VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

GEOREGISTRACE OBRAZU ZE STATICKÉ UAV PLATFORMY PRO LOKALIZACI POZEMNÍCH CÍLŮ

IMAGE REGISTRATION FROM STATIC UAV PLATFORM FOR GROUND OBJECTS LOCALIZATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **ADAM KUČERA**
AUTHOR

VEDOUCÍ PRÁCE **JAROSLAV ROZMAN, Ing. Ph.D.**
SUPERVISOR

BRNO 2020

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2019/2020

Zadání bakalářské práce

Student: **Kučera Adam**
Program: Informační technologie
Název: **Georegistrace obrazu ze statické UAV platformy pro lokalizaci pozemních cílů**
Image Registration from Static UAV Platform for Ground Objects Localization
Kategorie: Umělá inteligence

Zadání:

1. Nastudujte metody digitální stabilizace obrazu a georegistrace snímků.
2. Navrhněte vlastní robustní metodu pro georegistraci videa z ptačí perspektivy. Zaměřte se na odolnost vůči jasovým změnám.
3. Navrženou metodu implementujte. Proveďte testování algoritmu na několika reálných příkladech.
4. Vytvořte video a plakátek demonstrující vaši práci. Navrhněte možné pokračování vývoje.

Literatura:

- Aguilar, Wilbert G., Angulo, Cecilio: Real-time video stabilization without phantom movements for micro aerial vehicles, EURASIP Journal on Image and Video Processing, 2014, ISSN: 1687-5281

Pro udělení zápočtu za první semestr je požadováno:

- První dva body zadání.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Rozman Jaroslav, Ing., Ph.D.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 28. května 2020
Datum schválení: 31. října 2019

Abstrakt

Táto práca opisuje vývoj novej robustnej metódy na registráciu videa do spoločného priestoru. To je možné georegistrovať na satelitný obraz pomocou jednej lubovoľnej snímky. Vyvinutá vysokoúrovňová metóda je postavená na state-of-the-art nízkoúrovňových algoritmoch spracovania obrazu. Je robustná voči veľkým a/alebo skokovým zmenám svetelných podmienok v scéne a zmenám geometrie pohľadu. Problém globálnej chyby registrácie je prevedený na optimalizačný problém hľadania najkratšej cesty v grafe. Lokálna chyba je minimalizovaná fúziou dvoch prístupov ku stabilizácii videa.

Abstract

This paper describes development of new robust method for video registration into shared space. Then it is possible to georegister this video to satellite image using single arbitrary frame. Developed high-level method is based on state-of-the-art low-level image processing algorithms. It is robust to huge and/or instant changes in lighting conditions of the scene and changes in geometry of the view. Global error problem is converted to shortest path optimization problem. Local error is minimized via fusion of two approaches to video stabilization.

Klíčová slova

georegistrácia obrazu, UAV video, perspektívna transformácia, stabilizácia videa s využitím globálneho kontextu, registrácia obrazu, párovanie príznakov, lokálna fázová korelácia, optický tok, detekcia strihu vo videu

Keywords

image georegistration, UAV video, perspective transform, video stabilization using global context, image registration, feature matching, local phase correlation, optical flow, video cut detection

Citace

KUČERA, Adam. *Georegistrace obrazu ze statické UAV platformy pro lokalizaci pozemních cílů*. Brno, 2020. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Jaroslav Rozman, Ing. Ph.D.

Georegistrace obrazu ze statické UAV platformy pro lokalizaci pozemních cílů

Prohlášení

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Rozmana. Uviedol som všetky literálne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Adam Kučera
28. května 2020

Poděkování

Ďakujem môjmu vedúcemu J. Rozmanovi, Ing. Ph.D. za jeho čas a usmernenie vývoja.

Obsah

Kapitola 1

Úvod

Registrácia obrazu je jednou z mnohých podblastí počítačového videnia. Je to proces porovnávania zobrazení rovnakej scény z rôznych pohľadov a vo výsledku ich transformácia do spoločného súradnicového systému. V prípade georegistrácie je týmto spoločným priestorom geografický súradnicový systém. Digitálny obraz je len dvojrozmerné pole farebných hodnôt – pixelov. Procesom georegistrácie sa z týchto súradníc v obrazovej rovine stávajú geografické súradnice. V súčasnosti neexistuje žiaden univerzálny postup a jednotlivé riešenia sú závislé na možnostiach a obmedzeniach v konkrétnej aplikácii.

V tejto práci som sa venoval georegistrácii videa z malého bezpilotného lietadla (UAV), napr. dronu, vybaveného jedinou kamerou. Pomocou takéhoto videa je po georegistrácii možné na zachytenej scéne lokalizovať pozemné ciele a prípadne sledovať ich trajektórie, ak sa pohybujú. V praxi sa tieto dáta využívajú napr. na analýzu a následné optimalizácie v cestnej doprave. Je to lacné, jednoduché, požiteľné kdekoľvek a za predpokladu silného klasifikátoru a robustnej metódy pre georegistráciu, ktorej vývoju som sa v tejto práci venoval, aj veľmi spoľahlivé. Zameral som sa na odolnosť voči hlavným problémom, na ktorých súčasné prístupy zlyhávajú. Tými najväčšími sú:

1. Veľké a/alebo skokové zmeny svetelných podmienok
2. Veľké a/alebo skokové zmeny pohľadu
3. Pohyblivé a/alebo generické objekty v scéne

V súčasných prístupoch sa najprv celá video sekvencia registruje do spoločného referenčného priestoru, ktorý ešte nieje georegistrovaný. Ten sa následne georegistruje manuálne pomocou vhodne zvolenej referenčnej snímky. Automatická registrácia na satelitné snímky je v tomto prípade veľmi problematická, pretože tieto dva obrazy sú vo svojej povahe veľmi rozdielne. Videá z dronu sú totiž natáčané z malej výšky a zvyčajne pod uhlom, takže sa prejavuje perspektívne skreslenie. Limitujúcim faktorom však stále zostáva registrácia do spoločného referenčného priestoru, ktorá zlyháva na vyššie uvedených problémoch.

V tejto práci som sa zameral na dva ciele. Tým prvým a hlavným bol vývoj čo najrobustnejšej metódy pre registráciu do spoločného referenčného priestoru. Posledný krok samotnej georegistrácie ostáva síce manuálna, no zároveň triviálna záležitosť. Druhým cieľom bola rýchlosť – podľa možnosti sa čo najviac priblížiť k výpočtu v reálnom čase, čo by mohlo umožniť s určitými úpravami metódu použiť v real-time aplikáciách. Vďaka akcelerácii kritických výpočtov na grafickom procesore (GPU) a možnej predikcii pohybu sa mi nakoniec toto časové kritérium podarilo splniť.

Kapitola 2

Geometria rôznych pohľadov

Pretože registrácia obrazu je hľadanie transformácie medzi viacerými zobrazeniami jednej scény, je nevyhnutné nájsť vhodný geometrický model vyjadrujúci vzťah medzi týmito zobrazeniami.

2.1 Transformácie obrazu

Lineárna transformácia

Transformácia alebo zobrazenie je častým pojmom ako v matematike, tak v počítačovej grafike. Na tento abstraktný jav sa dá pozerať ako na funkciu, ktorá jednoznačne priraďuje ku každému bodu vo vektorovom priestore nejaký bod v inom vektorovom priestore. Najjednoduchšou transformáciou bodov v n-rozmernom priestore je linenárne zobrazenie, ktoré je definované maticou $n \times n$ prvkov. Medzi základné lineárne zobrazenia patrí napríklad zväčšenie, rotácia, skosenie alebo osová súmernosť. Rôzne lineárne zobrazenia je možné neobmedzene skladať pomocou operácie maticového násobenia, podľa reťazového pravidla v opačnom poradí, podobne ako pri skladaní matematických funkcii. Napríklad výraz 2.1 vyjadruje kompozíciu uniformnej λ - násobnej zmeny veľkosti a následnej rotácie proti smeru hodinových ručičiek o uhol γ.

$$A = \begin{pmatrix} \cos\gamma & -\sin\gamma \\ \sin\gamma & \cos\gamma \end{pmatrix} \cdot \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix} \tag{2.1}$$

Vďaka tejto vlastnosti je tiež možné jednoducho nájsť spätnú transformáciu ako inverznú maticu A^{-1}.

Afínna transformácia

Digitálny obraz je ako dvojrozmerné pole možné lineárne transformovať. Táto transformácia má však jedno veľké obmedzenie – viaže sa na stred súradnicovej sústavy. Posun alebo rotácia obrazu okolo ľubovoľného bodu nieje možná. Tu je potrebné rozšíriť základný priestor o jeden rozmer, avšak nie úplne. Ide o takzvaný lineárny podpriestor, pretože hodnota tohto nového rozmeru ostáva konštantná. Táto transformácia sa nazýva afínna.

Dá sa povedať, že afínna transformácia je lineárne zobrazenie rozšírené o t_x a t_y, t.j. horizontálny a vertikálny posun, resp. translácia. Zvyšná časť matice dedí všetky vlastnosti lineárneho zobrazenia. Pri premietaní vektoru je však nutné bod rozšíriť tiež o jeden rozmer,

aby bolo možne maticové násobenie. Schému tejto transformácie v dvojrozmernom priestore vyjadruje matica 2.2.

$$\begin{pmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2.2}$$

Projektívna transformácia

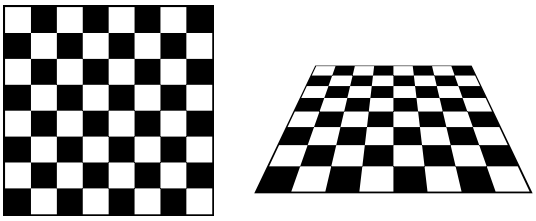
V prípade že je v obraze prítomná perspektíva, afínna transformácia ju nedokáže opísať. Transformácia zachytávajúca perspektívu sa nazýva projektívna. Ide v podstate o nadmnožinu afínnej transformácie – jediný rozdiel je v tom, že nieje obmedzená na lineárny podpriestor. Schému projektívnej transformácie v dvoch rozmeroch normalizovanú podľa posledného prvku vyjadruje matica 2.3.

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & 1 \end{pmatrix} \tag{2.3}$$

Na rozdiel od lineárnej a afínnej transformácie túto maticu už nieje možné deterministicky dekomponovať. Pri premietaní je nutné bod po prenásobení ešte normalizovať podľa posledného rozmeru, čiže napr. bod $[x, y]$ v rovine podľa rozmeru z. Vysvetlenia najmä 2D transformácii s množstvom príkladov je možné nájsť v literatúre o počítačovej grafike [7].

2.2 Homografia

V geometrii sa projektívna, prípadne afínna transformácia v minimálne dvojrozmernom priestore nazýva *homografia*. V počítačovom videní sa takto označuje vzťah medzi dvomi snímkami jednej roviny nachádzajúcej sa v trojrozmernom priestore, ako ilustruje obrázok 2.1. Uvažuje sa tzv. dierkový model kamery.[1]



Obrázek 2.1: Dva obrazy jednej roviny v 3D

V prípade leteckej platformy sledujúcej pozemné ciele je tým najvhodnejším modelom práve homografia, pretože povrch zeme sa z výšky vizuálne javí ako rovina. Existujú aj zložitejšie modely (napr. stereo videnie), tie sú ale výpočtovo náročnejšie a majú špeciálne požiadavky na to, aby sa vôbec dali použiť.

[1] https://staff.fnwi.uva.nl/r.vandenboomgaard/IPCV20162017/LectureNotes/CV/PinholeCamera/index.html#

Analytické riešenie

Pre nájdenie homografie medzi dvomi obrazmi je potrebné najprv nájsť dvojice zodpovedajúcich bodov a to presne 3 v prípade afínneho a 4 v prípade projektívneho modelu homografie. Tieto dvojice bodov definujú exaktné riešenie.

Metóda najmenších štvorcov

V praxi je kvalita týchto bodových korešpondecii v závislosti od postupu použitého k ich získaniu ďaleko alebo ešte ďalej od dokonalosti. Platí, že čím viac bodov, tým je väčšia pravdepodobnosť, že sa medzi nimi vyskytnú páry s vysokou kvalitou. Takto však vzniká preurčený systém (angl. overdetermined) a ten nemá exaktné riešenie.

V regresnej analýze sa zvyčajne na nájdenie modelu s najmenšou chybou voči dátam používa metóda najmenších štvorcov. Podobný prístup je možné aplikovať aj tu a to prostredníctvom algoritmu DLT (Direct Linear Transform) [5]. Riešenie je ďalej možné ešte vylepšiť algoritmom Levenberg-Marquardt [3]. Problém tohto prístupu spočíva v tom, že všetky body majú rovnaký vplyv na výsledné riešenie a ak chyba nemá normálne rozdelenie, odhadnutá transformácia sa môže výrazne líšiť od správneho riešenia.

