VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

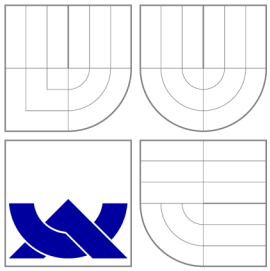
SLEDOVÁNÍ OBJEKTU VE VIDEU

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

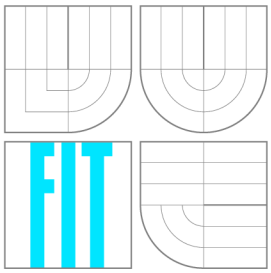
AUTOR PRÁCE
AUTHOR

PETER BEŤKO

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SLEDOVÁNÍ OBJEKTU VE VIDEU

OBJECT TRACKING IN VIDEO

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

PETER BEŤKO

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. ADAM HEROUT, Ph.D.

BRNO 2010

Abstrakt

Tato bakalářská práce se zabývá implementací a testováním různých algoritmů na sledování objektu ve videu, založených na principu Částicového filtru. Jejich vlastnosti jsou porovnávány a diskutovány pomocí vytvořeného nástroje, objektivně hodnotícího přesnost sledování. Dalšími tématy práce jsou dynamické přizpůsobování se aktuálnímu charakteru scény a zotavení se z dočasného překrytí sledovaného objektu jiným objektem.

Abstract

This bachelor's thesis is dealing with implementation and testing of algorithms used for object tracking in video, based on the principle of Particle filter. Their features are compared and discused using a tool, that objectively evaluates the accuracy of tracking. Other topics are dynamic adaptation on actual scene characteristics and recovery from object overlay.

Klíčová slova

sledování objektu, Částicový filtr, zpracování obrazu, histogram, překrývání objektů

Keywords

object tracking, Particle filter, image processing, histogram, object overlay

Citace

Peter Beťko: Sledování objektu ve videu, bakalářská práce, Brno, FIT VUT v Brně, 2010

Sledování objektu ve videu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Adama Herouta, Ph.D.

.....
Peter Beťko
19.05.2010

Poděkování

Chtěl bych poděkovat panu Ing. Adamu Heroutovi, Ph.D. za odbornou pomoc a cenné rady, které mi pomohly k vytvoření této bakalářské práce.

Obsah

Kapitola 1

Úvod

Vzrastajúci výkon počítačov, zvyšujúca sa dostupnosť kamier s vysokým rozlíšením spolu s potrebou automatickej analýzy videa dali podnet na rozvoj systémov sledovania objektu. Tie sú vo vysokej miere využívané ako súčasť vysokoúrovňových aplikácií, v ktorých je vyžadovaná detekcia polohy objektu v každej snímke videa. Príkladom môžu byť dohľadové systémy na letiskách, systémy na monitorovanie dopravnej situácie či aplikácie na ovládanie počítača pomocou gest.

Napriek intenzívne prebiehajúcemu výskumu v tejto oblasti má sledovanie objektu ešte mnoho slabých miest. Problémy môže spôsobovať pohyb kamery, prekrývanie sledovaného objektu iným objektom či zmena osvetlenia scény. Cieľom tejto práce je porovnať vlastnosti a zhodnotiť výhody rôznych prístupov k sledovaniu objektu. Dôraz je kladený na algoritmy založené na matematickom princípe Časticového filtra (z anglickej literatúry známy pod pojmom *Particle filter*). K ich objektívnemu porovnaniu slúži vytvorený prostriedok na ohodnocovanie presnosti sledovania. Práca tiež popisuje návrh, implementáciu a testovanie nástroja na automatické prispôsobovanie sa charakteru scény a nástroja na detekciu a vysporiadanie sa s dočasným prekrytím sledovaného objektu iným objektom.

Jadro práce je štrukturované do piatich kapitol. Prvá z nich sa zaoberá popisom problematiky sledovania objektu vo videu, na čo nadväzuje kapitola s teoretickými poznatkami o Časticovom filtri. Ďalšie dve časti sú venované návrhu a implementácií nástrojov potrebných pre účely práce. Posledná kapitola, na ktorú je kladený najväčší dôraz, sa zaoberá experimentami. Ich výsledky sú dokreslené ukážkovými videami, ktoré sú prílohou práce.

Kapitola 2

Sledovanie objektu vo videu

Sledovanie objektu, inými slovami odhad trajektórie telesa počas jeho pohybu v scéne, je dôležitou a rýchlo sa rozvíjajúcou problematikou z oblasti spracovania obrazu. Táto kapitola si dáva za cieľ stručne vysvetliť jej základy. Najskôr podáva informácie o možnostiach reprezentácie a detekcie objektov, ďalej sa zameriava na kategorizáciu jednotlivých algoritmov na ich sledovanie. Vybrané algoritmy sú bližšie popísané. Informácie k tejto kapitole boli čerpané z [14].

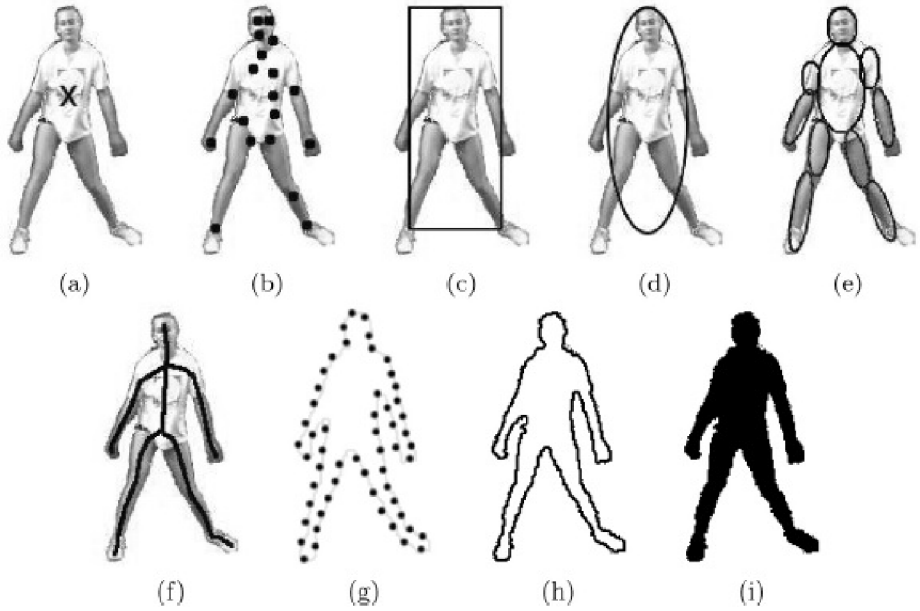
2.1 Reprezentácia objektu

Sledovaný objekt môže byť reprezentovaný viacerými spôsobmi, pričom vhodnosť ich použitia závisí od typu sledovanej scény a plánovaného nasadenia výsledného programu. Možné spôsoby reprezentácie objektu na základe tvaru zobrazuje obrázok 2.1. Bod (a) alebo skupina bodov (b) sa používa v prípade, že sledovaný objekt zaberá veľmi malú časť videa. Geometrické tvary (c) a (d) nájdu využitie najmä v scénach s objektami nemeniacimi svoj tvar, dajú sa však použiť aj v opačných prípadoch. Pre tie je však najoptimálnejšie použiť reprezentáciu pomocou kostry (f), kontúry (g) a (h) alebo siluety objektu (i).

Reprezentáciu na základe tvaru je možné kombinovať so vzhľadovou reprezentáciou. Okrem iných sa ponúka využitie histogramu alebo vzorky (v anglickej literatúre *template*). Časticový filter, implementovaný v tejto práci, kombinuje tieto dva prístupy s vyznačením polohy objektu pomocou obdĺžnika. V dokumente [14] je uvedené, že takéto riešenie je náchylné na zmenu osvetlenia v scéne, prístup s využitím vzorky je naviac problematický pri zmene tvaru alebo natočenia sledovaného objektu.

2.2 Detekcia objektu

Známym algoritmom je odčítanie pozadia (v anglickej literatúre *Background Subtraction*). Metóda je založená na vytvorení modelu pozadia a následného hľadania odchýlok pre prichádzajúce snímky. Každá významná zmena oproti pozadiu indikuje pohybujúci sa objekt. Okrem porovnávania presne korešpondujúcich bodov sa naskytá možnosť porovnávať bod s okolím jeho „dvojičky“, alebo pracovať s blokmi o rozmeroch napríklad 5x5 pixelov. Dôsledkom týchto postupov je menšia náchylnosť na chvenie kamery, zanedbateľný pohyb na pozadí alebo jemné zmeny osvetlenia scény. Fungovanie algoritmu približuje obrázok 2.2, ktorého ľavá časť reprezentuje zachytenú snímku videa, pravá výsledok po odčítaní pozadia.



Obr. 2.1: Reprezentácia objektu, prebrané z [14]. (a) ťažisko objektu, (b) viac bodov, (c) obdĺžniková vzorka, (d) elipsovitá vzorka, (e) čiastkové vzorky, (f) kostra objektu, (g) kontrolné body na kontúre objektu, (h) kontúra objektu, (i) silueta objektu

Iná možnosť detekcie objektu je založená na významných bodoch v obraze. Tieto body sa svojou farebnou intenzitou výrazne odlišujú od svojho okolia. Výhodou metódy je jej odolnosť voči zmenám osvetlenia. Známym spôsobom detekcie významných bodov je použitie *Moravcovho operátora* [8], ktoré môže fungovať nasledovne. Z okolia 4x4 body sa vypočítajú zmeny intenzity v horizontálnom, vertikálnom a diagonálnom smere. Pokiaľ je najmenšia z týchto hodnôt lokálnym maximom v okolí 12x12 bodov, bod je prehlásený za významný. Ďalšími metódami sú napr. *Harrisov* a *SIFT* (*Scale-Invariant Feature Transform*) detektor. Porovnanie prístupov možno nájsť v [7].

Obr. 2.2: Detekovanie objektu odčítaním pozadia, prebraté z [4]

2.3 Rozdelenie algoritmov na sledovanie objektu

Podľa [14] možno algoritmy na sledovanie objektu kategorizovať do troch skupín, tak ako to zobrazuje tabuľka 2.1. Ku každej kategórií je uvedených niekoľko konkrétnych príkladov algoritmov. Názvy sú ponechané v anglickom jazyku, nakoľko u väčšiny z nich nie je zaužívaný slovenský ekvivalent.

