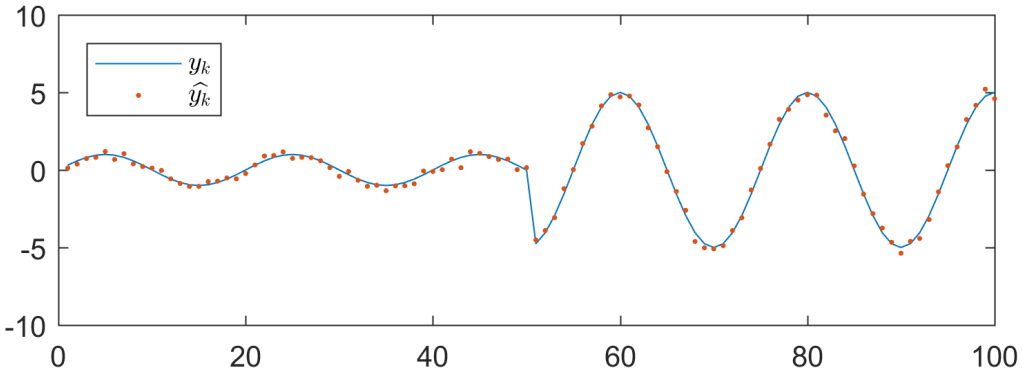
Obr. 2.1: Odhad efektívneho počtu častíc v priebehu procesu odhadu sínusovej vlny, v každom súbore bolo generovaných 100 častíc, na osi x index postupných realizácií sínusovej vlny.

Tento priebeh odhadu efektívneho počtu častíc poskytuje veľmi dobrú možnosť ukážky degenerácie vzorku. Pre prvú realizáciu sínusovej vlny sme algoritmu pravdivé hodnoty parametrov prezradili, nie sú generované žiadne častice. V časovom kroku 50, kde dochádza k náhlej zmene hodnoty oboch parametrov vidíme, že odhadu efektívneho počtu častíc prudko klesne na jedinú časticu. Potom hodnota odhadu efektívneho počtu častíc zase pomaly rastie.

Časticový filter je naozaj schopný poskytnúť presný odhad aj v mieste skoku a je výrazným zlepšením Kalmanovho filtra. Pri použití strednej absolútnej chyby celého procesu ako metriky presnosti, časticový filter pre 500 čatíc je porovateľný s rozšíreným Kalmanovým filtrom. Zmiešaný model sa v tomto prípade a pre tieto voľby parametrov neukázal ako lepší v porovnaní s Gaussovým. V tabuľkách sú zahrnuté aj chyby pre časticový filter pre malý rozptyl ($q = 0, 1$) keďže zatiaľ čo časticový filter s veľkým rozptlom poskytuje presné odhady sínusovej vlny, odhady jednotlivých zložiek stavového vektora sú veľmi nepresné.



Obr. 2.2: časticový filter pre Gaussovský model a 500 častíc



Obr. 2.3: časticový filter pre zmiešaný model a 500 častíc

2.5 Zložitejšie algoritmy vzorkovania

Základný algoritmus generovania častíc, ktorý bol doteraz diskutovaný označujeme BIS (basic importance sampling). Pre krátke pripomenutie BIS zhrnieme daný časový krok k nasledovne:

1. sekvenčné vzorkovanie
2. transfomácia váh
3. normalizácia váh
4. prípadné prevzorkovanie

Hlavným problémom BIS je voľba proposal distribúcie q, ak je zvolená proposal distribúcia ďaleko od ozajstej hustoty p, algoritmus sa stáva veľmi neefektívym, keďže sú generované nereprezentatívne častice. Často však nevieme zvoliť proposal distribúciu lepšie ako prechodnú hustotu ako to bolo popísané v rámci BIS. Riešením sú algoritmy popísané v tejto časti, keďže ich cieľom je automaticky

približovať proposal distribúciu ozajstej aposteriórnej hustote p. Umožnia nám teda generovať kvalitnejšie častice ale zase za cenu navýšenej výpočetnej náročnosti. Či sa použiť tieto zložitejšie algoritmy oplatí záleží na aplikácii a budeme to skúmať v kapitole 5.

V tejto časti teda predstavíme dva zložitejšie alternatívne algoritmy k BIS - adaptívne vzorkovanie podľa dôležitosti (adaptive importance sampling, skrátene AIS) [9] a hromadné adaptívne vzorkovanie podľa dôležitosti (adaptive multiple importance sampling, skrátene AMIS) [10]. Oba tieto algoritmy sú založené na postupnom spresňovaní hustoty q, z ktorej sa generujú častice a to v každom časovom kroku k. Tento proces sa uskutočňuje medzi krokmi 3 (normalizácia váh) a 4 (prípadné prevzorkovanie).

2.5.1 Adaptívne vzorkovanie podľa dôležitosti

Jedným z prvých zdrojov, v ktorých bol tento aloritmus popísaný je [9].

Cieľom je generovať častice z husototy $p(x)$, $x \in \mathbb{R}^n$, nie je to však možné. Chceme zvoliť husotu q tak, aby p čo najbližie odpovedala a bolo možné z nej generovať častice. Ďalej budeme požadovať aby q nemala ťažšie chvosty ako p.

Budeme predpokladať, že hustota q patrí do rodiny pravdepodobnostných hustôt $q(x;\lambda) \in Q = \{q_\lambda | \lambda \in \Lambda\}$. Práve parameter λ sa postupne spresňuje na základe jednotlivých generácii častíc aby q presnejšie odpovedala p.

Nech $\lambda = (\lambda_1, \ldots, \lambda_m)'$a definujeme funkciu $\xi(x) = (\xi_1(x), \ldots, \xi_m(x))$. Nech ďalej platí vzťah:

$$\lambda = \mathbb{E}_p[\xi(x)] = \frac{\int \xi(x) p(x)\, dx}{\int p(x)\, dx}. \tag{2.20}$$

Funckia $\xi(x)$ je zvolená tak, aby zachytávala požiadavky kladené na parameter λ, napríklad vo vzťahu k hustote p.

Algoritmus adaptívneho vzorkovania sa dá zhrnúť nasledovne. V tejto podobe bol uvedený v bakalárskej práci [1], prvotným zdrojom je [9].

- zvolíme kritérium zastavenia - podmienku, ktorá ak je splnená, proces spresňovania hustoty q ukončíme
- zvolíme prvotný odhad parametru λ, označíme $\lambda^{(0)}$
- za prvé priblíženie hustoty q zvolíme $q_{\lambda^{(0)}}$, označíme q_0
- generujeme častice $x_1^i,\ i \in \widehat{n_1}$ z hustoty q_0 a priradíme im váhy $w_1^i = \frac{p(x_1^i)}{q_0(x_1^i)}$
- definujeme funkcionál $N^{(1)}(h) = \sum_{i=1}^{n_1} h(x_1^i)\, w_1^i$, vyhodnotíme $N^{(1)}(\xi_j),\ j \in \widehat{m}$ a $N^{(1)}(1)$
- nová hodnota parametru λ je $\lambda^{(1)} = \left(\frac{N^{(1)}(\xi_1)}{N^{(1)}(1)}, \ldots, \frac{N^{(1)}(\xi_m)}{N^{(1)}(1)}\right)'$
- položíme $q^{(1)} = q(x; \lambda^{(1)})$
- l-tý krok tohto procesu prebieha nasledovne:
  - $x_l^1, \ldots, x_l^{n_l} \sim q^{(l-1)}(x) = q(x; \lambda^{(l-1)})$
  - $w_l^i = \frac{p(x_l^i)}{q^{(l-1)}(x_l^i)}$
  - $N^{(l)}(h) = \sum_{i=1}^{n_l} h(x_l^i)\, w_l^i$, vyhodnotíme $N^{(l)}(\xi_j),\ j \in \widehat{m}$ a $N^{(l)}(1)$
  - $\lambda^{(l)} = \left(\frac{N^{(l)}(\xi_1)}{N^{(l)}(1)}, \ldots, \frac{N^{(l)}(\xi_m)}{N^{(l)}(1)}\right)'$
  - $q^{(l)} = q(x; \lambda^{(l)})$
- tento krok sa opakuje kým nie je splnené kritérium zastavenia

2.5.2 Hromadné adaptívne vzorkovanie podľa dôležitosti

Hromadné adaptívne vzorkovanie podľa dôležitosti je ďalším krokom v zlepšovaní algoritmu vzorkovania. Obsahuje všetky kroky, ktoré sú obsiahnuté v adaptívnom vzorkovaní podľa dôležitosti ale odstraňuje slabinu tohto algoritmu, ktorou je plýtvanie vzorkov - po každom spresnení sa totiž častice zahodia a odhad stavového vektora je konštruovaný výlučne z posledného súboru po ukončení spesnenia hustoty q pre daný časový krok.

Na rozdiel od adaptívneho vzorkovania podľa dôležitosti teda hromadné adaptívne vzorkovanie podľa dôležitosti častice nezahadzuje. Namiesto toho iba aktualizuje ich váhy použitím aktuatualizovanej husoty q. Pri popise tohto algoritmu bol použitý zdroj [10] a algoritmus je v tejto podobe aj súčasťou bakalárskej práce [1]. Body, v ktorých sa algoritmus hromadného adaptívneho vzorkovania podľa dôležitosti líši od algoritmu adaptívneho vzorkovania podľa dôležitosti sú zvýraznené.

- zvolíme kritérium zastavenia - podmienku, ktorá ak je splňená, proces spresovania hustoty q ukončíme
- zvolíme prvotný odhad parametru λ, označíme $\lambda^{(0)}$
- za prvé priblíženie hustoty q zvolíme $q_{\lambda^{(0)}}$, označíme q_0

- generujeme častice $x_1^i,\ i \in \widehat{n_1}$ z hustoty q_0 a priradíme im váhy $w_1^i = \frac{p(x_1^i)}{q_0(x_1^i)}$
- spočteme $\underline{\delta_1^i = n_1 q_0(x_1^i)}$
- definujeme funkcionál $N^{(1)}(h) = \sum_{i=1}^{n_1} h(x_1^i)\, w_l^i$ a vyhodnotíme $N^{(1)}(\xi_j),\ j \in \widehat{m}$ a $N^{(1)}(1)$
- nová hodnota parametru λ je $\lambda^{(1)} = \left(\frac{N^{(1)}(\xi_1)}{N^{(1)}(1)}, \ldots, \frac{N^{(1)}(\xi_m)}{N^{(1)}(1)}\right)'$
- položíme $q^{(1)} = q(x; \lambda^{(1)})$
- l-tý krok tohto procesu prebieha nasledovne:
  - generujeme nový súbor častíc $x_l^1, \ldots, x_l^{n_l} \sim q^{(l-1)}(x) = q(x; \lambda^{(l-1)})$
  - vyhodnotíme $\underline{\delta_l^i = n_1 q_0(x_l^i) + \sum_{t=1}^{l} n_t q_t(x_l^i)}$
  - $\underline{w_l^i = p(x_l^i)\, \frac{\delta_l^i}{\sum_{t=1}^{l} n_t}}$
  - pre $t = 1, \ldots, l-1$ a $i = 1, \ldots, n_t$ váhy častíc generovaných v predchádzajúcich cykloch aktualizujeme nasledovne:

    $\underline{\delta_t^i \leftarrow \delta_t^i + n_t q_{l-1}(x_l^i)}$

    $\underline{w_l^i \leftarrow p(x_l^i)\, \frac{\delta_l^i}{\sum_{t=1}^{l} n_t}}$
  - $\underline{\text{sčítame cez všetky častice:}}\ \ N^{(l)}(h) = \sum_{t=1}^{l} \sum_{i=1}^{n_t} h(x_t^i)\, w_t^i,$ vyhodnotíme $N^{(l)}(\xi_j),\ j \in \widehat{m}$ a $N^{(l)}(1)$
  - $\lambda^{(l)} = \left(\frac{N^{(l)}(\xi_1)}{N^{(l)}(1)}, \ldots, \frac{N^{(l)}(\xi_m)}{N^{(l)}(1)}\right)'$
  - $q^{(l)} = q(x; \lambda^{(l)})$
- tento krok sa opakuje kým nie je splnené kritérium zastavenia

2.5.3 Adaptívne a hromadné adaptívne sekvenčné vzorkovanie

Prvotný odhad parametru $\lambda^{(0)}$ v kroku k je získaný zo súboru častíc generovaného sekvenčným vzorkovaním zo súboru generovaného a prevzorkovaného v kroku $k-1$ rovako ako pri nasledujúcich súboroch v kroku k určených na nasledovné spresnenie odhadu: $\lambda^{(0)} = \left(\frac{N^{(0)}(\xi_1)}{N^{(0)}(1)}, \ldots, \frac{N^{(0)}(\xi_m)}{N^{(0)}(1)}\right)'$.

Najjednoduchšou voľbou je v prvotnom súbore i všetkých nasledujúcich generovať rovnaký počet častíc - najmä ak predpokladáme napríklad model, ktorý nejakým spôsobom závisí na predchádzajúcom kroku $k-1$. V tomto prípade môže byť po skončení procesu spresňovania posledný generovaný súbor prevzorkovaný a použitý pre odhad v kroku $k+1$.

Kapitola 3

Slepá dekonvolúcia

Operácia konvolúcie je široko využívaná, jednou z oblastí, v ktorých je veľmi často používaná je oblasť spracovania obrazu. Jedným z jej využití v tejto oblasti je modelovanie rozmazania obrázku, ktoré môže byť spôsobené napríklad použitím digitálnej kamery alebo jej pohybom behom zaznamenania fotografie. Dôvodom je to, že konvolúciou vzniká rozmazaný obrázok - tento prípad sa vyskytuje i v úlohe sledovania robota - a dekonvolúcia umožňuje z rozmazaného obrázku znovu získať ako obrázok ostrý tak aj konvolučné jadro, ktoré môže pomôcť určiť smer pohybu robota, čo môže byť taktiež užitečné pri sledovaní robota [15].

