České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Petr Zuzánek

Studijní program: Softwarové technologie a management

Obor: Inteligentní systémy

Název tématu: Sledování skoro lineárních objektů v reálném čase

Pokyny pro vypracování:

1. Seznamte se s úlohou sledování objektů ze stabilizované kamerové hlavice nesené na vrtulníku nebo bezpilotním letadle a s předchozími relevantními pracemi. Zaměřte se na lineární objekty, jako např. silnice, řeky, lesní průseky.
2. Nastudujte optimalizační metodu, dynamické programování.
3. Navrhněte metodu pro sledování sledování lineárních objektů z videosekvencí pořízených letícím pozorovatelem.
4. Metodu implementujte v C++ a navažte na existující sw na pracovišti zadavatele.
5. Výslednou metodu i implementaci zdokumentujte z návrhového, programátorského i uživatelského hlediska a také experimentálně ověřte. Zdokumentujte i situace, kdy metoda přestává fungovat.

Seznam odborné literatury:

[1] Sonka, M.; Hlavac, V.; Boyle R.: Image Processing, Analysis, and Machine Vision. 3rd edition, Thomson Learning Toronto, April 2007, 821 p, ISBN 049508252X (2nd edition Brooks/Cole, Pacific Grove, CA, 1999, 1st edition Chapman & Hall, London 1993).
[2] Svoboda, T.; Kybic, J.; Hlavac V.: Image Processing, Analysis, and Machine Vision, A MATLAB Companion. Thomson Learning, Toronto, ISBN 0495295957, Sept 2007, 265 p. Learning, str. 73-76, 2007.

Další literaturu dodá vedoucí práce.

Vedoucí bakalářské práce: Ing. Karel Zimmermann, Ph.D.

Platnost zadání: do konce zimního semestru 2010/2011

L.S.

prof. Ing. Vladimír Mařík, DrSc.
vedoucí katedry

doc. Ing. Boris Šimák, CSc.
děkan

V Praze dne 30. 11. 2009

České vysoké učení technické v Praze
Fakulta Elektrotechnická

Bakalářská práce

Sledování skoro lineárních objektů v reálném čase

Praha, 26. května 2010 **Autor: Petr Zuzánek**

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze dne 26.5.2010

Petr Zvára

podpis

i

Poděkování

Na prvním místě velice děkuji vedoucímu bakalářské práce Ing. Karlovi Zimmermannovi, Ph.D, který mi umožnil pracovat na zajímavé úloze a poskytl mi spoustu cenných rad a triků spjatých s tvorbou práce. Děkuji mu také za odborné vedení práce a za čas, který mi věnoval. Děkuji prof. Ing. Václavu Hlaváčovi, CSc, za cenné komentáře a připomínky.

Abstrakt

Hlavním záměrem této bakalářské práce je návrh metody pro sledování skoro lineárních objektů z videosekvencích pořízených letícím pozorovatelem v reálném čase. Pro účely této konkrétní aplikace je jako nejvhodnější metodou určena poloautomatická metoda založená na porovnávání vzorů. Pro detekci lineárních objektů je použit rozšířený algoritmus hledající nejlevnější cestu mezi dolním a horním okrajem oblasti zájmu. Rozšířený algoritmus umožní sledovat objekty s podstatně nižšími nároky na jejich tvar a umístění v obraze, oproti standardnímu, což zvyšuje robustnost a variabilitu algoritmu.

Abstract

The main aim of this bachelor project is to design appropriate method for real-time tracking of nearly linear object from video sequences captured by flying observer. The semi-automated method based on template matching is chosen as a right solution for this particular kind of application. Linear objects are extracted using extended algorithm for finding minimal cost connected path between the bottom edge and top edge of a Region of interest. The extended algorithm for finding minimal cost allows tracking of object with less requirements for its orientation or location in a image. In comparison with basic algorithm it increases robustness and variability of the tracker.

Obsah

Kapitola 1

Úvod, motivace

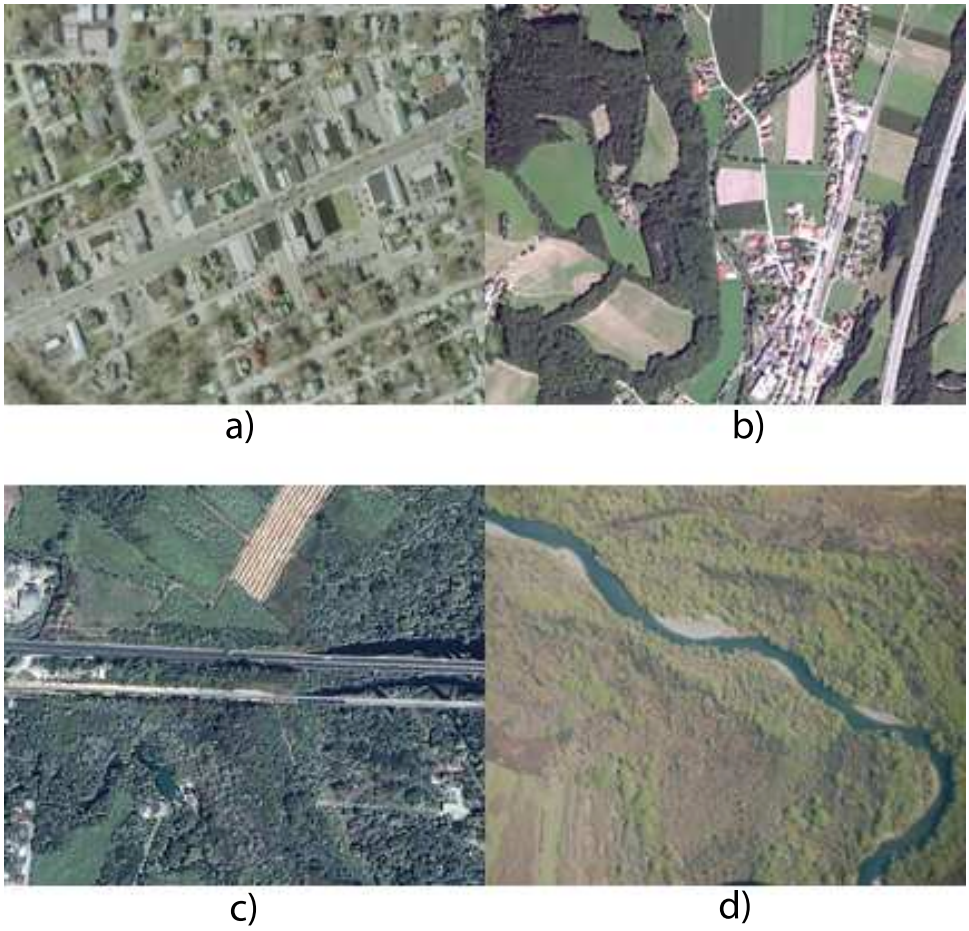
Tato práce je součástí projektu Centra strojového vnímání (CMP) Katedry kybernetiky Fakulty elektrotechnické ČVUT v Praze pro návrh a stavbu stabilizované kamerové hlavy využívané např. pro bezpilotní letadla. Tento projekt byl podpořen Ministerstvem průmyslu a obchodu České republiky v rámci projektu FR-TI1/265. Stabilizovaná kamerová hlava je zobrazena na obrázku 1.1

Obrázek 1.1: Stabilizovaná kamerová hlava.

Řízení kamerové hlavy je docíleno pomocí dvou zpětnovazebních okruhů. První, rychlý, zpětnovazební okruh stabilizuje hlavu vůči inerciální souřadné soustavě z gyro-

skopu a akcelerometru, čímž je kompenzován pohyb a vibrace letadla. Druhý, pomalejší, zpětnovazební okruh slouží ke sledování objektů na zemi, např. jedoucího auta.

V možných civilních i vojenských aplikacích hrají důležitou roli výrazně podlouhlé objekty, jako jsou např. silnice, řeky, lesní průseky, železnice, potrubní či elektrické vedení, atd. Tyto objekty jsou nepříliš přesně v literatuře dálkového průzkumu Země nazývány lineárními objekty. Termínu lineární objekt se přidržím i v této práci. Zástupci lineárních objektů jsou na obrázku 1.2.



Obrázek 1.2: Lineární objekty pozorované z leteckých snímků, kde
a), b) zobrazují silnici,
c) zobrazuje železnici,
d) zobrazuje řeku.

Metody pro extrahování lineárních objektů z digitálních, ať už letecký nebo satelitních snímků nacházejí uplatnění ve dvou rozdílných třídách aplikací. První třídou,

do které spadají i aplikace pro stabilizovanou kamerou hlavu, jsou aplikace pro automatickou navigaci nebo dohled. Jejich úkolem je zpravidla sledovat konkrétní lineární objekt. Druhou třídu pak tvoří aplikace využívané v oblasti získávání dat pro geografický informační systém (GIS), jejich databází a následnou aktualizaci. Cílem těchto aplikací je extrahovat všechny lineární objekty ve snímku.

Mým úkolem je vytvoření modulu pro detekci lineárních objektů potencionálně použitelného v kamerové hlavě.

Kapitola 2

Formulace úlohy

V této bakalářské práci navrhuji a implementuji vhodný algoritmus pro kamerovou hlavu, který bude schopný detekovat a následně sledovat lineární objekty. Lineárním objektem lze rozumět objekt, jenž splňuje následující vlastnosti:

1. Šířka objektu je ve snímku téměř konstantní.
2. Směr, kterým objekt pokračuje se mění plynule.
3. Vzhled podél objektu se liší jen velmi málo a mění se pomalu.
4. Rozdíl barvy objektu a barvy pozadí je výrazný (diskriminabilita).
5. Objekty jsou dlouhé, často prochází napříč celým snímkem.

Na aplikaci jsou kladeny tři hlavní požadavky:

1. *Reálný čas.*
   Uplatnění algoritmu na kamerové hlavě je k řízení polohy kamerového systému nebo bezpilotního letadla samotného v reálném čase, a to na základě polohy detekovaného objektu. Za tímto účelem je nutná koordinace s pilotem nebo operátorem, který vybere objekt ke sledování. Při zpracování videosignálu v reálném čase je nutné, aby měl pilot nebo operátor možnost označit sledovaný objekt v krátkém časovém úseku, tj. než sledovaný objekt ujede nebo se ztratí mimo zorné pole kamery. Kvůli požadavku na zpracování v reálném čase bude navržená metoda implementována v jazyce C++.

2. *Obecnost.*

   Cílem je schopnost sledovat obecný lineární objekt, nikoliv pouze určitou specifickou skupinu lineárních objektů jako např. silnice, řeky, atd. Obecným lineárním objektem je myšlen i objekt ve smyslu jeho orientace, směru, šířky, atd.

3. *Robustnost*

   Pro schopnost plynulého sledování označeného objektu musí být algoritmus robustní, čehož bude dosaženo vysokou diskriminativností modelu, čili schopností odlišit správný objekt od ostatních objektů nebo od pozadí.