Algoritmus RANSAC

V praxi neexistuje žiadna záruka, že chyba bude dokonale spadať do normálneho rozdelenia. V reálnych dátach sa často vyskytujú tzv. *ouliers*. Ide o body, ktorých chyba výrazne štandartnú odchýlku. Stále však platí prepoklad, že u väčšiny bodov bude chyba spadať do normálneho rozdelenia pravdepobnosti. Tieto body sa nazývajú tiež *inliers*.

Z tohto predpokladu vychádza algoritmus RANSAC (RANdom SAmple Consensus) [2], ktorý je základom pre celú skupinu algoritmov používaných tam, kde je predpoklad výskytu outliers. Ako názov vypovedá, hlavná myšlienka algoritmu je nájdenie konsenzu v dátach. RANSAC je iteratívny algoritmus a pozostáva z dvoch základných krokov:

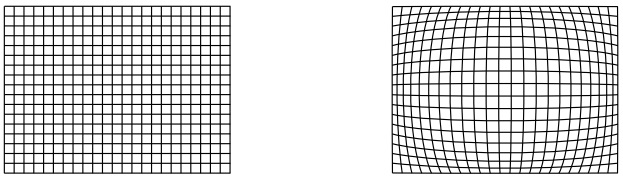
1. Výber vzorky n náhodných bodov z dát, kde n deterministicky definuje hľadaný model a následné zostavenie modelu z tejto vzorky.

2. Výpočet chyby pre každý bod v dátach voči tomuto modelu a zostavenie množiny konsenzu len z tých bodov, ktorých chyba neprekračuje stanovený prah.

Snahou je nájsť taký model, kde je veľkosť množiny konsenzu čo najväčšia. Kvalitu výstupu teda určujú dve veci – hodnotiaca funkcia určujúca chybu, a stanovený prah. Vhodná hodnotiaca funkcia v prípade homografie je tzv. chyba reprojekcie, kde sú body medzi obrazmi spätne premietané odhadnutou maticou. Čím bližsie bod z jedného obrazu padne k jeho korešpondecii v druhom, tým lepšie. Nájsť vhodnú hodnotu prahu je zvyčajne najlepšie empiricky. Pre dosiahnutie čo najlepšieho riešenia sa namiesto modelu z najlepšej vzorky výsledný model spočíta vyššie uvedenými metódami najmenších štvorcov z celej množiny konsenzu, t.j. inliers. Metrikou kvality odhadu je zvyčajne *pomer inlierov*, teda pomer veľkosti množiny konsenzu k počtu všetkých bodov v dátach.

Distorzia objektívu kamery

Dôsledkom optického dizajnu reálneho objektívu kamery je skreslenie, resp. *distorzia*. Ide o jav, kedy sa fyzicky rovné čiary javia ako zakrivené, ako ilustruje obrázok 2.2.



(a) Dokonalá mriežka (b) Mriežka s distorziou

Obrázek 2.2: Distorzia

Tento jav môže byť v závislosti od jeho intenzity pri hľadaní modelu homografie problém. Projekcie roviny sa pod distorziou nesprávaju v súlade s týmto modelom. Pri veľkej distorzii môžu byť odhadnuté modely veľmi nepresné a nestabilné. Distorzia je prítomná prakticky v každom obraze vyhotovenom fyzickou kamerou a preto je vhodným preprocesingom stabilizácie či georegistrácie videa odstránenie tejto distorzie.

Na odstránenie distorzie je potrebné získať vnútornú maticu kamery a jej distorzné koeficienty. Vnútorná matica kamery definovaná schémou 2.4 popisuje lineárny dierkový model kamery, kde f je ohnisková vzdialenosť a c je priesečník roviny obrazu s priamkou definujúcou natočenie kamery. Opisuje základnú charakteristiku kamery.

$$\begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2.4}$$

Distorzné koeficienty sú vektorom 5 až 14 reálnych čísel. Nulové koeficienty znamenajú model kamery bez distorzie. Postup hľadania vnútornej matice kamery a jej distorzných koeficientov sa nazýva *kalibrácia kamery* [6]. Pozostáva zo snímania šachovnicového vzoru z rôznych uhlov a analýzy projekcii rohových bodov tohto vzoru. Problém nemá analytické riešenie a nieje lineárny, používa sa preto metóda najmenších štvorcov realizovaná algoritmom Levenberg-Marquardt.

Po získaní vnútornej matice kamery a distorzných koeficientov už len stačí obraz na základe týchto parametrov transformovať tak, aby bola prítomná distorzia kompenzovaná.[2]

[2] https://docs.opencv.org/3.4/da/d54/group___imgproc___transform.html#ga69f2545a8b62a6b0fc2ee060dc30559d

Kapitola 3

Registrácia obrazu

Po zvolení vhodného modelu pre transformácie medzi obrazmi nasleduje hľadanie jeho parametrov, teda samotný proces registrácie obrazu. V terminológii sa obraz, z ktorého je transformácia hľadná, označuje ako *zdrojový* a obraz, do ktorého je z tohto zdrojového obrazu hľadaná, sa označuje ako *cieľový*. Metódy ktoré sa v súčasnosti používajú je možné rozdeliť do dvoch základných skupín – jasové (angl. intensity-based) a príznakové (angl. feature-based). Jasové metódy sú založené na korelácii medzi obrazmi ako celkom a to buď v priestorovej, alebo vo frekvenčnej doméne.[1] Tieto metódy sa pozerajú na obraz ako na signál a môžu dávať precízne výsledky, ale len v prípade, že sú si obrazy vizuálne dostatočne podobné a transformácie medzi nimi sú lineárne alebo afínne. Príznakové metódy sa pre zmenu nepozerajú na obraz ako na celok, ale len na jeho zaujímavé časti – príznaky. Nesústredia sa na to aký obraz je, ale na to, „čo na ňom je“. Vďaka tomu sú všeobecnejšie, robustnejšie a požiteľné aj pri hľadaní projektívnej transformácie. Pri snímkach meniacej sa scény s perspektívnym skreslením je teda vhodné použiť práve príznakové metódy. V tejto kapitole spomeniem tie najvýznamnejšie algoritmy, na ktorých okrem iného stojí aj moja metóda. Nakoniec spomeniem aj jednu jasovú metódu, ktorú som v modifikovanej verzii použil na vylepšenie výstupu z príznakových metód.

3.1 Vizuálne príznaky

V oblasti umelej inteligencie je príznak (angl. feature) veľmi dôležitý pojem pri riešení problémov klasifikácie či strojového učenia. Príznaky sú charakteristické črty, pomocou ktorých je možné dáta opisovať a porovnávať na vyššej úrovni abstrakcie. Napríklad kus oblečenia je možné definovať ako vektor dvoch príznakov – farba a typ.

V obrazových dátach sú objekty definované ich *hranami* a *rohmi*, teda časťami obrazu, kde dochádza k prudkej zmene hodnoty. Zvyčajne postačuje len samotná intenzita, obraz sa teda konvertuje do čiernobielej farebnej schémy, aby sa čo najviac urýchlil výpočet. Prudkú zmenu intenzity zachytáva napr. derivácia obrazu, preto sa v mnohých algoritmoch na detekciu príznakov počíta a to buď priamo Sobelovým operátorom, alebo sa nejakým spôsobom aproximuje.

[1] https://medium.com/vithelper/spatial-and-frequency-domain-image-processing-83ffa3fc7cbc

Harrisov a Shi-Tomasi algoritmus

Pri hľadaní bodových korešpondecii je najzaujímavejším príznakom taký, ktorý má charakter bodu a je nemenný voči rotácii či translácii. Takým je roh, ktorý sa dá definovať ako bod, v ktorého okolí existujú dva rôzne dominantné gradienty intenzity. Najznámejším detektorom rohov je Harrisov algoritmus [4]. Algoritmus najprv vypočíta parciálne derivácie intenzity obrazu podľa x a y. Následne pre každý bod výsledného gradientu vypočíta pomocou dvojrozmerného pohlyblivého okna definujúceho okolie tohto bodu štruktúrny tenzor, teda maticu o rozmeroch 2×2 odvodenú z tohto bodu a jeho okolia. Vlastné čísla λ štruktúrneho tenzoru opisujú prevládajúce zakrivenia gradientu v rámci daného okolia. Na základe ich hodnôt je možné vyvodiť nasledovné závery:

1. Ak obidve hodnoty λ sú malé, nejedná sa o zaujímavý bod.
2. Ak jedna hodnota λ je veľká a druhá malá, jedná sa o hranu.
3. Ak obidve hodnoty λ sú veľké, jedná sa o roh.

Pre Harrisov algoritmus je špecifickým jeho hodnotiace skóre, ktoré určuje kvalitu rohu. Nájdenie vlastných čísel matice štruktúrneho tenzoru A vyžaduje výpočet druhej odmocniny, čo je výpočtovo náročné. Harrisovo skóre preto využíva vzťah medzi vlastnými číslami, ktorý je spočítateľný výrazom 3.1 pozostávajúcim len z determinantu a stopy matice. Koeficient k sa odporúča nastaviť na 0.04.

$$M_c = \lambda_1 \lambda_2 - k\,(\lambda_1 + \lambda_2)^2 = \det(A) - k\ \mathrm{tr}^2(A) \tag{3.1}$$

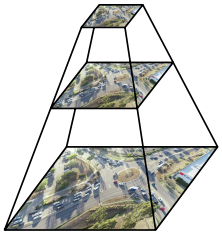
Toto skóre vyjadruje, do akej miery daný bod vykazuje vlasnosti rohu. Na podobnom princípe je založený ďalší detektor rohov známy ako Shi-Tomasi detektor alebo algoritmus GFTT [18]. Tento algoritmus je navrhnutý na detekciu najkvalitnejších rohov určených na stopovanie objektov v bezprostredne po sebe nasledujúcich snímkach videa. Algorimus zahadzuje nastavené percento bodov s najnižším skóre. Tiež v nastavenom okruhu ponecháva len lokálne najsilnejšie body, čím zabraňuje sústredeniu veľkého množstva bodov v malom regióne. Hoci algoritmus môže byť implementovaný s Harrisovým skóre, jeho autori zistili, že pri stopovaní objektov dáva lepšie výsledky skóre spočítané priamo z vlastných čísel ako ich najmenšia hodnota $min(\lambda_1, \lambda_2)$. Toto skóre sa zvykne označovať ako Shi-Tomasi skóre.

Algoritmus FAST

Novší algorimus FAST (Fast Accelerated Segment Test) [16] je veľmi zaujímavý najmä pre aplikácie, v ktorých je na prvom mieste rýchlosť. Tento detektor rohov nepočíta derivácie ale porovnáva intenzitu pixelov priamo. Pre každý bod obrazu definuje jeho okolie ako rasterizovanú kružnicu s polomerom 3 pixely a stredom v danom bode. Táto kružnica pozostáva zo 16 pixelov, každý z nich je označený číslenym indexom po smere hodinových ručičiek. Ak existuje v tomto zozname množina N súvislých po sebe nasledujúcich pixelov, pre ktoré platí buď že všetky sú svetlejšie ako pixel v strede kružnice plus zvolený prah, alebo že všetky sú tmavšie ako pixel v strede kružnice mínus zvolený prah, je daný bod klasifikovaný ako roh. Bežne používané hodnoty N sú 9 a 12. Kvalita nájdených bodov u algoritmu FAST je výrazne nižšia ako u pomalšieho algoritmu GFTT, autor však opísal ako je možné túto kvalitu zvýšiť s využitím strojového učenia.

Obrazová pyramída

Algoritmy na detekciu rohov posudzujú pevne dané okolie pixelov v obraze pevne s daným rozlíšením. Rohy ale môžu mať rôznu veľkosť. Príliš veľký roh môže byť v použitom okne vyhodnotený ako plocha s konštantným jasom. Požadovaná invariancia voči veľkosti, resp. rozlíšeniu je v praxi väčšinou realizovaná technikou známou ako *obrazová pyramída* 3.1.



Obrázek 3.1: Obrazová pyramída

Táto pyramída je tvorená z ľubovoľného počtu úrovní. Na každej úrovni je obraz z predchádzajúcej úrovne so zmenenou veľkosťou podľa nastavenej mierky. Na najspodnejšej úrovni je zvyčajne obraz v pôvodnom rozlíšení. Detekčný algoritmus sa vykoná na každej úrovni pyramídy.

3.2 Kľúčové body a ich deskriptory

Nájdené rohy alebo iné príznaky bodového charakteru sa označujú *kľúčové body*. Pri registrácii obrazu je potrebné nájsť nielen tieto body, ale v prvom rade zodpovedajúce dvojice týchto bodov v zdrojovom a cieľovom obraze. Na všeobecnú registráciu bez obmedzení existuje koncept lokálnych *deskriptorov* obrazu. Tieto deskriptory sú vektormi príznakov v n-rozmernom priestore. Vzdialenosť v tomto priestore príznakov určuje, ako veľmi sú si dva kľúčové body podobné. Najbližší susedia sú považovaní za najlepšiu zhodu.