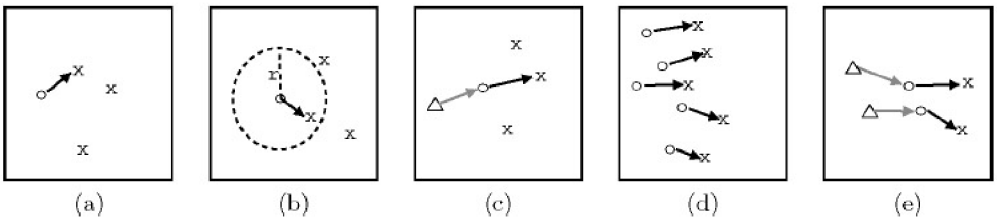
Point Tracking	
Deterministic methods	MGE tracker GOA tracker
Statistical methods	Kalman filter Particle filter JPDAF
Kernel Tracking	
Template and density based apperarance models	Mean-shift KLT Layering
Multi-view appearance models	Eigentracking SVM tracker
Silhouette Tracking	
Contour evolution	State space models Variational methods Heuristic methods
Matching shapes	Hausdorff Hough transform

Tabuľka 2.1: Rozdelenie algoritmov na sledovanie objektu, čiastočne prebraté z [14]

Prvá skupina, voľne preložiteľná ako sledovanie bodov, pracuje na základe reprezentácie sledovaných objektov pomocou bodov. Tie môžu byť vo všeobecnosti nahradené aj inými geometrickými útvarmi. Problémom je správne priradiť body z času t k bodom z času $t-1$ a tak zachytiť dynamiku objektu. Deterministické, v niektorých prípadoch aj štatistické metódy, to riešia kombináciou pravidiel zobrazených na obrázku 2.3:

- *blízkosť* – nová poloha objektu sa oproti predchádzajúcej výrazne nezmení
- *maximálna rýchlosť* – definuje hornú hranicu rýchlosti objektu, čím limituje kruhové okolie, v ktorom sa môže nachádzať bod z novej snímky
- *malá zmena rýchlosti* – rýchlosť a smer pohybu sa nemení vo veľkej miere
- *spoločný pohyb* – okolité body sa pohybujú podobnou rýchlosťou a smerom
- *tuhosť objektu* – predpokladá, že sledované objekty nemenia svoj tvar, preto sa vzdialenosť jednotlivých bodov zásadne nemení

Štatistické metódy zanášajú do sledovania objektu neurčitosti modelujúce šum a iné negatívne vplyvy vo videu. Základom fungovania sú dva modely. Prvý popisuje vzťah



Obr. 2.3: Pravidlá pre hľadanie odpovedajúcich si bodov, prebraté z [14]
(a) blízkosť, (b) maximálna rýchlosť, (c) malá zmena rýchlosti, (d) spoločný pohyb, (e) tuhosť objektu

nového stavu objektu na základe toho predchádzajúceho a šumu, druhý pojednáva o meraní závislom na aktuálnom stave a šume merania. Konkrétnejšie fungovanie je vysvetlené v ďalšej kapitole, na Časticovom filtri.

Druhá skupina algoritmov, anglicky označovaná ako *Kernel tracking*, realizuje sledovanie na základe porovnávania pomocou vzorky. Pohyb objektu je väčšinou charakterizovaný transformáciou, ako napríklad posunutím či rotáciou. Zástupcom tejto skupiny algoritmov je tzv. *Template matching*, ktorý sa pomocou korelácie pixelov snaží nájsť vhodného kandidáta pre sledovaný objekt, reprezentovaný referenčnou vzrokou. Výpočtová náročnosť tejto metódy je pomerne vysoká, dá sa však znížiť hľadaním objektu v okolí jeho predchádzajúcej pozície. Okrem klasickej korelácie pixelov je možné použiť prístup využívajúci histogramy, kombináciu týchto prístupov alebo počítanie priemernej hodnoty pixelov v referenčnej a kandidátnych vzorkách.

Ďalším zástupcom je algoritmus *Mean-shift*. Základnou myšlienkou je iteratívne maximalizovať podobnosť histogramov sledovaného objektu a jeho hypotetickej polohy. V každej iterácií sa okolo aktuálnej polohy vygenerujú kandidátne, ktorých histogramy sa porovnávajú s referenčným. Na základe najväčšej podobnosti sa vypočíta tzv. *mean-shift vector*, podľa ktorého sa upraví hypotetická poloha. Algoritmus zvyčajne konverguje po piatich až šiestich krokoch, po ktorých je nájdená pozícia s histogramom najviac podobným referenčnému histogramu.

Ďalšie algoritmy v tejto kategorii sú schopné pracovať s referenčnými snímkami objektu z viacerých uhlov, a tým sa vysporiadať s natáčaním objektu počas jeho sledovania.

Posledná kategória sa zaoberá sledovaním siluet. Tieto metódy poskytujú presný popis tvaru pre objekty, ktoré sa nedajú popísať jednoduchým geometrickým útvarom. Prvá podkategória, voľne preložiteľná ako porovnávanie tvaru, môže pracovať podobne ako už spomínaný *Template matching*. Model siluety objektu (hrany, kontúra objektu prípadne histogram) sú porovnávané s modelom z predchádzajúcej snímky. Referenčný model je reinicializovaný, aby sa predišlo problémom spôsobeným zmenami osvetlenia scény, natočenia a tvaru objektu.

Druhá podkategória by sa dala nazvať evolúcia kontúr. Funguje iteratívnym vyvíjaním kontúry z predchádzajúcej snímky. Táto evolúcia vyžaduje, aby sa časť objektu v aktuálnej snímke prekrývala s objektom v tej predchádzajúcej.

Kapitola 3

Časticový filter

Kapitola popisuje teoretické základy algoritmu implementovaného v tejto práci. Začína definovaním základného princípu, pokračuje popisom dvoch základných modelov a fáz, na ktorých je Časticový filter postavený. Tie sú vysvetlené na rovniciach prebratých z [1]. Ku koncu zhŕňa fungovanie algoritmu pomocou prehľadného obrázka.

Časticový filter prvýkrát predstavili páni M. Isard a A. Blake v roku 1998 [5]. Je odvodený od známeho algoritmu *Kalman filter* [10], ktorého základným problémom je fakt, že všetky rozloženia pravdepodobnosti sú *Gaussovské* a v čase t môže byť vybraný maximálne jeden kandidát sledovaného objektu [9]. Túto nevýhodu odstraňuje Časticový filter. Jeho hlavnou myšlienkou a odlišnosťou od spomínaného predchodcu je aproximovanie funkcie hustoty pravdepodobnosti výskytu sledovaného objektu (ďalej len *pdf – Probability Density Function*) metódou *Monte Carlo*, čo prebieha pomocou súboru častíc (anglicky *particle*). Meraním sú jednotlivým časticiam priradené váhy, ktoré označujú podobnosť so sledovaným objektom. Na základe váh dochádza k prevzorkovaniu častíc, kedy tie lepšie ohodnotené sú znásobené na úkor ostatných. Do ďalšej iterácie algoritmu „prežíva“ viac kandidátov na sledovaný objekt, ktorí môžu byť pozorovaní súbežne [9]. Medzi ďalšie výhody Časticového filtra patrí nenáchylnosť na pohyb kamery.

Na popis dynamiky sledovaného objektu sa používajú dva modely [1]. Prvým je systémový model (v anglickej literatúre známy ako *system model*), popisujúci vývoj stavov sledovaného objektu v čase. Popisuje ho rovnica (3.1):

$$x_k = f_k(x_{k-1}, v_{k-1}) \tag{3.1}$$

Z tej vyplýva, že aktuálny stav objektu x_k je funkčne závislý na predchádzajúcom stave x_{k-1} a šume v_{k-1}. Model sa dá v pravdepodobnostnej forme označiť takto: $p(x_k|x_{k-1})$.

Druhým modelom je merací model (*measurement model*). Popisuje spôsob priraďovania váh jednotlivým časticiam. Meranie z_k v čase k závisí na stave systému a meracom šume takisto v čase k. Model je popísaný vzťahom (3.2):

$$z_k = h_k(x_k, n_k) \tag{3.2}$$

Alternatívne sa dá označiť ako $p(z_k|x_k)$.

Na spomínané modely nadväzujú dve základné fázy, ktorými sú predpovedanie a aktualizácia (v anglickej literatúre *prediction* a *update*) [1].

V prvej spomínanej fáze sa predpovedá tvar apriórnej funkcie *pdf* pre aktuálnu snímku videa. Inými slovami, ide o odhad polohy objektu, tak ako to zobrazuje rovnica (3.3).

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})\mathrm{d}x_{k-1} \tag{3.3}$$

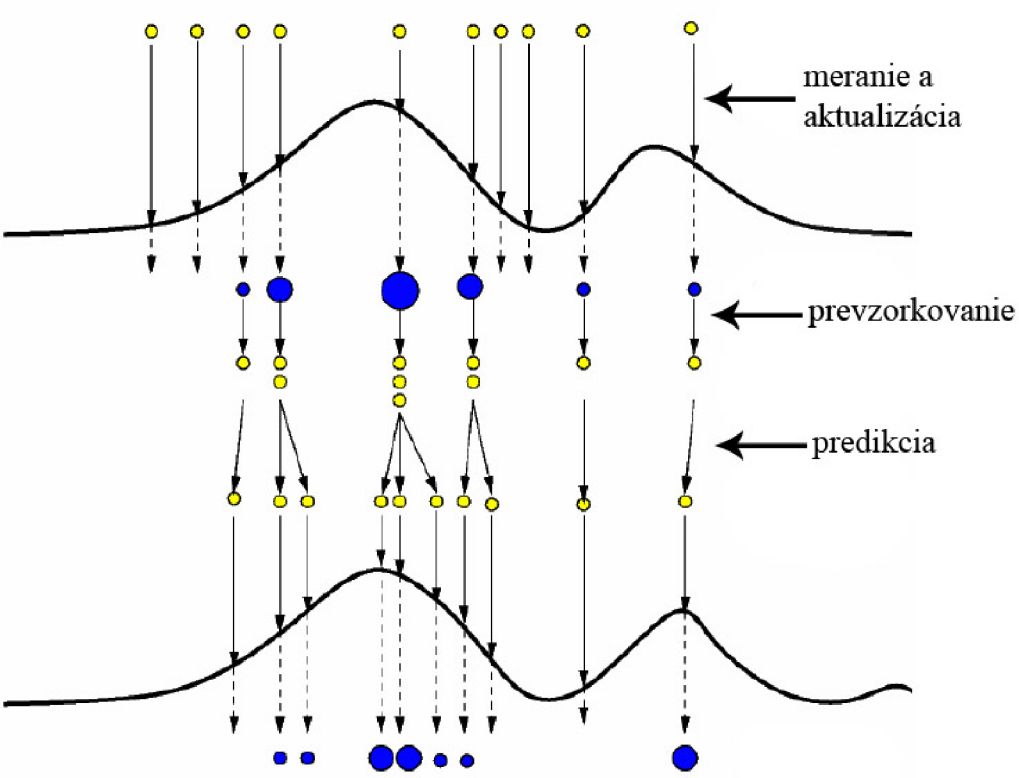
Apriórny stav závisí na aposteriórnom stave z predchádzajúcej iterácie $p(x_{k-1}|z_{1:k-1})$, opísaného rovnicou (3.4) z aktualizačnej fázy. $p(x_k|x_{k-1})$ je systémový model, definujúci závislosť nového stavu na predchádzajúcom.

Vo fáze aktualizácie sa aposteriórne, na základe merania z_k z času k, upravuje funkcia *pdf*, predpovedaná v predchádzajúcej fáze. Tohto je dosiahnuté pomocou *Bayesovho pravidla*, ktoré predstavuje mechanizmus obnovenia informácií o stave objektu použitím informácií z nového merania. Fázu bližšie popisuje rovnica (3.4), prebraná z [1].

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})} \tag{3.4}$$

Aposteriórny stav *pdf* sa získa pomocou aktuálnych meraní $p(z_k|x_k)$, popísaných meracím modelom a aktuálnou apriórnou funkciou $p(x_k|z_{1:k-1})$, popísanou rovnicou (3.3). Menovateľ vzorca (3.4) slúži ako normalizačná konštanta.

Po tejto fáze dochádza k prevzorkovaniu častíc (v anglickej literatúre sa stretneme s pojmom *resampling*). Častice s priradenou vysokou váhou sú znásobené, naopak tie s nízkou váhou zanikajú. Používanými metódami sú *SIS*(*Sequential Importance Sampling*) a *SIR* (*Sampling Importance Resampling*). Presný popis ich fungovania je možné nájsť v [1].



Obr. 3.1: Shéma fungovania Časticového filtra, čiastočne prebraté z [6]

Praktické fungovanie algoritmu zobrazuje obrázok 3.1. Pre prehľadnosť je zobrazenie jednorozmerné. Žltými kruhmi je reprezentovaných 10 častíc s rozličnými polohami. Tie sú následne ohodnotené pomocou meracieho modelu, poloha častice s najväčšou váhou reprezentuje novú polohu sledovaného objektu. Prevzorkovaním častíc dochádza k znásobeniu tých lepšie ohodnotených, naopak tie s nízkymi váhami zanikajú. Po prevzorkovaní dochádza k predikcii, kedy sa pripočítaním šumu upraví poloha častíc a cyklus môže začať odznovu pre ďalšiu snímku videa.