Kapitola 3

Stav vědění o úloze

Do dnešní doby je uvedeno množství metod, od různých výzkumníků, či výzkumných týmu, které popisují, jak z informací dostupných z digitálních snímků extrahovat lineární objekty.

Známé metody pro extrahování lineárních objektů lze dělit vzhledem k nutnosti inicializace algoritmu operátorem na automatické a poloautomatické metody:

1. *Automatické*
   Pracují zcela nezávisle, bez zásahu operátora. Využívají se např. pro aktualizaci GIS, či jiných map a systémů.

2. *Poloautomatické*
   Vyžadují interakci s uživatelem. Často je požadováno, aby uživatel inicializoval algoritmus předáním informací o objektu, který má být extrahován. Informacemi předanými uživatelem často bývá pozice objektu a směr, jakým z dané pozice objekt pokračuje.

Automatické metody, pracují nad obecným modelem objektu (např. silnice). Kvůli obecnosti modelu často selhávají například v městském prostředí. Zde jsou silnice na mnoha místech částečně překryty korunami stromů, či je jejich povrch ovlivněn dopadajícími stíny přilehlých budov, a silnice již neodpovídá modelu.

Toto částečně potlačují poloautomatické metody, které díky operátorově interakci dovolují částečně specifikovat vlastní model a jeho vlastnosti. Disponují tak větší variabilitou oproti metodám automatickým. Jsou více diskriminativní pro konkrétní typ objektu.

Pro aplikaci na stabilizované kamerové hlavě se tak poloautomatické metody, které fungují v interakci s operátorem, jeví jako dobrý kompromis kombinující přesnost a rychlost detekce.

Níže jsou uvedeny zástupci jak automatických, tak poloautomatických metod, resp. způsobů, jakým lze k problému detekce lineárních objektů v obraze přistupovat.

3.1 Automatické metody

Automatické metody jsou s výhodou využívány pro aktualizaci GIS, map, či jiných systémů, kde je požadována schopnost detekování všech objektů v obraze. Principy využívané v automatických metodách lze stejně využít i v poloautomatických metodách. Pouze jsou doplněné o zásah operátora. Jako zástupce této kategorie lze uvést automatickou metodu pro detekování hlavních silnic z leteckých snímků [1] navrženou Barzoharem a Cooperem, publikovanou v roce 1996. Metoda pracuje nad geometricko-stochastickým modelem silnic. Ten zohledňuje střed silnice, šířku silnice, její jasovou intenzitu a hranice, kterou je definována. Běh metody popisuje následující část.

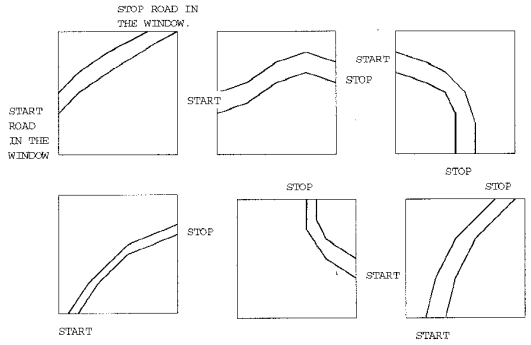
Princip činnosti algoritmu

Princip činnosti algoritmu lze rozdělit do tří následujících částí:

- *Rozdělení snímku na oblasti.*
  Snímek je rozdělen do čtvercových oblastí o stejné velikosti.

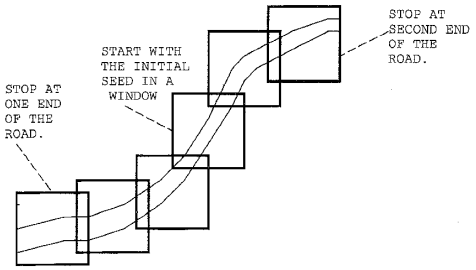
- *Odhad kandidátů v oblastech.*
  Kde kandidátem je rozuměna libovolná, naučenému modelu odpovídající spojnice dvou různých stran oblasti. Možné varianty znázorňuje obrázek 3.1. Mohou nastat případy, kdy je v oblasti více vhodných kandidátů. Pro nalezení všech kandidátů je algoritmus v každé oblasti spouštěn opakovaně, ovšem s tím rozdílem, že již detekovaný kandidát je zamaskován a tím ignorován. Takto se hledání v konkrétní oblasti opakuje dokud nejsou všichni kandidáti nalezeni.

- *Kombinování kandidátů.*
  Kandidáty získané v předešlém kroku je nutné vhodně spojit ve výsledný objekt.



Obrázek 3.1: Hledané varianty kandidátů.

Proces začíná v oblasti s vhodným kandidátem (např. v oblasti, kde se nachází nejlepší kandidát). Pro aktuální oblast se hledá její nejlepší rozšíření pomocí dynamického programování. Rozšiřování probíhá ve směru, kterým kandidát vstupuje a vystupuje z dané oblasti. Tím způsobem, že je k aktuální oblasti přidána oblast bezprostředně sousedící s aktuální oblastí v daném směru. Kombinování kandidátů je zobrazeno na obrázku 3.2. Rozšiřování je opakováno dokud nedojde



Obrázek 3.2: Kombinování kandidátů.

ke splnění ukončujících podmínek, nebo dokud extrahovaný objekt nenarazí na hranice obrazu.

3.2 Poloautomatické metody

Zde budou uvedeny tři hlavní přístupy, které se používají v oblasti poloautomatických metod detekce lineárních objektů. Se zaměřením pouze na jejich základní principy fungování. V dnešní době již byla publikována řada modifikací, vylepšení, či kombinací

všech těchto přístupů, které vedou k lepším výsledkům a robustnosti metod.

3.2.1 Metody založené na adaptaci aktivních obrysů

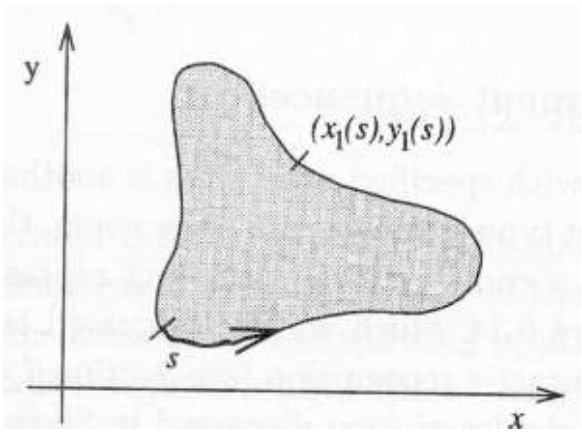
V odborných článcích uváděno pod názvem *Active Contours* neboli *Snakes*. Podrobný popis základního principu činnosti lze nalézt v [7].

Cílem těchto algoritmů je najít nejlepší aproximaci obrysu libovolného objektu v obraze. V inicializačním procesu je operátorem hrubě definován tvar objektu pomocí rozmístěných inicializačních bodů, tzv. aktivní obrys $\mathbf{c}$.

Způsob deformace určuje energetická funkce, která popisuje, jak dobře přiléhá aktivní obrys ke kontuře objektu. Cílem je naleznout aktivní obrys s minimální hodnotou energetické funkce. Přičemž minimum energetické funkce je hledáno gradientní minimalizací.

Energetická funkce

Parametrickým vyjádřením obrysu jako $c(s) = (x(s), y(s))$, kde $x(s)$ a $y(s)$ jsou souřadnice podél obrysu a $s \in [0, 1]$ lze energetickou funkci E vyjádřit jako integrál součtu členů. Parametrizace obrysu je znázorněna na obrázku 3.3. Každý člen odpovídá určité



Obrázek 3.3: Parametrizace obrysu.

síle působící na aktivní obrys. Hodnotu energetické funkce lze vypočítat jako:

$$E = \int (\alpha(s)E_{cont} + \beta(s)E_{curv} + \gamma(s)E_{image})\,\mathrm{d}s\,, \tag{3.1}$$

kde funkce α, β a γ přiřazují váhy ovlivňující význam jednotlivých energetických členů podél aktivní oblasti. Váhy se podél aktivní oblasti mohou lišit. Jednotlivé členy mají následující význam:

E_{image} přitahuje aktivní obrys k nejbližší hraně v obraze,

E_{cont} nutí aktivní obrys být kontinuální,

E_{curv} nutí aktivní obrys být hladký.

Metody pro extrahování lineárních objektů s pomocí aktivních obrysů lze nalézt např. v [3],[2].

3.2.2 Metody založené na porovnávání vzorů

Tyto algoritmy zpravidla fungují tak, že operátor označí v inicializačním procesu části obrazu reprezentující hledaný objekt - vzor. Vzorem může být pouze část samotného lineárního objektu. Dalším inicializačním krokem je přiřazení orientace ke každému vzoru, udávající postup objektu reprezentovaného daným vzorem.

Směr může být určen buď explicitně operátorem nebo vypočten vhodnou analýzou vzoru. Příkladem je metoda [4] navržená Parkem a Kimem, kde jsou shlukováním pixelů o podobné hranové orientaci a šířky hrany extrahovány všechny linie. Orientace nejdelší linie je považována za orientaci vzoru.

Tímto je inicializace vstupních dat pro algoritmus ukončena. Algoritmus začne v určeném směru (a obvykle i v opačném) hledat takové oblasti, které jsou podle definovaného kritéria podobné vzoru. V případě, že se shodují dostatečně přesně, lze prohlásit, že vzor s prohledávanou oblastí tvoří stejný objekt.

3.2.3 Metody založené na detekci hran

Zde je inicializační proces prováděný operátorem obdobný jako u metod založených na porovnávání vzorů. Operátorem je nutné zadat počáteční bod. Směr postupu ze startovního bodu může být zadán operátorem. Algoritmus uvedený v [5], který extrahuje silnice dokáže směr postupu vypočítat s pomocí orientace významných hran v blízkém

okolí počátečního bodu (hranice silnice). Tyto hrany také podají informaci o aktuální šířce objektu.

Ovšem v obecném případě nelze tento výpočetní postup aplikovat na libovolný lineární objekt. Výpočet by selhal například při snaze o extrahování železniční tratě, kde jsou i pražce reprezentovány významnými hranami.

Algoritmus uvedený v [5] postupuje v určeném směru. Na základě odhadu ceny zaplacené při přechodu na danou pozici, se algoritmus může vydat levnější cestou úkrokem vlevo či vpravo. Cena zohledňuje jak změnu intenzit pixelů tak směr postupu (snahou je jít co nejvíce zpříma a po oblastech se stejnou intenzitou pixelů). Pro nejlevnější pozici je přepočítána opět šířka silnice a orientace pro další expandování. Když je v porovnání s předchozí hodnotou šířka silnice podstatně vyšší, je přítomna křižovatka. V takovém okamžiku se algoritmus vydá jednou z cest a po jejím celém detekování, nebo po splnění ukončujících podmínek, bude pokračovat na dané křižovatce v jiném směru.