Algoritmus SIFT

Algoritmus SIFT (Scale Invariant Feature Transform) [11] je mnohými považovaný za najrobustnejší algoritmus registrujúci prostredníctvom deskriptorov, pretože vykazuje vysokú invarianciu voči veľkosti a orientácii príznakov. Kvalitné kľúčové body hľadá metódou DoG (Difference of Gaussians), ktorá je blízkou aproximáciou výpočtovo náročnejšieho postupu LoG (Laplacian of Gaussians). Algoritmus vytvorí zoradenú päticu obrazov, každý z nich je o niečo viac rozostrenou verziou pôvodného obrazu. Rozostrenie sa realizuje konvolúciou Gaussovým filtrom. Medzi susednými obrazmi tejto pätice spočíta rozdiely v takom smere, že vždy je odčítavaný viac rozostrený obraz od menej rozostreného. Takto aproximuje sumu druhých parciálnych derivácii týchto rozostrených obrazov. Výsledná množina je označovná ako *scale-space*. Každý obraz v nej reprezentuje podľa miery rozostrenia určitú veľkosť príznakov. Detekcia stabilných kľúčových bodov pozostáva z hľadania lokálnych extrémov – maxima a minima. Každý pixel každého obrazu zo scale-space je porovnávaný zo svojím okolím nielen v aktuálnom, ale aj v susedných obrazoch v rámci scale-space. Takto sú produkované stabilnejšie kľúčové body, než aké produkuje väčšina iných algoritmov (napr. Harrisov detektor). Ešte väčšia invariancia voči veľkosti je dosiahnutá pomocou obrazovej pyramídy. Algoritmus DoG sa vykoná na každej úrovni pyramídy samostatne.

Pre každý nájdený kľúčový bod je jeho deskriptor spočítaný z veľkostí a orientácii gradientu v jeho okolí, ako znázorňuje obrázok 3.2. Okolie pixelu je rozdelené na 4 subregióny a pre každý z nich je vypočítaný histogram orientácii vektorov gradientu rozdeľujúci 360° na 8 intervalov. Hodnota intervalu je akumulovaná sčítavaním veľkostí týchto vektorov vážených Gaussovým oknom, ktoré na obrázku 3.2a znázorňuje modrá kružnica.

(a) Veľkosti a orientácie gradientu (b) Histogramy orientácii

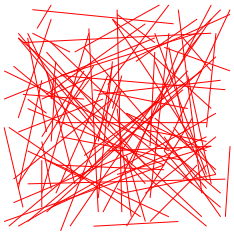
Obrázek 3.2: Výpočet SIFT deskriptoru

Hodnoty intervalov tejto štvorice histogramov tvoria výsledný deskriptor, teda vektor 64 reálnych čísel. Na hľadanie najbližších susedov sa používa Euklidovská vzdialenosť týchto vektorov. Pre invarianciu deskriptoru voči rotácii sa ešte vypočíta histogram orientácii vektorov gradientu z celého okolia bodu rozdeľujúci 360° na 36 intervalov. Hodnoty intervalov sú počítané rovnako ako pre histogramy deskriptoru. Nájde sa dominantný uhol a histogramy deskriptoru sú rotované o tento uhol. Algoritmus SIFT je síce veľmi robustný, je zároveň aj veľmi pomalý. Existuje jeho optimalizovaná verzia, algoritmus SURF (Speeded Up Robust Features), aj tento má však relatívne veľkú výpočtovú náročnosť.

Algoritmus BRIEF

Alternatívou k deskriptorom z gradientu obrazu sú binárne deskriptory. Ich extrakcia je výrazne rýchlejšia, pretože pozostáva z priameho porovnávania jasu pixelov, nieje počítaná derivácia. Hľadanie najbližších susedov je tiež podstatne rýchlejšie, pretože je používaná Hammingova vzdialenosť.[2] Na jej výpočet je potrebná len operácia XOR a funkcia na spočítanie jednotkových bitov, ktorá je niekedy implementovaná aj hardvérovo.

Algoritmus BRIEF (Binary Robust Independent Elementary Features) [1] je najznámejším algoritmom na výpočet binárnych deskriptorov. Funguje na jednoduchom princípe. Vygeneruje nastavený počet dvojíc z náhodne vybraných pixelov v okolí kľúčového bodu, ako je znázornené na obrázku 3.3. Tento počet je zvyčajne 128, 256 alebo 512.



Obrázek 3.3: Dvojice náhodne vybraných pixelov

[2] https://www.tutorialspoint.com/what-is-hamming-distance

Následne sú z týchto dvojíc vypočítané lokálne deskriptory ako bitové reťazce. Nech λ značí bitový reťazec a (p_1, p_2) značí intenzity pixelov z náhodne vygenerovanej dvojice. Pre každú dvojicu je vypočítaná hodnota príslušného bitu v λ podľa predpisu 3.2.

$$\lambda(p_1, p_2) = \begin{cases} 1 & \text{ak } p_1 < p_2 \\ 0 & \text{inak} \end{cases} \tag{3.2}$$

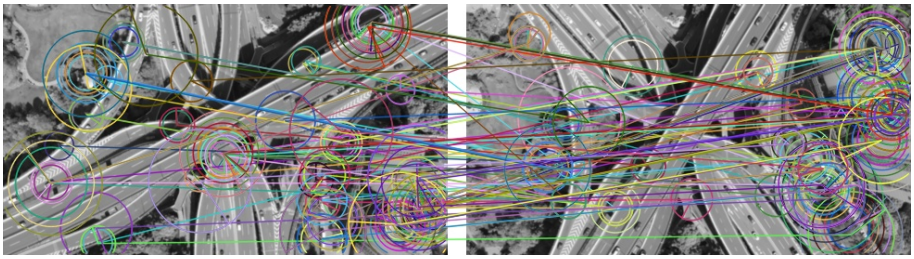
Algoritmus ORB

Narozdiel od komplexného algoritmu SIFT, algoritmus BRIEF vykonáva len extrakciu lokálnych deskriptorov. Detekciu kľúčových bodov je nutné realizovať iným špecializovaným algoritmom. Druhý nedostatok je tiež chýbajúca invariancia voči lineárnym transformáciám. Zložený algoritmus ORB (Oriented Fast Rotated Brief) [17] v sebe kombinuje exktraktor BRIEF s detektorom rohov algoritmom FAST a je považovaný za najrobustnejšiu alternatívu k algoritmu SIFT alebo SURF v aplikáciách zameraných na rýchlosť. Autori vo svojej práci uvádzajú, že môže byť až 100× rýchlejší než algoritmus SIFT, zároveň však v mnohých situáciách podáva rovnako kvalitné výsledky.

Invarianciu voči veľkosti algoritmus realizuje prostredníctvom obrazovej pyramídy. Pri algoritme SIFT vďaka metóde DoG stačil koeficient zmeny veľkosti 2. To znamená, že obraz na každej úrovni je obrazom z predošlej úrovne v polovičnom rozlíšení. V prípade algoritmu ORB je potrebný podstatne jemnejší prechod a na pokrytie špecifického rozsahu veľkostí je potrebné viac úrovní pyramídy. Detekcia rohov sa vykoná na každej úrovni algoritmom FAST. Na dosiahnutie invariancie voči rotácii je potrebné odhadnúť orientácie nájdených kľúčových bodov. Na to je použitý *centroid intenzity* [15], efektívna metrika, ktorá nevyžaduje výpočet derivácii obrazu. Tento centroid je vypočítaný z pixelov v okne definujúcom okolie rohu. Pre všetky pozície pixelov v okne je vypočítaný vážený priemer pre súradnicu x, kde váhou je normalizovaná intenzita obrazu na danej pozícii. Rovnako je vypočítaný aj vážený priemer pre súradnicu y. Centroid intenzity je bod $[x, y]$ zostavený z týchto dvoch priemerov. Za orientáciu rohu je považovaný uhol, ktorý zviera vektor smerujúci z rohu do tohto centroidu. Takto rozšírený algoritmus FAST autori nazvali oFAST.

Keď je známa orientácia kľučového bodu, je možné upraviť deskriptory BRIEF tak, aby bola dosiahnutá požadovaná invariancia voči rotácii. Pred výpočtom deskriptoru sa z vygenerovaných dvojíc pixelov zostaví matica, ktorej riadky sú súradnice x a y. Táto matica je transformovaná nájdenou maticou rotácie a z dvojíc definovaných týmito novými súradnicami je vypočítaný deskriptor, označovaný ako steered BRIEF. Vylepšením je algoritmus rBRIEF. Štatisticky má množina dvojíc pixelov pre výpočet deskriptoru dve žiadúce vlastnosti. Veľký rozptyl, ktorý zvyšuje rozlišovacie schopnosti deskriptoru, a nízka miera korelácie medzi dvojicami, ktorá zvyšuje počet bitov vplývajúcich na výsledok. Algoritmus rBRIEF aplikuje umelú inteligenciu a hladovým algoritmom (angl. greedy search) hľadá množinu dvojíc tak, aby minimalizoval koreláciu a maximalizoval rozptyl. Dvojice získané týmto algoritmom majú podstatne lepšie štatistické vlastnosti.

Obrázok 3.4 vizualizuje 100 párov príznakov nájdených algoritmom ORB. V závislosti od aplikácie sú bežne párované často až tisíce príznakov.



Obrázek 3.4: Korešpondencie príznakov nájdené algoritmom ORB

3.3 Optický tok

Metódy založené na lokálnych deskriptoroch a hľadaní najbližších susedov v priestore príznakov neprihliadajú na časový a fyzikálny kontext. Ak tento kontext nieje známy, sú dobrou voľbou. Ak tento kontext ale známy je, môže byť táto ich vlastnosť dokonca nevýhodou. V priestore príznakov môže byť bod nájdený ako najbližší sused falošná zhoda, a pretože neexistuje žiadne obmedzenie v priestore obrazu, bude táto korešpondencia akceptovaná. Tento problém čiastočne rieši algoritmus RANSAC. Všeobecne však platí, že výstup je podstatne viac zašumený než u metód, ktoré prihliadajú na časový a fyzikálny kontext.

Existuje predpoklad, že jeden kľúčový bod na dvoch v čase bezprostredne po sebe nasledujúcich snímkach bude mať rovnakú farebnú intenzitu. Tento koncept sa nazýva *optický tok*.[3] Opisuje zdanlivý pohyb objektov vo vizuálnej scéne. Vychádza z úvahy, že pre časový krok Δt bližiaci sa k nule sa bude zmena jasu objektu tiež blížiť k nule. Optický tok sa rozdeľuje na riedky a hustý. Hustý optický tok odhaduje pohyb všetkých pixelov obrazu, riedky sa sústreďuje len na pohyb príznakov, sleduje kľúčové body. Pre registráciu obrazu je preto vhodnejší riedky optický tok, ktorý je použiteľný na odhad pohybu kľúčových bodov v sekvencii videa.

V praxi sa optický tok používa na stopovanie objektov alebo vzájomnú registráciu po sebe nasledujúcich snímok. Výpočet optického toku je zvyčajne rýchlejší než extrakcia lokálnych deskriptorov a následné hľadanie najbližších susedov. Často sa preto využíva v real-time aplikáciách. V súčasnosti sa optický tok používa napríklad aj na real-time stabilizáciu videa z malého bezpilotného lietadla (UAV) [10], ide však len o vyhladenie prudkých pohybov a stabilizované video nieje možné georegistrovať pomocou ľubovoľnej snímky.

Algoritmus Lucas-Kanade

Myšlienku optického toku je možné matematicky vyjadriť rovnicou 3.3, kde I značí intenzitu bodu v čase a Δ značí zmenu.

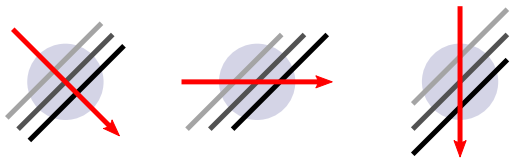
$$I\left(x, y, t\right) = I\left(x + \Delta x, y + \Delta y, t + \Delta t\right) \tag{3.3}$$

[3] https://nanonets.com/blog/optical-flow/

Matematickou analýzou je možné z tejto rovnice odvodiť vzťah 3.4, kde I_x, I_y a I_t značia parciálne derivácie intenzity podľa polohy a času, a V_x a V_y značia neznámu rýchlosť.

$$I_x V_x + I_y V_y + I_t = 0 \tag{3.4}$$

Problémom odhadu pohybu, resp. vektoru rýchlosti na základe optického toku je to, že táto rovnica obsahuje dve neznáme, a ako taká nieje riešiteľná. Tento problém je známy ako štrbinový problém (angl. aperture problem) a dá sa intuitívne ilustrovať na príklade pohybujúcej sa čiary pozorovanej skrze štrbinu, ktorá nedovoľuje vidieť koncové body čiary. Pohyb čiary je takto neurčiteľný, pretože ľubovoľný vektor pohybu tejto čiary bude produkovať identický vizuálny vstup, ako znázorňuje obrázok 3.5.