Konvolúcia je matematická operácia, ktorá dvom funkciám priradí tretiu. Konvolúcia ukazuje ako druhá z funkcií pôsobí na prvú, pričom táto druhá funkcia je obrátená a posunutá. Nasledujúce definície konvolúcie v spojitom a diskrétnom prípade sú prevzaté z [13]

Definícia: Nech f, g sú funkcie, $f, g : \mathbb{R} \mapsto \mathbb{C}$. Konvolúciou funkcií f a g nazývame funkciu $f * g$ definovanú ako

$$(f * g)(x) = \int_{-\infty}^{\infty} f(y - x)g(y)dy \tag{3.1}$$

Definícia: Nech f, g sú funkcie, $f, g : \mathbb{R} \mapsto \mathbb{C}$. Diskrétnou konvolúciou funkcií f a g nazývame funkciu $f * g$ definovanú ako

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n - m]. \tag{3.2}$$

Popíšeme princíp fungovania slepej dekonvolúcie. Šum býva modelovaný nasledovne:

$$g = u * h + n, \tag{3.3}$$

kde u je skutočný ostrý obrázok, g je rozmazaný obrázok, h reprezentuje proces rozmazania a n je aditívny šum, väčšinou modelovaný ako normálny. Funkcia h sa nazýva konvolučné jadro alebo point spread function (PSF). Jadro je nezáporné s malým nosičom v porovnaní s rozmermi obrázku.

Cieľom v rámci spracovania obrazu je najčastejšie získanie skutočného (ostrého) obrázku u, táto operácia sa v prípade použitia modelu (3.3) nazýva dekonvolúcia. V prípade, že h je neznáma funkcia ide o slepú dekonvolúciu, teda úlohou je odhadnúť šum i ostrý obrázok - nájsť dvojicu (u, h), ak je známy len rozmazaný obrázok g. Tento problém je zle podmienený - existuje nekonečne veľa dvojíc

(u, h) vysvetľujúcich g, medzi nimi aj alternatíva, kde $u = g$ a h je delta funkcia (teda nedochádza k rozmazaniu) - toto riešenie ale nehľadáme.

Najbežnejšie používaným odhahom (u, h) je maximum a posteriori odhad (ďalej MAP).

Predpokladáme, že u, g a h sú reprezentované náhodnými maticami a rozdeleniami pravdepodobnosti $p(u)$, $p(g)$ a $p(h)$. Použitím Bayesovho princípu platí:

$$p(u, h|g) \sim p(g|u, h)p(u)p(h). \tag{3.4}$$

MAP odhad spočíva k hľadaní argumentu maxima a posteriori distribúcie $p(u, h|g)$, teda nejakej dvojice (u, h).

Prvou alternatívou je hľadať túto dvojicu simultánne, ako je však ukázané v [14], tento prístup aplikovaný priamo bez akýchkoľvek úprav zlyháva a ako optimálne riešenie vyberie práve degenerované riešenie, kde h je delta funkcia.

Úspešnejším prístupom je striedavá maximalizácia aposteriornej hustoty osobitne pre jednu z veličín zatiaľčo druhá je konštantná - hľadá sa u ako argument maxima aposteriornej hustoty pričom h je konštantné a potom naopak, celý tento proces sa iteratívne opakuje.

Apriorná hustota ostrého obrázku býva založená na deriváciách hodnôt pixelov v smere x a y, označme operátory týchto smerových derivácií pre pixel i ako $[D_x]_i$ a $[D_y]_i$. Samotná hustota je zvolená tak, aby prípadu, že norma týchto derivácií cez celý obrázok je nula priradila vysokú pravdepodobnosť. [14] ako častú voľbu uvádza

$$\log(p(u)) = -\sum_i \left|[D_x u]_i\right|^\alpha + \left|[D_y u]_i\right|^\alpha + C, \tag{3.5}$$

kde voľba $\alpha < 1$ produkuje žiadúce apriorné hustoty, C = konšt. Pre $\alpha = 1$ tento predpis odpovedá Laplaceovmu, pre $\alpha = 2$ Gaussovskému rozdeleniu.

Apriornú hustotu kernelu je možné voliť rôzne, kernel má byť pozitívny.

3.1 Použitý algoritmus slepej dekonvolúcie

Algoritmus slepej dekonvolúcie skúmaný ako možná alternatíva časticového filtra je detailne popísaný v [15]. Je založený na striedavom MAP odhade u a h, pričom druhá z veličín je držaná ako konštanta. Hľadanie argumentu maxima aposteriornej hustoty $p(u, h|g)$ je ekvivalentné hľadaniu argumentu minima negatívneho logaritmu tejto hustoty. Predpokladaný je i.i.d Gaussovský šum, teda $p(g|u, h) \sim \exp(-\frac{\gamma}{2}\|u*h-g\|^2)$. Po zlogaritmovaní teda hľadáme argument minima nasledujúceho výrazu, ktorý sa niekedy nazýva aj energia:

$$L(u, h) = -\log(p(u, h|g)) = \frac{\gamma}{2}\|u * h - g\|^2 + Q(u) + S(h) + \text{konšt.}, \tag{3.6}$$

kde $Q(u) = -\log(p(u))$ a $S(h) = -\log(p(h))$ sú regularizované hustoty odpovedajúce apriorným hustotám u a h.

Apriorná hustota ostrého obrázku je zvolená tak, že

$$Q(u) = \alpha_u \sum_i |[Du]_i|^P, \tag{3.7}$$

kde $D = [D_x^T, D_y^T]^T$ je operátor derivácií, reps. gradientný operátor a α_u má význam presnosti - prevrátenej hodnoty rozptylu hustoty derivácií pixelov. Voľba parametrov bola nasledujúca: $\alpha_u = 2$, $P = 0.5$. Táto voľba apriornej hustoty má ťažšie chvosty ako Laplaceovo rozdelenie.

Apriorná hustota kernelu je zvolená ako Laplaceovo rozdelenie na kladných hodnotách a 0 na záporných hodnotách:

$$S(h) = \alpha_h \sum_i \psi(h_i), \quad \psi(h_i) = \begin{cases} h_i & h_i \geq 0 \\ 0 & h_i < 0 \end{cases} \tag{3.8}$$

s voľbou $\alpha_h = 1$. Minimalizovaný výraz teda prechádza do tvaru:

$$L(u,h) = \frac{\gamma}{2}\|u * h - g\|^2 + \alpha_u \sum_i |[Du]_i|^P + \alpha_h \sum_i \psi(h_i) + \text{konšt.} \tag{3.9}$$

Nech pozorovaný obrázok g má rozmery $b_1 \times b_2$, kernel h má rozmery $s_1 \times s_2$. Potom u musí mať rozmery $b_1 + s_1 - 1 \times b_2 + s_2 - 1$, označme ich $a_1 \times a_2$, aby bol g jednoznačne určený - táto alternatívna je označovaná ako validná konvolúcia. Označme maticu validnej konvolúcie s h ako $\hat{H}$, jej rozmery sú $b_1 b_2 \times a_1 a_2$.

Z hľadiska efektívnosti výpočtov je výhodné rozložiť $\hat{H}$ ako $\hat{H} = MH$, kde H rozmerov $a_1 a_2 \times a_1 a_2$ odpovedá kruhovej konvolúcii, ktorá je veľmi rýchla a M rozmerov $b_1 b_2 \times a_1 a_2$ je tzv. cropping matrix - matica, kde sa v každom riadku nachádza 1 najviac raz, ostatné prvky sú nulové.

Touto úpravou sa dá model šumu vyjadrený rovnicou 3.3 zapísať v podobe

$$g = MHu + n \tag{3.10}$$

a výsledná podoba minimalizovaného výrazu $L(u,h)$ je nasledovná:

$$L(u,h) = \frac{\gamma}{2}\|MHu - g\|^2 + \alpha_u \sum_i |[Du]_i|^P + \alpha_h \sum_i \psi(h_i) + \text{konšt.} \tag{3.11}$$

Celý proces nachádzania argumentu minima $L(u,h)$ je možné popísať pomocou nasledujúceho pseudoalgoritmu:

1: $h^0 = \delta$, $j = 0$, $\gamma_0 = \gamma_{min}$

2: **while** nie je uspokojené kritérium zastavenia **do**

3: nájsť $u_{j+1} = \arg\min_u L_{\gamma_j}(u, h^j)$

4: nájsť $h_{j+1} = \arg\min_h L_{\gamma_j}(u^{j+1}, h)$

5: $\gamma_{j+1} = \frac{\gamma_{max}}{\left(\frac{\gamma_{max}}{\gamma_{min}} - 1\right)e^{-\kappa j} + 1}$

6: **end while**

7: **return** u^j, h^j

Voľby parametrov: $\gamma_{min} = 2$, $\gamma_{max} = 2^{13}$, $\kappa = 1.25$.

Hľadanie argumentu minima v bodoch 3 a 4 sa v oboch prípadoch uskutočňuje tzv. alternating direction method of multipliers (ďalej ADMM), čo je popísané podrobne v [16]. Táto metóda spočíva v použití tzv. augmented lagrangian method (ďalej ALM) a zavedení kvadratickej penalty pre väzby vzniknuté zavedením substitúcie.

Bod 3. je metódou ADMM riešený nasledovne:

- $\min_u L(u,h) = \min_u \frac{\gamma}{2}\|MHu - g\|^2 + \alpha_u \sum_i |[Du]_i|^P$
- zavedie sa substitúcia $v = Du$, $w = Hu$, čím vznikne optimalizačný problém s dvoma väzbami a problém sa taktiež výpočetne zjednoduší, keďže členy závisia na iných premenných a minimalizáciu je možné uskutočňovať osobitne:

$$min_{u,v,w}\frac{\gamma}{2}\|Mw - g\|^2 + \alpha_u \sum_i |v_i|^p, \quad v = Du, w = Hu$$

- pridaním kvadratických penált má výsledný funkcionál, ktorý je potrebné minimalizovať nasledujúci tvar:

$$L_u(u,v,w) = \frac{\gamma}{2}\|Mw - g\|^2 + \alpha_u \sum_i |v_i|^P + \frac{\beta_u}{2}\|v - Du - \alpha_u\|^2 + \frac{\beta_w}{2}\|w - Hu - \alpha_w\|^2$$

- minimalizácia $L_u(u,v,w)$ prebieha počítaním derivácie v jednom smere pričom druhé dve premenné sú konštantné a položením tejto derivácie nule, tento proces sa uskutoční pre všetky tri premenné a po aktualizácii ich hodnôt sa celý opakuje

Presný postup v rámci bodu 3. je zhrnutý v nasledujúcom pseudoalgoritme:

1: $v^0 = 0, w^0 = 0, a_u^0 = 0, a_w^0 = 0, j = 0$
2: **while** nie je uspokojené kritérium zastavenia **do**
3: vyriešiť $(\gamma M^T M + \beta_w I)w^{j+1} = \gamma M^T g + \beta_w(Hu^j + a_w^j)$ pre w^{j+1}
4: $v_i^{j+1} \leftarrow [Du^j + a_u^j]_i \ \ \forall i$ pre dané P, α_u, β_u
5: vyriešiť $(\beta_w H^T H + \beta_u D^T D)u^{j+1} = \beta_w H^T(w^{j+1} - a_w^j) + \beta_u D^T(v_u^{j+1} - a_u^j)$ pre u^{j+1}
6: $a_u^{j+1} \leftarrow a_u^j - v^{j+1} + Du^{j+1}$
7: $a_w^{j+1} \leftarrow a_w^j - w^{j+1} + Hu^{j+1}$
8: $j \leftarrow j + 1$
9: **end while**
10: **return** u^j

Voľby parametrov: $\alpha_u = 2, \beta_u = 2^4, \beta_w = 2^{13}$.

Bod 4 je riešený analogicky:

- $\min_h L(u,h) = \min_u \frac{\gamma}{2}\|MHu - g\|^2 + \alpha_h \sum_i \psi(h_i)$

- zavedie sa substitúcia $v_h = h$, $w = Uh$, čím vznikne optimalizačný problém s dvoma väzbami a minimalizácia sa opäť zjednoduší:

$$min_{h,v_h,w} \frac{\gamma}{2}||Mw - g||^2 + S(v_h) + \alpha_h \sum_i \psi([v_h]_i), \quad v_h = h, w = Uh$$

- pridaním kvadratických penált má výsledný funkcionál, ktorý je potrebné minimalizovať nasledujúci tvar:

$$L_u(u, v, w) = \frac{\gamma}{2}||Mw - g||^2 + S(v_h) + \alpha_h \sum_i \psi([v_h]_i) + \frac{\beta_h}{2}||v_h - h - a_h||^2 + \frac{\beta_w}{2}||W - Uh - a_w||^2$$

Presný postup v rámci bodu 4. je zhrnutý v nasledujúcom pseudoalgoritme:

1: $v_h^0 = 0$, $w^0 = 0$,$a_h^0 = 0$, $a_w^0 = 0$, $j = 0$

2: **while** nie je uspokojené kritérium zastavenia **do**

3: vyriešiť $(\gamma M^T M + \beta_w I)w^{j+1} = \gamma M^T g + \beta_w(Uh^j + \alpha_w^j)$ pre w^{j+1}

4: $[v_h^{j+1}]_i \leftarrow \max(0, [h^j + a_h^j]_i - \frac{\alpha_h}{\beta_h}), \ \forall i$

5: vyriešiť $(\beta_w U^T U + \beta_h I)h^{j+1} = \beta_w U^T(w^{j+1} - a_w^j) + \beta_h(v_h^{j+1} - a_h^j)$ pre h^{j+1}

6: $a_h^{j+1} \leftarrow a_h^j - v_h^{j+1} + h^{j+1}$

7: $j \leftarrow j + 1$

8: **end while**

9: **return** h^j

Voľby parametrov: $\alpha_h = 1, \beta_h = 2^{13}$.