Kapitola 4

Volba vhodné metody

Tato část bakalářské práce je zaměřena na volbu vhodné metody pro detekování lineárních objektů z videosekvencí pořízených letícím pozorovatelem a obhájení vhodnosti této metody v porovnání s ostatními využívanými metodami.

Pro účely této konkrétní aplikace, kterým je sledování lineárních objektů z videosekvencí pořízených letícím pozorovatelem, jsem se rozhodl využít poloautomatickou metodu založenou na porovnávání vzorů popsanou v kapitole 3.

Zde se jeví volba poloautomatické metody jako vhodná. Zejména z toho důvodu, že výsledky běhu algoritmu poslouží především pro řízení polohy kamerového systému a případně bezpilotního letadla samotného (let podél silnice atd.). Přítomnost operátora je zde tedy i tak potřeba k vybrání konkrétního objektu.

V případě, že by snímky pořízené z letícího pozorovatele sloužily pro aktualizaci map nebo GIS, pak je vhodné použít automatickou metodu.

Následující fakta hovořila ve prospěch volby metody založené na porovnávání vzorů. Ty popisují výhody a nevýhody poloautomatických metod popsaných v kapitole 3 vztažené pro tuto konkrétní aplikaci:

1. *Metody založené na adaptaci aktivních obrysů*
   Vyžadují rozmístění inicializačních bodů podél objektu, což zpravidla od operátora vyžaduje více zásahů. Při zpracování videosignálu v reálném čase se tak může stát, že při inicializaci objekt ujede od již rozmístěných inicializačních bodů. Gradientní minimalizací je dosažena konvergence do lokálního minima. Tak může při větším pohybu (např. při turbulencích) dojít ke ztrátě objektu.

2. *Metody založené na detekci hran*

   Algoritmy založené na detekci hran nejsou dostatečně obecné pro sledování libovolného lineárního objektu. Např. při sledování železničních tratí, kde výrazné hrany reprezentují jak hranice objektu (železniční trať), tak železniční pražce. Výhodou je ovšem rychlá, takřka okamžitá inicializace operátorem.

3. *Metody založené na porovnávání vzorů*

   Vystačí s nenáročnou a rychlou inicializací operátora. Případnou částečnou nelinearitu či náhlou změnu objektu lze postihnout vhodnou aktualizací vzoru, či prací s více vzory najednou. V obecném měřítku nabízí velkou míru variability a parametrizace hledaného objektu například změnou velikosti a orientace vzoru, aktualizací vzoru, atd. Variabilitu také nabízí možnost volby kriteriální funkce hodnotící míru podobnosti vzoru na porovnávané pozici snímku.

   Výhodou metod založených na porovnávání vzorů je možnost sledovat obecné lineární objekty a to dostatečně diskriminativně.

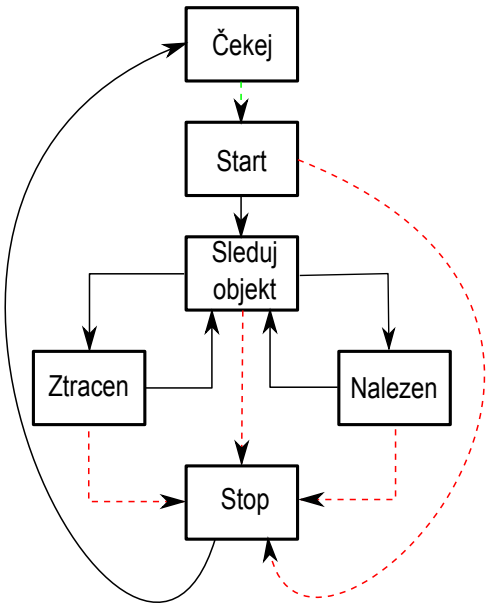
Kapitola 5

Řešení

Tato část je zaměřena na podrobný popis navržené metody vhodné pro sledování lineárních objektů pomocí porovnávání vzorů.

5.1 Algoritmus pro detekci lineárních objektů

V této sekci je popsán funkční popis navrženého algoritmu a k němu odpovídající stavový automat, který bude implementován.



Obrázek 5.1: Stavový automat pro sledování.

5.1.1 Funkční popis algoritmu

1. *Inicializační proces prováděný operátorem*
   Od operátora je požadováno:

   - Označit část snímku jako reprezentativní část objektu (vzor).
   - Definovat oblast zájmu.
     Oblast zájmu je malá část snímku, ve které se daný objekt nachází. Vlastnosti a parametry oblasti zájmu jsou popsány v kapitole 5.4.

2. *Sledovací proces*
   Po inicializačním procesu algoritmu nastává sledovací proces, který automaticky vykonává v každém snímku tyto kroky:

   (a) *Porovnávání vzoru v oblasti zájmu.*
   Algoritmus porovná získaný vzor na všech pozicích, které se nacházejí uvnitř oblasti zájmu a pro každou takovou pozici určí, jak moc vzor odpovídá části snímku na definované pozici. Podobnost porovnávaných oblastí je určena kriteriální funkcí O, jejíž popis je v kapitole 5.2.

   (b) *Nalezení cesty v oblasti zájmu nejvíce podobné vzoru.*
   Pro nalezení objektu je použit modifikovaný algoritmus hledající nejpodobnější cestu mezi dolním a horním okrajem definované oblasti. Přesný popis algoritmu pro nalezení cesty nejvíce podobné vzoru je popsán v kapitole 5.3.

   (c) *Aktualizace parametrů oblasti zájmu*
   Nové parametry oblasti zájmu jsou vypočítány z informací aktuálně nalezeného objektu. Přesný popis výpočtu nových parametrů lze naleznout v kapitole 5.5.

   (d) *Aktualizace vzoru*
   Způsoby pro aktualizaci vzoru využité v této práci, jsou popsány v kapitole 5.6.

5.1.2 Stavový automat pro sledování

Zde je popsán stavový automat, pro tuto aplikaci. Navržený automat je na obrázku 5.1, kde přechod mezi stavy automatu může být buď přirozený (černá plná hrana) nebo způsobený vnější událostí (zásahem operátora). Zelená přerušovaná hrana signalizuje přechod při události nastalé po ukončené inicializaci operátorem. Červená přerušovaná hrana značí přechod při nastalé události pro ukončení sledování. Pokud nebude řečeno jinak, automat přejde do koncového stavu hrany až v následujícím snímku.

Popis stavů stavového automatu

- Stav *Čekej*.
  Žádný objekt není sledován. Slouží pro poskytnutí inicializačních parametrů od operátora.

- Stav *Start*.
  Na základě inicializačních parametrů je získán vzor a parametry oblasti zájmu.

- Stav *Sleduj objekt*.
  Hledání cesty nejpodobnější vzoru v dané oblasti zájmu. Nalezený objekt je ohodnocen, jak velkou měrou vyhovuje požadavkům. Hodnocením je v této práci průměrná hodnota kriteriální funkce O podél nalezené cesty W, která je porovnávána s definovaným prahem Θ.

  $$\frac{1}{N} \cdot \sum_{i \in W} (O(\mathbf{x_w})) \leq \Theta \,, \tag{5.1}$$

  kde N značí počet prvků množiny W. Bod $\mathbf{x_w}$ je definován jako $\mathbf{x_w} = (W_h(i), i)^\top$, kde W_h označuje množinu všech řádků i, na kterých je cesta definována. Pokud je požadavek splněn, ještě v tomtéž snímku, automat přechází do stavu *Nalezen*. Pokud objekt požadavky nesplňuje, pak přechází, do stavu *Ztracen* (v tomtéž snímku).

- Stav *Nalezen*.
  V případě, že je objekt nalezen následuje aktualizace parametrů oblasti zájmu a aktualizace vzoru, které jsou popsány v kapitole 5.5 a 5.6.

- Stav *Ztracen*.
  Vzor ani oblast zájmu se nemění.

- Stav *Stop*.
  Ukončení sledování objektu. Vymazání používaného vzoru a parametrů oblasti zájmu.

5.2 Kriteriální funkce O

Kriteriální funkce O určuje podobnost vzoru T s odpovídající částí obrazu I. Vzor je definován souřadnicemi obrazových bodů $\mathbf{x_i}$, které jsou relativní vůči středu vzoru. Kritérium určující podobnost vzoru s porovnávanou oblastí je definováno vztahem:

$$O(\mathbf{p}) = \min_{\alpha} \left(\sum_{\mathbf{x_i}} o\left(I(G(\mathbf{x_i}, [\mathbf{p}; \alpha])), T(\mathbf{x_i})\right) \right) , \tag{5.2}$$

kde $T(\mathbf{x_i})$ určuje jasové intenzity vzoru na pozici $\mathbf{x_i}$, a $G(\mathbf{x_i}; [\mathbf{p}, \alpha])$ posouvá souřadnice x_i otočené kolem středu vzoru o úhel α na pozici $\mathbf{p} = (p_x, p_y)^\top$.

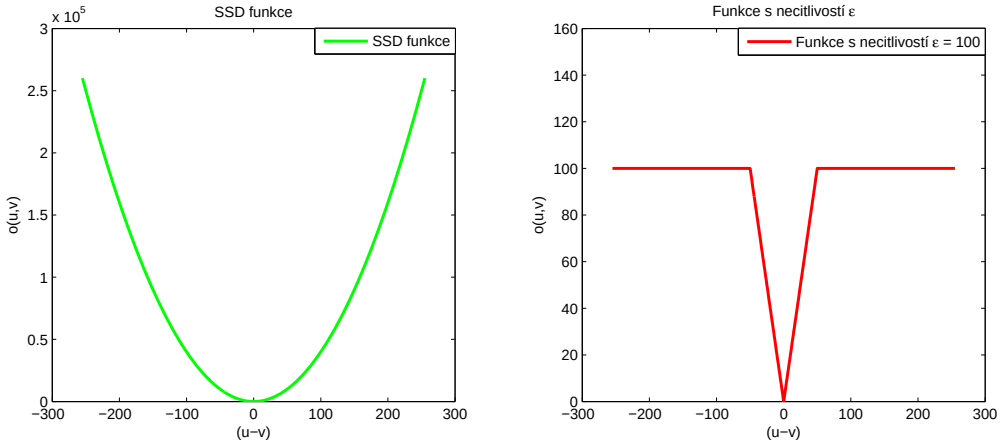
5.2.1 Funkce o

Funkce o může být definována následujícími předpisy:

1. *SSD (sum-squared-difference)*
   Je dán předpisem:
   $$o(u, v) = (u - v\ )^2 \ . \tag{5.3}$$

2. *Funkce s necitlivostí ϵ*
   Může být dána předpisem:
   $$o(u, v) = \begin{cases} \epsilon & |u\ - v\ | > \epsilon \ , \\ |u\ - v\ | & \text{jinak} \ . \end{cases} \tag{5.4}$$
   pro $\epsilon \geq 0$, který značí definovaný penalizační práh.