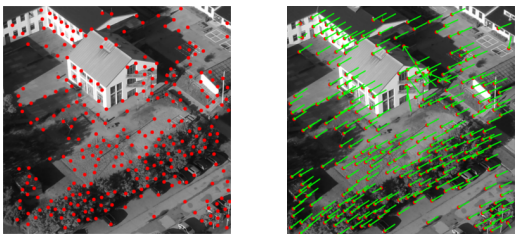


Obrázek 3.5: Problém neurčitosti pohybu

Pri odhade riedkeho optického toku je najčastejšie používaným riešením tohto problému algoritmus Lucas-Kanade [12]. Problém neurčitosti je riešený pridaním ďalšieho obmedzenia, ktorým je predpoklad, že posun pixelov v rámci okolia kľúčového bodu medzi dvomi v čase blízkymi snímkami bude malý a približne konštantný. Rovnica 3.4 by teda mala platiť pre všetky pixely v rámci okna definujúceho okolie bodu. Z neurčeného systému takto vzniká preurčený systém a ten je riešený metódou najmenších štvorcov. Odhad optického toku pre každé okno prebieha nasledovne. Z gradiendu obrazu je v rámci okna vypočítaný štruktúrny tenzor A. Vypočíta sa záporná suma súčinov parciálnej derivácie podľa polohy a času pre obidve súradnice polohy x a y. Z týchto súm sa zostaví stĺpcový vektor B. Výsledné riešenie vektoru rýchlosti je dané rovnicou 3.5.

$$\begin{pmatrix} V_x \\ V_y \end{pmatrix} = A^{-1} \cdot B \tag{3.5}$$

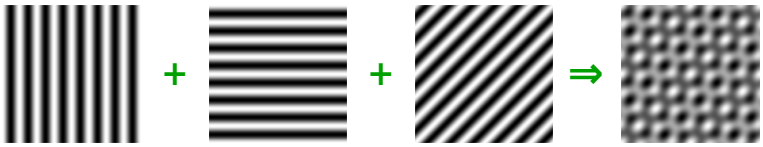
Invariancia voči veľkosti je dosiahnutá opäť použitím obrazovej pyramídy. Detekcia kľúčových bodov vhodných pre stopovanie je väčšinou realizovaná algoritmom GFTT s použitím Shi-Tomasi skóre. Na obrázku 3.6 je ukážka odhadnutých vektorov pohybu medzi snímkami.



Obrázek 3.6: Odhad optického toku metódou Lucas-Kanade

3.4 Spektrálna analýza

V priestorovej doméne je obraz definovaný (farebnými) hodnotami jednotlivých pixelov. Každý obraz je ale možné opísať aj vo frekvenčnej doméne ako súčet množiny funkcii $\sin(x)$ a $\cos(x)$ rôznych periód, amplitúd a orientácii. Na obraz sa dá teda pozerať ako na dvojrozmerný signál zložených frekvencii. Obrázok 3.7 znázorňuje kompozíciu troch rôzne orientovaných kosínusoviek o rovnakej frekvencii. Ich sčítaním vzniká interferenčný obrazec. Každý obraz, bez ohľadu na to ako zložito môže vizuálne pôsobiť, je v podstate takýmto interferenčným obrazcom zloženým z tisícov či miliónov kosínusoviek s rôznymi parametrami.



Obrázek 3.7: Kompozícia interferenčného obrazca

V praktických aplikáciách kde sú prítomné len afínne transformácie, ako napríklad lekárske zobrazovanie, sú metódy registrácie obrazu vo frekvenčnej doméne preferované. Je to najmä z toho dôvodu, že odhadujú transformácie na subpixelovej úrovni. Sú tiež rýchlejšie než ekvivalentne robustné jasové metódy v priestorovej doméne.

DFT obrazu

Na rozklad signálov do frekvenčnej domény sa používa diskrétna Fourierova transformácia alebo len skrátene DFT [19]. Táto lineárna transformácia konvertuje diskrétnu funkciu reálnych hodnôt na komplexnú diskrétnu funkciu frekvencii. Použitie komplexných čísel zabezpečuje invarianciu voči fázovým posunom. Reálna zložka reprezentuje funkciu $\cos(x)$ a imaginárna jej fázový posun o polovicu periódy, teda funkciu $\sin(x)$. Zvyčajne sa táto transformácia používa pri jednorozmerných signáloch v čase, akým je napríklad zvukový signál. DFT je ale abstraktná matematická myšlienka a voľnou premennou môže byť čokoľvek. V prípade obrazu sú voľné premenné dve – priestorové súradnice x a y. Matematický podklad z ktorého táto transformácia vychádza je pomerne rozsiahly, výsledný postup je však veľmi jednoduchý a ľahko implementovateľný sofvérovo alebo dokonca hardvérovo. Vďaka Eulerovej formule je pre obraz $f(x,y)$ o rozmeroch $N \times M$ transformácia do frekvenčnej domény $F(u,v)$ realizovateľná podľa rovnice 3.6.

$$F(u,v) = \sum_{y}\sum_{x} f(x,y) \cdot \left(\cos\left[2\pi\left(\frac{xu}{N}+\frac{yv}{M}\right)\right] - i \cdot \sin\left[2\pi\left(\frac{xu}{N}+\frac{yv}{M}\right)\right]\right) \tag{3.6}$$

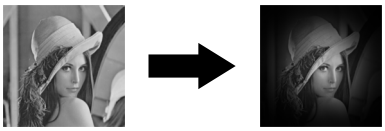
Fázová korelácia

Ak existuje medzi dvomi podobnými obrazmi translácia, je možné ju nájsť pomocou krížovej korelácie v priestorovej doméne. Krížová korelácia je veľmi podobná konvolúcii s tým rozdielom, že posúvaný signál nieje prevrátený. Najlepší odhad posunu medzi týmito dvomi signálmi je argument maxima výslednej funkcie. Normalizovaná krížová korelácia je vysoko invariantná voči zmenám jasu a odolná voči šumu, je však veľmi výpočtovo náročná. Vo frekvenčnej doméne je translácia medzi obrazmi pozorovateľná ako rôzne fázové posuny

jednotlivých frekvenčných zložiek. Metóda, ktorá hľadá transláciu medzi obrazmi vo frekvenčnej doméne, sa preto nazýva fázová korelácia. Je založená na Fourierovej transformácii (FT, nie DFT) krížovej korelácie, ktorá sa nazýva krížová spektrálna hustota. V prípade obrazu, teda dvojrozmerného diskrétneho signálu, je táto operácia opäť pomerne jednoduchá. Obidva obrazy sa najprv transformujú pomocou DFT. Pre druhý obraz, ktorého translácia voči prvému je hľadaná, je vypočítaný komplexný konjugát. To znamená, že imaginárna zložka je vynásobená číslom -1. Prvý obraz a tento komplexný konjugát druhého obrazu sú vynásobené pomocou Hadamardovho súčinu. Výsledkom tejto maticovej operácie je opäť matica, ktorej prvky sú definované výrazom 3.7. Matice A a B značia prvý obraz a komplexný konjugát druhého obrazu.

$$(A \circ B)_{ij} = A_{ij} \cdot B_{ij} \tag{3.7}$$

Každý prvok výslednej matice je normalizovaný sám sebou, teda vydelený jeho absolútnou hodnotou. Normalizovaná krížová korelácia v priestorovej doméne je získaná inverznou DFT vypočítanej krížovej spektrálnej hustoty. Translácia medzi obrazmi je nájdená ako argument maxima výslednej funkcie. Pre odstránenie ostrých okrajov signálu sa pred výpočtom DFT zvykne na signál aplikovať okno, ktoré zoslabí intenzitu signálu na jeho okrajoch.[4] Na obrázku 3.8 je znázornené použitie Hammingovho okna na obrazový signál. Podobné a často používané je tiež Hannovo okno.



Obrázek 3.8: Použitie Hammingovho okna

Pomocou fázovej korelácie je možné odhadnúť aj uniformnú zmenu veľkosti a rotáciu. V log-polárnej sústave súradníc sú totiž tieto dve transformácie obyčajnou transláciou [20]. Stačí obraz konvertovať z Karteziánskej sústavy do log-polárnej sústavy a odhadnúť transláciu fázovou koreláciou. Rotácia a logaritmus veľkosti výsledného vektoru budú definovať túto rotáciu a zmenu veľkosti medzi obrazmi.

[4] https://community.sw.siemens.com/s/article/window-types-hanning-flattop-uniform-tukey-and-exponential

Kapitola 4

Georegistrácia

Automatická georegistrácia satelitných alebo leteckých snímok vyhotovených z veľkej výšky býva často realizovaná fázovou koreláciou vo frekvenčnej doméne. Toto je možné vďaka tomu, že tieto obrazy sú si veľmi podobné a vďaka vertikálnemu pohľadu sú medzi nimi prítomné len parciálne afínne transformácie. V matematike sa objekty, medzi ktorými je takáto transformácia, označujú za podobné. V prípade videa natáčaného z malej výšky je prítomné veľké perspektívne skreslenie a automatická registrácia na satelitné snímky je len veľmi ťažko predstaviteľná. Rozhodne nieje nemožná, tieto obrazy sú však vo svojej podstate natoľko rozdielne, že ani tie najrobustnejšie v súčasnosti známe príznakové metódy založené na lokálnych deskriptoroch niesú schopné tento proces s nejakou rozumnou presnosťou a istotou automatizovať. Tiež môžu nastať extrémne situácie, napríklad že video je natočené v zime. Registrovať obraz zasneženej krajiny na satelitnú snímku vyhotovenú v lete je ešte obtiažnejšie. Vďaka mojej metóde je ale možné ľubovoľne dlhé video georegistrovať pomocou jedinej snímky a preto nutnosť manuálnej anotácie bodov nieje až tak vážny problém.

Tradičný geografický súradnicový systém definujúci polohu na zemskom povrchu je zložený z dvoch parametrov – zemepisná šírka (angl. latitude) a zemepisná dĺžka (angl. longitude). Sú to uhly, ktoré zviera spojnica stredu zeme a daného bodu na zemskom povrchu so vzájomne kolmými rovinami. Tieto roviny sú určné konvenciou ako rovina rovníka a rovina nultého poludníka. Tieto súradnice opisujú zakrivenie povrchu zeme. Zvolený model homografie však zanedbáva toto zakrivenie, pretože povrch zeme sa lokálne javí ako rovina. Preto sa v tomto prípade väčšinou používa alternatívny geografický súradnicový systém, a to UTM (Universal Transverse Mercator). Tento rozdeľuje zemepisnú dĺžku zeme na 60 zón a každá z týchto zón je rozdelená na 20 pásiem zemepisnej šírky. Vzniká takto mriežka pozostávajúca z rovných plôch. Súradnice na týchto plochách sú definované v Karteziánskej sústave ako x a y. Sú teda kompatibilné so súradnicami v obrazovej rovine.

4.1 Manuálna anotácia

Proces georegistrácie, ktorého sa budem držať v tejto práci, pozostáva z manuálnej anotácie bodových korešpondencii medzi zvolenou snímkou z videa a satelitnou snímkou danej lokality. Anotáciu je možné vykonať pomocou bežne dostupnej geografickej aplikácie ktorá podporuje súradnice UTM. Takou je napríklad Google Earth. Z týchto korešpondencii je nájdená georegistračná matica homografie metódou najmenších štvorcov alebo algoritmom RANSAC. Každá snímka stabilizovaného videa má priradenú stabilizačnú maticu, ktorá definuje transformáciu snímky do referenčného priestoru videa. Stabilizačná matica snímky

zvolenej na georegistráciu sa invertuje a skombinuje s nájdenou georegistračnou maticou. Všetky stabilizačné matice videa sú skombinované s touto maticou a výsledkom je georegistrované video. Pri kombinovaní transformačných matíc je samozrejme potrebné dodržať správne poradie, matice sa násobia v opačnom poradí, než v akom majú byť aplikované transformácie.

Na obrázku 4.1 je znázorený tento proces na príklade snímky z reálneho videa a satelitnej snímky z programu Google Earth. Výsledná matica bola normalizovaná tak, aby bolo možné obraz videa na ilustráciu transformovať a vizuálne „priložiť“ na satelitnú snímku.

(a) Snímka videa (b) Satelitná snímka (c) Georegistrácia

Obrázek 4.1: Proces georegistrácie

Kapitola 5

Datová sada

Nová robustná metóda ktorej vývoju som sa v tejto práci venoval, mala čo najviac automatizovať georegistráciu videa z UAV platformy vybavenej jednou kamerou. Táto metóda mala čo najlepšie stabilizovať, resp. registrovať video do spoločného referenčného priestoru. Mala byť odolná najmä voči veľkým zmenám svetelných podmienok, ale tiež voči veľkým alebo skokovým geometrickým zmenám. Aby bolo možné túto metódu otestovať, validovať a prípadne kalibrovať rôzne jej parametre, bolo potrebné pripraviť vhodnú dátovú sadu videí. Pripravil som si dve dátove sady.