Kapitola 4

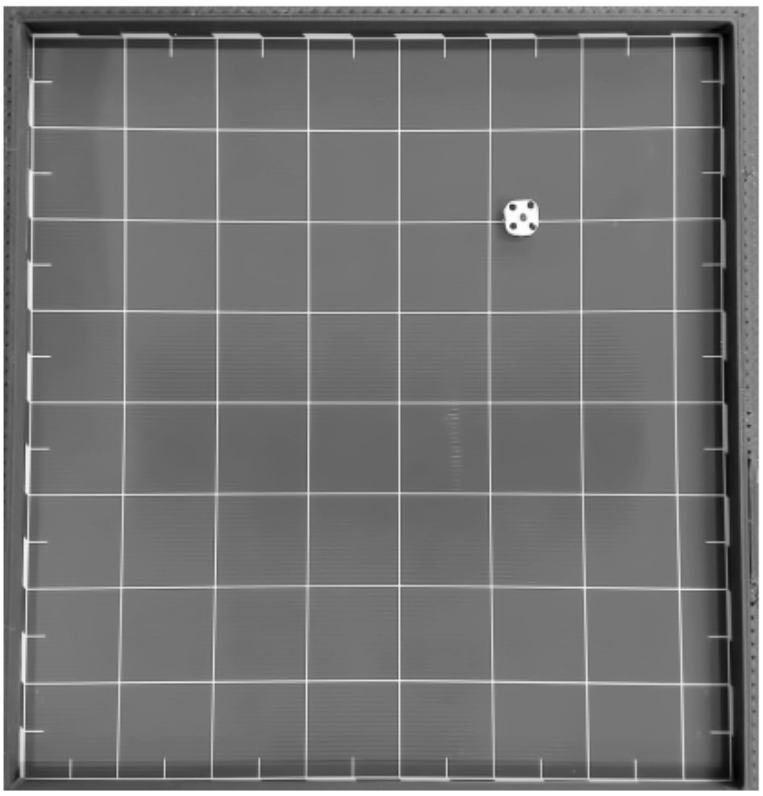
Detekcia polohy robota

V tejto časti sa pokúsime aplikovať časticový filter a slepú dekonvolúciu na sledovanie polohy vo videu magneticky ovládaného mikrorobota pohybujúceho sa v rovine xy. Tento problém bol modelovaný a riešený numericky v [11].

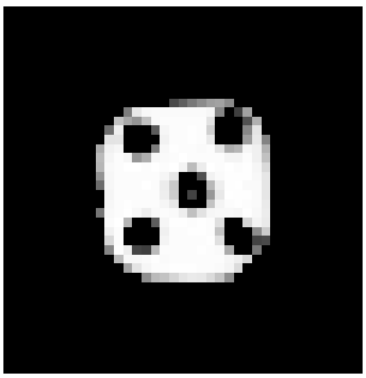
Úloha detekcie magneticky poháňaného mikrorobota z video záznamu prináša niekoľko ťažkostí:

- magnetický pohon spôsobuje nerovnomerný pohyb, ktorý je náhly, robot je schopný okamžite zrýchliť a zastaviť, pohyb nevykazuje zotrvačnosť
- kamera sníma v pravidelných intervaloch, nie je synchronizovaná s pohybom robota a nevieme určiť v akej fáze pohybu sa robot nachádza alebo kedy presne sa začal pohybovať, dôsledkom sú rozmazané obrázky

Výzvou je teda dobrá reprezentácia tohto procesu. O to sa pokúšame zavedením nesledujúcich modelov, následne vyhodnotíme, ktoré sú vhodné.



Obr. 4.1: Robot na pozadí pred začiatkom pohybu



Obr. 4.2: Šablona robota

4.1 Časticový filter

Stavový vektor je pre úlohu detekcie poloha a rýchlosť: (p_x, p_y, v_x, v_y). Stavový model má tvar

$$v_{k+1} = v_k + C\, a_k + \varpi, \tag{4.1}$$

kde a_k je akcelerácia v čase k, ktorá je neznáma a mení sa v čase, C je konštanta. Tento model aproximujeme náhodnou prechádzkou, ktorý má po rozpísaní na zložky v smere x a y nasledujúci tvar:

$$\begin{aligned} v_{x,k+1} &= v_{x,k} + \varpi \\ v_{y,k+1} &= v_{y,k} + \varpi \end{aligned} \tag{4.2}$$

$$\begin{aligned} p_{x,k+1} &= p_{x,k} + v_{x,k+1}\, dt \\ p_{y,k+1} &= p_{y,k} + v_{y,k+1}\, dt, \end{aligned} \tag{4.3}$$

kde ϖ má štandardné normálne rozdelenie. Rozptyl rozdelenia šumu procesu bol volený ako 1, menší rozptyl produkuje častice len blízko predchádzajúcej polohy a nezachytí náhlu zmenu rýchlosti, pri voľbe väčšieho rozptylu bol často efektívny počet častíc N_{eff} nízky a teda takto zvolený model poskytoval opäť nepresné odhady. $dt = \frac{1}{\text{FPS}}$, FPS (frames per second) kamery je 29.992442. V praxi miesto rýchlostí $v_{x,k+1}$, $v_{y,k+1}$ odhadujeme veličiny $v_{x,k+1}\, dt$, $v_{y,k+1}\, dt$, ktoré majú význam posunu v smeroch x, y.

Pri použití všetkých algoritmov volíme Gaussovský model - hustota p je normálna a q volíme rovnako ako p. Pod časovým krokom rozumieme jeden snímok (frame) videa. V každom kroku k je teda generovaný súbor častíc $\{(v^i_{x,k}, v^i_{y,k}, x^i_k, y^i_k)\}_{i=1}^N$ z prechodnej hustoty, úlohu pozorovania z_k má k-tý snímok videa, ktorý označujeme O_k:

$$q_v(v_k|v^i_{k-1}, z_k) = q_v(v_k|v^i_{k-1}, O_k) = p(v_k|v^i_{k-1}) = \mathcal{N}(v_k; v^i_{k-1}, \Sigma_v)$$

Predpokladáme, že v_x, v_y sú nezávislé náhodné veličiny a teda

$$\Sigma_v = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix} \tag{4.4}$$

pre nejaké σ^2.

Pre tento jeden model procesu navrhujeme dva rôzne modely pozorovaní, prvý základný a používaný doteraz a druhý o niečo zložitejší, ktorý berie do úvahy ťažkosti zmienené v úvode kapitoly a mohol by poskytovať presnejšie odhady pre rozmazané obrázky.

4.1.1 Základný model pozorovania

Model pozorovaní:

$$O_k = O_1 + T^i + n, \tag{4.5}$$

kde O_1 je pozadie videa bez prítomnosti robota a T^i označuje snímok, ktorý vznikol posunom šablóny mikrorobota o $(p^i_{x,k}, p^i_{y,k})$.

4.1.2 Model pozorovania s rozmazaním

Model pozorovaní:

$$O_k = O_1 + 0.1 \sum_{j=0}^{9} T_j^i + n, \tag{4.6}$$

kde O_1 je pozadie videa bez prítomnosti robota a T_j^i označuje snímok, ktorý vznikol ako suma posunov šablóny mikrorobota o $(p^i_{x,k-1} + 0.1jv^i_{x,k}, p^i_{y,k-1} + 0.1jv^i_{y,k})$. Teda výsledná posunutá šablóna je konštruovaná ako súčet postupných posunutí.

Tento model bol testovaný ako možnosť reprezentovať rozmazaného robota ak sa v čase snímania kamery pohyboval. Jeho predpokladom je však rovnomerný pohyb, čo v prípade magnetického pohonu nemusí byť naplnené.

4.1.3 Výpočet váh častíc

V oboch prípadoch je váha generovaným časticiam pripisovaná na základe porovnania daného framu videa a snímku, ktorý by vznikol posunom šablóny mikrorobota podľa danej častice. Ako je zvolený rozptyl σ^2 rozdelenia šumu procesu, akú hypotézu vyslovíme o šume procesu ϖ a akú normu zvolíme pri výpočte váh veľmi výrazne ovplyvňuje fungovanie algoritmu.

- $\sigma^2 = 1$
- rozdelenie šumu pozorovaní n predpokladáme štandardné normálne. Hustota pravdepodobnosti má podľa [17] tvar

  $$p(O_k|x_k^i) = \frac{1}{\sqrt{2\pi}} \exp\left(- \frac{\|O_k - O_1 - T\|^2}{2} \right), \tag{4.7}$$

  kde $\|O_k - O_1 - T\|$ je L2 norma na oblasti G posunutej šablóny robota spolu s jej okolím veľkosti 10 pixelov, rozmery robota sú približne 19x18 pixelov. $(O_k - O_1)(m,n)$ je hodnota pixelu (m,n) v k - tom snímku videa po odčítaní pozadia a $T^i(m,n)$ je hodnota rovnakého pixelu v obrázku posunutej šablóny. Teda

  $$\|O_k - O_1 - T\| = \sqrt{\sum_{(m,n)\in G} \Big((O_k - O_1)(m,n) - T(m,n) \Big)^2} \tag{4.8}$$

Váha i-tej častice v k-tom kroku procesu je teda spočítaná nasledovne:

$$w_k^i \propto w_{k-1}^i \, p(O_k|x_k^i) \propto w_{k-1}^i \exp\left\{ - \frac{1}{2} \sum_{(m,n)\in G} \Big((O_k - O_1)(m,n) - T(m,n) \Big)^2 \right\} \tag{4.9}$$

4.1.4 Adaptívne a hromadné adaptívne vzorkovanie podľa dôležitosti

Pri použití AIS a AMIS je prvý krok rovnaký ako pri BIS - v každom kroku k sa generuje prvotný súbor častíc z prechodnej hustoty. Následne sa hustota q spresňuje. Cieľom je zvoliť hustotu q tak, aby čo najbližšie odpovedala skutočnej hustote pravdepodobnosti p, pričom predpokladáme, že q je z rodiny normálnych hustôt.

$$q_v(v_k|v^i_{k-1}, z_k) = q_v(v_k|v^i_{k-1}, O_k) = \mathcal{N}(v_k; \mu, \Sigma), \text{ kde } \mu = \begin{pmatrix} \mu_x \\ \mu_y \end{pmatrix}, \Sigma = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix}.$$

Parameter $\lambda = (\lambda_1, \ldots, \lambda_4) = (\mu_x, \mu_y, \sigma_x^2, \sigma_y^2)$. Keďže má q čo najbližšie odpovedať p, funkcia ξ je zvolená nasledovne:

$$\xi(v, \lambda) = (\xi_1(v), \ldots, \xi_4(v)) = (v_x, v_y, (v_x - \lambda_1)^2, (v_y - \lambda_2)^2),$$

$\xi(v, \lambda)$ se vyhodnotí dosadením aktuálneho odhadu parametru λ. Funkcia $\xi(v, \lambda)$ v tomto tvare splňuje rovnost 2.20:

$$\begin{aligned}
\mu_x &= \lambda_1 = \mathbb{E}[\xi_1(v, \lambda)] = \mathbb{E}[v_x] \\
\mu_y &= \lambda_2 = \mathbb{E}[\xi_2(v, \lambda)] = \mathbb{E}[v_y] \\
\sigma_x^2 &= \lambda_3 = \mathbb{E}[\xi_3(v, \lambda)] = \mathbb{E}[(v_x - \lambda_1)^2] = \mathbb{E}[(v_x - \mathbb{E}[v_x])^2] \\
\sigma_y^2 &= \lambda_4 = \mathbb{E}[\xi_4(v, \lambda)] = \mathbb{E}[(v_y - \lambda_2)^2] = \mathbb{E}[(v_y - \mathbb{E}[v_y])^2]
\end{aligned}$$

Za kritérium zastavenia volíme jednoducho určitý počet spresnení odhadu parametru λ. Za prvotný odhad hustoty q_v je volená prechodná hustota, teda $\mu^{(0)} = (v^i_{k-1,x}, v^i_{k-1,y})', \Sigma^{(0)} = \Sigma_v$. Počet častíc je v každom kroku spresňovania q_v rovnaký. AMIS bol aplikovaný s použitím základného modelu pozorovania.

4.2 Slepá dekonvolúcia

Ako bolo podrobne popísané v časti 3 predpokladáme nasledujúci model šumu:

$$O_k = O_1 + T * h + n, \tag{4.10}$$

kde T je šablóna, n je normálne rozdelený šum a h je odhadované konvolučné jadro. Pri detekcii polohy robota úlohu obecného pozorovaného obrázku g plní k-tý frame videa O_k, kde obrázok robota býva často rozmazaný. O_1 je opäť prvný snímok videa, na ktorom je pozadie bez prítomnosti robota.

Algoritmus dekonvolúcie neuvažuje dynamiku modelu alebo vývoj stavového vektora, zaoberá sa len konkrétnym pozorovaním v čase k. Rovnica 4.10 je teda alternatívou k 4.5, resp. 4.6, keďže má k dispozícii rovnaké pozorovanie O_k, ale používa iný model pozorovania.