Průběh jak funkce SSD tak funkce s necitlivostí ϵ znázorňuje obrázek 5.2. G může značit obecnou geometrickou transformaci, kde prvním argumentem jsou souřadnice původního bodu a druhým argumentem jsou parametry transformace.



Obrázek 5.2: Průběh funkce SSD a funkce s necitlivostí $\epsilon = 100$.

5.2.2 Geometrické transformace G vzoru

Transformace G je funkce $G(\mathbf{x}, \mathbf{p})$ transformující množinu bodů $\mathbf{x}$ pomocí parametrů $\mathbf{p}$ transformační matice T. Např. $G(\mathbf{x}, \mathbf{p}) = \mathbf{x} + \mathbf{p}$, kde $\mathbf{p}$ jsou parametry transformační matice popisující translaci.

V mnoha případech lze změnu tvaru objektu předpokládat či očekávat. To je přesně případ detekce lineárních objektů, neboť objekt podél celé své délky může měnit svůj tvar. Mezi základní geometrické transformace patří:

1. *Otočení obrazu*
2. *Změna měřítka obrazu*
3. *Translace obrazu*
4. *Zkosení obrazu*

Důsledkem geometrické transformace je změna polohy obrazového bodu. Tuto změnu popisuje obecná tzv. *afinní transformace*. Afinní transformací lze popsat všechny výše zmíněné transformace. Obecný tvar afinní transformace G, s koeficienty transformace $a_{0..2}$ a $b_{0..2}$, je dán vztahem:

$$x' = a_0 + a_1 x + a_2 y \,, \tag{5.5}$$

$$y' = b_0 + b_1 x + b_2 y \,. \tag{5.6}$$

Zde $(x', y')^\top = X'$ značí novou pozici transformovaného bodu $(x, y)^\top = X$.

S použitím homogenních souřadnic lze afinní transformaci vyjádřit v maticovém tvaru

$$\begin{pmatrix} X' \\ 1 \end{pmatrix} = \mathrm{T}_{[2\times 3]} \cdot \begin{pmatrix} X \\ 1 \end{pmatrix} . \tag{5.7}$$

Po dosazení je maticový tvar následující:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a_1 & a_2 & a_0 \\ b_1 & b_2 & b_1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} . \tag{5.8}$$

Jednotlivé výše popsané transformace se liší koeficienty transformační matice T. Tvar transformační matice pro různé afinní transformace je následující:

Otočení obrazu $G(\mathbf{x}, \alpha)$

$$\mathrm{T} = \begin{pmatrix} \cos(\alpha) & -sin(\alpha) & 0 \\ \sin(\alpha) & cos(\alpha) & 0 \end{pmatrix} , \tag{5.9}$$

kde parametr α značí úhel rotace.

Změna měřítka obrazu $G(\mathbf{x}, \mathbf{s})$

$$\mathrm{T} = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \end{pmatrix} , \tag{5.10}$$

kde s_x značí změnu měřítka v ose x a koeficient s_y uvádí změnu měřítka v ose y.

Translace obrazu $G(\mathbf{x}, \mathbf{p})$

$$\mathrm{T} = \begin{pmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \end{pmatrix} , \tag{5.11}$$

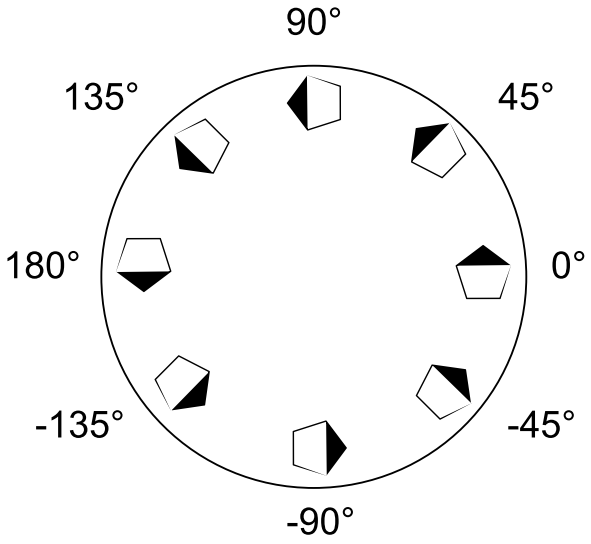
kde p_x značí posunutí v ose x a p_y posunutí v ose y.

Zkosení obrazu $G(\mathbf{x}, \mathbf{c})$

$$\mathrm{T} = \begin{pmatrix} 1 & c_x & 0 \\ c_y & 1 & 0 \end{pmatrix} , \tag{5.12}$$

kde koeficient c_x udává zkosení podél osy x. Obdobně c_y podél osy y.

V této práci bude experimentováno pouze s rotací, kdy bude získaný vzor transformován dle různých hodnot parametru α. Očekávaným výsledkem je lepší postihnutí změny směru objektu, což povede na přesnější výsledky sledování. Obrázek 5.3 znázorňuje rotaci obecného objektu kolem svého středu o definovaný úhel.



Obrázek 5.3: Znázornění rotace pro obecný objekt.

5.3 Algoritmus pro hledání cesty s nejnižší cenou

Popis algoritmu lze naleznout v [6]. Cílem algoritmu je nalezení cesty takové, pro které je součet hodnot kriteriální funkce O na dané cestě minimální. Při hledání minimální cesty algoritmus postupuje po řádcích od dolní hrany oblasti zájmu nahoru (cesta prochází každým řádkem obrazu pouze jedním pixelem). Cesta do pixelu $Y_{i,j}$ na řádku i a ve sloupci j může vést pouze z pixelu $X_{i-1,j-1}$, $X_{i-1,j}$ nebo $X_{i-1,j+1}$ (Předchůdci pixelu $Y_{i,j}$). Pro každý pixel $X \in P(Y_{i,j})$, kde P značí množinu všech předchůdců pixelu $Y_{i,j}$ je definována cena $F(X)$. Potom optimální cesta do pixelu $Y_{i,j}$ vede přes:

$$\arg \min_{X \in P(Y_{i,j})} (F(X)) \,. \tag{5.13}$$

Cena na pozici $Y_{i,j}$ tak bude definována jako:

$$F(Y_{i,j}) = \min_{X \in P(Y_{i,j})} (F(X)) + O(Y_{i,j}) \,, \tag{5.14}$$

kde $O(Y_{i,j})$ značí hodnotu kriteriální funkce O na pozici $Y_{i,j}$.

Algoritmus pracuje v následujících krocích:

1. Inicializace ceny cesty F v prvním řádku je určena hodnotou kriteriální funkce O na příslušných pozicích.

$$F(Y_{1,j}) = O(Y_{1,j}), pro\ \forall j\ . \tag{5.15}$$

2. Po řádcích o počtu `h` je počítána cena cesty F do každého obrazového bodu $Y_{i,j}$ v řádku i, který obsahuje `w` obrazových bodů, s pomocí ceny F jejich předchůdců a hodnot kriteriální funkce pro daný bod $O(Y_{i,j})$.

```
for each row i from 2 to h
    for each column j from 1 to w
```

$$F(Y_{i,j}) = \min_{X \in P(Y_{i,j})} (F(X)) + O(Y_{i,j})\ , \tag{5.16}$$

$$M(Y_{i,j}) = \arg \min_{X \in P(Y_{i,j})} (F(X)))\ , \tag{5.17}$$

kde $M(Y_{i,j})$ reprezentuje pole o velikosti počtu obrazových bodů ve snímku pro ukládání nejlevnějšího předchůdce bodu $Y_{i,j}$ (pro možnost rekonstrukce cesty).

3. Na horní hraně oblasti zájmu je nalezen cíl nejlevnější cesty c^*,

$$c^* = \arg \min_{Y \in R^T} (F(Y))\ , \tag{5.18}$$

kde R^T značí množinu všech bodů oblasti zájmu R obsažených v jeho horní hraně.

4. Z cíle je rekonstruována nejlevnější cesta W přes zapamatované předchůdce,

$$W = c^*\ ;\ c = M(c^*)\ ; \tag{5.19}$$

```
do
```

$$W = W \cup c\ ;\ c = M(c)\ ; \tag{5.20}$$

```
while(c > 0)
```

kde c je pomocná proměnná reprezentující předchůdce.

Výhodou algoritmu je jeho rychlost a přesnost (nevyužívá heuristický odhad). Nevýhodou takto popsaného algoritmu je poměrně velké omezení na směr, kterým objekt musí procházet obrazem, aby byl správně detekován. Omezení na směr objektu je částečně řešen zavedením lineární penalizační funkce uvedené v kapitole 5.4.

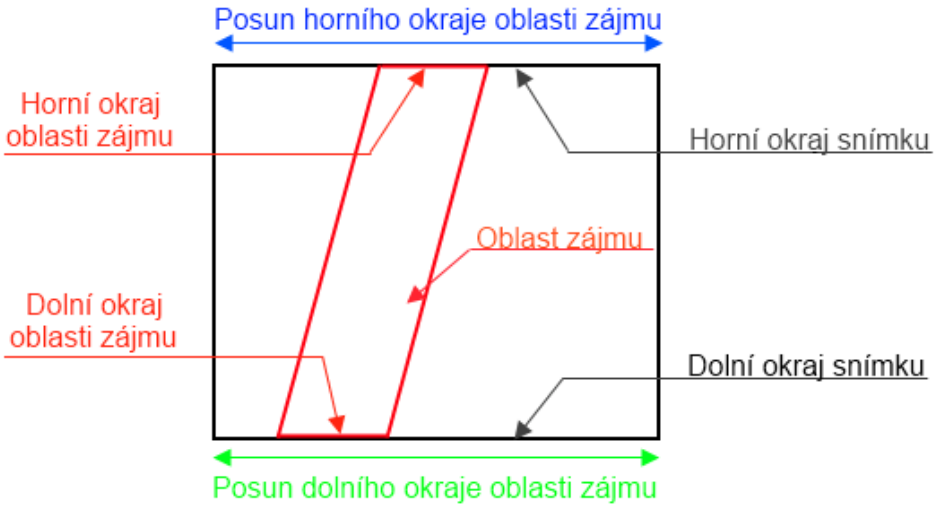
5.4 Oblast zájmu

V této části je popsán význam použití oblasti zájmu a jsou definována omezení pro její tvar a umístění.

Oblastí zájmu je rozuměna část snímku, ve které algoritmus hledá cestu s nejnižší cenou.

Výhodou použití oblasti zájmu je zvýšení rychlosti algoritmu (dynamické programování je omezeno pouze na oblast zájmu) a zvýšení robustnosti algoritmu (v malé oblasti existuje méně objektů, se kterými si lze sledovaný objekt splést).