5.1 Reálne videá z praxe

Prvá dátová sada pozostáva z reálnych videí z praxe, presnejšie ide o videá rôznych dopravných uzlov natáčané z dronu. Cieľom týchto videí je analýza dopravy. Na videách boli klasifikátorom (neurónová sieť) detekované pozemné ciele z množiny tried objektov účastniacich sa cestnej premávky, ktoré boli následne stopované a boli získané ich trajektórie v čase. Tu prichádza na rad georegistračná metóda, ktorá umožní zo súradníc v obrazovej rovine vypočítať geografické súradnice v zobrazenej rovine a napríklad namiesto rýchlostí v pixeloch za hodinu získať z trajektórii skutočné rýchlosti v kilometroch za hodinu.

Z tejto rozsiahlej dátovej sady som sa snažil vybrať tie videá, na ktorých súčasné prístupy nedávajú dobré výsledky a na určitých úsekoch zlyhávajú. Zostavil som z týchto náročných videí malú validačnú sadu, ku ktorej sa vrátim v kapitole 7 o testovaní implementácie navrhnutej metódy.

5.2 Ground-Truth – počítačom generované videá

V oblastiach štatistiky, strojového učenia či počítačového videnia je očakávaný ideálny výsledok bežne označovaný anglickým termínom *ground-truth*. Na to aby bolo možné objektívne vyhodnotiť chybu metódy v čase, je potrebná práve takáto ground-truth informácia. Pri reálnych videách táto informácia nieje dostupná a najlepší spôsob ako na nich metódu hodnotiť je vizuálny posudok. Je možné vytvoriť si vizuálnu pomôcku, napr. vykresliť do registrovaného videa obrysy oblasti záujmu, napr. kontúry križovatky.

Pre objektívne hodnotenie chyby a možnosť sledovania jej vývoja v čase vo forme grafu som sa rozhodol použiť počítačom generované video. O pomoc som sa obrátil na R. Pazderku, ktorý vo svojej diplomovej práci [14] implementoval generátor syntetických dát z počítačovej hry Grand Theft Auto V (GTAV). Výslednú dátovú sadu používal k tré-

novaniu a validácii neuronových sietí. S malými úpravami mi pomohol týmto generátorom vygenerovať moju druhú dátovú sadu, ktorá simuluje typ videí z praxe. Ku každému videu som dostal množinu cca 100 bodov na rovine vozovky spolu s ich korešpondenciami v obrazovej rovine. Z toho som následne pre každú snímku vypočítal maticu homografie metódou najmenších štvorcov prostredníctvom funkcie implementovanej v knižnici OpenCV. Túto množinu matíc som použil ako ground-truth. Výsledná generovaná dátová sada, ktorú som použil, pozostáva z troch videí. Vo všetkých sa kamera plynule pohybuje po kružnici v určitej výške sledujúc stred križiatky, čo simuluje veľké geometrické zmeny. Na obrázkoch 5.1 sú momentky z týchto videí vystihujúce ich podstatu. Videá sa odlišujú vo svetelných podmienkach. Prvé video (5.1a) má statické, nemeniace sa osvetlenie. Súčasné prístupy sú na tomto videu aj napriek veľkým geometrickým zmenám pomerne úspešné. Toto video má dĺžku trvania 19 minút. Náročné video (5.1b), na ktorom súčasné prístupy zlyhávajú, je video s intenzívnymi zmenami počasia. Na robustnosť voči svetelným podmienkach a situáciám simulovaným v tomto videu je moja metóda zameraná najviac. Dĺžka trvania tohto videa je 40 minút. Tretie je video (5.1c) s prechodmi dňa do noci a noci do dňa. Dĺžka jeho trvania je 18 minút. Je to extrémne náročné video, najmä nočná scéna je veľmi problematická. Aj takéto krajné prípady by moja metóda mala zvládať.

(a) Video so statickým osvetlením

(b) Video s intenzívnymi zmenami počasia

(c) Video s prechodmi deň – noc

Obrázek 5.1: Generovaná dátová sada

V nasledujúcej kapitole, kde budem popisovať vývoj robustnej metódy na registráciu videa do spoločného referenčného priestoru, budem videá z tejto počítačom generovanej dátovej sady s ground-truth informáciou označovať skrátene len ako *ground-truth video*.

Kapitola 6

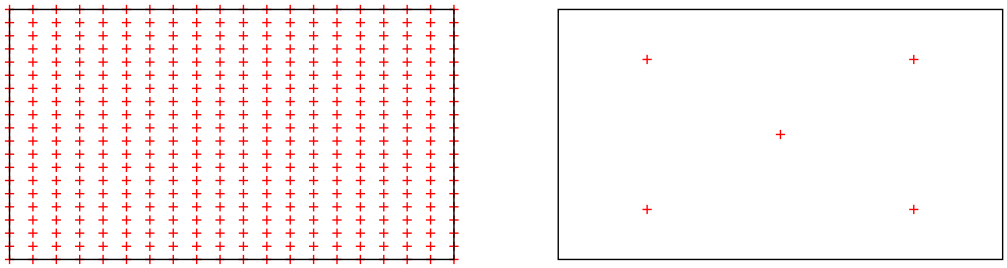
Nová robustná metóda

Cieľom tejto práce nebol návrh novej nízkoúrovňovej techniky spracovania obrazu, ale kompozícia tých najlepších algoritmov používaných v súčasnoti do komplexného celku, efektívne využívajúceho ich silné a kompenzujúceho ich slabé stránky. Pretože problematika spracovania obrazu je výpočtovo náročná, bolo potrebné zvoliť implementačný jazyk prekladaný priamo do strojového kódu. Rozhodol som sa pre jazyk C++, pretože je multiparadigmatický a umožňuje písať čitateľnejší a ľahšie udržiavateľný kód ako väčšina iných kompilovaných jazykov. V implementácii metódy som využil štandartné knižnice jazyka a knižnicu OpenCV (Open Source Computer Vision Library). Táto knižnica obsahuje implementácie objektov lineárnej algebry (matice a vektory) a tiež väčšinu nízkoúrovňových algoritmov pre spracovanie obrazu. Na paralelizáciu výpočtov na procesore som použil knižnicu OpenMP (Open Multi-Processing). Paralelizáciu výpočtov na grafickom procesore som realizoval prostredníctvom rozhrania CUDA, je teda viazaná na grafické procesory od firmy NVIDIA.

6.1 Dekompozícia homografie

Pred tým ako som mohol začať vývoj samotného algoritmu, bolo potrebné nájsť vhodný spôsob, ako porovnávať matice homografie, aby som mohol porovnať maticu odhadnutú mojou metódou s ground-truth maticou. Pretože hľadaný model je perspektívna transformácia, nieje možné tieto matice porovnávať priamo, ani žiadnym triviálnym spôsobom dekomponovať na základné zložky, ako by to bolo možné napríklad pri parciálnej afínnej transformácii. Aj napriek tomu že existujú postupy [13], ako maticu homografie dekomponovať na sadu matíc rotácie, vektorov translácii a normál kamery, matematicky existujú až 4 riešenia a je tiež nutné poznať vnútornú maticu kamery. Z tejto štvorice riešení je možné vybrať to najlepšie aplikovaním fyzikálnych obmedzení, neexistuje však záruka jeho správnosti.

Rozhodol som sa preto ísť inou cestou a to porovnávaním projekcii množiny bodov rozložených v obraze podľa vhodnej geometrie. Na hodnotenie chyby voči referenčnej matici som navrhol mriežku 400 bodov. V stabilizačnom algoritme som neskôr tento prístup použil tiež a to na odhad a filtráciu pohybu, tu som však použil geometriu 5 bodov, ktorá sa ukázala ako postačujúca. Obrázok 6.1 znázorňuje tieto dve mriežky bodov. Aj keď toto riešenie nieje dokonalé, ide o dobrú a deterministickú aproximáciu.



Obrázek 6.1: Projekčné mriežky

6.2 Kontinuálna stabilizácia

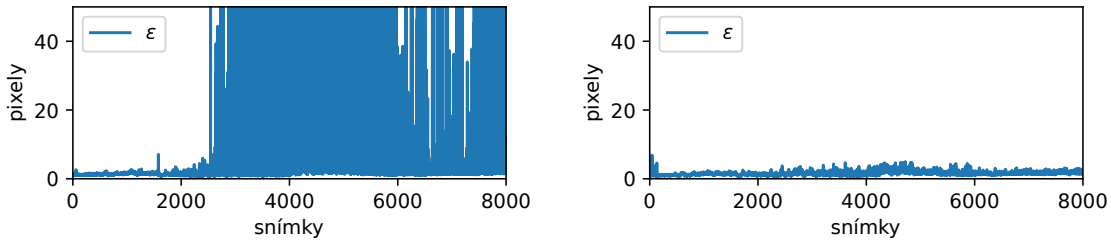
Prvým krokom pri vývoji komplexného algoritmu bola implementácia dvoch fundamentálnych v súčasnosti používaných prístupov a analýza ich správania na dátovej sade.

Stabilizácia algoritmom ORB

Prvým prístupom bola klasická príznaková registrácia na prvú snímku ako referenčný obraz. Tá pozostáva z detekcie kľúčových bodov v zdrojovom a cieľovom obraze, extrakcie deskriptorov týchto bodov a hľadania najbližších susedov v priestore príznakov. Ako detektor a extraktor som sa rozhodol použiť algoritmus ORB, ktorý je robustný a zároveň rýchly. Na párovanie bodov som použil algoritmus brute-force kNN, teda hľadanie najbližších susedov hrubou silou, a na odhad výslednej homografie algoritmus RANSAC. Aj napriek tomu, že rBRIEF deskriptory používané algoritmom ORB majú pri použití pyramídy vysokú invarianciu voči afínným transformáciám a zmenám jasu, registrácia na referenčný obraz sa s meniacou sa scénou stávala rýchlo nestabilnou. Hlavným dôvodom boli falošné zhody.

Tento fakt ma viedol k implementácii filtrovania nájdených dvojíc na základe najlepších v súčasnosti známych metód na zvýšenie kvality korešpondencii z algoritmov ako SIFT alebo ORB. Tou najjednoduchšou je krížová kontrola (angl. cross-check). To znamená, že su akceptované len tie páry, kde pre obidva deskriptory platí, že ten druhý je jeho najbližší sused. Algoritmus kNN sa teda vykoná dvakrát. Ďalšia metóda, ktorú som implementoval, je obmedzenie pomeru vzdialeností dvoch najbližších susedov. Algoritmus kNN nájde dvoch najbližších susedov a akceptované sú len tie páry, u ktorých je pomer vzdialeností prvého najbližšieho suseda k druhému menší ako určitý prah. Lowe [11] odporúča nastaviť hodnotu tohto prahu na 0.8. Táto metóda mala na dátovej sade najväčší vplyv na zvýšenie kvality registrácie. Poslednou použitou metódou je obmedzenie veľkosti (angl. scale restriction) [21]. Každý kľúčový bod je detekovaný na určitej úrovni obrazovej pyramídy a má teda určitú veľkosť okolia. Po nájdení najlepších dvojíc bodov sa zostaví histogram pomerov veľkostí ich okolí a nájde sa dominantný interval. Tie body, ktoré nespadajú do tohto intervalu v rámci určitého prahu, sú zahodené. Táto metóda sa však neukázala byť príliš významnou, pretože v dátovej sade sa nenachádzali problematické prvky, na ktoré je táto metóda zameraná. Tými sú najmä oblasti s intenzívnou distribúciou príznakov.

Na grafoch 6.2 zobrazujúcich chybu v kritickej časti ground-truth videa je vidieť ako tieto filtračné metódy výrazne zvýšili stabilitu a invariantnosť algoritmu. Stále však len oddialili neodvratné – zlyhanie. V určitom bode pri príliš veľkej zmene osvetlenia alebo geometrie už jednoducho nieje možné snímku registrovať priamo na referenčný obraz. Práve na riešenie tohto problému je zameraná zvyšná časť mojej práce.