Kapitola 4

Návrh programu

Cieľom tejto kapitoly je detailne popísať návrh programu. Vychádza z teoretických poznatkov popísaných v časti 3 a praktických postrehov získaných pri implementácii a testovaní, ktoré samotný návrh spätne ovplyvnili. Kapitola začína popisom základnej myšlienky realizácie Časticového filtra, následne bližšie popisuje jeho najdôležitejšie súčasti. Ku koncu vysvetľuje mechanizmy navrhnuté nad rámec základnej funkcionality algoritmu.

4.1 Kostra programu

Program bude pracovať ako konzolová aplikácia ovládaná z príkazového riadku. S grafickým užívateľským prostredím sa pre jednoduchosť a zameranie práce na experimentovanie nepočíta.

V počiatočnej fáze programu dôjde k spracovaniu parametrov zadaných pomocou príkazového riadku. Tie budú zaznamenané tak, aby k nim bol možný jednoduchý prístup z viacerých miest programu. Následne dôjde k načítaniu videa špecifikovaného pri spustení. Zobrazí sa prvá snímka a užívateľ interaktívne, pomocou myši, vyberie sledovaný objekt. Ten sa uloží ako referenčný obrázok, ktorého veľkosť bude zmenená podľa hodnoty zadanej užívateľom. To bude mať za následok zníženie náročnosti výpočtu pri veľkých rozmeroch sledovaného objektu. S referenčným obrázkom budú porovnávané všetky častice vytvorené počas behu programu. V prípade, že bol zvolený spôsob ohodnotenia častíc pracujúci s využitím histogramu (bližšie popísané v časti 4.2), dôjde k jeho vytvoreniu. Je to z dôvodu optimalizácie programu, pretože je zbytočné aby sa histogram z referenčného obrázka znovu vytváral pri porovnávaní s každou časticou.

Po všetkých potrebných inicializáciach dôjde k samotnému sledovaniu vybraného objektu. Program iteratívne, pre všetky snímky zdrojového videa, prevedie nasledujúce kroky. Najskôr dôjde k predikcii novej polohy sledovaného objektu. Každej častici sa upraví poloha pričítaním náhodne vygenerovaného šumu v rozmedzí zadanom vstupnými parametrami programu. Analogicky sa môžu zmeniť aj rozmery častice a uhol jej natočenia. Takto upravený súbor častíc bude následne ohodnotený jedným z ohodnocovacích mechanizmov. Častica s najväčšou váhou, predstavujúca novú polohu sledovaného objektu, bude zobrazená. Poslednou operáciou bude prevzorkovanie častíc. Na základe ich ohodnotenia budú vygenerované nové, určené pre ďalšiu iteráciu. Jednotlivé snímky s vyznačenou polohou sledovaného objektu budú postupne zapisované do videosúboru, ktorý bude pred ukončením programu uložený na pevný disk počítača.

4.2 Ohodnocovanie častíc

Ohodnocovanie častíc spočíva v ich porovnávaní s referenčným obrázkom sledovaného objektu. Výsledkom porovnania bude váha častice reprezentovaná desatinným číslom, pričom vyššia hodnota znamená väčšiu podobnosť. Okrem schopnosti správne určiť váhu častice je dôležitá aj časová náročnosť porovnania, keďže táto operácia bude vykonaná pre každú časticu v každej snímke videa.

Prvým spôsobom ohodnocovania častíc bude odčítavanie hodnôt korešpondujúcich pixelov častice a referenčného obrázka. Konkrétne pôjde o sumu druhých mocnín rozdielov, tak ako to popisuje vzorec (4.1) prebratý z [3]:

$$\|arr1 - arr2\| = \sum_{x,y} (arr1_{x,y} - arr2_{x,y})^2 \tag{4.1}$$

Odpočítanie prebehne pre každú farebnú zložku *RGB modelu* zvlášť. Suma bude nakoniec vynásobená hodnotou -1, aby sa dodržalo pravidlo, že vyššie hodnoty značia lepšie ohodnotenie. Vo zvyšku práce bude tento spôsob označovaný ako základný algoritmus ohodnocovania častíc.

Ďalším spôsobom bude porovnávanie na základe histogramu vytvoreného zo šedotónového, resp. farebného obrázku (ďalej len šedotónový, resp. farebný histogram). Pre každú časticu sa vypočíta histogram, ktorý bude následne porovnaný s histogramom referenčného obrázka, vytvoreným v počiatočnej fáze programu. Na porovnanie bude použitá jedna z metód, známych z anglickej literatúry ako *Chi-square* alebo *Intersection*, ktoré popisujú vzorce (4.2) a (4.3) prebraté z [3].

$$d_{chi-square}(H_1, H_2) = \sum_i \frac{(H_1(i) - H_2(i))^2}{H_1(i) + H_2(i)} \tag{4.2}$$

$$d_{intersection}(H_1, H_2) = \sum_i min(H_1(i), H_2(i)) \tag{4.3}$$

Posledný spôsob bude kombináciou predchádzajúcich, vo zvyšku práce označovaný ako hybridný prístup. Referenčný obrázok a častica sa rozdelia na obecne MxN častí a z každej z nich bude následne vytvorený farebný histogram. Odpovedajúce si histogramy sa porovnajú, pričom celkové ohodnotenie častice bude rovné zápornej hodnote súčtu týchto porovnaní. Znovu teda platí že vyššie ohodnotenie znamená väčšiu podobnosť. V prípade, že hodnoty M a N budú nastavené na 1, pôjde o analógiu použitia farebného histogramu. Opačným extrémom bude nastavenie týchto hodnôt na rozmery referenčného obrázka, čo bude podobné základnému algoritmu ohodnocovania častíc.

4.3 Ohodnocovanie presnosti programu

Kvôli zámeru práce porovnať presnosť jednotlivých spôsobov ohodnocovania častíc a vplyv ostatných parametrov, bude potrebné vytvoriť nástroj na ohodnocovanie presnosti fungovania programu ako celku. Ten bude fungovať na základe porovnávania optimálnej trajektórie s hypotetickou, vypočítanou programom. Túto metódu popisuje článok [2], v ktorom je využitá na sledovanie viacerých objektov.

Užívateľovi bude umožnené spustiť program v špeciálnom *vyznačovacom* móde. Vyberie objekt, ktorý chce sledovať a následne, po uplynutí stanoveného počtu snímok, znovu vyznačí polohu sledovaného objektu. Takto bude pokračovať až do skončenia videa. Vyznačovanie bude prebiehať interaktívne, pomocou myši. Následne program zapíše do súboru rozmery sledovaného objektu, jeho počiatočnú polohu a takisto všetky jeho polohy, ktoré užívateľ vyznačil. K samotnému sledovaniu objektu nedôjde.

Po spustení programu v *testovacom* móde dôjde k načítaniu údajov zo súboru. Na základe zistenej počiatočnej polohy a rozmerov objektu sa bez zásahu užívateľa spustí sledovanie objektu. Ohodnotenie presnosti bude založené na rozdiele korešpondujúcich polôh objektu, vyznačených užívateľom v prvom móde a vypočítaných programom v druhom móde. Po jednorazovom vyznačení polôh objektu pre dané video bude možné spustiť sériu testov zameraných na vplyv rôznych nastavení na presnosť programu.

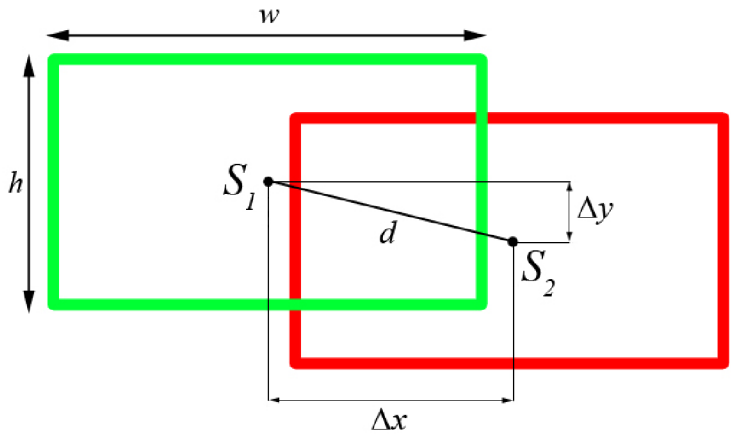
V článku [2] je spomenutá možnosť počítať odchýlku absolútne v pixeloch. Tá bude využitá v prípade, že budú zadané nenulové hodnoty rozptylu šumu pre rozmery častice, čo spôsobí zmeny ich veľkosti počas behu programu. Keďže absolútna odchýlka nezohľadňuje veľkosť sledovaného objektu, nebude možné objektívne porovnávať presnosť v rôznych scénach.

V prípade, že sa veľkosť častíc meniť nebude, odchýlka nebude počítaná absolútne, ale bude normalizovaná podľa veľkosti častice. To znamená, že presnosť algoritmu bude vyjadrená tým, o koľko percent veľkosti sledovaného objektu sa program odchýli. Vďaka tomu bude možné porovnávať medzi sebou rôzne scény s rozdielne veľkými objektami. Výpočet sa bude riadiť podľa vzorca 4.4:

$$d_n = \sqrt{\left(\frac{\Delta x}{w}\right)^2 + \left(\frac{\Delta y}{h}\right)^2} \tag{4.4}$$

kde d_n je normalizovaná vzdialenosť, Δx rozdiel x-ových súradníc, w šírka častice, Δy rozdiel y-ových súradníc a h výška častice. Normalizujú sa vzdialenosti v obidvoch osiach a následne sa pomocou Pytagorovej vety vypočíta normalizovaná vzdialenosť.

Vzorec dokresľuje obrázok 4.1, zelenou farbou je znázornená ideálna poloha objektu so stredom S_1, červenou odhadovaná poloha so stredom S_2.



Obr. 4.1: Odchýlka ideálnej a odhadovanej polohy

4.4 Dynamická zmena parametrov

Podkapitoly 6.2 a 6.3 dokážu, že optimálny počet a rozptyl častíc je úzko spätý s rýchlosťou sledovaného objektu. Rýchlejší pohyb vyžaduje vyššie hodnoty týchto parametrov, na optimálne sledovanie pomalšieho pohybu stačia hodnoty nižšie. Užívateľ je teda nútený správne nastaviť tieto parametre pre každé video podľa povahy scény, ktorú zobrazuje. Naviac, ak sa v inak pomalej scéne nachádza prudký pohyb, je treba nastaviť spomínané parametre na vysoké hodnoty. To je však neefektívne v ostatných, pomalých častiach videa. Kvôli týmto dôvodom bude program obsahovať nástroj na dynamickú zmenu parametrov, konkrétne počtu častíc a rozpylu šumu pre x-ovú a y-ovú súradnicu polohy. Cieľom tohto nástroja bude dynamické prispôsobovanie sa aktuálnemu charakteru scény a zvýšenie užívateľského komfortu.

Počas behu aplikácie sa bude zaznamenávať niekoľko posledných polôh sledovaného objektu, z ktorých sa následne vypočíta rýchlosť zvlášť pre x-ovú a y-ovú os. Na základe týchto hodnôt sa budú nastavovať hore uvedené parametre. K tomu bude slúžiť spojitá funkcia, ktorá zabráni prudkým skokovým zmenám počtu častíc a ich rozptylu.

Nevýhodou riešenia môže byť nestály počet častíc, ktorý má za následok premenlivú náročnosť výpočtu jednej snímky.