Základní algoritmus

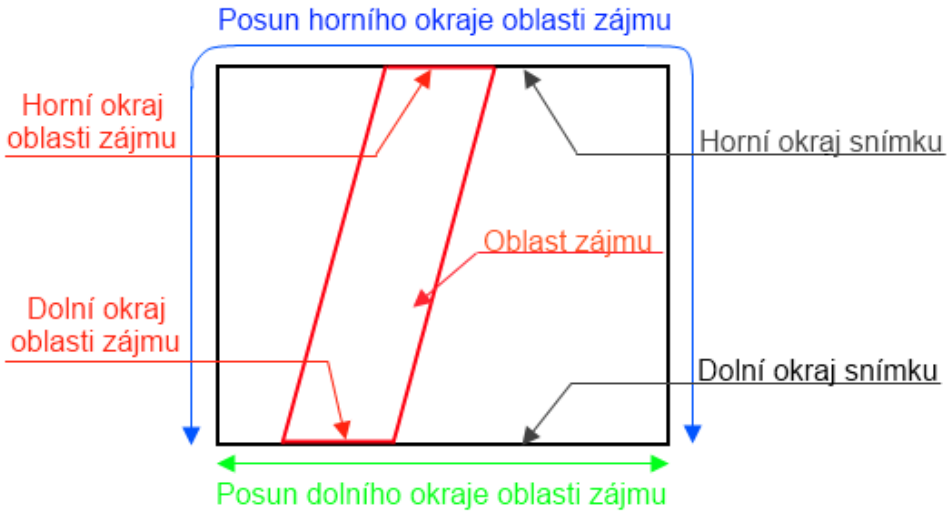


Obrázek 5.4: Možnosti základního algoritmu pro nalezení nejlevnější cesty.

V aktuální implementaci algoritmu pro hledání nejnižší cesty je nutné, aby objekt vždy začínal na dolním okraji oblasti zájmu. A to pro možnost provedení inicializačního kroku algoritmu (viz. kapitola 5.3). Standardní verze algoritmu také požaduje,

aby oblast zájmu končila na posledním řádku snímku, respektive aby všechny body, určující horní hranici, končily na stejném řádku snímku. To je z důvodů komparability cen. Taková implementace je pouze schopna detekovat objekty s velkým omezením na jejich směr a pozici v obraze. Obrázek 5.4 popisuje možnost posouvání horní a dolní hrany oblasti zájmu v obraze, se základní implementací algoritmu. Obrázek 5.4 zároveň popisuje požadavky na směr a umístění lineárního objektu.

Rozšířený algoritmus



Obrázek 5.5: Možnosti rozšířeného algoritmu pro nalezení nejlevnější cesty.

Komparabilitu cen mezi různými řádky snímku lze docílit lineární penalizační funkcí, jejíž funkční hodnota pro daný řádek bude nepřímo úměrná vzdálenosti daného řádku od prvního řádku snímku. Penalizační funkce znevýhodňuje kratší cesty. Hodnota penalizační funkce l pro řádek i může být dána předpisem:

$$l(i) = (1 - i) \cdot \lambda \,, \tag{5.21}$$

kde λ je konstanta definovaná uživatelem.

S využitím penalizační funkce l oblast zájmu může končit jak na posledním řádku snímku, tak na stranách snímku. To značně rozšiřuje použitelnost algoritmu v závislosti na směru a umístění hledaného objektu. Algoritmus je schopný detekovat objekty,

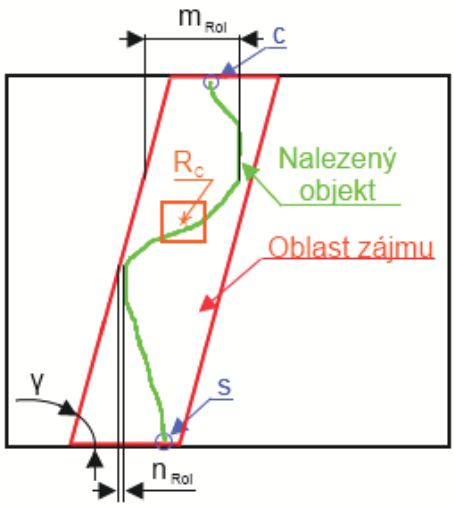
jejichž jediným omezením je přítomnost objektu na dolním okraji snímku. Možnosti rozšířeného algoritmu zachycuje obrázek 5.5. Penalizační funkce l není uvažována při výpočtu kriteriální funkce (ceny) F. Penále je připočítáváno dodatečně až při hledání cíle nejlevnější cesty c^*

$$c^* = \arg\min_{Y \in R^T} (F(Y) + l(v(Y)) , \tag{5.22}$$

kde funkce $v(Y)$ přiděluje bodu Y vertikální vzdálenost od prvního řádku snímku.

5.5 Aktualizace parametrů oblasti zájmu

Parametry popisující oblast zájmu jsou šířka, sklon a její střed. Výpočet parametrů pro aktualizaci oblasti zájmu popisuje obrázek 5.6.



Obrázek 5.6: Výpočet parametrů pro oblast zájmu.

5.5.1 Aktualizace středu oblasti zájmu

Střed oblasti zájmu r_c je totožný se středem oblasti, která se nejlépe schoduje se vzorem. Čili střed oblasti zájmu je aktualizován na pozici v obraze, kde hodnota kriteriální funkce nabývá globálního minima.

$$r_c = \arg\min_{\mathbf{p} \in R} O(\mathbf{p}) , \tag{5.23}$$

kde R značí množinu všech bodů $\mathbf{p}$ uvnitř oblasti zájmu.

Význam středu oblasti zde není geometrický, pouze se jedná o bod, kolem kterého lze otáčet oblast zájmu.

5.5.2 Aktualizace sklonu oblasti zájmu

Sklon oblasti zájmu určují pozice počátečního bodu detekovaného objektu $\mathbf{s} = (s_x, s_y)^\top$ a koncového bodu $\mathbf{c} = (c_x, c_y)^\top$. Směr oblasti zájmu je definován směrnicí přímky, která prochází počátečním a koncovým bodem detekovaného objektu. Výpočet směru oblasti zájmu k_R popisuje následující rovnice:

$$k_R = \begin{cases} \infty & c_x = s_x \ , \\ \dfrac{c_y - s_y}{c_x - s_x} & \text{jinak} \ . \end{cases} \tag{5.24}$$

Jedná se o tangentu úhlu γ popisující odklon oblasti zájmu od horizontální osy obrazu.

Hraniční sklon oblasti zájmu je určen jejím umístěním v obraze a podmínkou, že musí začínat na prvním řádku snímku. Další výslovné omezení na sklon oblasti zájmu není definováno. Ovšem v případě uvažování čtvercových pixelů a vzhledem k možnosti hledání cesty ve třech směrech by sklon neměl přesáhnout odchylku $\pm 45°$ od svislice.

5.5.3 Aktualizace šířky oblasti zájmu

Také šířka oblasti zájmu záleží na vlastnostech detekovaného objektu. Jelikož se nepředpokládá výrazná změna tvaru objektu oproti stávajícímu, vychází šířka oblasti z aktuální šířky objektu. Tou je myšlena jeho maximální šířka s_a, která se vypočítá následujícím způsobem:

$$n_R = \min_{i \in C_R^l} \left(W_h(i) - R_h^l(i) \right) \ , \tag{5.25}$$

$$m_R = \max_{i \in C_R^l} \left(W_h(i) - R_h^l(i) \right) \ , \tag{5.26}$$

$$s_a = m_R - n_R \ , \tag{5.27}$$

kde C_R^l označuje množinu všech řádků i, na kterých je definován levý okraj oblasti zájmu. W_h je funkce přiřazující na každém řádku i horizontalní souřadnici cesty. Funkce R_h^l přiděluje řádku i horizontální pozici levého okraje oblasti zájmu.

Nová šířka oblasti zájmu s_n je definována:

$$s_n = s_a + \kappa \,, \tag{5.28}$$

kde κ značí kladnou konstantu, jejíž hodnota přidává okraj oblasti zájmu. Ta zaručí možnost hledání i širších objektů oproti objektu aktuálně nalezenému.

5.6 Aktualizace vzoru

Volba vhodné aktualizace vzoru může mít zásadní vliv na kvalitu a přesnost algoritmu. Existují různé způsoby jakým vzor aktualizovat. Zde jsou popsány dva netriviální způsoby aktualizace vzoru. V prvním případě se jedná o exponenciální zapomínání a v druhém o průměrování hodnot přes různé pozice.

Exponenciální zapomínání

Exponenciální zapomínání je proces, který aktualizuje uložený vzor s pomocí zapomínajícího parametru β a nově nalezené odpovídající části stávajícího obrazu. Aktualizace uloženého vzoru $T(\mathbf{x_i})$ odpovídajícími hodnotami na pozici $\mathbf{p}$ v obraze I je provedena způsobem

$$T(\mathbf{x_i}) = T(\mathbf{x_i}) + \beta \cdot (I(G(\mathbf{x_i}, [\mathbf{p}; \alpha])) - T(\mathbf{x_i})) \,. \tag{5.29}$$

Parametr β, který určuje míru zapomínání, je nutné volit v intervalu $(0; 1)$.

Výhodou exponenciálního zapomínání je:

1. *Volitelnost parametru β*
   Nastavení dle výsledků experimentů. Případně lze tuto hodnotu přepočítávat dynamicky při běhu algoritmu (např. závislá na hodnotě kriteriální funkce pro oblast na pozici $\mathbf{p}$)

2. *Plynulá přeměna originálního vzoru*
   Model je aktualizován plynule, neskokově. To poskytuje částečnou bezpečnost proti aktualizování vzoru nevhodnou oblastí. Nevhodnou ve smyslu oblast s vysokou hodnotou kriteriální funkce O.

Na druhou stranu se může jevit exponenciální zapomínání v některých případech jako nedostatečně rychle adaptabilní, což v důsledku vede na zvýšení hodnot kriteriální funkce O.

Toto částečně řeší aktualizace vzoru prostřednictvím průměrných hodnot přes odpovídající oblasti v obraze.

Průměrování hodnot přes odpovídající oblasti v obraze

Je proces aktualizace, který nahradí stávající vzor novým vzorem, který je aritmetickým průměrem zvolených N odpovídajících oblastí. Matematicky lze proces aktualizace vzoru na pozici $\mathbf{x_i}$ odpovídajícími body na pozicích $\mathbf{p_k}$ popsat rovnicí

$$T(\mathbf{x_i}) = \frac{1}{N} \cdot \sum_{k=1}^{N} I(G(\mathbf{x_i}; [\mathbf{p_k}, \alpha])) \,. \tag{5.30}$$

V oblasti detekování lineárních objektů lze vhodné oblasti např. rozmístit rovnoměrně podél aktuálně extrahovaného objektu.

Výhodou tohoto způsobu aktualizace je, že je udržován vzor podobný všem vybraným oblastem určených pozicemi $\mathbf{p_k}$ a nikoliv pouze jednomu konkrétnímu vzoru, jako tomu je u exponenciálního zapomínání.