Výrazným zjednodušením oproti obecnému algoritmu popísanému v kapitole 3 je, že jedinou odhadovanou veličinou je konvolučné jadro h, u v obecnom algoritme odpovedá v našej aplikácii šablóne T a tú neodhadujeme. Celý krok 3. obecného algoritmu teda vynechávame.

Veľkou výhodou použitého algoritmu slepej dekonvolúcie je jej rýchlosť, naviac slepá dekonvolúcia je deterministickou metódou, zatiaľ čo úspešnosť časticových filtrov do istej miery závisí na konkrétnych generovaných súboroch. Odhadnuté jadro je preložené priamkou, ktorá môže naznačovať smer pohybu robota.

Naopak potenciálnou nevýhodou je obmedzenie sa na to, že rolu pozorovania plní snímok videa. Na rozdiel od slepej dekonvolúciu je možné ako pozorovanie uvažovať napríklad meranie detekčnej cvievky.

V rámci základného algoritmu dekonvolúcie navrhujeme dve potenciálne zlepšenia.

4.2.1 Spresnenie pozadia

Napriek tomu že pozadie je v skutočnosti konštantné, behom videa dochádza k miernym zmenám, čo by mohlo negatívne ovplyvniť presnosť odhadu.

Riešením by mohlo byť ako pozadie miesto prvého snímku zvoliť medián susedných snímkov. Takáto konštrukcia zachytí malé lokálne zmeny alebo posunutia kamery behom pohybu robota.

Zvolíme medián z 15 snímkov pred a po snímku, na ktorom odhadujeme polohu. Ak je medzi mediánom a snímkom, na ktorom odhadujeme polohu dostatočný rozdiel, proces končí, v opačnom prípade okno cez ktoré sa medián konštruuje zdvojnásobíme a znova overíme podmienku, ktorá je ekvivalentná tomu, že robot na mediáne susedných snímkov už nie je viditeľný.

4.2.2 Normalizácia jadra

Častou podmienkou kladenou na konvolučné jadro je aby bola suma pixelov rovná 1. Toto obmedzenie je zmysluplné napríklad v aplikáciách kedy sledovaný objekt ostáva v rovnakej vzdialenosti.

Napriek tomu že robot sa pohybuje v rovnakej vzdialenosti od kamery, je na kontrastnom pozadí na ktorom sa nenachádzajú žiadne iné objekty a je zaujímavou možnosťou toto obmedzenie rozvoľniť.

Kapitola 5

Výsledky a porovnanie algoritmov

5.1 Presnosť určenia polohy

5.1.1 Zvolené metriky

Keďže nie je známa presná poloha robota, presnosť jednotlivých metód je vyhodnotená na základe rozdielu medzi $O_k - O_1$, teda k-tým framom od ktorého je odčítané pozadie a obrázkom, ktorý vznikne na základe nášho odhadu polohy na nejakom širšom okolí robota, ktoré je vybrané pre každý snímok zvlášť a je rovnaké pre všetky odhady. Odhady poskytnuté jednotlivými metódami sú porovnané na základe sumy hodnôt pixelov tohto rozdielu na danom okolí robota v snímku.

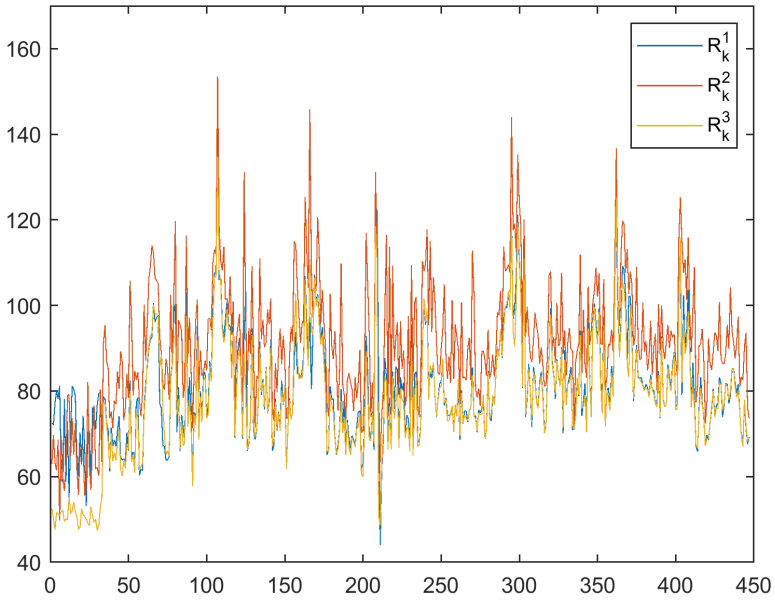
V tomto duchu sme navrhli tri metriky, ktoré si kladú za cieľ porovnať medzi sebou jednotlivé odhady.

- Označme obrázok $r_k^1 = ||O_k - O_1 - T^{mean}||$, kde T^{mean} je posunutie šablóny do odhadu polohy, ktorý pre časticové filtre vznikne ako vážený priemer častíc po prevzorkovaní a pre slepú dekonvolúciu ako stred priamky, ktorou je preložené odhadnuté jadro. Takto definovaný odhad je intuitívny a v oblasti časticových filtrov používaný najčastejšie. Prvú metriku označíme R_k^1 kde k označuje poradové číslo snímku videa pre ktorý tento rozdiel konštruujeme, R_k^1 je suma hodnôt pixelov obrázku r_k^1 na danom okolí polohy robota v k-tom snímku.

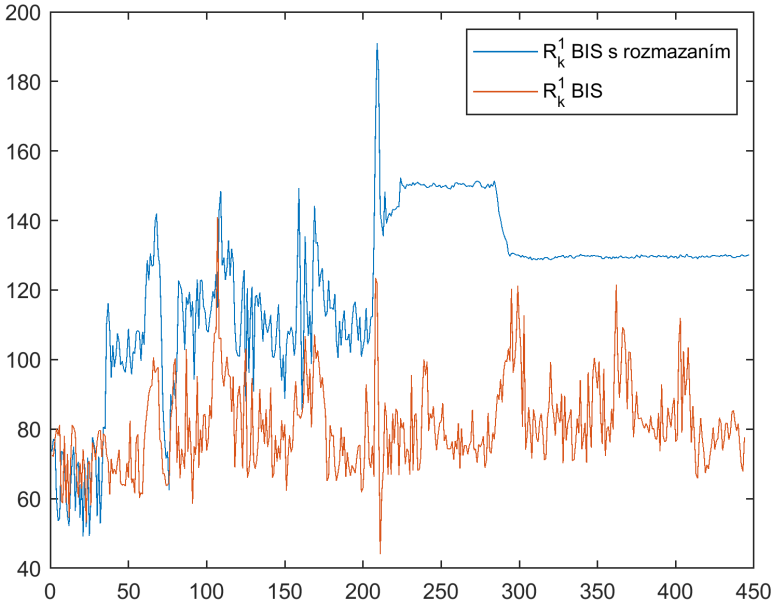
- Označme obrázok $r_k^2 = ||O_k - O_1 - T^{end}||$, kde T^{end} je posunutie šablóny do odhadu polohy, ktorý je pre $k-$ty snímok pre časticové filtre zvolený ako tá z prevzorkovaných častíc, ktorá má najväčšiu euklidovskú vzdialenosť od odhadnutej polohy pre snímok $k-1$. Podobne pre slepú dekonvolúciu je ako odhad polohy volený ten krajný bod úsečky, ktorý má väčšiu euklidovskú vzdialenosť od odhadnutej polohy pre snímok $k-1$. V miestach, kde sa náhle zmení veľkosť alebo smer rýchlosti by sa tento odhad mohol ukázať ako presnejší. Metriku označíme R_k^2 a je to opäť suma pixelov obrázku r_k^2 na okolí skutočnej polohy.

- Označme obrázok $r_k^3 = ||O_k - O_1 - T^{repr}||$, kde T^{repr} je pre časticové filtre obrázok, ktorý vznikne ako priemer posunutí šablóny do všetkých častíc po prevzorkovaní, pre slepú dekonvolúciu je T^{repr} konvolúcia šablóny a odhadnutého konvolučného jadra. Metriku označíme R_k^3 a je to opäť suma pixelov obrázku r_k^3 na okolí skutočnej polohy.

5.1.2 Porovnanie metrík pre každý uvažovaný algoritmus



Obr. 5.1: R_k^1, R_k^2, R_k^3 pre algoritmus BIS a 100 častíc, na osi x je uvedené poradové číslo snímku vo videu

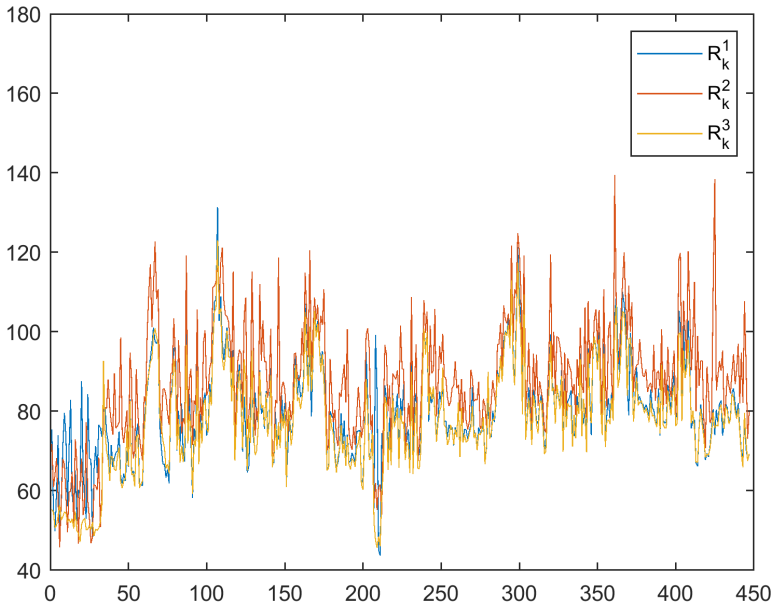


Obr. 5.2: R_k^1 pre algoritmus BIS pre 100 častíc a BIS s rozmazaním pre 100 častíc, na osi x je uvedené poradové číslo snímku vo videu

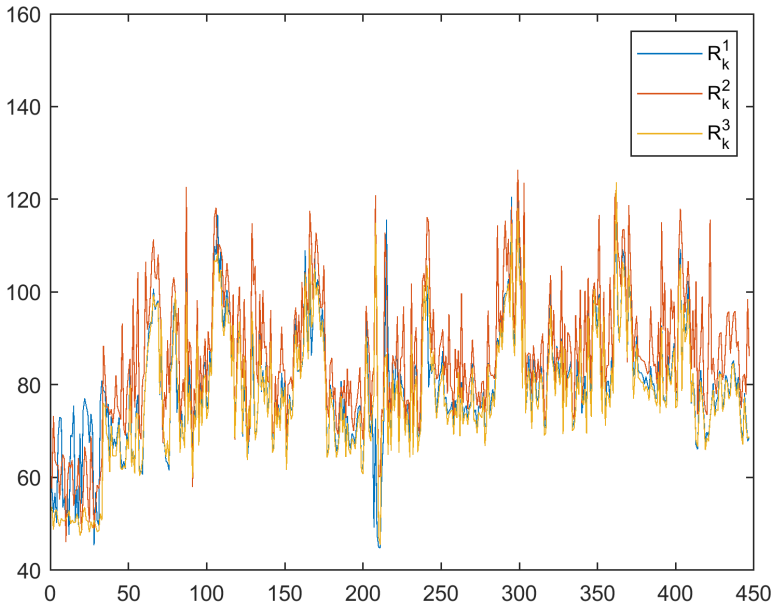
Z obrázku 5.2 je zrejmé, že pri použití modelu s rozmazaním a 100 častíc častícový filter úplne zlyhá. Krátko po 200. snímku vidíme, že norma obrázku r_k^1 je osciluje minimálne. Tento jav nastane preto, že odhad sa úplne odchýli od skutočnej polohy a r_k^1 je obrázok šablóny posunutej na nesprávne miesto. Krátko po snímku 300 sa hodnota chyby zase zníží a je skoro konštantná. V tomto mieste odhad polohy dospeje úplne do rohu plochy po ktorej sa mikrorobot pohybuje a plocha obrázku r_k^1 sa zmenší. Odhad ostane rovnaký až do konca videa.

Zlyhanie BIS s rozmazaním na snímkoch 210 až 214 je viditeľné na obrázku 5.3. V tomto úseku dochádza k prudkej zmene smeru. Z priebehu metriky R_k^1 je jasné, že všetky častice sú úplne nesprávne a vykresľovať ostatné metriky nemá zmysel.