4.5 Zotavenie sa z prekrývania

Program bude schopný vysporiadať sa s dočasným prekrytím sledovaného objektu iným objektom. Navrhnuté riešenie popisuje algoritmus 1. K detekcii prekrývania dôjde na základe poklesu váhy najlepšie ohodnotenej častice w_{p_best} pod prahovú hodnotu w_{min} (1. riadok). Inicializačná časť algoritmu pozostáva z výpočtu priemernej zmeny polohy Δp na základe predchádzajúceho pohybu telesa a z nastavenia počtu častíc na maximálnu hodnotu n_{p_max}. Samotný cyklus bude iterovať, kým sa nezvýši ohodnotenie najlepšej častice nad prahovú hodnotu, čo bude indikovať lokalizáciu už odkrytého objektu. V každej iterácii cyklu sa posunie stred generovania častíc c_p o hodnotu Δp a zvýši sa rozptyl šumu v_p (5. a 6. riadok). Vďaka tomu bude algoritmus odolný voči drobným zmenám rýchlosti a smeru prekrytého objektu. Následne sa vygenerujú častice p pre ďalšiu snímku videa a ohodnotia sa.

Algorithm 1 Zotavenie sa z prekrývania

```
1:  if w_p_best < w_min then
2:      comp(Δp)
3:      n_p ← n_p_max
4:      while w_p < w_min do
5:          c_p ← c_p + Δp
6:          inc(v_p)
7:          gen(p)
8:          eval(p)
9:      end while
10: end if
```

Kapitola 5

Implementácia programu

Táto kapitola začína popisom všeobecných rysov programu, nástrojov a knižníc použitých pri jeho vývoji. Následne približuje implementáciu jeho jednotlivých súčastí, spomenutých v kapitole 4. Príklady ich použitia spolu s návodom na ovládanie programu a prehľadom vstupných parametrov sú zhrnuté v prílohe A.

Program bol vytvorený na platforme *Microsoft Windows*, vo vývojovom prostredí *Microsoft Visual Studio 2008*. Implementovaný bol v jazyku *C++* bez použitia objektového prístupu. Využitá bola tiež knižnica *OpenCV* [12] (*Open Source Computer Vision*), zaoberajúca sa počítačovým videním.

5.1 OpenCV

Informácie o *OpenCV* boli čerpané jednak zo stránok projektu [12], ale najmä z knižnej publikácie venujúcej sa komplexnému popisu vlastností a funkcionality knižnice, ktorou je [3].

Zo širokej plejády dostupných funkcií boli využité najmä tie, ktoré umožňujú načítanie, zápis a zobrazenie videa v okne programu. Veľmi užitočné boli štruktúry `IplImage` a `CvHistogram` na spracovanie obrázku, resp. histogramu a funkcie k nim pridružené. Za zmienku stojí funkcia `cvWaitKey` zachytávajúca stlačenie klávesy a `cvSetMouseCallback` umožňujúca nadefinovanie vlastnej *callback* funkcie, reagujúcej na udalosti myši.

5.2 Kostra programu

Za základ programu bola použitá jednoduchá, voľne dostupná implementácia Časticového filtra zo stránok [11]. Využíva nadstavbu štandardnej knižnice *OpenCV* s názvom *OpenCVX* [13]. Tá poskytuje množstvo nových funkcií, z ktorých niektoré sú využiteľné pri implementácii Časticového filtra.

Použitá kostra programu obsahovala načítanie a zobrazenie videa, funkcie na predikciu a prevzorkovanie častíc a základný algoritmus ich ohodnocovania.

Hlavnými zdrojovými súbormi implementácie sú:

- `track.cpp` – inicializácia algoritmu, načítanie a zobrazenie videa, rozširujúce nástroje
- `observetemplate.h` – pozorovací model, funkcie na ohodnocovanie častíc
- `cvparticle.h` – systémový model, predikcia a prevzorkovanie

Po preložení vznikne jediný spustiteľný súbor `track.exe`. Vstupom programu je videosúbor s príponou `.avi` prípadne video zachytené pomocou pripojenej kamery. Výstupom je súbor `output.avi`, obsahujúci video s výsledkom sledovania objektu.

K predikcii novej polohy objektu dochádza na začiatku cyklu, iterujúceho cez všetky snímky videa, volaním funkcie `cvParticleTransition(...)`. Cyklus pokračuje ohodnocovaním častíc popísaným nižšie a na jeho konci dôjde k ich prevzorkovaniu. To zabezpečuje funkcia `cvParticleResample(...)`.

5.3 Ohodnocovanie častíc

Funkcie na ohodnocovanie častíc sa volajú z cyklu prechádzajúceho cez jednotlivé snímky videa, v zdrojovom súbore `track.cpp`. Podľa parametra zadaného pri spustení programu sa zavolá jedna zo štyroch funkcií.

`particleEvalDefault(...)` implementuje základný algoritmus ohodnocovania častíc. Podobnosť referenčného obrázka a častice sa vypočíta ako suma druhých mocnín rozdielov všetkých pixelov, tak ako to popisuje vzorec 4.1. Na to sa využíva vstavaná funkcia knižnice *OpenCV* s názvom `CvNorm()`. Takto vypočítaná hodnota sa zapíše do štruktúry typu `CvParticle`, ktorá obsahuje informácie o všetkých časticiach.

Funkcia na ohodnocovanie častíc `particleEvalGrayHist(...)` pracuje na báze šedotónového histogramu. Pred jej prvým použitím dôjde k inicializácií zavolaním funkcie `initializeGrayHist(...)`, ktorá z optimalizačných dôvodov vytvorí histogram z referenčného obrázka. Ten sa predá ako parameter do funkcie `particleEvalGrayHist(...)`, ktorá ho pomocou funkcie `cvCompareHist(...)` porovná s vytvoreným histogramom častice.

Obdobne funguje ohodnocovanie častíc založené na farebnom histograme. Použité funkcie sú `particleEvalRGBHist(...)` a `initializeRGBHist(...)`. Na základe experimentov na testovacích videách bolo zvolené porovnávanie histogramov metódou *Intersection*.

Posledným spôsobom je ohodnocovanie hybridnou metódou. K inicializácii slúži funkcia `initializeHyb(...)`, následne sa pre každú snímku videa volá `particleEvalHybrid(...)`. Tá pre každú z MxN častí častice vypočíta histogram. Z dôvodu optimalizácie sú komplexné štruktúry `CvHistogram` nahradené `CvMat` s rozmermi 1 x počet prvkov histogramu. Hodnota pixelu sa vydelí konštantou určenou vzorcom $k = 256/počet\ prvkov$, výsledok je index položky matice `CvMat`, ktorá sa má inkrementovať. Počet prvkov bol empiricky zvolený na hodnotu 64. Neskoršie testy ukázali, že zmena tejto hodnoty nemá zásadný vplyv na presnosť algoritmu.

5.4 Ohodnocovanie presnosti prgramu

Prvú fázu ohodnocovania zabezpečuje funkcia `testLearn(...)`. Vyznačovanie polohy objektu je uskutočnené použitím funkcie `icvGetRegion(...)`. Po skončení videa dôjde k ukončeniu programu. Priebeh fázy zobrazuje obrázok 5.1.

V druhej fáze sa zavolá funkcia `testExecute(...)`. Tá načíta zo súboru polohy objektu z predchádzajúcej fázy a uloží ich do poľa bodov typu `CvPoint`. Po samotnom sledovaní objektu sa predá riadenie programu funkcii `evaluateSuccess(...)`, ktorá porovná vzdialenosť korešpondujúcich bodov ideálnej (v obrázku 5.2 znázornené zelenou farbou) a vysledovanej (zobrazené červenou farbou) trajektórie absolútne, alebo podľa vzorca 4.4.

Obr. 5.1: Prvá fáza nástroja na ohodnocovanie presnosti

5.5 Dynamická zmena parametrov

O dynamickú zmenu parametrov na základe rýchlosti pohybu sledovaného telesa sa stará funkcia `dynamicParamMotion(...)`. Hodnota šumu sa získa pripočítaním empiricky zistenej konštanty 5 k aktuálnej rýchlosti objektu a to zvlášť pre x-ovú a y-ovú os. Počet častíc sa vypočíta nasledovne:

$$minCastic + \frac{maxCastic - minCastic}{maxRychlost} * \frac{dx + dy}{2}$$

Kde $minCastic$ a $maxCastic$ je minimálny, resp. maximálny počet častíc, $maxRychlost$ je hraničná rýchlosť. Hodnoty boli zvolené empiricky, v poradí 100, 600 a 20. Jedná sa vlastne o namapovanie rýchlosti objektu v intervale $< 0, 20 >$ na interval počtu častíc $< 100, 600 >$. Pre hodnoty rýchlosti vyššie ako 20 sa nastaví maximálny počet častíc.

Obr. 5.2: Druhá fáza nástroja na ohodnocovanie presnosti

5.6 Zotavenie sa z prekrývania

Detekciu a vysporiadanie sa s prekrývaním má na starosti funkcia `dynamicParamEval(...)`. Kľúčovou úlohou je správne nastaviť prahovú hodnotu detekujúcu prekrývanie objektov. Experimentovaním s testovacími videami bolo zistené, že táto hodnota je pre každé video odlišná. Preto ju nie je možné nastaviť konštantne a musí sa zadať ako parameter pri spustení programu. Pokiaľ ohodnotenie najlepšej častice klesne pod prahovú hodnotu, z poľa predchádzajúcich polôh objektu typu `CvPoint` sa vypočíta optimálna trajektória. Stred generovaných častíc pre n-tú snímku od začiatku prekrývania sa vypočíta podľa 5.1:

$$p_n = p_0 + n * \Delta p \tag{5.1}$$

kde p_n je nová poloha stredu častíc, p_0 je poloha poslednej častice s nadprahovým ohodnotením a Δp je priemerná zmena polohy objektu z predchádzajúcich snímok.

Kapitola 6

Testovanie a experimenty

Vďaka nástroju na ohodnotenie presnosti sledovania objektu z časti 4.3 je možné objektívne ohodnotiť fungovanie programu pri jeho rôznych nastaveniach a použitých algoritmoch. Táto kapitola porovnáva implementované metódy výberu najlepšej častice a popisuje experimenty s počtom potrebných častíc a nastavením šumu. Taktiež sa zaoberá testovaním nástroja na dynamickú zmenu parametrov a zotavenia sa z prekrývania objektov. Všetky tabuľky, grafy a obrázky použité v tejto kapitole boli pre jej účely vytvorené, žiadne z nich neboli prevzaté z iných dokumentov.

Experimenty prvých štyroch podkapitol boli realizované pomocou dávkového súboru, ktorý postupne spúšťal program v *testovacom* móde s rozličnými nastaveniami parametrov. Program bol dočasne upravený tak, aby výsledky testovania zapisoval do textového súboru, čo uľahčilo ich spracovanie. Každý experiment prebehol päťkrát a výsledné hodnoty sú priemerom týchto piatich pokusov.

Ku každej podkapitole bolo vytvorené ukážkové video dokresľujúce vykonané experimenty a ich výsledky. Súbory s videosekvenciami sú uložené v adresári `\vid\experimenty` na priloženom DVD. Adresár `\vid` obsahuje video `demo.wmv`, ktoré demonštruje základné vlastnosti implementovaného riešenia a je vhodné na rýchle zoznámenie sa s jeho funkcionalitou a typmi vykonaných experimentov.

6.1 Porovnanie prístupov k ohodnocovaniu častíc

Táto podkapitola obsahuje sériu experimentov zameraných na čo najkomplexnejšie porovnanie fungovania jednotlivých prístupov k ohodnocovaniu častíc. Taktiež sú diskutované vlastnosti implementovaného Časticového filtra, spomenuté v kapitole 2 a 3. Na konci podkapitoly sú zhrnuté charakteristické črty jednotlivých spôsobov ohodnocovania častíc. Výsledky experimentov sú zobrazené vo forme tabuliek s normalizovanou odchýlkou od ideálnej trajektórie. Hodnota N/A znamená, že došlo k strate objektu a sledovanie skončilo neúspechom. Použité testovacie videá sa nachádzajú v adresári `\experimenty\zdrojove` na priloženom DVD. Pokiaľ nie je uvedené inak, experimenty boli uskutočnené s použitím 300 častíc a optimálnym rozptylom pre danú scénu. Veľkosť referenčného obrázka a častíc bola zmenená na 24x24 pixelov.