(a) Bez filtrovania falošných zhôd

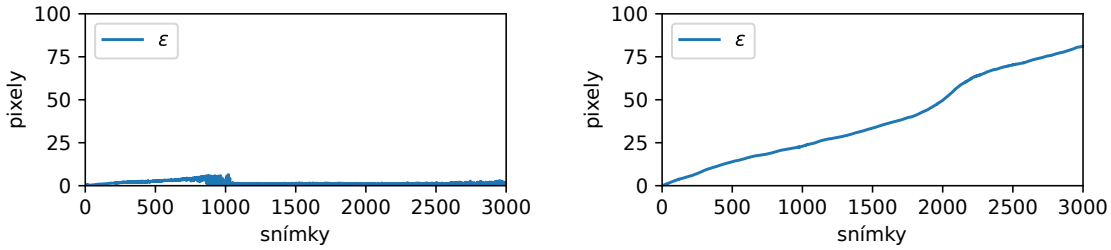
(b) S filtrovaním falošných zhôd

Obrázek 6.2: Chyba algoritmu ORB

Stabilizácia optickým tokom

Druhým prístupom bola registrácia prostredníctvom optického toku, presnejšie jeho odhadu metódou Lucas-Kanade. Tento prístup má výhodu, že je stabilnejší a na rozdiel od registrácie pomocou deskriptorov nieje v jeho výstupe prítomný šum. Dokáže sa tiež lepšie vysporiadať s veľkými zmenami geometrie, ak sú plynulé. Napríklad ak kamera výrazne zmení výšku, spôsobí to degradáciu kvality deskriptorov algoritmu ORB a ten sa stane nestabilným. Optický tok sa v takýchto situáciách ukázal ako veľmi stabilný.

V praxi sa optický tok používa na stopovanie objektov alebo práve na stabilizáciu videa. Ide však o stabilizáciu v zmysle vyhladenia pohybu. Cieľom mojej metódy je ale stabilizácia v zmysle čo najlepšie video registrovať do spoločného priestoru tak, aby ho bolo možné následne precízne georegistrovať. V tomto má optický tok jednu veľkú nevýhodu – kumuluje v čase chybu. Ďalšia slabá stránka optického toku je nutnosť redetekcie kľúčových bodov. Konkrétne na datovej sade sa totiž stávalo že bod, ktorý bol detekovaný na vozovke bol „odnesený“ autom, ktoré prechádzalo priamo cez okno definujúce jeho okolie. Prvá verzia, ktorú som implementoval, bol teda jednosmerný optický tok s redetekciou každých 5 snímok. Na detekciu kľúčových bodov som použil algoritmus GFTT a na odhad výslednej homografie opäť algoritmus RANSAC. Grafy 6.3 zobrazujú na úseku ground-truth videa so statickým osvetlením porovnanie vývoja chyby týchto dvoch prístupov v čase. Je vidieť kumulatívny charakter chyby optického toku.



(a) Algoritmus ORB

(b) Optický tok

Obrázek 6.3: Vývoj chyby v čase

Pre čo najväčsiu robustnosť je odporúčané vykonávať ešte spätný optický tok. Preto som vo finálnej verzii implementoval obojsmerný optický tok s redetekciou na každej snímke.

Algoritmus ako zvyčajne najprv odhadne vektory pohybu z predošlej snímky na nasledujúcu algoritmom Lucas-Kanade. Následne sa z nimi posunutých bodov odhadnú vektory pohybu spätne z nasledujúcej snímky do predošlej. Tie body, ktoré sa ocitnú na predošlej snímke inde než boli pôvodne, sú zahodené. Takto sa výrazne zvyšuje stabilita odhadu pohybu medzi jednotlivými snímkami, kumulácia chyby sa však pochopiteľne zachováva.

Akcelerácia na grafickom procesore

Pri realizácii mojej metódy som využil implementácie algoritmov ORB, brute-force kNN, GFTT a pyramídovej verzie algoritmu Lucas-Kanade z knižnice OpenCV. Tieto algoritmy zväčša pozostávajú z objemným a navzájom nezávislých výpočtov, čo ich robí skvelým adeptom pre beh na vysoko paralelizovanom hardvéri – grafickom procesore. Knižnica OpenCV obsahuje aj moduly napísané v rozhraní CUDA, kde sú okrem iného implementované aj tieto tri algoritmy. Rozhodol som sa teda použiť tieto existujúce implementácie. V tabuľke 6.1 sú orientačné hodnoty trvania jednotlivých implementácii týchto algoritmov na referenčnom stroji s procesorom Intel Core i7 a grafickým procesorom GeForce GTX 1060. Číslo N vyjadruje počet spracovávaných príznakov.

	N	**Procesor**	**Grafika**
GFTT	500	267.9 ms	14.5 ms
GFTT	5000	276.4 ms	15.3 ms
LK	500	48.6 ms	3.6 ms
LK	5000	77.0 ms	8.7 ms
ORB	500	185.2 ms	63.7 ms
ORB	5000	192.0 ms	65.5 ms
kNN	500	1.0 ms	0.1 ms
kNN	5000	42.7 ms	3.9 ms

Tabulka 6.1: Približné trvanie operácii – procesor vs. grafický procesor

Z nameraných hodnôt okrem iného vyplýva, že vďaka efektívnej implementácii algoritmu GFTT na grafickom procesore je stabilizácia optickým tokom viac ako dvojnásobne rýchlejšia než algoritmom ORB a to aj pri redetekcii na každej snímke. Pri registrácii algoritmom ORB zas došlo k najväčšiemu urýchleniu u následného hľadania najbližších susedov. Pre ešte väčšiu rýchlosť je dobré podľa možnosti vykonávať operácie asynchrónne a tiež minimalizovať objem dát prenášaný po zbernici PCI medzi procesorom a grafickým procesorom.

6.3 Odhad chyby a kontrolné body

Ako základ novej metódy som zvolil algoritmus ORB. Ten však aj napriek tomu že nekumuluje chybu môže zlyhať. Množinu snímok registrovaných priamo na spoločný referenčný obraz týmto algoritmom budem označovať ako *referenčný priestor*. Za zlyhanie algoritmu sa považuje stav, keď je vizuálna zmena oproti aktuálnemu referenčnému obrazu tak veľká, že nieje možné úspešne pokračovať v registrácii do aktuálneho referenčného priestoru. Navrhol som jednoduché riešenie tohto problému – rozdelenie na viacero referenčných priestorov a uchovávanie trasformácii medzi nimi. Časové body, ktoré sú blízko zlyhania, budem označovať ako *kontrolné body*. V takomto bode sa vytvorí nový referenčný priestor z aktuálnej snímky a algoritmus je opäť stabilný.

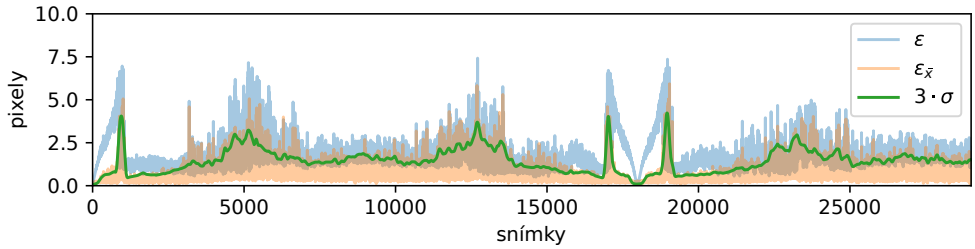
Šum ako metrika chyby

Pretože v praxi pochopiteľne nieje k dispozícii ground-truth, nieje jednoduché toto zlyhanie detekovať. Analýzou správania algoritmu ORB na dátovej sade som odpozoroval, že jeho výstup obsahuje šum, ktorý má v lokálnom čase Gaussovský charakter. V globálnom čase sa však štandartná odchýlka tohto šumu mení. Hodnota tejto odchýlky sa ukázala ako najlepšia metrika, na základej ktorej určiť zlyhanie algoritmu a prípadne sa rozhodnúť, ako veľmi jeho výstupu dôverovať. Na výstup z algoritmu ORB sa dá pozerať ako na signál v čase. Nájdenú transformačnú maticu je ale potrebné nejakým spôsobom dekomponovať. Tu som použil navrhnutú projekciu 5 bodov. Mojím prvým pokusom ako odhadnúť tento šum bolo použitie konvolúcie priemerovacím oknom na trajektórie týchto bodov. Odhadnutá chyba bola vzdialenosť pôvodného bodu od jeho priemernej polohy v rámci časového okna.

Problém tohto prístupu spočíva v tom, že výstupom je odhad okmažitej chyby, teda šum. Dôležitejšie než odhad okamžitej chyby na určenie stability algoritmu je odhad lokálnej štandartnej odchýlky tejto chyby. Toto je už „hladká“ funkcia plynule meniaca sa v čase. Štandartná odchýlka normálneho rozdelenia je odmocninou jeho rozptylu. Rozhodol som sa teda v rámci posuvného okna tento rozptyl odhadovať. Tu sa ale vynára zásadná otázka. Rozpyl čoho? Zmena polohy neznamená zlyhanie algoritmu, chaotická zmena tejto zmeny, čiže jej derivácia, však vykazuje vysokú koreláciu s chybou voči ground-truth. Experimentoval som s dvomi metrikami – rýchlosť resp. prvá derivácia polohy, a zrýchlenie resp. druhá derivácia polohy. Po experimentoch a teoretických úvahách som sa nakoniec ako najrobustnešiu metriku rozhodol použiť práve rozptyl zrýchlení. Tento odhad štandartnej odchýlky v diskrétnom čase, resp. $\sigma_x[n]$ abstraktne vyjadruje rovnica 6.1, kde x je poloha, n je index snímky a w značí pohyblivé okno. V mojej implementácii som použil Gaussovo okno normalizované tak, aby platilo $\sum w(n) = 1$, čo viedlo k hladšiemu výsledku než u obyčajného priemerovacieho okna. Jedna polovica odmocniny je v rovnici preto, že ide o odhad odchýlky polohy na základe jej druhej derivácie.

$$\sigma_x[n] = \frac{1}{2} \cdot \sqrt{\left((x'' - (x'' * w)[n])^2 * w \right)[n]} \tag{6.1}$$

Experimentoval som tiež s rôznymi veľkosťami okna. Príliš malé časové okno viedlo k šumu, príliš veľké naopak zanedbávalo lokálne zmeny. Okná o rozmeroch niekde medzi 200 a 300 snímkov, teda cca 10 sekúnd, sa ukázali ako najvhodnejšie. Graf 6.4 zobrazuje odhad chyby v ground-truth videu so statickým osvetlením oknom o veľkosti 251 snímkov. Modrá v grafe znázorňuje chybu voči ground-truth, oranžová odhad okamžitej chyby pohyblivým priemerovacím oknom a zelená trojnásobok odhadnutej funkcie hodnôt štandartnej odchýlky σ. Je vizuálne viditeľné pravidlo troch sigma vzhľadom na odhad okamžitej chyby.

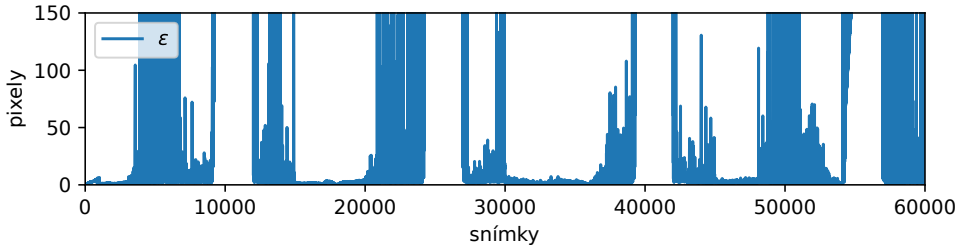


Obrázek 6.4: Odhad parametrov šumu

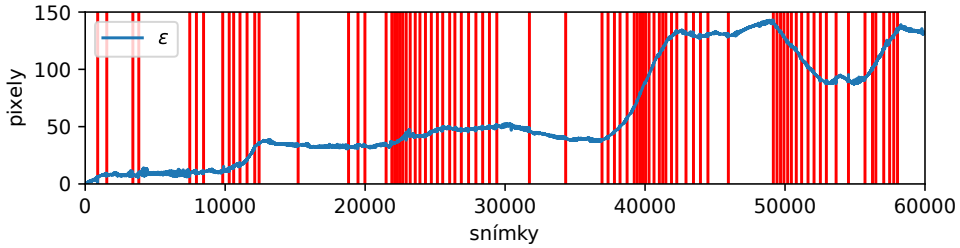
V mojej implementácii som pre čo najväčšiu robustnosť použil centrované okno, ktorého polovica zasahuje do budúcnosti. Kontrolný bod pre vytvorenie nového referenčného priestoru je teda detekovaný s opozdením polovice veľkosti tohto okna. Proces zotavenia sa z chyby pri takomto okne budem označovať ako *rollback*. Pri procese rollback je potrebné vykonať algoritmus kNN pre snímky za kontrolným bodom znovu, algoritmus ORB však nie. Implementoval som preto posuvnú vyrovnávaciu pamäť pre nájdené príznaky.

6.4 Graf z kontrolných bodov

Aj keď vytváranie nových referenčných priestorov v kontrolných bodoch zabránilo zlyhaniu, prinieslo nový problém – kumuláciu chyby. Chyba sa takto kumuluje podobným spôsobom, ako u optického toku, samozrejme oveľa pomalšie. Na grafe 6.5a je vývoj chyby u náročného ground-truth videa s častými a intenzívnymi zmenami počasia. Vo väčšine videa došlo k absolútnemu zlyhaniu algoritmu ORB. Graf 6.5b zobrazuje vývoj chyby pri použití kontrolných bodov a vytvorení celkovo až 81 referenčných priestorov. Ku zlyhaniu nedošlo, je však viditeľná výrazná kumulácia chyby.