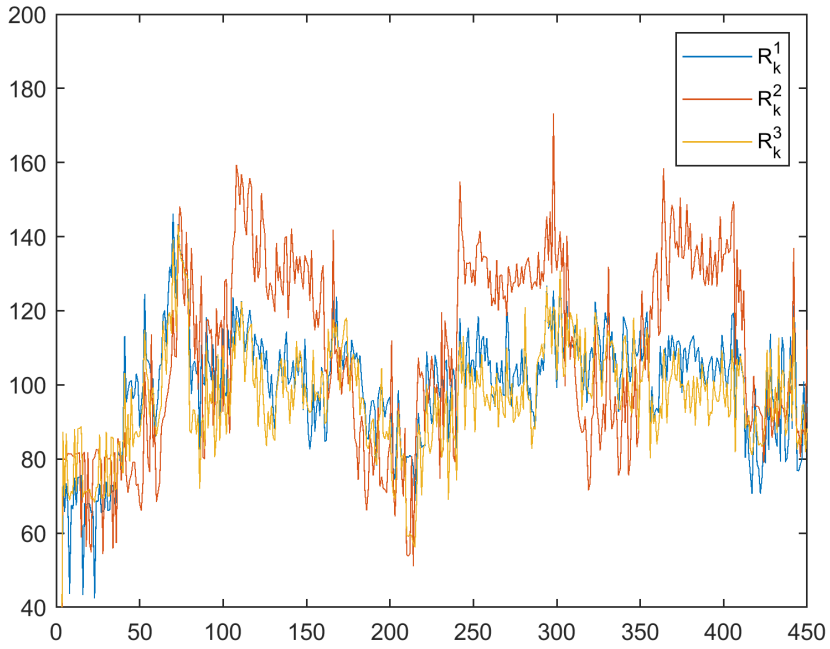
Obr. 5.3: r_{210}^1 až r_{214}^1 pre BIS s rozmazaním a 100 častíc



Obr. 5.4: R_k^1, R_k^2, R_k^3 pre algoritmus AIS a 2x50 častíc, na osi x je uvedené poradové číslo snímku vo videu



Obr. 5.5: R_k^1, R_k^2, R_k^3 pre algoritmus AMIS a 2x50 častíc, na osi x je uvedené poradové číslo snímku vo videu



Obr. 5.6: R_k^1, R_k^2, R_k^3 pre algoritmus slepej dekonvolúcie, na osi x je uvedené poradové číslo snímku vo videu

Pre BIS, AIS a AMIS sa ukázalo, že metrika R_k^3, tzn. priemer posunutí do všetkých častíc má najmenšiu chybu, nasleduje R_k^1, posunutie do váženého priemeru a najhoršie výsledky poskytuje posunutie do najvzdialenejšej častice R_k^2. Tento výsledok nie je prekvapivý. Priemer posunutí umožňuje lepšiu reprezentáciu obrázkov které sú mierne rozmazané vplyvom pohybu kým kamera sníma. R_k^1 je nejpoužívanejšia konštrukcia odhadu pre časticové filtre obecne.

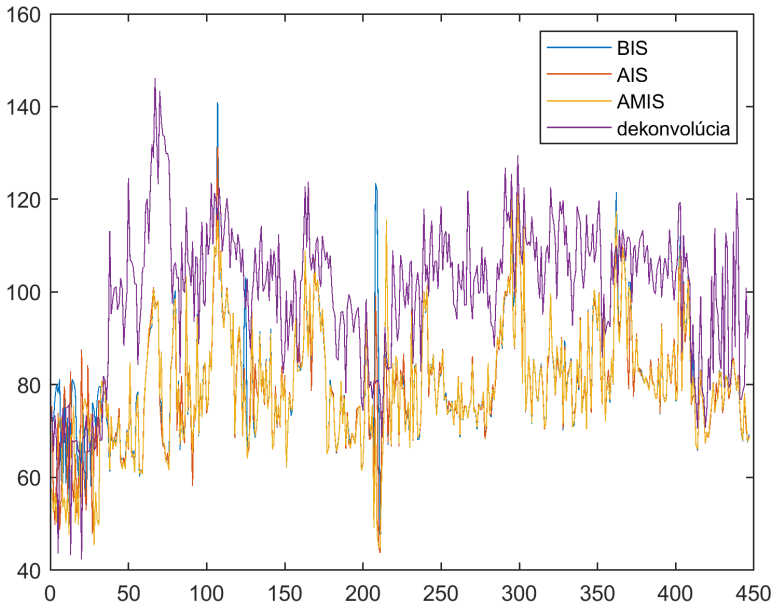
Pre slepú dekonvolúciu obrázok konštruovaný ako konvolúcia odhadnutého jadra a šablóny má za následok nejmenší rozdiel vo väčšine procesu analogicky k časticovým filtrom.

metóda	$\frac{1}{K}\sum_{k=1}^{K} R_k^1$	$\max\left\{R_k^1\right\}$	$\frac{1}{K}\sum_{k=1}^{K} R_k^2$	$\max\left\{R_k^2\right\}$	$\frac{1}{K}\sum_{k=1}^{K} R_k^3$	$\max\left\{R_k^3\right\}$
BIS	80.73	141	91.46	153.44	78.45	134.81
AIS	79.94	131.37	87.79	139.47	77.65	123
AMIS	79.72	120.67	86.44	126.43	77.88	123.68
slepá dekonvolúcia	99.69	146.2	108.07	173.25	95.88	143.08

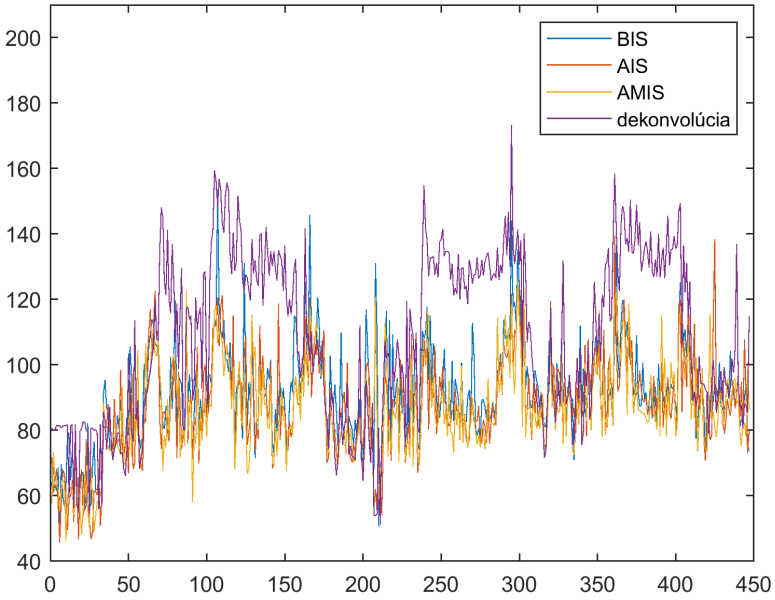
Tabuľka 5.1: Priemer a maximum metrík pre uvažované algoritmy

5.1.3 Porovnanie algoritmov na základe zvolených metrík

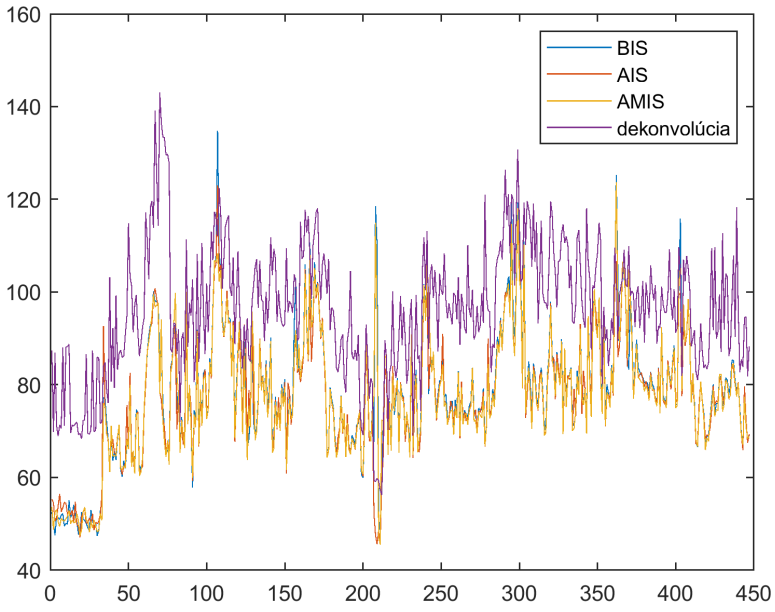
Okrem tabuľky 5.1 znázorníme rozdiely medzi algoritmami aj graficky. Všetky algoritmy časticového filtra sú pre 100, resp. 2x50 častíc.



Obr. 5.7: R_k^1 pre uvažované algoritmy, na osi x je uvedené poradové číslo snímku vo videu



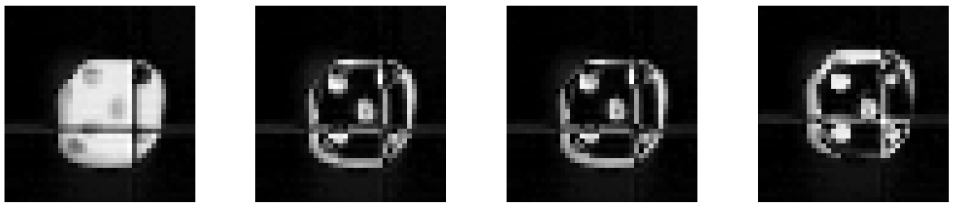
Obr. 5.8: R_k^2 pre uvažované algoritmy, na osi x je uvedené poradové číslo snímku vo videu



Obr. 5.9: R_k^3 pre uvažované algoritmy, na osi x je uvedené poradové číslo snímku vo videu

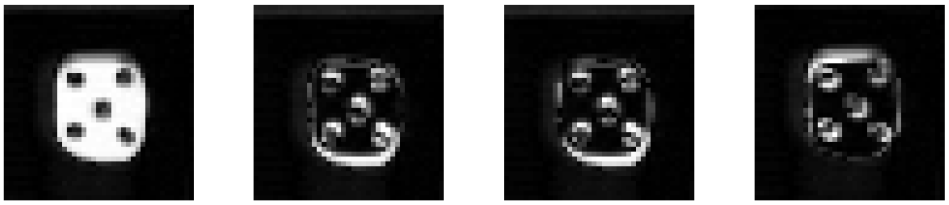
Vidíme, že pri použití všetkých metrík sú všetky varianty časticového filtra veľmi podobné a dekonvolúcia je výrazne menej presná.

Vykreslíme taktiež obrázky r_k^1 pre niekoľko snímkov.

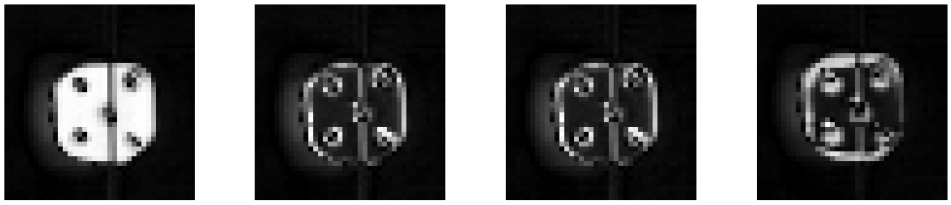


Obr. 5.10: Snímok 90. Snímok po odčítaní pozadia, r^1_{90} pre BIS 100 častíc, AMIS 2x50 častíc a dekonvolúciu.

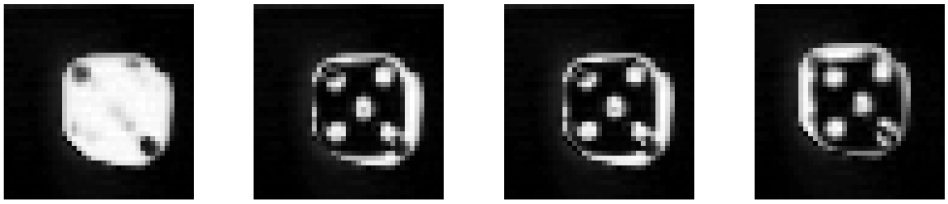
Obr. 5.11: Snímok 107. Snímok po odčítaní pozadia, r^1_{107} pre BIS 100 častíc, AMIS 2x50 častíc a dekonvolúciu.



Obr. 5.12: Snímok 213. Snímok po odčítaní pozadia, r^1_{213} pre BIS 100 častíc, AMIS 2x50 častíc a dekonvolúciu.



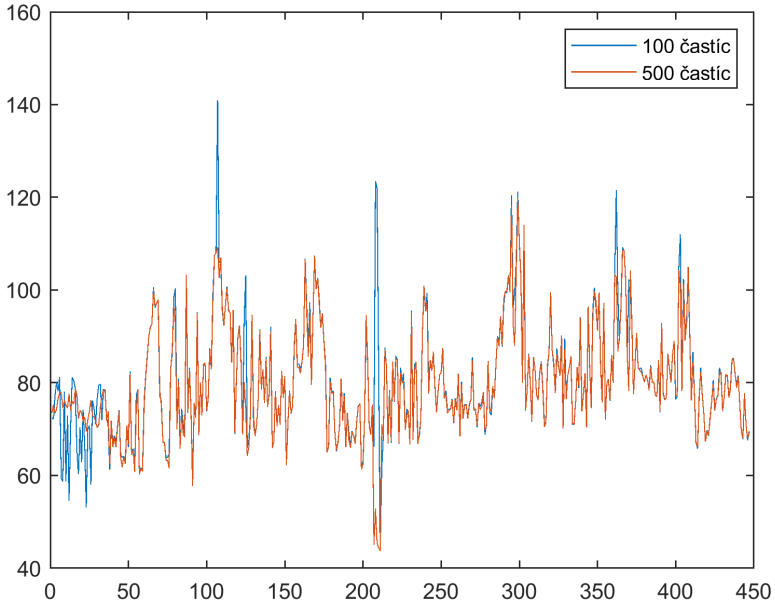
Obr. 5.13: Snímok 303. Snímok po odčítaní pozadia, r^1_{303} pre BIS 100 častíc, AMIS 2x50 častíc a dekonvolúciu.