V testovacích videách *Vid01* a *Vid02* sa vyskytujú objekty nemeniace svoj tvar. V prvom z nich dochádza k rýchlemu pohybu a prudkým zmenám smeru sledovaného objektu, ktorým je časť ľudskej tváre. Základný algoritmus pracujúci so vzorkom objektu dosahuje veľmi dobré výsledky s priemernou normalizovanou odchýlkou 0,14 pri použití 300 častíc.

Histogramové prístupy sú na tom s presnosťou o poznanie horšie. Najlepší výkon podal hybridný prístup, nasledovaný farebným histogramom. Ohodnocovanie častíc pomocou šedotónového histogramu skončilo neúspechom. Nepriaznivým vplyvom na histogramy je fakt, že bola vybraná iba časť tváre, čo spôsobuje, že okolie sledovaného objektu sa farebne podobá na objekt samotný. V druhom videu dochádza k sledovaniu objektu s jednou prevládajúcou farbou na bielom pozadí. Poradie presnosti jednotlivých prístupov je totožné s predchádzajúcim experimentom.

Videá *Vid03* a *Vid04* sa zameriavajú na otestovanie prípadov, v ktorých dochádza k zmene tvaru alebo natočeniu sledovaného objektu. V prvom videu je sledovaná tenisová hráčka, ktorá počas hry neustále mení pózu tela, v druhom je sledovaná natáčajúca sa tvár. Podľa očakávaní z kapitoly 2, základný algoritmus pracujúci so vzorkou objektu v takýchto podmienkach nedosahuje dobré výsledky. Ani v jednom prípade sa nepodarilo objekt sledovať až do konca testovacieho videa. Ostatné algoritmy dosiahli uspokojujúcejšie výsledky. Najvyššiu presnosť dosiahol farebný histogram, tesne nasledovaný hybridným prístupom. Úspešnosť algoritmov založených na histograme sa dá pripísať faktu, že napriek zmene tvaru a natočenia objektu nedochádza k veľkej zmene percentuálneho zastúpenia farieb. Výsledky prvých štyroch pokusov sú prehľadne zhrnuté v tabuľke 6.1.

	def	gray	rgb	hyb 8 8
Vid01	0,14	N/A	1,88	0,45
Vid02	0,09	N/A	0,54	0,29
Vid03	N/A	0,65	0,37	0,40
Vid04	N/A	0,29	0,23	0,23

Tabuľka 6.1: Odchýlka prístupov k ohodnocovaniu častíc

Predchádzajúce experimenty ukázali univerzálnosť hybridného prístupu ktorý s akceptovateľnou presnosťou dokázal sledovať objekt v obidvoch typoch videí, s tuhými objektami, aj s tými, ktoré menia svoj tvar a natočenie. Tabuľka 6.2 ukazuje presnosť tohto prístupu vzhľadom na počet častí, na ktoré je objekt rozdelený.

	hyb 2 2	hyb 4 4	hyb 8 8	hyb 12 12	hyb 24 24
Vid01	0,93	0,55	0,45	0,52	1,10
Vid02	0,35	0,31	0,29	0,24	0,26
Vid03	0,49	0,41	0,40	0,44	N/A
Vid04	0,23	0,21	0,23	0,25	N/A

Tabuľka 6.2: Odchýlka jednotlivých nastavení hybridného prístupu

Z tabuľky možno odvodiť nasledovné závery. V prvých dvoch videách dosahoval hybridný algoritmus lepšie výsledky pri rozdelení objektu na väčší počet častí. Tu možno nájsť istú analógiu s výsledkami predchádzajúci experimentov, ktoré ukázali, že na videá *Vid01* a *Vid02* je najvhodnejší základný prístup ohodnocovania častíc. Hybridný prístup s parametrami 24 24, v tomto prípade rozdeľujúci referenčný snímok na časti o veľkosti $1px$ x $1px$ dosahoval vo väčšine prípadov slabé výsledky. Predchádzajúce experimenty ukázali, že u videí *Vid03* a *Vid04* sa viac ako základný prístup vyplatí použitie histogramov. To demonštruje aj o niečo vyššia úspešnosť hybridného algoritmu pracujúceho s menším

počtom častí, čo sa svojou podstatou blíži k „čisto“ histogramovému prístupu. Výnimkou bol hybridný prístup s parametrami 2 2, ktorý dosahoval horšie výsledky. Vykonané boli aj experimenty s rozdielnym počtom horizontálnych a vertikálnych častí. Z výsledkov vyplýva, že pokiaľ nie je počet častí v jednom smere výrazne väčší ako v druhom, nemá toto nastavenie výrazný vplyv na kvalitu sledovania. Pri extrémnom nastavení typu „hyb 24 1“ dochádzalo k drobným vertikálnym nepresnostiam, naopak pri nastavení „hyb 1 24“ sa nepresnosti prejavili v horizontálnom smere.

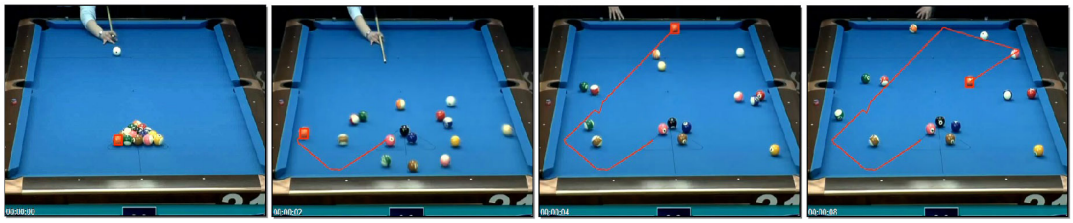
V kapitole 3 bolo uvedené, že Časticový filter nevyžaduje použitie statickej kamery. Túto vlastnosť potvrdzujú experimenty s videami *Vid05* a *Vid06*, zobrazujúcimi športujúce osoby pohyblivou kamerou. Naviac, v druhom spomínanom videu sa podarilo základným algoritmom úspešne sledovať hokejového hráča. To dokazuje, že v niektorých prípadoch môže byť tento prístup úspešný aj pri objektoch, ktoré v menšej miere menia svoj tvar.

Video *Vid07* zachytáva scénu v miestnosti so slabším osvetlením. Okrem zvýšeného šumu v obraze dochádza aj k postupnej zmene osvetlenia objektu z dôvodu jeho natáčania voči zdroju svetla a tiež dvom skokovým zmenám spôsobeným rozsvietením a zhasnutím stropnej lampy. Takto náročná scéna ma fatálny vplyv na úspešnosť sledovania za použitia algoritmov založených na histograme. Výsledok je možno vidieť na obrázku 6.1. Sledovaný objekt sa stráca v momente, keď dôjde k jeho natočeniu smerom od zdroja svetla. Subjektívne nie je zmena farieb veľmi výrazná, algoritmy si s ňou však nevedia poradiť ani pri použití referenčnej vzorky s vyšším rozlíšením alebo extrémne vysokého počtu častíc. Základný algoritmus dokázal sledovať objekt o niečo dlhšie, ale nakoniec tiež došlo k jeho strateniu.

Obr. 6.1: Neúspech vo videu *Vid07* pri použití prístupu založeného na histograme

V prípade správneho nastavenia parametrov je Časticový filter schopný sledovať objekt veľmi prudko meniaci rýchlosť a smer pohybu. Video *Vid08* zachytáva úvodný rozstrel v biliardovej partii. Sledovaným objektom je jedna z gúľ, ktorá z pokoja náhle zrýchli a odráža sa od mantinelov hracej plochy. Pre úspešné sledovanie bolo nutné nastaviť rozptyl pre pozíciu častíc na hodnotu 30 (pre x-ovú aj y-ovú os) a počet častíc na hodnotu 800 a viac. Použitý bol základný spôsob ohodnocovania častíc. Výsledok experimentu zachytáva obrázok 6.2. Drobná nepresnosť trajektórie v ľavej časti posledných dvoch obrázkov je spôsobená dočasným prekrytím objektu inou guľou.

Podkapitola sa zaoberala najmä vplyvom jednotlivých prístupov k ohodnocovaniu častíc na presnosť programu. Výsledky experimentov potvrdili domienky z kapitoly 2 a 3. Základný algoritmus podával veľmi dobré výsledky v prípade sledovania objektov nemeniacich svoj tvar alebo natočenie. Na ostatné typy scén sa viac hodili histogramové prístupy. Farebný histogram podával obecne lepšie výsledky ako šedotónový, obidva však v niektorých prípadoch vykazovali vysoké odchýlky oproti optimálnej trajektórii. Optimálnym riešením sa ukázal byť hybridný prístup, ktorý síce nedosahoval presnosti základného, zato dokázal úspešne sledovať objekt u viacerých typov scén. Ďalšie experimenty ukázali, že implemen-



Obr. 6.2: Sledovanie objektu s prudkými zmenami rýchlosti a smeru vo videu *Vid08*

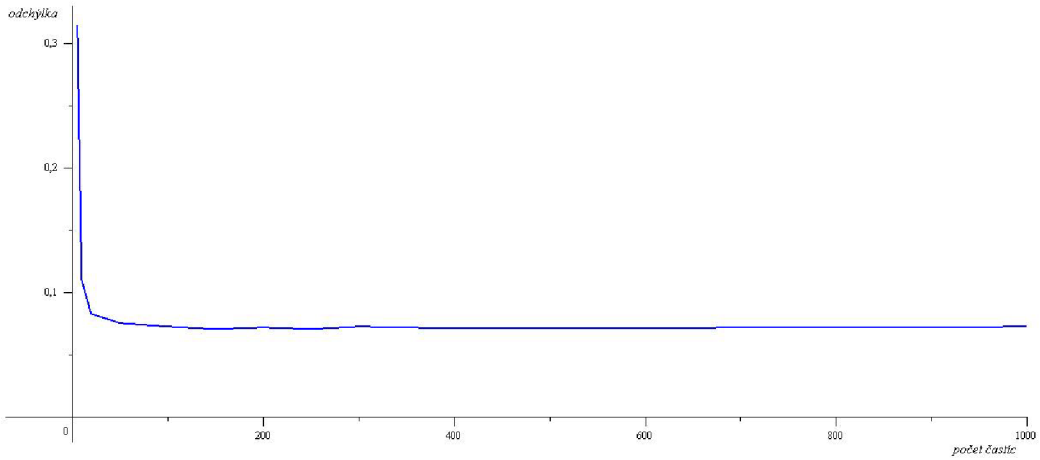
tovaný Časticový filter nie je náchylný na pohyb kamery, či zmeny rýchlosti a smeru objektu. Problémy spôsobuje vysoký šum vo videu a zmena osvetlenia.

6.2 Experimentovanie s počtom častíc

Podkapitola sa v prvej časti venuje vplyvu počtu častíc na presnosť sledovania v pomalej a rýchlej scéne. Výsledky sú zobrazené pomocou grafov a diskutované. Druhá časť približuje, akým spôsobom vplýva počet častíc na náročnosť výpočtu a dobu behu programu.

Na experimenty boli použité testovacie videá *Vid09* a *Vid02*. Obidve zachytávajú veľmi jednoduchú a podobnú scénu, s tým rozdielom, že v druhom prípade je pohyb sledovaného objektu rýchlejší. Výber tohto typu videí je zámerný. Ich jednoduchosť je dôležitá preto, aby sa minimalizovali rušivé vplyvy, ktoré by mohli spôsobovať prudké výkyvy presnosti medzi jednotlivými meraniami. Podobnosť umožňuje priame porovnanie vplyvu počtu častíc pri rozdielne rýchlych scénach.