Kapitola 6

Implementace navržené metody pro sledování lineárních objektů v C++

Tato sekce se zabývá popisem implementace algoritmu pomocí frameworku BUDEČ a popisem ovládání algoritmu prostřednictvím uživatelského rozhraní. Framework BUDEČ je vytvořený Ing. Lukášem Cermanem a je dostupný na pracovišti CMP.

Framework slouží pro podporu psaní aplikací v jazyce C++, které se zabývají zpracováním videa. BUDEČ je funkční video aplikací zpracovávající data ze *zdroje videa* (tím může být živý zdroj nebo video soubor) prostřednictvím `procesoru snímků` a jeho následné `renderování`, buď na obrazovku nebo do video souboru.

Framework umožňuje programátorovi implementovat tzv. `filtry` a umístit je do `procesoru snímků` a tím transformovat, či jinak zpracovávat vstupní videosignál ve výstupní.

`Filtry` pracují nad tzv. `framem`, který může obsahovat vedle originálního snímku poskytnutého `zdrojem videa` ještě další, různě transformovaná obrazová data. Data jsou ukládána do tzv. `slotů`. Každý `filtr` definuje, z jakého slotu mají být použita vstupní obrazová data pro daný `filtr`.

V následující části jsou popsány filtry implementované pro tento algoritmus.

6.1 Implementované filtry ve frameworku BUDEČ

Při implementaci algoritmu pro sledování lineárních objektů jsou použity čtyři filtry. Filtry jsou popisovány v pořadí, v jakém se nacházejí v `procesoru snímků`.

Filtr pro převod originálního RGB obrázku do šedotónového

a) b)

Obrázek 6.1: *a*) vstupní snímek, *b*) transformovaný snímek.

Obecně je možné pracovat přímo nad barevným snímkem, ovšem při prvotním experimentování s algoritmem jsem se rozhodl používat šedotónový obrázek. V případě rozšíření algoritmu tak, aby pracoval nad barevným obrázkem, se výsledky detekce určitě nezhorší. Barevný kanál poskytuje více obrazových informací, což vede ke zvýšení přesnosti sledování. Ovšem délka výpočtu algoritmu se při přechodu z šedotónového snímku na RGB kanál zvýší průměrně $3\times$.

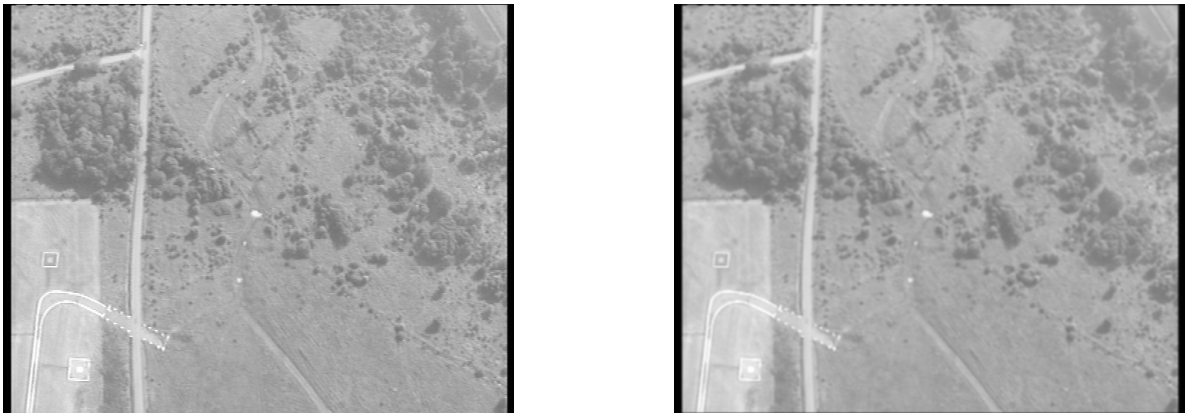
Transformace RGB obrázku s barevnými složkami r, g a b do šedotónového g_g lze pro každý pixel $X_{i,j}$ popsat vztahem

$$g_g(X_{i,j}) = 0.3 \cdot r(X_{i,j}) + 0.59 \cdot g(X_{i,j}) + 0.11 \cdot b(X_{i,j})\,. \tag{6.1}$$

Vstupem filtru jsou originální obrazová data (barevný snímek). Transformovaný, šedotónový, snímek je uložen do nového slotu a připraven pro další použití. Vstupní snímek a transformovaný snímek, se kterými pracuje tento filtr, jsou znázorněny na obrázku 6.1.

Filtr pro potlačení šumu

Pro potlačení šumu v obraze je použit Gaussovský filtr. Potlačení šumu lze dosáhnout konvolucí (značené $*$) obrazové funkce I s konvolučním jádrem v podobě Gaussovské



Obrázek 6.2: *a*) vstupní snímek, *b*) transformovaný snímek.

masky. Konvoluce dvourozměrné obrazové funkce s konvoluční maskou o rozměrech $[2 \cdot k + 1 \times 2 \cdot k + 1]$ má tvar

$$I(x, y) * h(x, y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} I(x + i, y + j) \cdot G(i, j)\,. \tag{6.2}$$

Zde je použita následující konvoluční maska G

$$G = \frac{1}{159} \begin{pmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{pmatrix}. \tag{6.3}$$

Potlačení šumu a rozostření snímku povede k vyhlazení obrazové funkce a stane se plynulejší, což povede na lepší výsledky porovnávání vzoru s částí obrazu a tím i k lepšímu sledování. Obrázek 6.2 demonstruje, s jakými snímky filtr pracuje.

Vstupem filtru je šedotónový obrázek vzniklý v předešlém snímku (Obrázek 6.2 *a*). Obrázek rozostřený Gaussovskou konvoluční maskou (dále uváděn pouze jako rozostřený snímek) je uložen ve volném slotu pro další použití (Obrázek 6.2 *b*).

a) b)

Obrázek 6.3: *a*) snímek pro hledání nejlevnější cesty,
b) snímek pro vykreslení nalezené cesty.

Filtr zajišťující hlavní činnost algoritmu

Vstupními daty pro tento obrázek jsou data z rozostřeného snímku a data originálního barevného obrázku. Filtr nejprve pracuje nad rozostřeným snímkem, kde hledá v definované oblasti zájmu cestu s minimální cenou. Nalezenou cestu poté vykreslí do původního barevného snímku, jak lze pozorovat na obrázku 6.3.

Filtr pro vykreslení polohy vzoru a oblasti zájmu

Do filtru vstupuje originální barevný snímek s již zvýrazněnou nejlevnější cestou. Úkolem filtru je do snímku, před jeho renderováním na obrazovku (do souboru), zakreslit polohu vzoru a oblasti zájmu. Parametry vzoru a oblasti zájmu jsou vykresleny dle výpočtů uvedených v kapitole 5.5 a 5.6. Vstupní snímek a snímek vzniklý jeho upravením, který bude renderován, znázorňuje obrázek 6.4

6.2 Implementace ovládání algoritmu

Pro ovládání algoritmu byl implementován vlastní *plugin* do uživatelského rozhraní standardního frameworku, pro možnost ovládání algoritmu přímo z uživatelského rozhraní. *Plugin* poskytuje operátorovi v inicializační části následující možnosti:

Obrázek 6.4: *a*) vstupní snímek *b*) konečný renderovaný snímek

- *Označit část snímku jako reprezentativní část objektu.*
  Lze docílit kliknutím `levého tlačítka myši` kdekoliv po snímku.

- *Měnit velikost reprezentativní části objektu - vzoru.*
  Pomocí tlačítek `Patch +` (zvětšení oblasti pro vzor) a `Patch -` (zmenšení oblasti pro vzor).

- *Měnit šířku oblasti zájmu.*
  Změna šířky oblasti zájmu je docílena tlačítky `RoI +` (zvětšení šířky oblasti zájmu) a `RoI -` (zmenšení šířky oblasti zájmu).

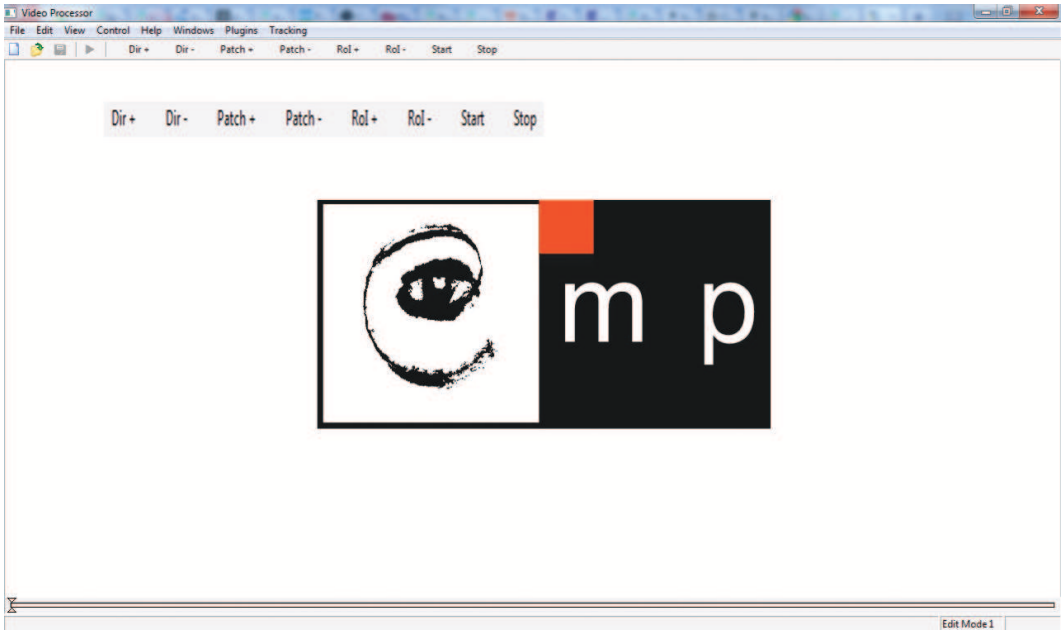
- *Změnit sklon oblasti zájmu.*
  Sklon oblasti zájmu lze určit buď kliknutím `pravého tlačítka myši` kdekoli v obraze nebo tlačítky `Dir +` (zvětšování úhlu) a `Dir -` (zmenšování úhlu). Sklon oblasti zájmu pomocí kliknutí myši je vypočítán stejným způsobem jako je popsáno v kapitole 5.5 v odstavci věnovaném aktualizaci sklonu oblasti zájmu. Ovšem s tím rozdílem, že souřadnice $\mathbf{s}$ značí pozici středu vzoru a $\mathbf{c}$ jsou souřadnice bodu označeného pravým kliknutím myši.

- *Zahájit sledování objektu.*
  Zahájit sledování objektu v oblasti zájmu s označenou reprezentativní částí ob-

jektu lze pomocí tlačítka `Start`.

- *Ukončit sledování objektu.*
  Sledování lze ukončit stiskem tlačítka `Stop`.