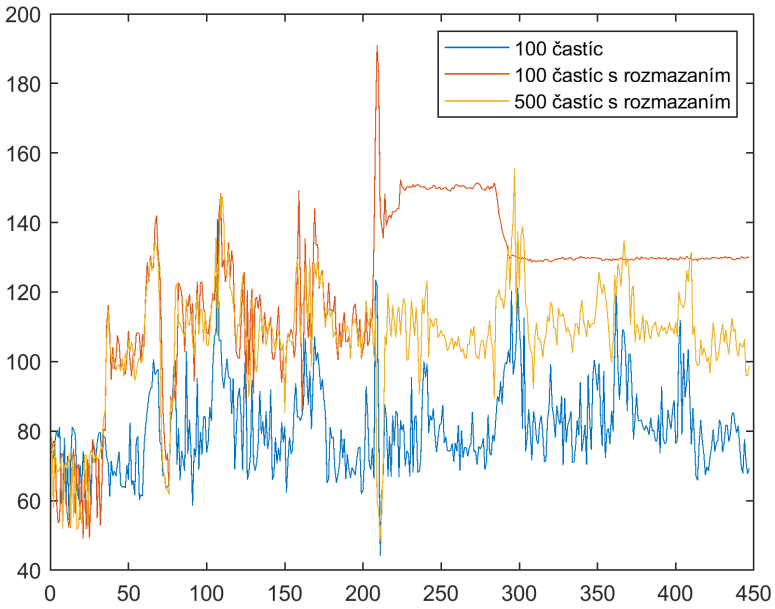
Obr. 5.14: Snímok 362. Snímok po odčítaní pozadia, r^1_{362} pre BIS 100 častíc, AMIS 2x50 častíc a dekonvolúciu.

5.2 Konvergencia časticového filtra

Z teórie popísanej v [3] vyplýva, že so zvyšujúcim sa počtom generovaných častíc by mali odhady poskytnuté časticovými filtrami konvergovať k skutočnej hodnote stavového vektora. Porovnáme algoritmy pre 100, resp. 2x50 častíc a 500, resp. 5x100 častíc na základe štatistiky R_k^1. 100 je najmenší počet častíc pre ktorý odhad pomocou časticového filtra sleduje pohyb robota počas celého videa aj pre opakujúce sa realizácie. Okrem overenia teórie je cieľom tejto časti aj zistenie, či zvýšenie výpočetnej náročnosti prináša zvýšenie presnosti odhadu, ktoré by ho opodstatnilo.



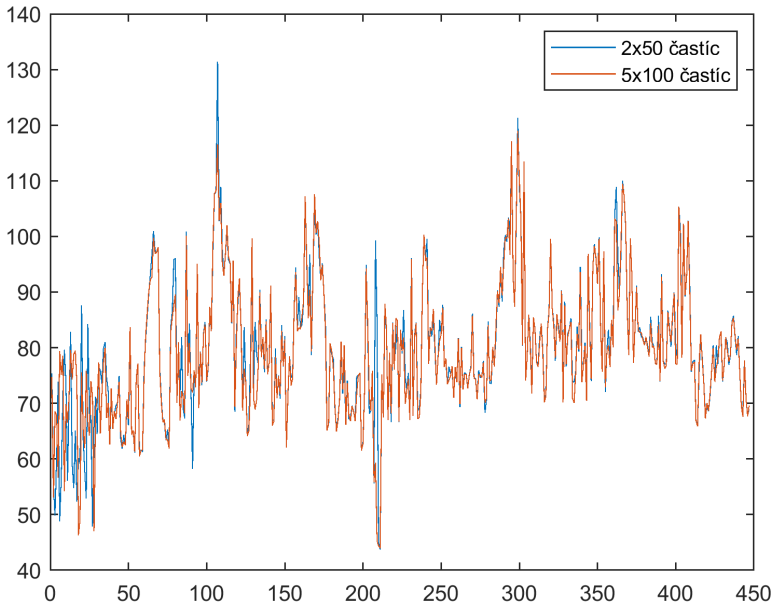
Obr. 5.15: R_k^1 pre BIS 100 a 500 častíc, na osi x je uvedené poradové číslo snímku vo videu



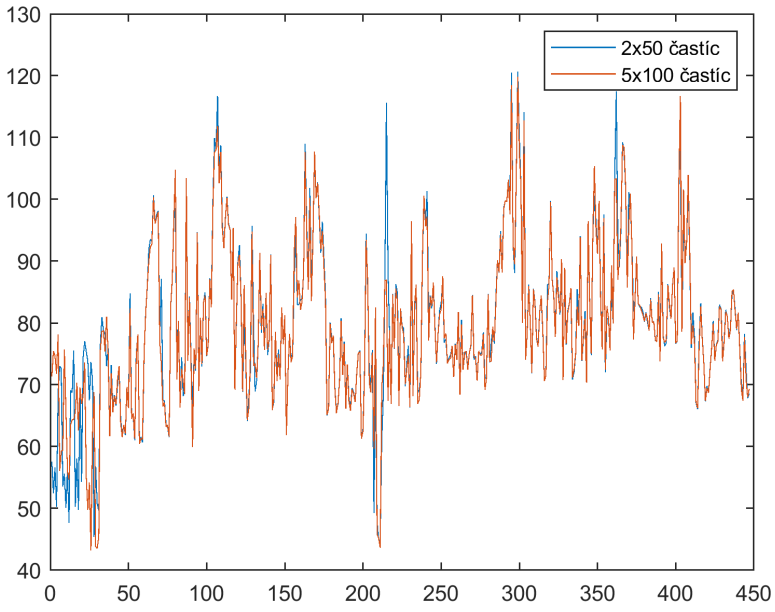
Obr. 5.16: R_k^1 pre BIS s rozmazaním 100 a 500 častíc a BIS 100 častíc, na osi x je uvedené poradové číslo snímku vo videu

Pre navýšený počet 500 častíc časticový filter s rozmazaním až do konca pohybu robota sleduje. Jeho presnosť je však výrazne horšia v porovnaní so základnou verziou časticového filtra pre len 100 častíc. Na obrázku 5.17 vidíme, že pre 500 častíc sa filter s rozmazaním v oblasti 210. snímku neodchýli.

Obr. 5.17: r_{210}^1 až r_{217}^1 pre BIS s rozmazaním a 500 častíc

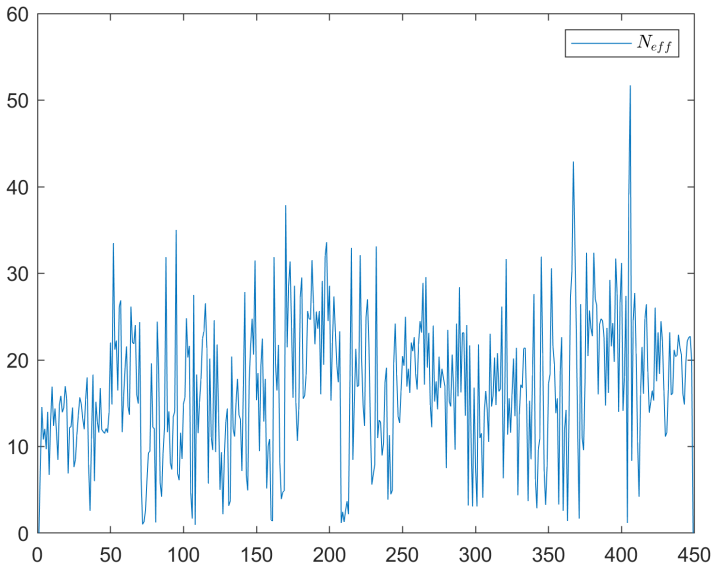


Obr. 5.18: R_k^1 pre AIS 100 a 500 častíc, na osi x je uvedené poradové číslo snímku vo videu

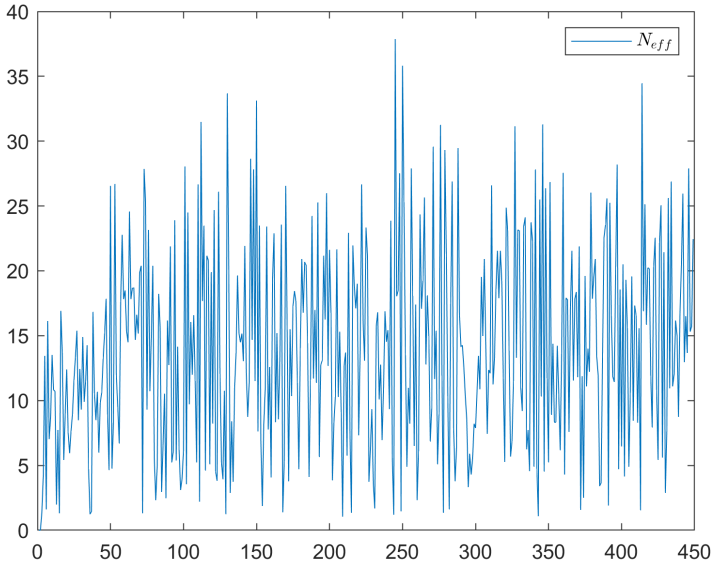


Obr. 5.19: R_k^1 pre AMIS 100 a 500 častíc, na osi x je uvedené poradové číslo snímku vo videu

Na tomto mieste tiež vzkreslíme pribeh efektívneho odhadu počtu častíc pre BIS a AIS.



Obr. 5.20: Odhad efektívneho počtu častíc v priebehu procesu detekcie polohy robota pomocou BIS, v každom súbore bolo generovaných 100 častíc, na osi x index postupných realizácií sínusovej vlny.



Obr. 5.21: Odhad efektívneho počtu častíc v priebehu procesu detekcie polohy robota pomocou AIS, v každom súbore bolo generovaných 100 častíc, na osi x index postupných realizácií sínusovej vlny.

Ako vidíme aj v grafe odhadu efektívneho počtu častíc je viditeľná veľká nepresnosť v okolí snímku 210, odhad efektívneho počtu častíc klesne skoro na 0 a na rozdiel od iných časti procesu, kde toto nastane, trvá viac časových krokov než hodnota odhadu efektívneho počtu častíc zase vzrastie. Ďalším pozorovaním v grafe BIS je, že k prevzorkovaniu dochádza skoro pre každý snímok.

Do kontrastu môžme postaviť graf odhadu efektívneho počtu častíc pre AIS. Pri tomto algoritme vygenerujeme prvotný súbor častíc (pri ňom nedochádza k prevzorkovaniu) a z neho následne ďalší jeden alebo viac súborov častíc po postupnom spresňovaní hustoty, z ktorej sú častice generované. Prevzorkovaný je až posledný vygenerovaný súbor častíc a teda odhad efektívneho počtu častíc je taktiež zostrojený až z tohto finálneho súboru. Konkrétny graf je výsledkom procesu s voľbou jedného spresnenia, generovali sme teda 2-krát 50 častíc. Vidíme, že k prevzorkovaniu nedochádza tak často a celkové percento zachovaných častíc je vyššie.

Okrem grafického znázornenia je možné konvergenciu časticového filtra vyjadriť aj numericky. Pre vykreslené realizácie algoritmov časticového filtra platí:

filter	počet častíc	$\frac{1}{K}\sum_{k=1}^{K} R_k^1$	$\max\left\{R_k^1\right\}$
BIS	100	80.73	141
	500	80.03	119.9
AIS	2x50	79.94	131.37
	5x100	79.57	118.98
AMIS	2x50	79.72	120.67
	5x100	79.52	119.83

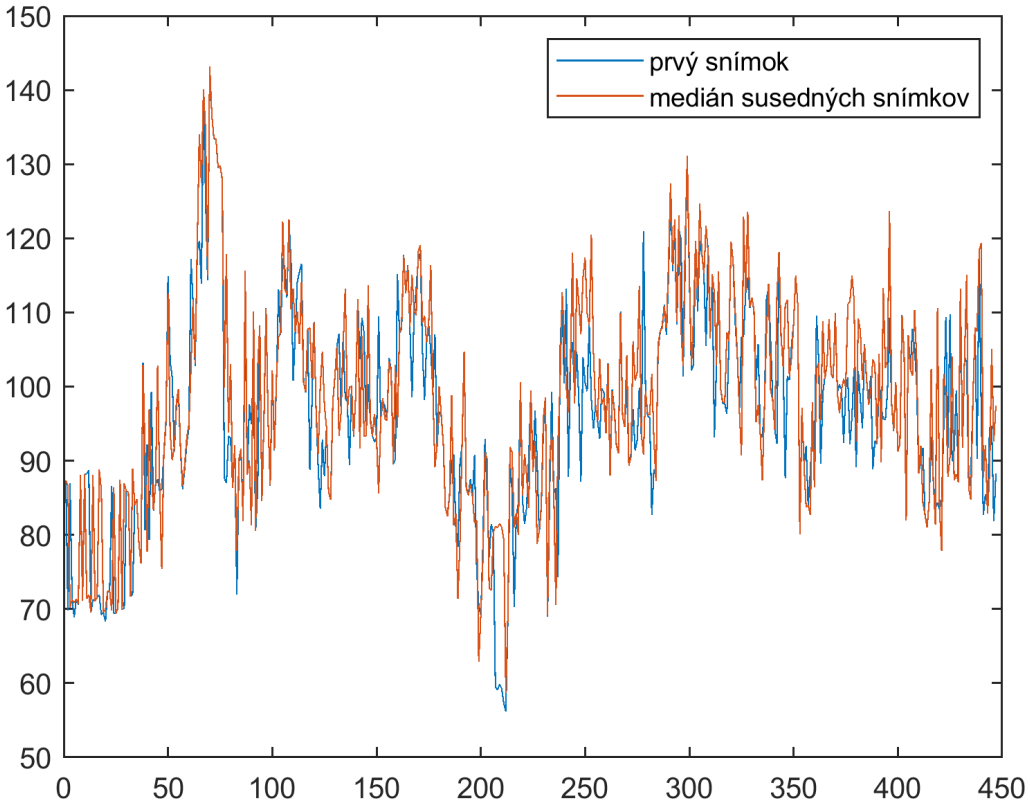
Tabuľka 5.2: Porovnanie BIS, AIS a AMIS pre zvyšujúci sa počet častíc

Z grafických aj numerických porovnaní vidíme, že algoritmy splňujú teóriu. Presnosť sa zvyšuje so zvyšujúcim sa počtom častíc a pre zložitejšie algoritmy vzorkovania, avšak nie veľmi výrazne.