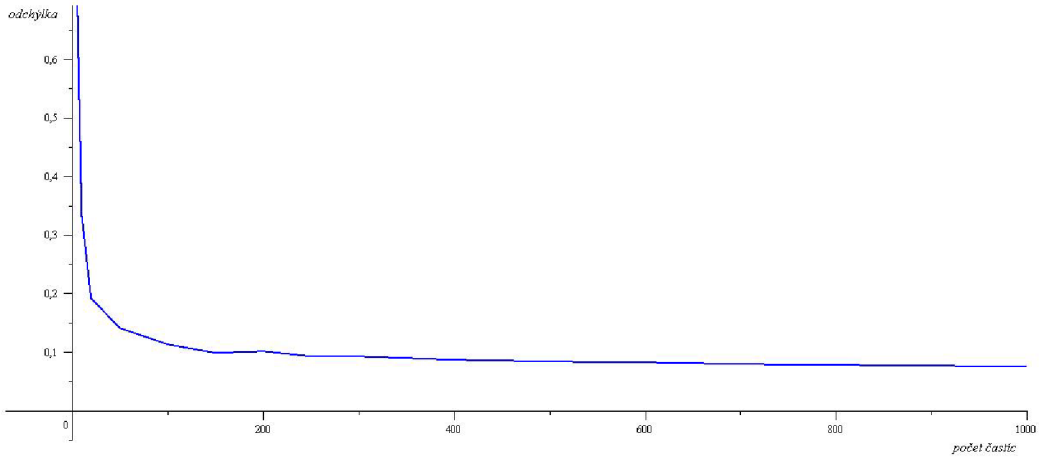
Grafy 6.3 a 6.4 ukazujú vplyv počtu častíc na sledovanie objektu v testovacích videách *Vid09* a *Vid02*. Na x-ovú os je vynesený počet častíc, na y-ovú os odchýlka normalizovaná veľkosťou referenčného obrázku, tak ako to bolo vysvetlené v časti 4.3. Nižšia hodnota teda značí lepší výsledok. Na ohodnocovanie častíc bol použitý základný prístup, stredná hodnota náhodného šumu pre obidve osi bola zvolená tak, aby bola optimálna pre každú scénu.



Obr. 6.3: Graf závislosti odchýlky na počte častíc pre *Vid09*

Nízka rýchlosť sledovaného objektu vo videu *Vid09* umožnila nastaviť rozptyl šumu pre x-ovú a y-ovú os na pomerne nízku hodnotu 3. To sa prejavilo na ohodnotení úspešnosti programu. Zatiaľ čo pri extrémne nízkom počte častíc má funkcia na grafe 6.3 silne klesajúcu tendenciu, už od počtu približne 100 častíc sa s drobnými výchylkami udržiava na konštantnej hodnote. Nízky rozptyl šumu spôsobuje, že pri veľkých počtoch častíc dochádza k ich generovaniu na identické súradnice. Mnohé z nich sú potom nadbytočné a ďalšie zvyšovanie ich počtu nemá za následok zlepšovanie presnosti algoritmu. Kvôli lepšiemu porovnaniu so zostávajúcimi dvomi experimentami bude užitočné označiť si nejakú hodnotu presnosti za uspokojujúcu. Pri jej dosiahnutí sa dá považovať použitý počet častíc za optimálny. Zvolená hodnota bola 0, 1, čo pri tomto extrémne pomalom pohybe znamená optimálny počet častíc na úrovni 20.

Výsledky experimentovania s videom *Vid02* zobrazuje graf 6.4. Z dôvodu rýchlejšieho pohybu bolo nutné nastaviť rozptyl šumu na vyššiu hodnotu ako v predchádzajúcom prípade, konkrétne na 10. To bol hlavný dôvod odlišností oboch experimentov. Z grafu 6.4



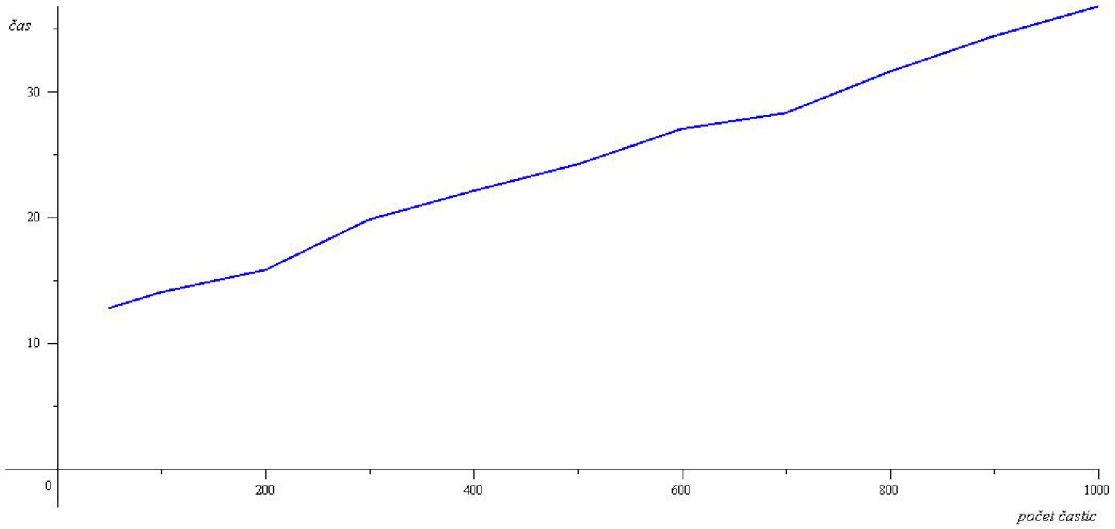
Obr. 6.4: Graf závislosti odchýlky na počte častíc pre *Vid02*

vyplýva, že algoritmus dosiahol uspokojujúcich výsledkov pri počte častíc 250, čo je vyššia hodnota ako v predchádzajúcom prípade. Porovnaním obidvoch grafov sa dá zistiť, že na dosiahnutie ekvivalentných hodnôt presnosti treba v druhom experimente až niekoľkonásobný počet častíc. Zvýšený rozptyl šumu má za následok generovanie častíc do väčšieho priestoru. Kvôli tomu je ich treba väčší počet, ktorý zaručí ich väčšiu koncentráciu v okolí sledovaného objektu, a teda aj presnejšie výsledky. Z grafu ďalej vidno, že v rámci testovaného počtu častíc sa funkcia neustáli na jednej hodnote a má stále klesajúcu tendenciu. To znamená, že zvyšovaním počtu častíc je možno dosiahnuť ešte väčšej presnosti. Ďalšími experimentami bolo zistené, že funkcia sa ustáli od približne 5000 častíc.

So zväčšujúcim sa počtom častíc narastá náročnosť výpočtu. Pre každú snímku videa je potrebné porovnať referenčný obrázok sledovaného objektu s každou časticou. Od ich určitého počtu prestáva program pracovať v reálnom čase. Graf 6.5 popisuje vplyv počtu častíc na dobu behu programu. Namerané hodnoty sú pre úplnosť zobrazené v tabuľke 6.3. Experiment bol uskutočnený na počítači s dvojjadrovým procesorom *Intel Core Duo* s frekvenciou 2 x 1,86GHz, 2 GB pamäte RAM a nainštalovaným operačným systémom *Windows 7* v 32-bitovej verzii. Použité bolo video *Vid02* s dĺžkou 18 sekúnd, častice boli ohodnocované základným prístupom. Z grafu 6.5 vidno, že závislosť dĺžky behu programu

počet častíc	50	100	200	300	400	500	600	700	800	900	1000
doba behu [s]	12,8	14,0	15,8	19,8	22,1	24,2	27,0	28,3	31,6	34,4	36,8

Tabuľka 6.3: Tabuľka závislosti doby behu programu na počte častíc



Obr. 6.5: Graf závislosti doby behu programu na počte častíc

na počte častíc je s menšími odchýlkami lineárna. Ďalej sa z neho dá odvodiť, že program je schopný fungovať na danom počítači v reálnom čase a to za použitia približne 200 a menej častíc. V ostatných prípadoch už doba behu programu presiahne 18 sekúnd – dĺžku testovacieho videa. Z grafu 6.4, opisujúceho rovnakú scénu vyplýva, že použitiu 200 častíc odpovedá ohodnotenie na úrovni 0,1, ktoré sa ešte dá považovať za uspokojujúce. Výsledky týchto pokusov dokazujú, že program je schopný podať kvalitné výsledky v reálnom čase aj na málo výkonnom počítači.

Ďalšie testovanie prebiehalo na počítači s procesorom *Intel Core 2 Duo* s frekvenciou 2 x 3,0GHz, 2GB pamäte RAM a operačným systémom *Windows XP*. Pri tejto konfigurácii dokázal program pracovať v reálnom čase za použitia maximálne 800 častíc.

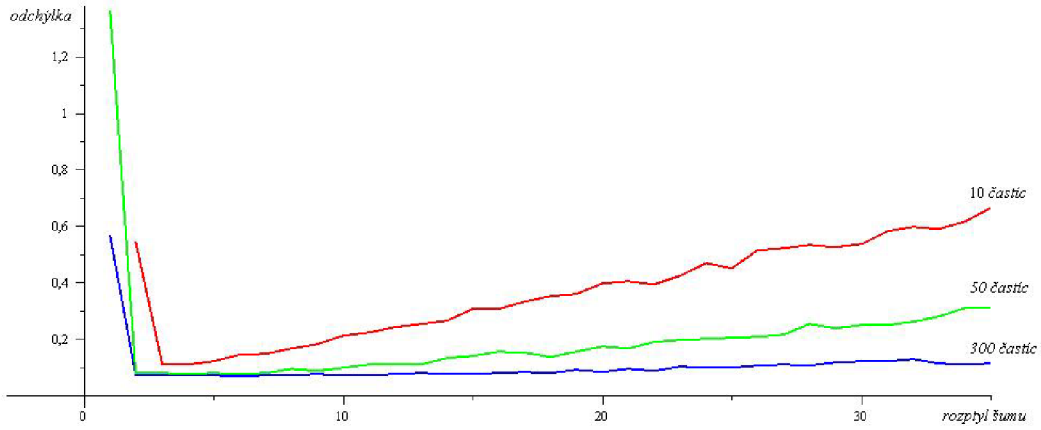
Experimenty dokázali, že so zväčšujúcim sa počtom častíc sa zlepšuje presnosť programu. Výnimkou je stav, kedy sa častice vo veľkom množstve generujú na rovnakých súradniciach a zvyšovaním ich počtu sa už presnosť programu nezlepšuje. Toto môže nastať v dôsledku nízkeho rozptylu šumu. Veľký počet častíc sa negatívne odzrkadľuje na výpočtovej náročnosti, ktorá rezultuje v predĺženie doby behu programu.

6.3 Experimentovanie s nastavením šumu pre polohu častíc

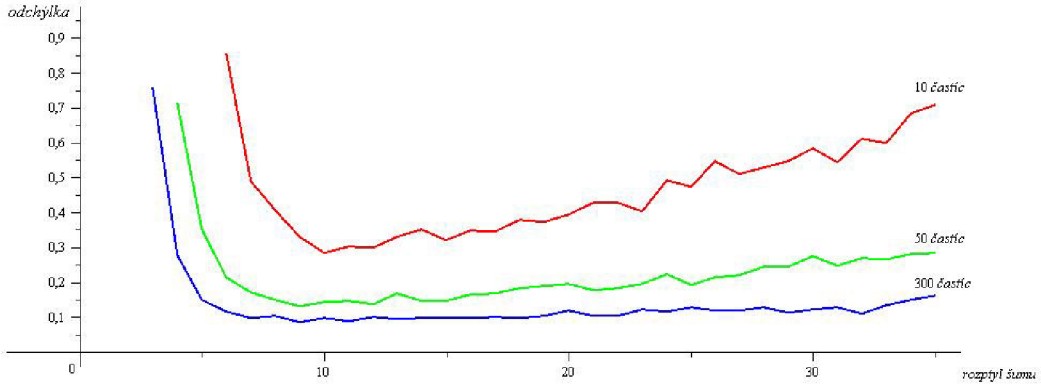
Táto podkapitola sa zaoberá vplyvom rozptylu šumu pre polohy častíc na presnosť sledovania objektu. Testovanie bude prebiehať na tých istých videách ako v predchádzajúcej podkapitole, teda *Vid09* a *Vid02*. Dôvod ich použitia je rovnaký.