Grafické rozhraní se zvýrazněnými tlačítky pro ovládání implementovaného algoritmu ukazuje obrázek 6.5



Obrázek 6.5: Grafické rozhraní frameworku s implementovaným pluginem, zobrazen pouze ilustrační obrázek loga CMP.

Kapitola 7

Experimenty

Součástí této práce je ověření funkčnosti navrženého algoritmu, včetně zhodnocení výsledků a zdokumentování případů, kdy metoda selhává, či pracuje chybně. Na obrázku 7.1 jsou uvedeny příklady správné detekce z video sekvencí, s nimiž bylo pracováno.

Experimentováno je s více modifikacemi algoritmu s cílem zdokumentovat úspěšnost každého z nich. Modifikací se rozumí použití různé techniky pro aktualizaci vzoru a použití více vzorů vzniklých otočením vzoru o úhel α.

Jako měřítko úspěšnosti sledování objektu je zde uvažována odchylka polohy nalezeného objektu od skutečné pozice objektu. Skutečná pozice objektu je reprezentována souřadnicemi obou jeho konců $\hat{\mathbf{s}} = (\hat{s_x}, \hat{s_y})^\top$ a $\hat{\mathbf{c}} = (\hat{c_x}, \hat{c_y})^\top$. Data popisující skutečný objekt se nazývají *Ground truth data*.

7.1 Získání Ground truth dat

Získání *Ground truth dat* probíhá manuálně a spočívá v označení koncových bodů $\hat{\mathbf{s}}$ a $\hat{\mathbf{c}}$ objektu. V tomto případě poslouží pro získání *Ground truth dat* algoritmus, který prochází chronologicky celý video soubor snímek po snímku. Na každém snímku $\hat{f}$ nechá uživatele označit počáteční bod $\hat{\mathbf{s}}$ a koncový bod $\hat{\mathbf{c}}$ objektu.

Výstupem celého algoritmu tak bude množina trojic $\{\hat{f}, \hat{\mathbf{s}}, \hat{\mathbf{c}}\}$.

Algoritmus pro získání *Ground truth dat* lze naleznout v příloze (GroundTruthDataMiner.m). Jeho jediným rozdílem oproti výše popsanému algoritmu je ten, že nena-

Obrázek 7.1: Příklady správné detekce lineárních objektů.
Rovnoběžné červené čáry ohraničují oblast zájmu.
Červený čtverec uvnitř oblasti zájmu reprezentuje oblast nejvíce podobnou vzoru.

čítá celý video soubor, který by musel dále dekomponovat na snímky, ale pracuje již s vyextrahovanými snímky, které byly ve formátu jpg k dispozici na pracovišti CMP.

Tímto způsobem bylo označeno v pěti sekvencích 817 snímků. Ovšem ne na každém snímku lineární objekt splňuje požadavky na umístění kladené navrženou metodou.

7.2 Experimenty s různými modifikacemi

Modifikacemi jsou rozuměny varianty algoritmů skládající se z různých kombinací použitých funkcí o a použitých vzorů v závislosti na pootočení α.

Použité funkce o:

1. *SSD funkce (zn. SSD).*
2. *Funkce s necitlivostí ϵ (zn. ϵ).* Práh ϵ je experimentálně nastaven na hodnotu 25.

Použité vzory vzhledem k pootočení α

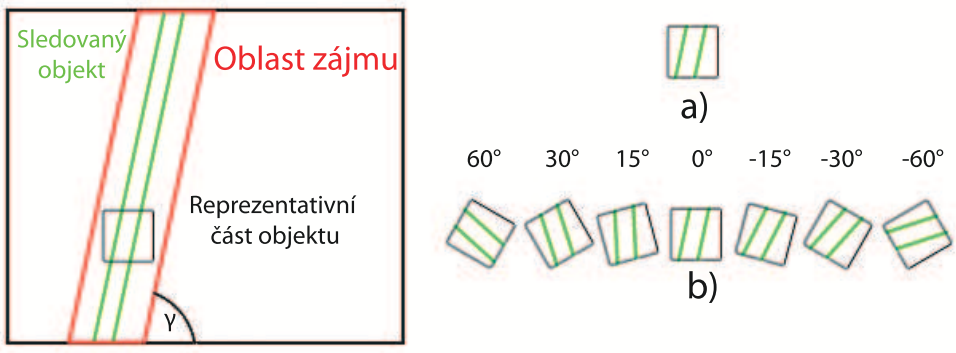
V této práci jsou použity čtyři způsoby, které lze dělit do dvou skupin. První skupinou jsou statické vzory (varianta A) a druhou skupinou jsou vzory adaptující se dle sklonu oblasti zájmu (varianta B).

Statické vzory (varianta A)

Využívají:

1. *Jeden vzor (varianta 1A)* s $\alpha = (0)^\top$.
2. *Sedm vzorů (varianta 7A)* s $\alpha = (-60, -30, -15, 0, 15, 30, 60)^\top$.

Během běhu algoritmu se nemění jejich úhly pootočení. Vzor je získán při inicializaci a dále se nemění. Jaké vzory jsou získány,demonstruje obrázek 7.2, kde jsou znázorněna



Obrázek 7.2: Statické vzory:
a) jeden vzor (varianta 1A), *b*) sedm vzorů (varianta 7A).

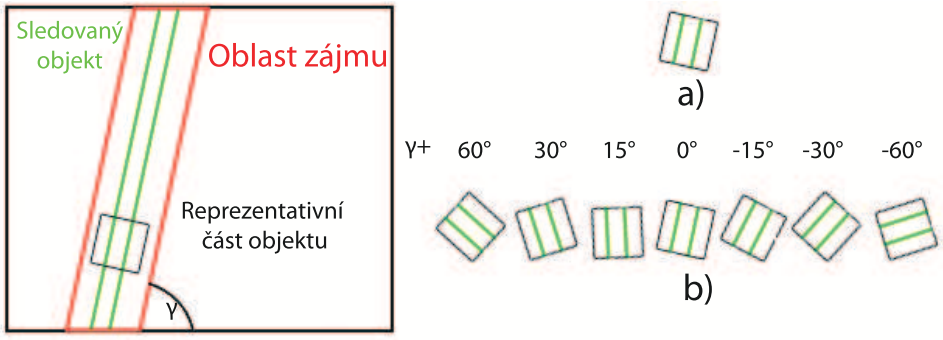
otočení vzorů při použití a) právě jednoho vzoru (1A) nebo b) sedmi vzorů (7A).

Vzory adaptující se sklonu oblasti zájmu (varianta B)

Dělí se na varianty využívající:

1. *Jeden vzor (varianta 1B)* s $\alpha = (\gamma)^\top$, kde γ je úhel sklonu oblasti zájmu.
2. *Sedm vzorů (varianta 7B)* s $\alpha = \gamma + (-60, -30, -15, 0, 15, 30, 60)^\top$.

Tyto vzory se přizpůsobují sklonu oblasti zájmu. Vzor je získán při inicializaci a je natáčen podle aktuálně vypočítaného sklonu oblasti zájmu. Získané vzory, jejichž otočení je přizpůsobeno sklonu oblasti zájmu, popisuje obrázek 7.3, kde a) demonstruje



Obrázek 7.3: Vzory adaptující se sklonu oblasti zájmu:
a) jeden vzor (varianta 1B), b) sedm vzorů (varianta 7B).

natočení vzoru a použití právě jednoho vzoru (1B) a b) znázorňuje sedm vzorů (7B) otočených dle určených úhlů.

Hodnoticí kritérium Λ

Kritérium udávající úspěšnost sledování objektu pro každou modifikaci algoritmu lze popsat následujícím vztahem

$$\Lambda = \frac{1}{N} \cdot \sum_{i=0}^{N} \left(\max \left(\{ |\mathbf{s_i} - \hat{\mathbf{s}}_\mathbf{i}|, |\mathbf{c_i} - \hat{\mathbf{c}}_\mathbf{i}| \} \right) < \Phi \right), \tag{7.1}$$

kde N značí počet snímků, na kterých jsou porovnávány souřadnice $\mathbf{s}$ a $\mathbf{c}$ získané při běhu algoritmu se souřadnicemi *Ground truth dat* $\hat{\mathbf{s}}$ a $\hat{\mathbf{c}}$ na odpovídajícím snímku i.

Kritérium Λ tedy vypovídá, v kolika případech ze všech algoritmus správně detekoval objekt. Hranice Φ byla nastavena na hodnotu 25, což je vzhledem k šířce objektu ve videosekvencích, se kterými se pracovalo, odpovídající hodnota.

Tyto experimenty používaly pro aktualizace vzoru exponenciální zapomínání.

Výsledky experimentů

Experimenty byly prováděny na pěti nezávislých videosekvencích získaných na pracovišti CMP, pro získání co nejobjektivnějších výsledků. Těchto pět sekvencí dohromady činí 792 snímků, na kterých je sledován lineární objekt. V pořízených sekvencích se jako lineární objekt vyskytují pouze silnice či polní cesty.

Experiment je proveden jednotlivě pro každou modifikaci a následně pouze v závislosti na volbě funkce o.

Výsledky pro každou modifikaci

Výsledné hodnoty kritérií Λ pro jednotlivé modifikace algoritmu popisuje tabulka 7.1. Tabulka 7.1 ukazuje průměrnou snímkovací frekvenci (fps) pro jednotlivé modifikace

Tabulka 7.1: Kritérium Λ pro každou modifikaci.

	$1A$	$7A$	$1B$	$7B$
SSD	0.562	0.706	0.553	0.722
ϵ	0.665	0.628	0.766	0.697
fps	2	16	2	16

Dle kritéria Λ je tak nejlepší modifikací algoritmu varianta, která pracuje nad jedním vzorem adaptujícím se dle sklonu oblasti zájmu s funkcí s necitlivostí ϵ kriteriální funkcí. Tato modifikace dosahuje úspěšnosti sledování přes 76 %. To vede k závěru, že s rostoucím počtem porovnávaných vzorů nemusí být docíleno lepších výsledků. To je optimistický závěr, neboť s rostoucím počtem porovnávaných vzorů rostou také výpočetní nároky algoritmu - to je patrné z uvedených snímkovacích frekvencí.

Z tabulky 7.1 lze usoudit, že lepších výsledků, v závislosti na použité variantě porovnávání vzoru, dosahuje varianta B. Ta dosahuje vždy lepších výsledků, kromě jednoho případu, kdy jsou výsledky téměř shodné.

Výsledky vzhledem ke zvolené funkci o

Nad stejnými daty jako v předchozím případě byl vykonán experiment, jehož výsledky byly vztažené ke zvolené funkci o. Hodnoty kritéria Λ pro použité funkce o popisuje tabulka 7.2 Lepších výsledků téměř o 6% dosahuje funkce s necitlivostí ϵ, jejíž úspěšnost

Tabulka 7.2: Kritérium Λ vzhledem k funkci o.