5.3 Spresnenie odhadu slepou dekonvolúciou

Pre lepšie vizuálne znázornenie prípadného zlepšenia modifikáciou algoritmu slepej dekonvolúcie sa pozrieme na niekoľko riadkov rezidua r_3^k, teda snímkov videa po odčítaní pozadia a konvolúcie jadra a šablóny.

5.3.1 Spresnenie pozadia



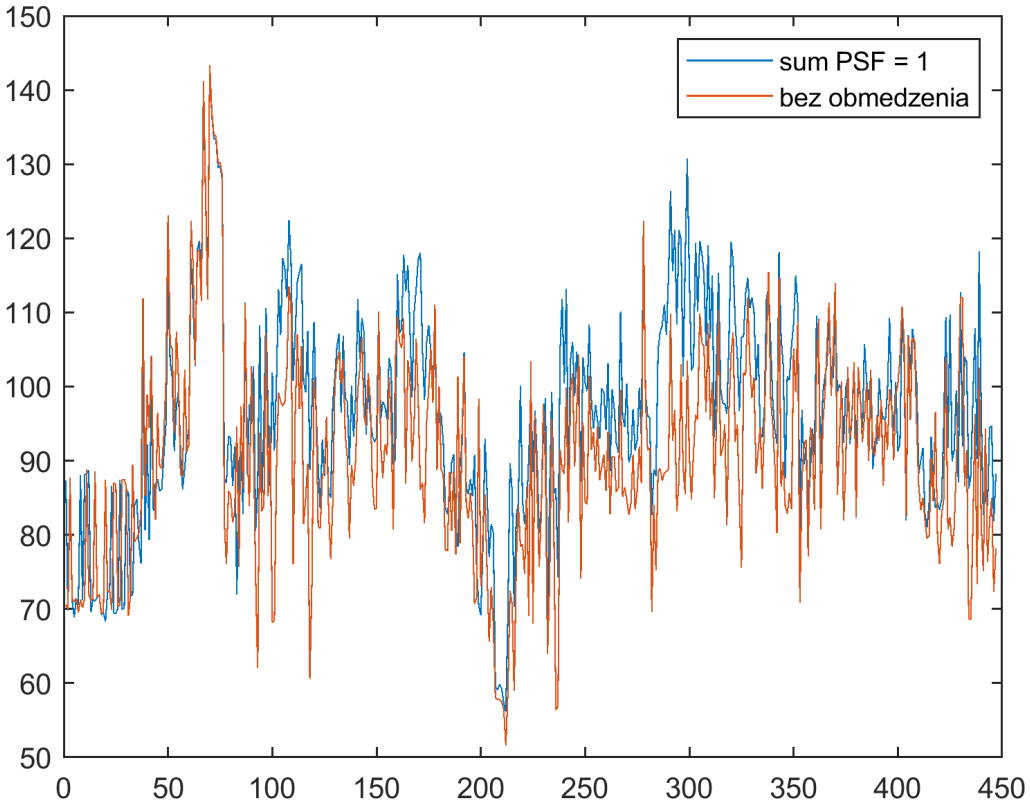
Obr. 5.22: R_k^3 pre dekonvolúciu s pozadim ako prvný snímok a medián susedných snímkov, na osi x je uvedené poradové číslo snímku vo videu

pozadie	$\frac{1}{K}\sum_{k=1}^{K} R_k^3$	$\max\left\{R_k^3\right\}$
prvý snímok	95.89	143.08
medián susedných snímkov	97.52	143.14

Tabuľka 5.3: Porovnanie dekonvolúcie pre dve konštrukcie pozadia

Ako je viditeľné z grafu a štatistík, konštrukcia pozadia ako medián susedných snímkov nezlepšuje presnosť slepej dekonvolúcie. Pozadie má veľmi blízko konštantnému.

5.3.2 Normalizácia jadra



Obr. 5.23: R_k^3 pre dekonvolúciu s obmedzením na sumu jadra a bez obmedzania, na osi x je uvedené poradové číslo snímku vo videu

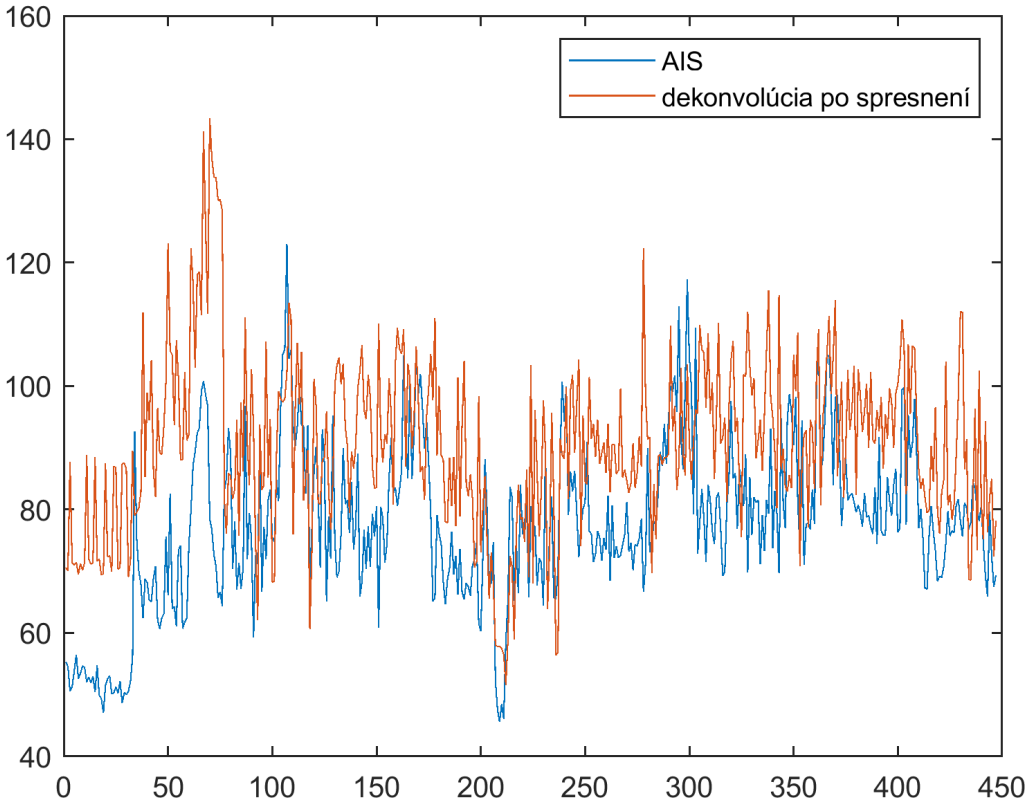
suma jadra	$\frac{1}{K}\sum_{k=1}^{K} R_k^3$	$\max\left\{R_k^3\right\}$
1	95.88	143.08
bez obmedzenia	90	143.36

Tabuľka 5.4: Porovnanie dekonvolúcie pri dve obmedzenie na sumu jadra

Na rozdiel od spresnenia pozadia táto modifikácia výrazne zlepší presnosť slepej dekonvolúcie vo veľkom počte snímkov.

5.3.3 Výsledky spresnenia

Za spresnenie považujeme rozvoľnenie obmedzenia na sumu odhadnutého konvolučného jadra, konštrukcia pozadia ako medián susedných snímkov sa pre túto aplikáciu neoplatila. Dekonvolúciu sme porovnali s AIS keďže zložitejšie algoritmy vzorkovania sa ukázali byť presnejšie. Ak algoritmy posudzujeme podľa metriky R_k^3, AIS je presnejší ako AMIS, ako je zreteľné v tabuľke 5.1. Priemer posunutí do všetkých častíc je pochopiteľne presnejší pre AIS, kde berieme do úvahy len druhú, presnejšiu generáciu na rozdiel od AMIS.

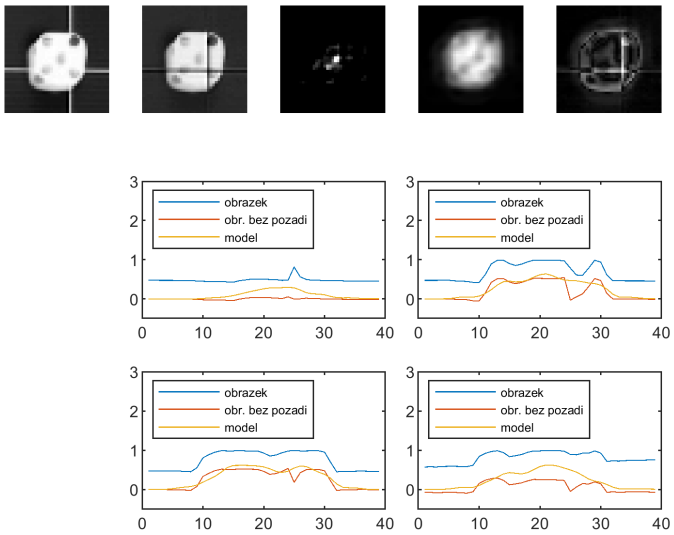


Obr. 5.24: R_k^3 pre AIS 2x50 častíc a dekonvolúciu po spresnení

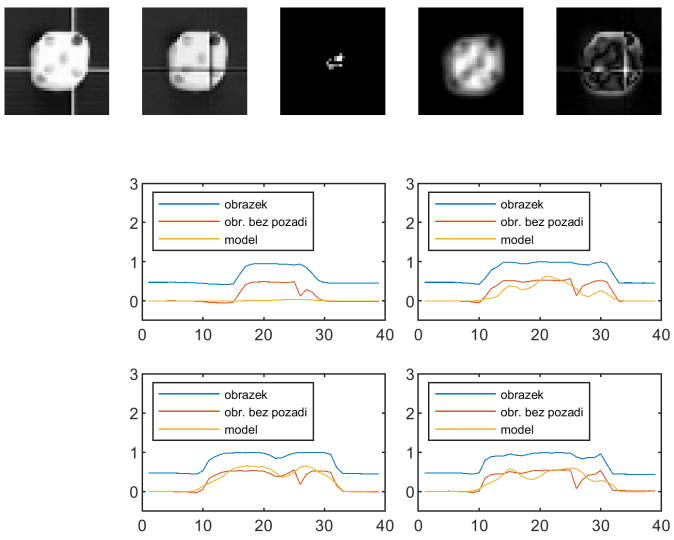
algoritmus	$\frac{1}{K}\sum_{k=1}^{K} R_k^3$	$\max\left\{R_k^3\right\}$
AIS 2x50	77.65	123
dekonvolúcia po spresnení	90	143.36

Tabuľka 5.5: Porovnanie spresnenej dekonvolúcie a AIS

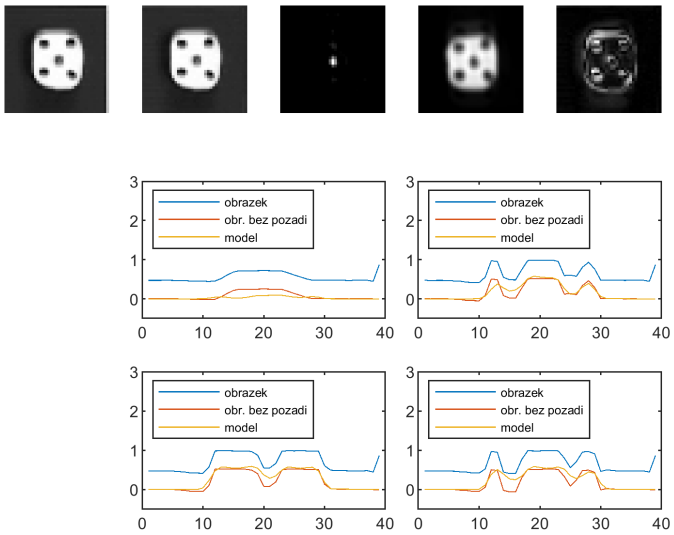
Dekonvolúcia po spresnení stále zaostáva za časticovými filtrami. Rozdiel ukážeme aj na 3 snímkoch podobne ako v prípade časticových filtrov. Pre každé reziduum vykreslíme 10., 15., 20. a 25. riadok, ktoré približne odpovedajú oblasti tesne nad robotom a tri riadky v ktorých sa nachádzajú diery v robotovi ako je viditeľné napríklad na nerozmazanom snímku 213 v obrázku 5.27



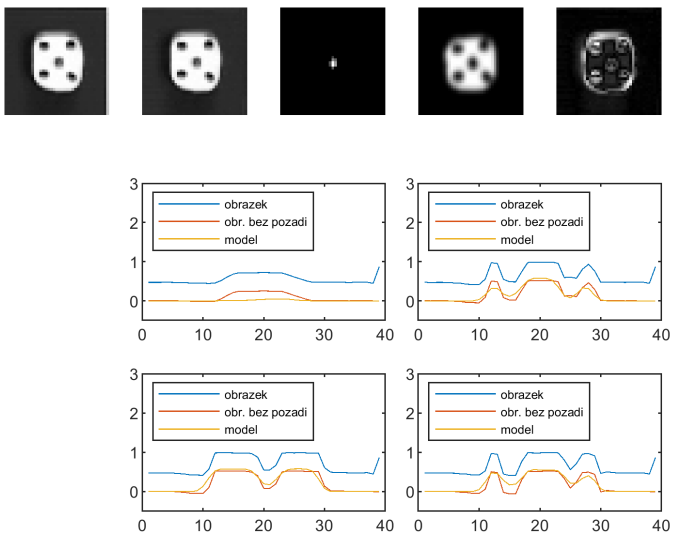
Obr. 5.25: Snímok 90, základná verzia dekonvolúcie. Obrázok videa, obrázok po odčítaní pozadia, odhadnuté jadro, konvolúcia jadra a šablóny, reziddum. Riadky 10, 15, 20 a 25, obrázok, obrázok po odčítaní pozadia a konvolúcia jadra a šablóny.