Grafy 6.6 a 6.7 opisujú odchýlku od ideálnej trajektórie v závislosti na veľkosti šumu pre pomalú, resp. rýchlu scénu. V každom grafe sú tri krivky, reprezentujúce merania pre 10,

50 a 300 častíc. Zámerom nie je znovu dokazovať vplyv počtu častíc na presnosť programu, ale vplyv na tvar krivky popisujúcej odchýlku programu ako funkciu rozptylu šumu.



Obr. 6.6: Graf závislosti odchýlky od ideálnej trajektórie na nastavení šumu pre *Vid09*



Obr. 6.7: Graf závislosti odchýlky od ideálnej trajektórie na nastavení šumu pre *Vid02*

V obidvoch grafoch majú závislosti podobný charakter. Pri veľmi nízkych hodnotách šumu je odchýlka od ideálnej trajektórie vysoká, čiže presnosť algoritmu nízka. Malý rozptyl častíc nie je schopný zachytiť rýchlejší pohyb, kedy sa sledovaný objekt za jednu snímku posunie o väčšiu vzdialenosť ako je samotný rozptyl. Pri vysokých hodnotách sa častice rozptýlia do širšieho okolia okolo objektu, čo zabraňuje presnejšie určiť jeho polohu.

Z grafov sa dá tiež vyvodiť, že rozsah nastavenia šumu pri ktorom bola dosiahnutá minimálna odchýlka, je pre každú scénu jedinečný a v rámci jednej scény nie je ovplyvnený počtom použitých častíc. Pre video *Vid09* je to okolie hodnoty 3, pre *Vid02* okolie hodnoty 10.

Počet častíc totiž vplýva na tvar krivky, nie na hodnotu, v ktorej dosahuje minimum. Z grafov si je možné všimnúť, že pri väčšom počte častíc, so zväčšujúcou sa hodnotou šumu, nestúpa krivka odchýlky tak prudko. Napríklad v grafe 6.7 došlo v intervale $< 20, 35 >$ k nárastu odchýlky o $0, 04$ pre 300 častíc, pre 10 častíc bol nárast až $0, 28$. V praxi to znamená, že vyšší počet častíc znižuje rozdiely v ohodnotení pre rôzne hodnoty šumu a tým

aj dôležitosť jeho správneho nastavenia. Naopak, pri nízkom počte častíc je správny výber hodnoty šumu kritický.

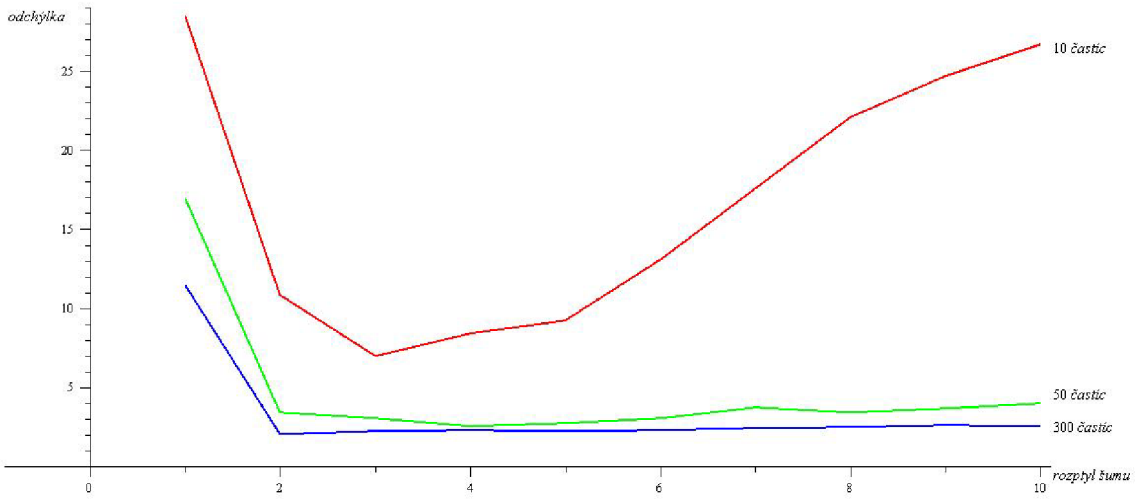
6.4 Experimentovanie s nastavením šumu pre rozmery častíc

Počas sledovania sa môže meniť vzdialenosť objektu od kamery, prípadne vplyvom natáčania zvlášť jeho šírka alebo výška. Dôsledkom toho dochádza k zmene veľkosti plochy, ktorú objekt vo videu zaberá. Na prispôsobenie sa tomuto javu slúži nastavenie šumu pre rozmery častíc, ktorým sa táto podkapitola zaoberá.

Prvý experiment bol uskutočnený na videu *Vid10*, zobrazujúcom vzďaľujúci sa obdĺžnikový objekt. Použitý bol základný prístup ohodnocovania častíc, v prvom prípade bol šum pre výšku a šírku častíc nastavený na hodnotu 2, v druhom prípade na 0. Algoritmus bol dočasne upravený tak, aby aj pri nulovom šume vypočítal absolútnu odchýlku, čo umožnilo porovnanie obidvoch prístupov. Výsledky zobrazuje tabuľka 6.4. Vyplýva z nej, že pri použitom testovacom videu dosahuje algoritmus vyššej presnosti s nastavením šumu pre rozmer častíc na nenulové hodnoty.

	10 častíc	50 častíc	300 častíc
hodnota šumu: 0	35,6	35,5	35,3
hodnota šumu: 2	6,9	3,0	2,2

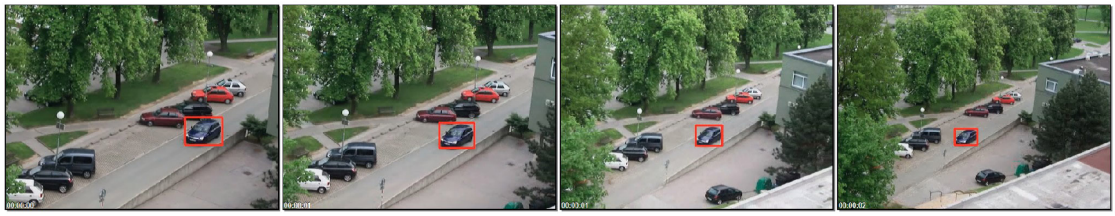
Tabuľka 6.4: Tabuľka absolútnych odchýlok v pixeloch pre video *Vid10*



Obr. 6.8: Graf závislosti odchýlky na nastavení šumu pre veľkosť častíc vo videu *Vid10*

Ďalšie experimenty s videom *Vid10* sa zaoberajú vplyvom veľkosti nastaveného šumu na odchýlku. Výsledky zobrazuje graf 6.8. Vyplýva z neho, že so šumom neúmerne sa zvyšujúcim voči povahe scény, sa odchýlka zväčšuje a to tým prudšie, čím menší počet častíc je použitý. Naopak, nízka hodnota šumu nedokáže zachytiť zmenu veľkosti objektu.

Obrázok 6.9, zobrazujúci video *Vid11* ukazuje, že algoritmus sa dokáže vysporiadať aj s náhlym oddialením objektu, spôsobeným zmenou ohniskovej vzdialenosti objektívu kamery.



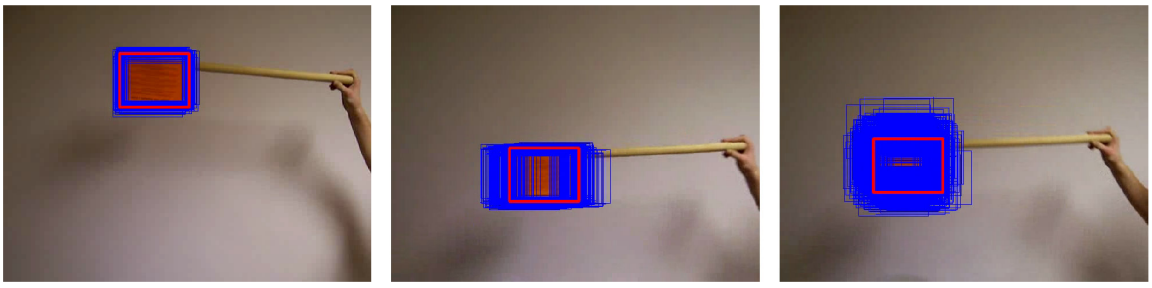
Obr. 6.9: Zmena rozmerov častíc vo videu *Vid39*

6.5 Testovanie nástroja na dynamickú zmenu parametrov

Podkapitoly 6.2 a 6.3 potvrdili dôležitosť správneho nastavenia parametrov pre presnosť a efektívnosť sledovania objektu. Táto podkapitola sa zameriava na testovanie nástroja na dynamickú zmenu počtu a rozptylu častíc s ohľadom na aktuálny charakter scény.

Testovacím videom je *Vid02*, ktoré sa svojím charakterom hodí pre testovanie dynamických parametrov. Zachytáva objekt mnohokrát meniaci svoju rýchlosť a smer pohybu, čomu testovaný nástroj okamžite prispôsobuje parametre sledovania. V okamihoch spomalenia a zastavenia objektu klesá počet a rozptyl častíc na minimálnu hodnotu, čo znižuje výpočtovú náročnosť. Naopak, v rýchlych pasážach sa spomínané parametre nastavia na vysoké hodnoty, čo zabezpečuje vysokú presnosť. Spomínané javy ukazuje obrázok 6.10. Sledovanie bolo ukončené za 15,7 sekundy a dosiahnutá priemerná odchýlka bola na úrovni 0,08. Pri použití statických parametrov nastavených tak, aby bola odchýlka totožná s predchádzajúcim výsledkom, sa doba behu programu zväčšila na 19,5 sekundy, čo znamená skoro 1,25-násobné zrýchlenie v prospech testovaného prístupu.

Naviac, použitie tohto nástroja nevyžaduje nastavovanie parametrov sledovania, čo zvyšuje užívateľskú prívetivosť programu.

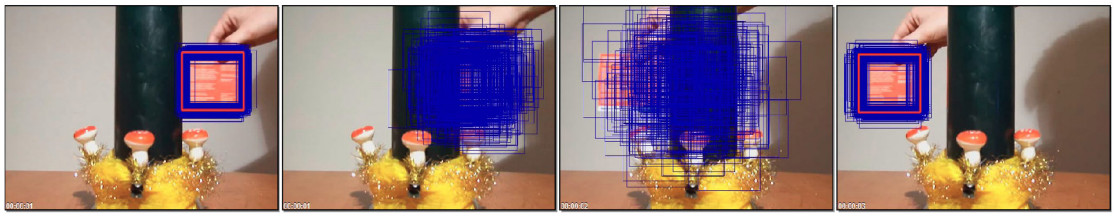


Obr. 6.10: Použitie dynamických parametrov vo videu *Vid02* (a) objekt v pokoji, (b) objekt pohybujúci sa horizontálnym smerom, (c) objekt pohybujúci sa horizontálnym aj vertikálnym smerom

6.6 Testovanie nástroja na zotavenie sa z prekrývania objektov

V mnohých scénach dochádza k prekrývaniu sledovaného objektu iným, čo môže viesť až k jeho strate a neúspešnosti sledovania. Táto podkapitola sa zaoberá testovaním navrhnutého nástroja na detekciu a zotavenie sa z prekrývania.