(a) 1 referenčný priestor



(b) 81 referenčných priestorov

Obrázek 6.5: Použitie kontrolných bodov (červené vertikálne čiary)

Ak by bolo možné nájsť funkciu, ktorá by bola schopná dobrým spôsobom ohodnotiť cenu prechodu medzi dvomi referenčnými priestormi bez ohľadu na to, ako ďaleko sú od seba v čase, bolo by možné previezť problém minimalizácie kumulovanej chyby na optimalizačný problém hľadania najkratšej cesty v grafe.

Budovanie grafu

Graf som sa rozhodol budovať nasledovne. Pri vytváraní nového referenčného priestoru sa algoritmus pokúsi nový referenčný obraz registrovať na všetky referenčné obrazy z predchá-

dzajúcich priestorov. Takto sa buduje graf, ktorého uzly reprezentujú jednotlivé referenčné priestory a každý uzol je spojený hranou so všetkými ostatnými uzlami. Je dobré ale odstrániť tie hrany, kde je príliš nízka kvalita odhadu algoritmom RANSAC. To znamená, že pomer inlierov je menší ako určité percento (v mojej implementácii som použil minimum 50%). Nesmie však ísť o hranu medzi uzlom a jeho priamym predchodcom v čase.

Váhy deskriptorov

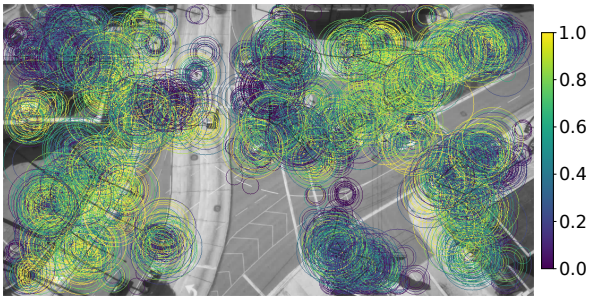
Prvým kandidátom na ohodnotenie hrán bol pomer inlierov, chýbal mu však časový kontext. Ďalšiu časť výskumu som teda venoval hľadaniu kvalitnej metriky na ohodnotenie hrán tohto grafu, ktorú bude možné vyhodnotiť len z dvojice zdrojový a cieľový obraz, no zároveň bude v sebe zahŕňať časový kontext. Spočiatku som sa orientoval na priestor obrazových dát, prišiel som s myšlienkou „tepelnej mapy“ (angl. heatmap). Vždy keď by bol kľúčový bod kvalitne spárovaný s iným, zvýšila by sa teplota mapy v tomto bode a jeho okolí. Globálne by zas táto mapa v čase „chladla“. Nedostatkom tohto prístupu je ale fakt, že detektor kľúčových bodov algoritmus FAST má tendenciu detekovať pomerne veľké množstvá kľúčových bodov sústredené v pomerne malých regiónoch.

Omnoho zmysluplnejšie je preto orientovať sa na priestor príznakov, teda na deskriptory. Postup, ktorý som navrhol, pozostáva z počítania opakovaných výskytov deskriptoru v časovom intervale referenčného priestoru a následného odhadu exponenciálneho rozdelenia zo strednej doby jeho výskytu. Na výpočet váhy som použil kumulatívnu distribučnú funkciu (CDF) tohto rozdelenia. Váha w je určená rovnicou 6.2, kde n je počet výskytov a Δ je počet snímok referenčného priestoru. Výsledná váha je pravdepodobnosť, že daný deskriptor sa znovu objaví o ďalších x snímok. Parameter x doporučujem nastaviť ako medián exponenciálneho rozdelenia podľa rovnice 6.3 s vhodne zvolenou strednou dobou μ.

$$w = 1 - e^{-\frac{n}{\Delta}x} \tag{6.2}$$

$$x = \ln(2) \cdot \mu \tag{6.3}$$

Na obrázku 6.6 je vizualizácia váh na príklade z dátovej sady s hodnotu $\mu = 15$ použitou v mojej implementácii. To znamená, že deskriptor so strednou dobou výskytu 15 snímok dostane váhu 0.5. Polomery kružníc sú dané veľkosťami okolí kľúčových bodov.



Obrázek 6.6: Vizualizácia váh

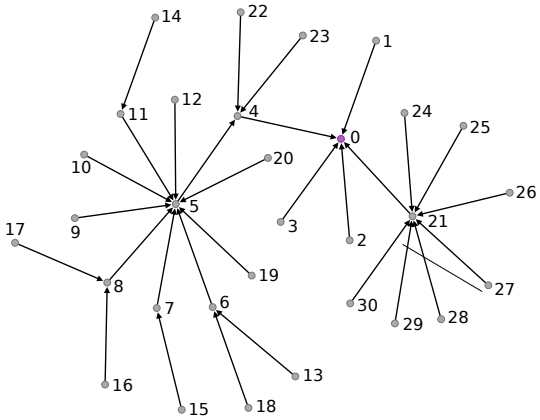
Rozhodnutie, či sa jedná o ten istý príznak, som realizoval pomocou prahu maximálnej vzdialenosti v priestore príznakov. Pre 256 bitové binárne deskriptory rBRIEF, aké bežne používa algoritmus ORB, som empiricky zvolil hodnotu 20.

Hľadanie najkratšej cesty

Na hľadanie najkratšej cesty grafom existuje rada algoritmov, výber toho najvhodnejšieho závisí od vlastností grafu, či je graf konečný, nekonečný, či je známa nejaká heuristika a podobne. V mojej metóde som sa rozhodol použiť Dijkstrov algoritmus [8], ktorý je najčastejšie používaným algoritmom na hľadanie najkratšej cesty. Implementoval som jeho základnú verziu, ktorá sa dá zjednodušene opísať v nasledovne. Na začiatku sa vytvorí množina nenavštívených uzlov Q. Vzdialenosti všetkých uzlov sa nastavia na ∞, pre počiatočný uzol na 0. Nasleduje hlavný cyklus. Algoritmus vyberie uzol s najmenšou vzdialenosťou z množiny Q a nastaví ho ako aktuálny. Pre každý jeho susedný uzol, ktorý je ešte v množine Q, spočíta vzdialenosť cez aktuálny uzol. Ak je táto vzdialenosť menšia ako predošlá hodnota, nastaví susednému uzlu novú hodnotu. Cyklus sa opakuje, kým množina Q nieje prázdna.

Pomer inlierov a váhy sú metrikami podobnosti. Tie je potrebné previezť na vzdialenosť, aby bolo možné postaviť graf a hľadať najkratšiu cestu. Prvý postup, s ktorým som experimentoval, bola jednoduchá zmena znamienka spolu s malou úpravou Dijkstrovho algoritmu, a to odčítavanie vzdialeností namiesto ich sčítavania. Pri pomere inlierov mal tento postup zlé výsledky. Ako som očakával, algoritmus mal takto tendenciu nachádzať priame cesty. V prípade váh boli výsledky podstatne lepšie, stále však nepôsobili príliš optimálne. Teoretickou úvahou som usúdil, že pre čo najrealistickejšie výsledky je vhodné použiť nelineárnu funkciu, pre ktorú platí, že jej limita pre $x \to 0$ je rovná ∞. Takáto funkcia zosilňuje vysoké a zoslabuje nízke hodnoty. Je vhodné, aby parameter tejto funkcie bol v intervale $\langle 0, 1 \rangle$. Pre pomer inlierov je toto prirodzená vlastnosť. Pretože celkovú váhu som definoval ako sumu všetkých váh silných párov, tie som sa rozhodol deliť počtom všetkých príznakov. Je možné, že to nieje najlepší spôsob normalizácie, v mojom výskume sa však ukázal ako postačujúci. Nevenoval som sa teda tejto problematike hlbšie. Experimentoval som s dvomi funkciami – parabola a záporný prirodzený logaritmus. Pre obidve metriky v experimentoch dávala najlepšie výsledky funkcia záporný prirodzený logaritmus, resp. $-\ln(x)$.

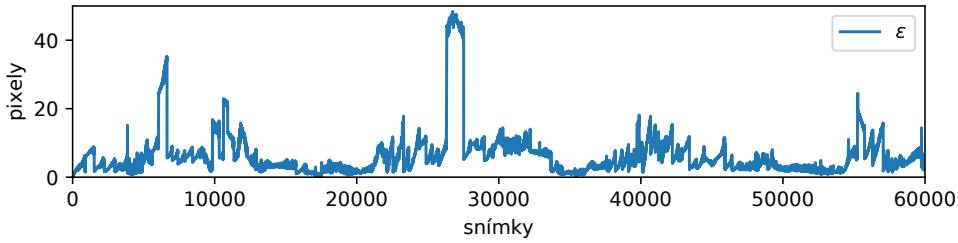
Na obrázku 6.7 je ukážka výsledného grafu pre prehľadnosť prvých 30 najkratších ciest z videa so zmenami počasia 6.5b. Čísla uzlov sú indexy referenčných priestorov.



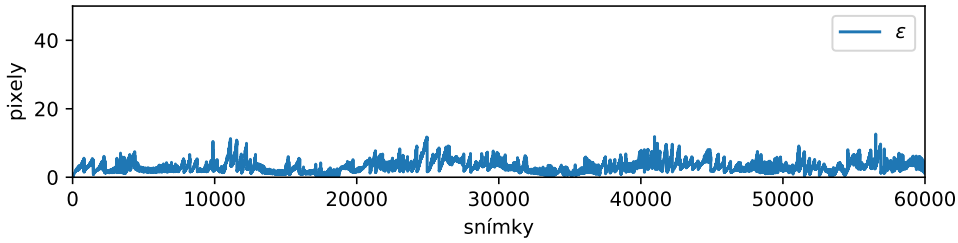
Obrázek 6.7: Graf z najkratších ciest

6.5 Zarovnanie fázovou koreláciou

Váhy ako metrika podobnosti dáva všeobecne lepšie výsledky než pomer inlierov, je však náchylnejšia na zlyhanie. Stalo sa, že aj keď nebolo možné algoritmom ORB dvojicu obrazov kvalitne registrovať, v malom regióne sa objavili silné korešpondencie výrazných príznakov a výsledkom toho bola nenulová váha aj tam, kde globálne došlo k pomerne veľkej chybe. Tento jav, keď nejaká cesta obsahuje nekvalitne registrovaný pár, je pozorovateľný v chybovej funkcii ako „schodovitý“ útvar. Časť práce som preto venoval vývoju algoritmu, ktorý je schopný túto chybu odhaliť a minimalizovať. Na grafe 6.8a zobrazujúceho výslednú chybu u videa so zmenami počasia po aplikovaní najkratších ciest je vidieť, že na niektorých cestách došlo k veľkej chybe. Graf 6.8b zobrazuje túto chybu minimalizovanú pomocou navrhnutého algoritmu.



(a) Cesty obsahujúce zlyhanie



(b) Cesty po korekcii

Obrázek 6.8: Minimalizácia globálnej chyby

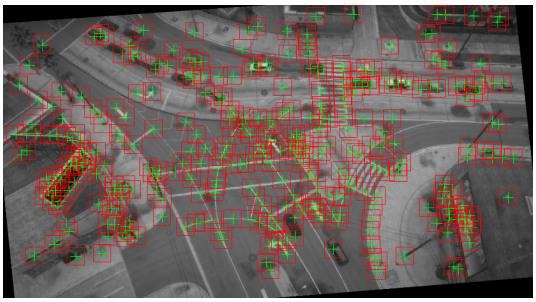
Algoritmus, ktorý som navrhol, sa pokúsi pomocou transformácie získanej z algoritmu ORB použitím metódy na hľadanie lokálnych posunov nájsť ešte presnejšiu transformáciu, než akú našiel algoritmus ORB. Ak tento pokus zlyhá, je to silná indikácia toho, že výstup algoritmu ORB nebol kvalitný. V takom prípade je jednoducho daná hrana odstránená z grafu ešte predtým, než sa vykoná Dijkstrov algoritmus. Gro algoritmu je teda hľadanie lokálnych posunov medzi zdrojovým a cieľovým obrazom. Optický tok v tomto prípade nieje použiteľný, pretože tieto dva obrazy majú zvyčajne veľmi rozdielny jas a je preto potrebné použiť metódu, ktorá je invariantná voči jasu a kontrastu. Registrácia vo frekvenčnej doméne má túto vlastnosť, preto som sa rozhodol použiť fázovú koreláciu. Zdrojový obraz sa najprv transformuje maticou homografie získanou z algoritmu ORB. Nasleduje detekcia kvalitných kľučových bodov v transformovanom obraze algoritmom GFTT. V tomto prípade doporučujem použiť Harrisovo skóre, ktoré sa zdá byť pre daný problém lepším než skóre Shi-Tomasi. Na mieste detekovaných bodov sa vytvoria okná o veľkosti, ktorú je

vhodné nastaviť napríklad ako mocninu 2, pre optimálnu rýchlosť výpočtu DFT [1]. Veľké okná niesú vhodné, pretože čím väčšie okno, tým viac sa prejaví perspektívna deformácia. Pre Full HD videá mali najväčší úspech okná o veľkosti 64×64 pixelov. Každé okno definuje dvojicu podobrazov v zdrojovom a cieľovom obraze. Fázovou koreláciou vo frekvenčnej doméne sa odhadne translácia medzi týmito podobrazmi na subpixelovej úrovni. V mojej implementácii som na tento odhad použil funkciu dostupnú v knižnici OpenCV.