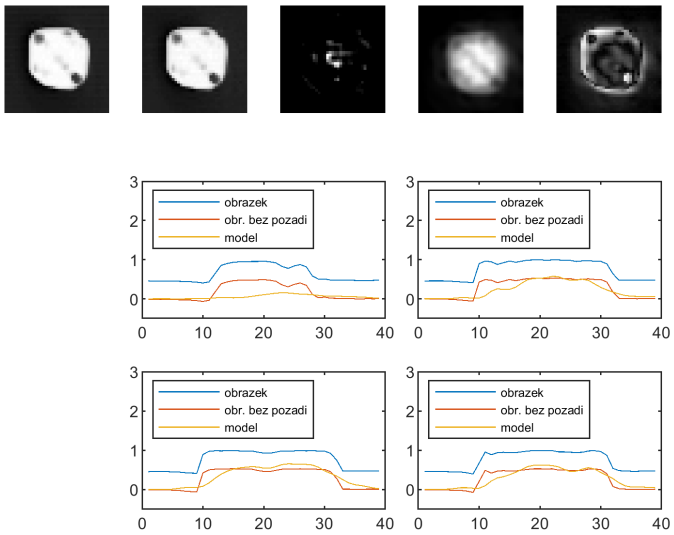
Obr. 5.26: Snímok 90, spresnená dekonvolúcia. Obrázok videa, obrázok po odčítaní pozadia, odhadnuté jadro, konvolúcia jadra a šablóny, reziduum. Riadky 10, 15, 20 a 25, obrázok, obrázok po odčítaní pozadia a konvolúcia jadra a šablóny.



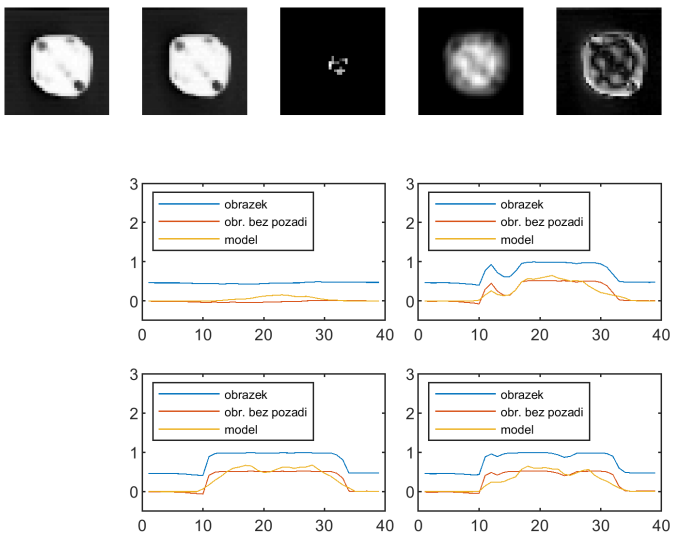
Obr. 5.27: Snímok 213, základná verzia dekonvolúcie. Obrázok videa, obrázok po odčítaní pozadia, odhadnuté jadro, konvolúcia jadra a šablóny, reziduum. Riadky 10, 15, 20 a 25, obrázok, obrázok po odčítaní pozadia a konvolúcia jadra a šablóny.



Obr. 5.28: Snímok 213, spresnená dekonvolúcia. Obrázok videa, obrázok po odčítaní pozadia, odhadnuté jadro, konvolúcia jadra a šablóny, reziduum. Riadky 10, 15, 20 a 25, obrázok, obrázok po odčítaní pozadia a konvolúcia jadra a šablóny.



Obr. 5.29: Snímok 362, základná verzia dekonvolúcie. Obrázok videa, obrázok po odčítaní pozadia, odhadnuté jadro, konvolúcia jadra a šablóny, reziduum. Riadky 10, 15, 20 a 25, obrázok, obrázok po odčítaní pozadia a konvolúcia jadra a šablóny.

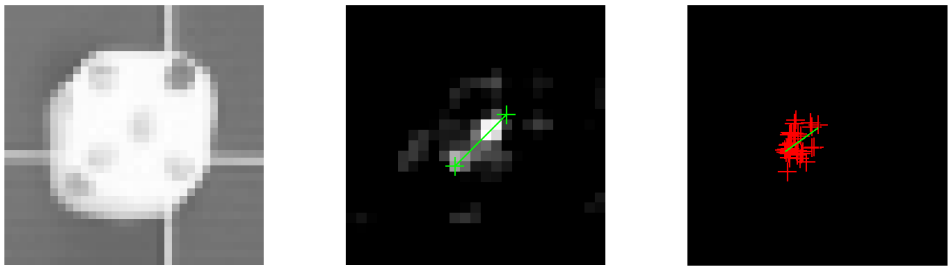


Obr. 5.30: Snímok 362, spresnená dekonvolúcia. Obrázok videa, obrázok po odčítaní pozadia, odhadnuté jadro, konvolúcia jadra a šablóny, reziduum. Riadky 10, 15, 20 a 25, obrázok, obrázok po odčítaní pozadia a konvolúcia jadra a šablóny.

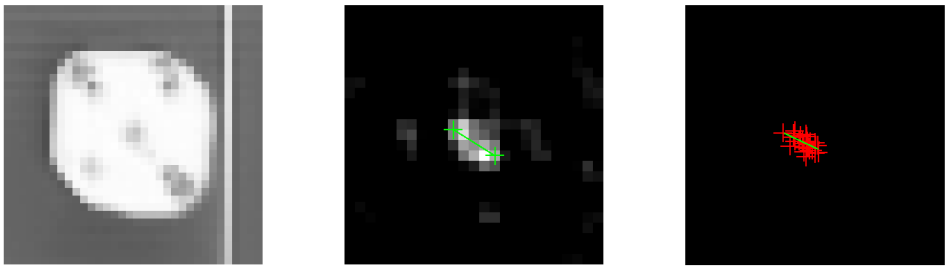
5.4 Konvolučné jadro

Konvolučné jadro odhadnuté slepou dekonvolúciou a priamka ním preložená môže naznačovať smer pohybu v prípade rozmazaného obrázku. Pre porovnanie boli vykreslené častice pre BIS a 100 častíc po prevzorkovaní a nimi preložená priamka.

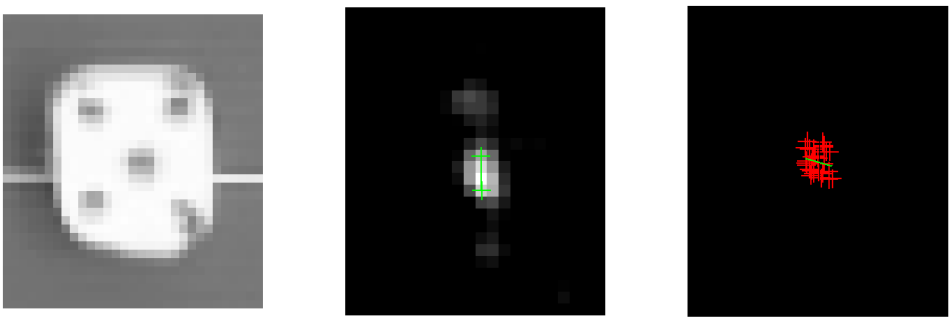
Ako vidíme z obrázkov, dekonvolúcia je úspešnejšia v určení správneho smeru pohybu.



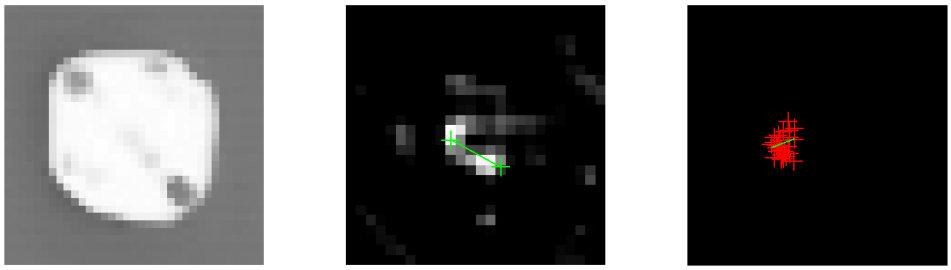
Obr. 5.31: Snímok 90. Konvolučné jadro a častice BIS.



Obr. 5.32: Snímok 107. Konvolučné jadro a častice BIS.



Obr. 5.33: Snímok 315. Konvolučné jadro a častice BIS.



Obr. 5.34: Snímok 362. Konvolučné jadro a častice BIS.

Záver

Cieľom práce bolo zoznámiť sa s algoritmami slepej dekonvolúcie a bayesovskej filtrácie, aplikovať ich pri detekcii polohy magneticky poháňaného mikrorobota z videozáznamu a následne algoritmy porovnať.

Keďže nie je známa presná poloha robota, zvolili sme metriky, na základe ktorých sme porovnali presnosť jednotlivých algoritmov. Časticový filter sa pri použití základného modelu ukázal ako najpresnejší. So zvyšujúcim sa počtom častíc sa jeho presnosť zvyšuje. Zložitejšie obdoby časticového filtra sa ukázali ako presnejšie.

Slepá dekonvolúcia je však taktiež použiteľnou metódou, keďže je deterministická, rýchlejšia a lepšie reprezentuje rozmazané obrázky. Pomocou slepej dekonvolúcie je taktiež možné najčastejšie správne odhadnúť smer pohybu mikrorobota v danom okamihu. Navrhli sme taktiež zlepšenie presnoti slepej dekonvolúcie.

Literatúra

[1] Z. Sabolová. *Odhadování pohybu mikrorobotů v magnetickém poli*. Praha, 2019. Bakalářská práce. České vysoké učení technické, Fakulta jaderná a fyzikálně inženýrská. Vedoucí práce: doc. Ing. Václav Šmídl, Ph.D.

[2] F. Gustafsson. *Particle filter theory and practice with positioning applications* IEEE Aerospace and Electronic Systems Magazine, vol. 25, no. 7, pp. 53-82, 2010.

[3] B. Ristic, S. Arulampalam, N. Gordon. *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Addison-Wesley, Norwood, Massachusetts, 2004.

[4] E. Wan, R. Merwe. (2000). *The Unscented Kalman Filter for Nonlinear Estimation*. Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium, pp. 153-158, 2000.

[5] J. Hammersley, K. Morton. *Poor man´s Monte Carlo* Journal of the Royal Statistical Society, pp. 23, 1954.

[6] N.J. Gordon, D.J. Salmond, A.F.M. Smith. *Novel approach to nonlinear/non-Gaussian Bayesian state estimation*. IEE Proceedings F - Radar and Signal Processing, vol. 140, no. 2, pp. 107-113, 1993.

[7] X. Hu, T. B. Schon, L. Ljung. *A Basic Convergence Result for Particle Filtering*. IEEE Transactions on Signal Processing, vol. 56, no. 4, pp. 1337-1348, 2008.

[8] T. Li, M. Bolic, P. Djuric. *Resampling Methods for Particle Filtering: Classification, implementation, and strategies* IEEE Signal Processing Magazine, vol. 32, no. 3, pp. 70-86, 2015.

[9] M.-S. Oh, J. O. Berger. *Adaptive importance sampling in Monte Carlo integration*. Journal of Statistical Computation and Simulation, 41(3-4):143-168, 1992.

[10] J.-M. Cornuet, J.-M. Marin, A. Mira, C. P. Robert. *Adaptive Multiple Importance Sampling*. Scandinavian Journal of Statistics, 39. , 2009.

[11] J. Kuthan, F. Mach. *Magnetically Guided Actuation of Ferromagnetic Bodies on the Planar Surfaces: Numerical Modeling and Experimental Verification*. 2017 18th International Conference "Computational Problems of Electrical Engineering"(CPEE) 1-4, 2017.

[12] Y. Xu, K. Xu, J. Wan, Z. Xiong, Y. Li. *Research on Particle Filter Tracking Method Based on Kalman Filter*. 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), 1564-1568, 2018.

[13] S. Damelin, W. Miller. *The Mathematics of Signal Processing*. Cambridge University Press, 2011.

[14] A. Levin, Y. Weiss, F. Durand, W. T. Freeman. *Understanding Blind Deconvolution Algorithms*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 12, pp. 2354-2367, Dec. 2011.

[15] J. Kotera, F. Šroubek. *Blind deconvolution of images with model discrepancies using maximum a posteriori estimation with heavy-tailed priors*. Proceedings of SPIE - The International Society for Optical Engineering. 9404. 10.1117/12.2077158, 2015.

[16] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein. *Distributed optimization and statistical learning via the alternating direction method of multipliers* Foundations and Trends in Machine Learning, vol. 3, no. 1, pp. 1–122, 2011.

[17] S. Nadarajah. *A generalized normal distribution*. Journal of Applied Statistics, 32:7, 685-694, 2005.