	SSD	ϵ
Λ	0.633	0.689

sledování činí bezmála 69 %.

7.3 Experiment s aktualizací vzoru průměrováním

Cílem experimentu je porovnat vliv aktualizace vzoru na přesnost a kvalitu detekce.

Zatímco aktualizace vzoru pomocí exponenciálního zapomínání disponuje velice dobrými výsledky, jak lze vidět v předešlé sekci, u aktualizace vzoru průměrováním přes zvolené oblasti tomu tak není. Implementovaná metoda aktualizuje vzor oblastmi ležícími podél nalezené cesty W. I tak je výsledný vzor velkou mírou ovlivněn případnou špatnou detekcí a sledovaný objekt je lehce ztracen.

Jistým zlepšením v tomto ohledu bylo zvětšení vzoru (z používaných $[30 \times 30]$ na nových $[60 \times 60]$ obrazových bodů). Což ve vzoru zvýraznilo sledovaný objekt a algoritmus ho dokázal lépe sledovat.

Dalšího pokroku bylo docíleno výběrem vhodných oblastí pro průměrování. Pouze takových, které se od aktuálního vzoru neliší více než o definovanou mez. I tak ovšem úspěšnost sledování objektu nepřesáhla 46 %.

Z důvodů neuspokojivých výsledků sledování s využitím tohoto typu aktualizace vzoru bylo experimentováno pouze s jednou konfigurací, a to s konfigurací $\{SSD, 1B\}$, která vedla k výše zmíněné úspěšnosti.

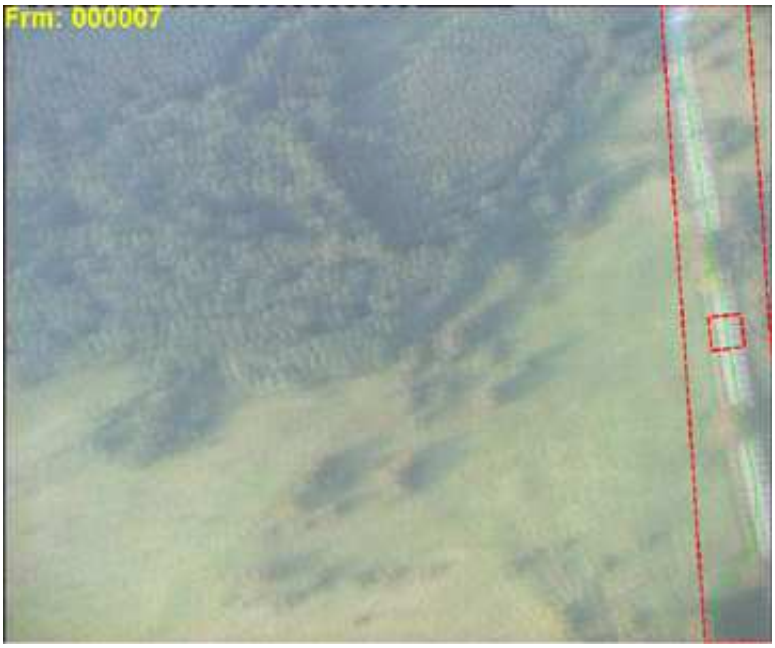
7.4 Případy selhání algoritmu

Tato část je zaměřena na popsání případů a příkladů objektů, u kterých navržená metoda selhává a nedokáže daný objekt sledovat.

Požadavky na tvar a polohu objektu

Zde plynou omezení již ze samotné implementace algoritmu pro hledání nejlevnější cesty. Algoritmus nedokáže správně detekovat objekty, které:

1. *Nezačínají na dolním okraji oblasti zájmu.* Příklad je na obrázku 7.4, kde byl



Obrázek 7.4: Objekt nezačíná na dolním okraji oblasti zájmu.

   očekáván začátek objektu na dolní hraně oblasti zájmu, a z toho důvodu je dolní konec objektu detekován špatně.

2. *Neprocházejí každým obrazovým řádkem pouze jednou (vracejí se).* Situaci demonstruje obrázek 7.5, kde detekce objektu selhala, neboť cesta se v obraze vracela. Algoritmus tak byl schopen detekovat pouze tu část objektu, která postupovala směrem k hornímu okraji oblasti zájmu.

Obrázek 7.5: Objekt neprochází každým obrazovým řádkem pouze jednou.

3. *Mají velký sklon (změna jejich horizontální pozice na dvou sousedních řádcích je více jak jeden obrazový bod).* Příklad selhání detekce ukazuje obrázek 7.6, kde algoritmus pro nalezení nejlevnější cesty (viz. 5.3 počítal chybně cenu na ak-

Obrázek 7.6: Objekt má příliš velký sklon.

tuální pozici, kvůli nedostatečnému počtu předchůdců (využíval neinicializované předchůdce). Toto lze potlačit otáčením směru průchodu dynamického programování podle směru oblasti zájmu.

Požadavky na diskriminativitu objektu

Dále algoritmus selhává, pokud nelze objekt dostatečně odlišit od pozadí, či od jiného objektu. Příklad nedostatečně odlišeného objektu od pozadí demonstruje obrázek 7.7.

Obrázek 7.7: Objekt není dostatečně odlišen od pozadí.

Kapitola 8

Zhodnocení

V této bakalářské práci byla navržena metoda pro sledování lineárních objektů z videosekvencí pořízených letícím pozorovatelem. Cílem bylo navrhnout obecnou, robustní metodu schopnou detekovat libovolný lineární objekt v reálném čase. Za tímto účelem je využit přístup založený na porovnávání vzorů, jenž dostatečně splňuje kladené požadavky. Poloautomatické metody založené na porovnávání vzorů splňují požadavky tím, že je lze jednoduše inicializovat (reálný čas) a zároveň jsou, díky možnosti získání libovolného vzoru, diskriminativní (robustnost) a zároveň obecné (obecnost).

Pro zvýšení robustnosti metody a rychlosti sledování objektu je objekt hledán pouze v malé části snímku, tzv. oblasti zájmu, jejíž parametry jsou definovány uživatelem při inicializaci.

Pro nalezení hledaného objektu je použit modifikovaný algoritmus hledající cestu nejvíce podobnou vzoru procházející mezi dolním a horním okrajem oblasti zájmu. Rozšíření algoritmu spočívá v zavedení penalizační funkce, která zajistí komparabilitu cen. Komparabilitu cen zajistí na pozicích, jenž se nacházejí na rozdílných řádcích snímku. To umožňuje hledat konec objektu nejen na horním okraji snímku, ale také po jeho krajích. To je značné rozšíření oproti základnímu algoritmu. Jistým omezením takto implementovaného algoritmu ale zůstává nutnost, aby objekt procházel dolním okrajem oblasti zájmu (nutné pro inicializační krok algoritmu hledající nejlevnější cestu).

Výsledná cena cesty na horním okraji oblasti zájmu je součtem hodnot kriteriální funkce O podél cesty. Ta uvádí, jak moc odpovídá daná oblast reprezentativnímu vzoru.

V další části bakalářské práce byla ověřena funkčnost navržené metody a testování

algoritmu s využitím různých technik. V testovaných videosekvencích nejlépe fungoval algoritmus využívající:

- Jeden vzor, jehož sklon se adaptuje sklonu oblasti zájmu.
- Kriteriální funkci s necitlivostí ϵ.
- Aktualizaci vzoru pomocí exponenciálního zapomínání.

Úspěšnost sledování této modifikace algoritmu činila $76,6\,\%$.

Naopak neočekávaně špatných výsledků dosáhl experiment s využitím aktualizace vzoru průměrováním přes oblasti ležících na nalezené cestě. Úspěšnost sledování v tomto případě činila necelých 46 %. Tím se jeví volba aktualizace vzoru exponenciálním zapomínáním jako vhodná pro danou aplikaci.

Experimentováno bylo i s modifikacemi porovnávající více vzorů (pootočených od původního vzoru o definované úhly). Ovšem i přesto, že tyto modifikace vybíraly v každé oblasti vzor nejvíce odpovídající dané oblasti, jejich úspěšnost sledování zaostala za výše popsanou modifikací.

To se může jevit jako optimistický výsledek, neboť s každým přidaným vzorem se zvyšuje časová náročnost algoritmu. A to zpravidla lineárně.

Náměty pro vylepšení

Námětem pro vylepšení této práce může být experimentovaní s dalšími geometrickými transformacemi vzorů. V praxi využitelná změna měřítka vzoru, která vykompenzuje změnu letové výšky pozorovatele, poskytne větší robustnost algoritmu.

Lepší variability bude docíleno náhradou algoritmu pro hledání nejlevnější cesty za algoritmus, který nebude klást požadavky na umístění objektu.

Dalším námětem pro vylepšení může být využívání více aktuálních vzorů získaných podél nalezeného objektu. To zvýší robustnost algoritmu v případě náhlé překážky (např. jedoucího auta) nebo náhlé změny vzhledu objektu (např. vjezd cesty do lesa).

Ještě vyšší robustnosti algoritmu může být docíleno tím, že cesta nebude znovu detekována v každém novém snímku. Pouze se po získání vzoru nalezne cesta jemu nejpodobnější, která se pro další snímky prohlásí za vzor. Tento vzor se v následujícím

snímku co nejlépe nasadí na oblast, která vzoru nejlépe odpovídá. Zbytek cesty se případně dopočítá dynamickým programováním.

Pro zvýšení robustnosti metody lze do stávající metody zapracovat upřednostňování lineárních objektů, které jsou nalezeny blízko lineárního objektu detekovaného v předešlém snímku.

Literatura

[1] Meir Barzohar and David B. Cooper. Automatic Finding of Main Roads in Aerial Images by using Geometric - Stochastic Models and Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:707–721, July 1996.

[2] Armin Gruen and Hailong Li. Semi-Automatic Linear Feature Extraction by Dynamic Programming and LSB-Snakes. *Photogrammetric Engineering and Remote Sensing*, 63(8):985–995, August 1997.

[3] Helmut Mayer, Ivan Laptev, and Albert Baumgartner. Multi-Scale and Snakes for Automatic Road Extraction. In Springer-Verlag Berlin Heidelberg 1998, editor, *European Conference on Computer Vision*, volume 1, pages 720–733, 1998.

[4] Seung-Ran Park and Taejung Kim. Semi-Automatic Road Extraction Algorithm from IKONOS Images Using Template Matching. *Paper presented at the 22nd Asian Conference on Remote Sensing*, November 2001.

[5] Vandana Shukla, R. Chandrakanth, and R. Ramachandran. Semi-Automatic Road Extraction Algorithm for High Resolution Images Using Path following Approach. In *Indian Conference on Computer Vision, Graphics and Image Processing*, December 2002.

[6] T. Svoboda, J. Kybic, and V. Hlavac. Image processing, analysis and machine vision: A matlab companion. 2008.

[7] Emanuele Trucco. Deformable/Active Contours (or Snakes). *Study material at the University of Nevada*, 2003.