Prvý experiment bol uskutočnený na videu *Vid12*, ktorého priebeh zobrazuje obrázok 6.11. Jedná sa o jednoduchú scénu, v ktorej sa červený štvorcový objekt pohybuje smerom doľava, postupne celý zachádza za tmavozelený objekt a vynára sa na jeho druhej strane. Akonáhle program zdetekuje prekrytie, kandidátne častice (zobrazené modrými štvoruholníkmi) sa začnú generovať po predpokladanej trajektórii a zvyšuje sa ich rozptyl, čo je v tomto konkrétnom prípade dôležité. Objekt totiž nemá konštantnú rýchlosť, jeho pohyb sa zrýchľuje a to aj počas prekrytia. Z toho rezultuje jeho skoršie odkrytie, s čím sa však algoritmus dokáže vysporiadať a to vďaka zväčšujúcemu sa rozptylu častíc. Dôležitým faktorom vplývajúcim na detekciu prekrývania je správne nastavenie prahovej hodnoty. V tomto prípade ju bolo potrebné nastaviť na hodnotu 2400 a to za použitia základného prístupu k ohodnocovaniu častíc. Priveľmi vysoká hodnota spôsobí, že prekrývanie nie je zdetekované vôbec, v prípade priveľmi nízkej hodnoty je zase detekované skôr ako k nemu dôjde. Optimálna prahová hodnota stúpa s náročnosťou scény a veľkosťou, na ktorú bude zmenená referenčná snímka objektu.



Obr. 6.11: Fungovanie nástroja na detekciu a vysporiadanie sa s prekrývaním objketu

Ďalšie testovanie prebiehalo na videu *Vid13*. Náročnosť scény okrem prekrývania spôsobuje fakt, že zachytáva dve osoby pri chôdzi, čo z pohľadu programu znamená sledovanie objektu s premenlivým tvarom. Pri použití hybridného prístupu, rozdeľujúceho objekt na 8 x 8 častí a nastavení prahovej hodnoty na 550, bolo sledovanie úspešné aj v tomto prípade. Výsledok testu ukazuje obrázok 6.12.

Obr. 6.12: Prekrývanie vo videu *Vid13*

Kapitola 7

Záver

V práci boli popísané základy problematiky sledovania objektu. Opierajúc sa o množstvo vedeckých článkov práca prehľadne kategorizovala a bližšie charakterizovala vybrané algoritmy. Z teoretických poznatkov vychádzal návrh a implementácia algoritmov pracujúcich na matematickom základe Časticového filtra.

Hlavný prínos práce je v experimentovaní s týmito algortimami a to na širokom spektre videí, zameraných na dôkladné otestovanie ich vlastností. Pre maximálnu objektívnosť testovania bol vytvorený nástroj pre ohodnocovanie presnosti. Základný algoritmus, pracujúci na báze porovnávania intenzít pixelov, dosahoval vysokú presnosť pri sledovaní objektov nemeniacich svoj tvar. V ostatných videách sa presadili prístupy pracujúce s využitím histogramu. Ako dostatočne robustným riešením sa ukázal byť hybridný prístup, kombinujúci výhody predchádzajúcich. Ďalšie experimenty ukázali vplyv počtu a rozptylu častíc na odchýlku od ideálnej trajektórie. Diskutovaná bola tiež schopnosť algoritmov vyspriadať sa so zmenou vzdialenosti sledovaného objektu od kamery.

Úspechom skončilo testovanie nástroja na dynamickú zmenu parametrov, ktorému sa podarilo dosiahnuť až $1,25$-násobné zrýchlenie oproti konvenčnej metóde a to pri zachovaní presnosti. Uspokojivé výsledky podával nástroj na zotavenie sa z prekrývania objektov.

Z pohľadu ďalšieho vývoja práce by bolo dobré zamerať sa na optimalizáciu rýchlosti programu. Jednou z možností je využitie metódy *Integral histogram*, ktorá optimalizuje prácu s histogramami. Ďalším smerom, ktorým by sa mohla práca rozvíjať, je zlepšovanie detekcie prekrývania objektu. K aktuálnemu riešeniu by mohol byť doplnený súbor pravidiel, ktorý by pomohol pri rozhodovaní či k prekrývaniu došlo alebo nie.

Z osobného hľadiska mala pre mňa práca veľký prínos. Osvojil som si základy popisovanej problematiky a hlbšie som prenikol do fungovania jedného typu algoritmov. Prácu som nebral ako povinnosť, ale so záujmom som jej venoval množstvo voľného času. Moja snaha bola ocenená 1. miestom na študentskej konferencií *Student EEICT 2010* v kategórii Grafika, multimédiá a inteligentné systémy, čo ma motivovalo k ďalšiemu vzdelávaniu sa v tejto problematike.

Literatúra

[1] Arulampalam, S.; Maskell, S.; Gordon, N.; aj.: A Tutorial on Particle Filters for On-line Non-linear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing*, ročník 50, č. 2, 2002: s. 174–188, ISSN 1053-587X.

[2] Bernardin, K.; Stiefelhagen, R.: Evaluating multiple object tracking performance: the CLEAR MOT metrics. *J. Image Video Process.*, 2008: s. 1–10, ISSN 1687-5176.

[3] Bradski, G.; Kaehler, A.: *Learning OpenCV*. O'Reilly Media, 2008, ISBN 978-0-596-51613-0.

[4] Dharamadhat, T.; Thanasoontornlerk, K.; Kanongchaiyos, P.: Tracking object in video pictures based on background subtraction and image matching. In *ROBIO '09: Proceedings of the 2008 IEEE International Conference on Robotics and Biomimetics*, Washington, DC, USA: IEEE Computer Society, 2009, ISBN 978-1-4244-2678-2, s. 1255–1260.

[5] Isard, M.; Blake, A.: CONDENSATION – Conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision*, ročník 29, č. 1, 1998: s. 5–28, ISSN 0920-5691.

[6] van der Merwe, R.; de Freitas, N.; Doucet, A.; aj.: The Unscented Particle Filter. In *Advances in Neural Information Processing Systems 13*, 2001.

[7] Mikolajczyk, K.; Schmid, C.: An affine invariant interest point detector. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part I*, London, UK: Springer-Verlag, 2002, s. 128–142.

[8] Moravec, H. P.: Visual mapping by a robot rover. In *IJCAI'79: Proceedings of the 6th international joint conference on Artificial intelligence*, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1979, ISBN 0-934613-47-8, s. 598–600.

[9] Trucco, E.; Plakas, K.: Video Tracking: A Concise Survey. *IEEE Journal of Oceanic Engineering*, ročník 31, č. 2, 2006: s. 520–529, ISSN 0364-9059.

[10] Welch, G.; Bishop, G.: An Introduction to the Kalman Filter. Technická zpráva, Chapel Hill, NC, USA, 1995.

[11] WWW stránky: ParticleFilter - opencvx - Project Hosting on Google Code. 2010-02-04 [cit. 2010-05-13].
URL http://code.google.com/p/opencvx/wiki/ParticleFilter

[12] WWW stránky: Welcome - OpenCV Wiki. 2010-04-06 [cit. 2010-05-05].
URL http://opencv.willowgarage.com/wiki

[13] WWW stránky: opencvx - Project Hosting on Google Code. [cit. 2010-05-13]. URL http://code.google.com/p/opencvx

[14] Yilmaz, A.; Javed, O.; Shah, M.: Object Tracking: A Survey. *ACM Computing Surveys*, ročník 38, č. 4, 2006: s. 1–45, ISSN 03600300.

Zoznam príloh

Príloha A: Ovládanie programu

Príloha B: Adresárová štruktúra DVD

Príloha C: DVD

Príloha D: Plagát

Príloha A

Ovládanie programu

Program sa spúšťa z príkazového riadku zadaním `track` a príslušných prepínačov. Tie sú všetky nepovinné, v prípade ich vynechania sa použijú prednastavené hodnoty a program sa pokúsi načítať video z pripojenej kamery. Program pracuje s videami typu `.avi`, jeho výstupom je súbor `output.avi`.

Po spustení programu užívateľ pomocou myši vyberie sledovaný objekt a stlačením medzerníku spustí samotné sledovanie. Beh programu možno prerušiť stlačením klávesy Esc.

Parametre programu:

`<vid>`
: Zdrojový súbor s videom

`-et <def>|<gray>|<rgb>|<hyb m n>`
: Výber spôsovu ohodnocovania častíc
  - `<def>` Základný prístup
  - `<gray>` Prístup založený na šedotónovom histograme
  - `<rgb>` Prístup založený na farebnom histograme
  - `<hyb m n>` Hybridný prístup s parametrami `m` a `n`

`-p <n_castic>`
: Počet častíc
  Prednastavená hodnota: 300

`-sx <rozptyl_x>`
: Rozptyl normálneho rozloženia šumu so strednou hodnotou 0 pre x-ovú súradnicu polohy častice
  Prednastavená hodnota: 5

`-sy <rozptyl_y>`
: Rozptyl pre y-ovú súradnicu polohy častice
  Prednastavená hodnota: 5

`-sw <rozptyl_w>`
: Rozptyl pre šírku častice
  Prednastavená hodnota: 0

`-sh <rozptyl_h>`
: Rozptyl pre výšku častice
Prednastavená hodnota: 0

`-sr <rozptyl_r>`
: Rozptyl pre rotáciu častice
Prednastavená hodnota: 0

`-rs <x> <y>`
: Rozmery obrázku na ktorý sa prevedie každá častica

`-dt`
: Vykreslenie trajektórie pohybu

`-dap`
: Vykreslenie všetkých kandidátnych častíc

`-tl <n_snimiek>`
: Prvá fáza ohodnocovania presnosti programu s nastavením počtu snímiek, po ktorých sa ma znovu vyznačiť poloha objektu

`-te`
: Druhá fáza ohodnocovania presnosti programu

`-dpm`
: Dynamická zmena parametrov na základe rýchlosti objektu

`-dpe <prah_prekryvania>`
: Dynamická zmena parametrov na základe ohodnotenia častíc s nastavením prahovej hodnota detekujúcej prekrývanie

`-h`
: Vypísanie nápovedy

Príklady použitia:

`track video1.avi -p 300 -sx 8 -sy 8`
: Načítanie videa `video1.avi`, nastavenie počtu častíc na 300, rozptylu šumu pre x-ovú a y-ovú polohu na 8

`track video1.avi -et hyb 4 8 -dap -dt`
: Načítanie videa `video1.avi`, použitie hybridného algoritmu s parametrami 4 a 8, zobrazenie všetkých častíc, vykreslenie trajektórie

`track -dpm`
: Načítanie videa z pripojenej kamery, použitie nástroja na dynamickú zmenu parametrov na základe rýchlosti pohybu

Príloha B

Adresárová štruktúra DVD

Príloha popisuje adresárovú štruktúru priloženého DVD do druhej úrovne.

`\bin`	binárne súbory
`track.rar`	spustiteľná verzia programu
`\doc`	dokumenty
`manual.pdf`	manuál k programu
`sprava.pdf`	správa
`plagat.jpg`	plagát
`\src`	zdrojové súbory
`\manual`	zdrojové súbory manuálu
`\sprava`	zdrojové súbory správy
`\track`	zdrojové súbory programu track
`README.txt`	informácie o možnostiach prekladu
`\vid`	videá
`\ukazky`	ukážkové videá ku kapitole 6
`\zdrojove`	zdrojové videá ku kapitole 6
`demo.wmv`	demonštračné video k programu track
`README.txt`	základné informácie