Homografia opisuje transformáciu roviny, v tomto prípade pomyselnej roviny povrchu zeme. Regióny obrazu, v ktorých sa nachádzajú plochy kolmé na túto rovinu, napríklad výškové budovy či stromy, budú po tranformácii obrazu maticou nájdenej homografie značne deformované. Regióny, v ktorých sa nachádzajú plochy rovnobežné s touto rovinou, napríklad vozovka, budú po tejto tranformácii vizuálne veľmi podobné ich korešpondenciám v cieľovom obraze. Pre okná v nich sa úspešne podarí nájsť fázový posun. Nová homografia je v najhoršom prípade rovnaká, ako tá najdená algoritmom ORB. Väčšinou je však lepšia. Na obrázkoch 6.9a a 6.9b je detail pred a po zarovnaní. Je vidieť, že po zarovnaní cieľová rovina homografie viac zodpovedá rovine vozovky. Na obrázku 6.9c sú vizualizované okná.

(a) Pred zarovnaním

(b) Po zarovnaní



(c) Použité okná

Obrázek 6.9: Zarovnanie fázovou koreláciou

Odhad matice homografie sa opäť vykoná algoritmom RANSAC. Je všeobecne dobré zabezpečiť čo najväčšiu kvalitu dvojíc vstupujúcich do tohto algoritmu. Výber kvalitných párov je založený na dvoch kritériach. Najprv sa zahodia tie vzorky, kde odozva z fázovej korelácie je menšia ako medián. V druhom kroku sa ešte zahodia tie vzorky, kde nájdený posun je väčší ako polovica veľkosti použitého okna. Po vykonaní algoritmu RANSAC je pre úspešné zarovanie typický vysoký pomer inlierov, zvyčajne okolo 90% až 100%. To že algoritmus ORB zlyhal, resp. ním získaná transformácia nieje dostatočne kvalitná pre nájdenie lokálnych posunov, indikuje nízky pomer inlierov, často menej ako 40%. V mojej implementácii

[1] https://docs.opencv.org/master/d2/de8/group___core___array.html#ga6577a2e59968936ae02eb2edde5de299

som na rozhodnutie či k zlyhaniu došlo empiricky určil prah 80%, ktorý sa aj v extrémnych prípadoch z dátovej sady ukázal ako záruka kvalitného výsledku bez excesov. Tiež som pre maximálnu robustnosť implementoval bilaterálne hľadanie posunov, to znamená že sa inverznou maticou transformuje tiež cieľový obraz do zdrojového, nájdu sa posuny, transformujú sa naspäť pôvodnou maticou, obidve množiny dvojíc bodov sa konkatenujú a vykoná sa algoritmus RANSAC.

6.6 Kalmanov filter

Napriek úspešnej minimalizácii globálnej chyby je vo výstupe stále prítomná lokálna chyba, ktorou je šum algoritmu ORB. Tento šum je potrebné nejakým spôsobom odstrániť, resp. minimalizovať. To je možné docieliť použitím vhodného filtru vyhladzujúceho pohyb projekcii 5 bodov a následnej rekonštrukcie matice homografie z ich nových pozíc. Tým najjednoduchším je pohyblivý priemer. Tento lineárny filter síce dáva pomerne dobré výsledky, pristupuje k pohybu primitívne. Optimálna veľkosť okna je totiž závislá na charaktere pohybu a tam, kde pohyb nieje rovnomerný, je veľmi problematické toto okno určiť. Malé okno odstráni len tie najvyššie frekvencie, veľké zase vyhladí aj prudké pohyby, ktoré niesú súčasťou šumu a celkovo spôsobí viditeľné skreslenie pohybu.

Spojenie algoritmu ORB a optického toku

Kalmanov filter [9], známy aj ako LQE (Linear Quadratic Estimation) je algoritmus hľadajúci optimálny odhad stavu procesu z meraní obsahujúcich Gaussovský šum. Výsledný odhad má menšiu chybu resp. šum, než ktorékoľvek z týchto meraní osamote. Používa sa najmä na fúziu meraní z rôznych senzorov. Algoritmus pozostáva z dvoch krokov – predikcia (angl. predict) a aktualizácia (angl. update). Predikcia je odhad nového apriórneho stavu na základe modelu prechodu a aktuálneho stavu. Aktualizácia pozostáva z porovnania tejto predikcie so stavom získaným z nového merania a odhadu nového aposteriórneho stavu. Výsledná Gaussova funkcia definujúca chybu odhadu je súčinom Gaussových funkcii definujúcich chyby merania a predikcie.

Tento Kalmanov filter som sa rozhodol použiť na spojenie registrácie algoritmom ORB a registrácie optickým tokom. Algoritmus ORB ako senzor polohy, ktorý nekumuluje v čase chybu, je však zašumený. Optický tok ako senzor rýchlosti, ktorý kumuluje v čase chybu ale v malom časovom okne je schopný kvalitne odhadnúť pohyb prakticky bez šumu. Takýto filter využíva potenciál optického toku minimalizovať lokálnu chybu a netriviálnym spôsobom tak znižuje mieru šumu vo výstupe. Zároveň navrhnutý fyzický model umožňuje dobrú predikciu pohybu len na základe optického toku. S malým znížením kvality je takto možné registráciu urýchliť vynechaním algoritmu ORB napríklad každú druhú snímku, čo sa môže hodiť pri real-time aplikáci s nedostatočne silným grafickým procesorom.

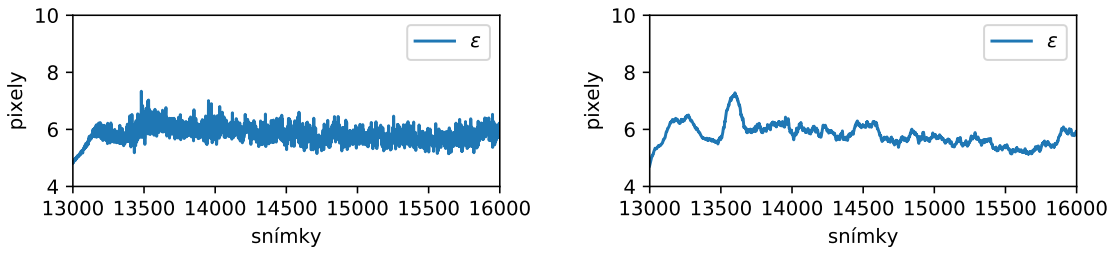
Návrh modelu

Predikcia a aktualizácia sú v podstate len súbory matematických rovníc pozostávajúcich z maticových operácii. Doležité je však správne nastaviť tieto matice pre daný systém. Stĺpcový vektor x definujúci stav a maticu prechodu F som navrhol tak, ako opisuje rovnica 6.4, kde p_x a p_y sú súradnice bodu v Karteziánskej sústave súradníc a p'_x a p'_y sú ich derivácie.

$$x_{k+1} = F \cdot x_k = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} p_x \\ p_y \\ p'_x \\ p'_y \end{pmatrix} = \begin{pmatrix} p_x + p'_x \\ p_y + p'_y \\ p'_x \\ p'_y \end{pmatrix} \tag{6.4}$$

Maticu meraní H som navrhol ako maticu identity I o rozmeroch matice F, pretože vektor merania je rovnaký ako vektor stavu x (meraná je nielen poloha ale aj jej derivácia). To ako bude Kalmanov filter jednotlivým meraniam dôverovať určuje kovariančná matica šumu meraní R_k. Pre polohu z algoritmu ORB som sa rozhodol použiť mnou navrhnutý odhad štandartnej odchýlky z rozptylu zrýchlení. Pre rýchlosť z optického toku som použil nízku empiricky určenú hodnotu $5 \cdot 10^{-4}$, nakoľko jeho odhadu rýchlosti sa dá vysoko dôverovať. Kovariančnú maticu šumu procesu Q_k je možné nastaviť empiricky, v mojej implementácii som použil konštantu na hlavnej diagonále $I \cdot 10^{-5}$.

Časový kontext optického toku použitý k odhadu rýchlosti je dobré nastaviť niekde medzi 1 až 10 snímkami, je možné odhadovať len na základe minulosti alebo aj na základe budúcnosti, ak nejde o real-time aplikáciu. Implementácia Kalmanovho filtru je relatívne jednoduchá, v knižnici OpenCV je ale tento algoritmus už implementovaný a preto som použil túto existujúcu implementáciu. Na grafe 6.10a je zobrazená chyba v zašumenom úseku ground-truth videa a na grafe 6.10b je chyba v danom úseku po vyhladení Kalmanovým filtrom.



(a) Lokálna chyba

(b) Lokálna chyba po filtrácii

Obrázek 6.10: Filtrácia Kalmanovým filtrom

Kalmanov filter je rekurzívny algoritmus a preto je možné túto filtráciu vykonávať priebežne počas spracovávania streamu videa, čo je dôležité najmä v prípade real-time aplikácie.

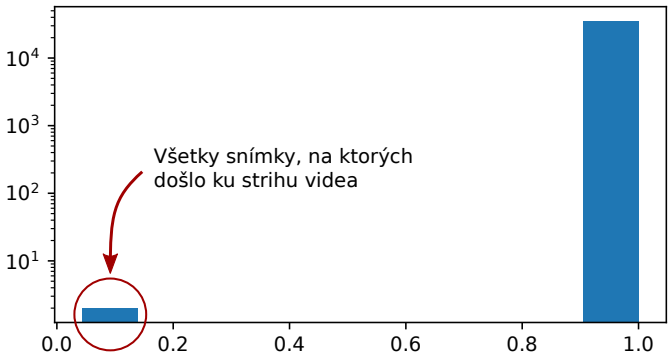
6.7 Detekcia strihu vo videu

Posledný problematický jav, ktorý mala moja metóda zvládať, boli skokové zmeny pohľadu a/alebo svetelných podmienok – strih videa. Ten je špecifický pre niektoré videá z praxe. Jeden dron sa totiž nemôže vznášať nekonečne dlho. Po určitom čase je potrebné dron vymeniť. Tieto dve videá sú následne konkatenované do jedného a je očakávané, že výsledné video bude celé registrované do spoločného referenčného priestoru a následne georegistrované. Aj napriek snahe vykonať túto výmenu čo najhladšie je ale tento strih vo videu často pozorovateľný ako veľmi výrazná skoková zmena scény.

Kontinuálna stabilizácia tu zlyháva rovnako, ako pri postupných zmenách. Moja metóda bez detekcie tohto strihu zlyháva na odhade šumu vo výstupe algoritmu ORB. Ten sa

v bode tohto strihu náhle zvýši. Vykoná sa teda rollback a vytvorí sa nový referenčný priestor. Algoritmus ORB by mal byť opäť stabilný. To však v tomto prípade neplatí. Kontrolný bod sa totiž nachádza pred týmto strihom a registrácia do nového referenčného priestoru opäť narazí na tento strih. Algoritmus ORB sa stane nestabilným, vytvorí sa kontrolný bod a takto to pokračuje až kým sa stredová snímka použitého časového okna nenachádza za týmto strihom. Nejde teda o fatálne zlyhanie, metóda je v konečnom dôsledku schopná video stabilizovať. Problém je, že v okolí tohto strihu dôjde k vytváraniu nového referenčného priestoru na každej snímke, čo má za následok postupné spomalenie algoritmu až do neprijateľnej miery.

Na základne vlastností optického toku som uvážil, že by bolo možné ho okrem odhadu pohybu využiť aj na detekciu tohto strihu. Pri skokovej zmene svetelných podmienok alebo pohľadu sa totiž u väčšiny bodov v rámci ich okolí skokovo zmení jas. Optický tok by mal teda zlyhať. Na videách z praxe sa mi na základe tejto úvahy podarilo pomocou obojsmerného optického toku tento strih vo videu dokonale detekovať. Určujúcou metrikou je *pomer úspešnosti optického toku*, teda počet bodov, pre ktoré sa podarilo odhadnúť vektor pohybu. U pomeru inlierov tiež dôjde k poklesu, nieje však prítomná až tak výrazná separácia. Z histogramu 6.11 vytvoreného z dvoch videí so strihom je vidieť, ako dobre optický tok dokázal tento strih detekovať.



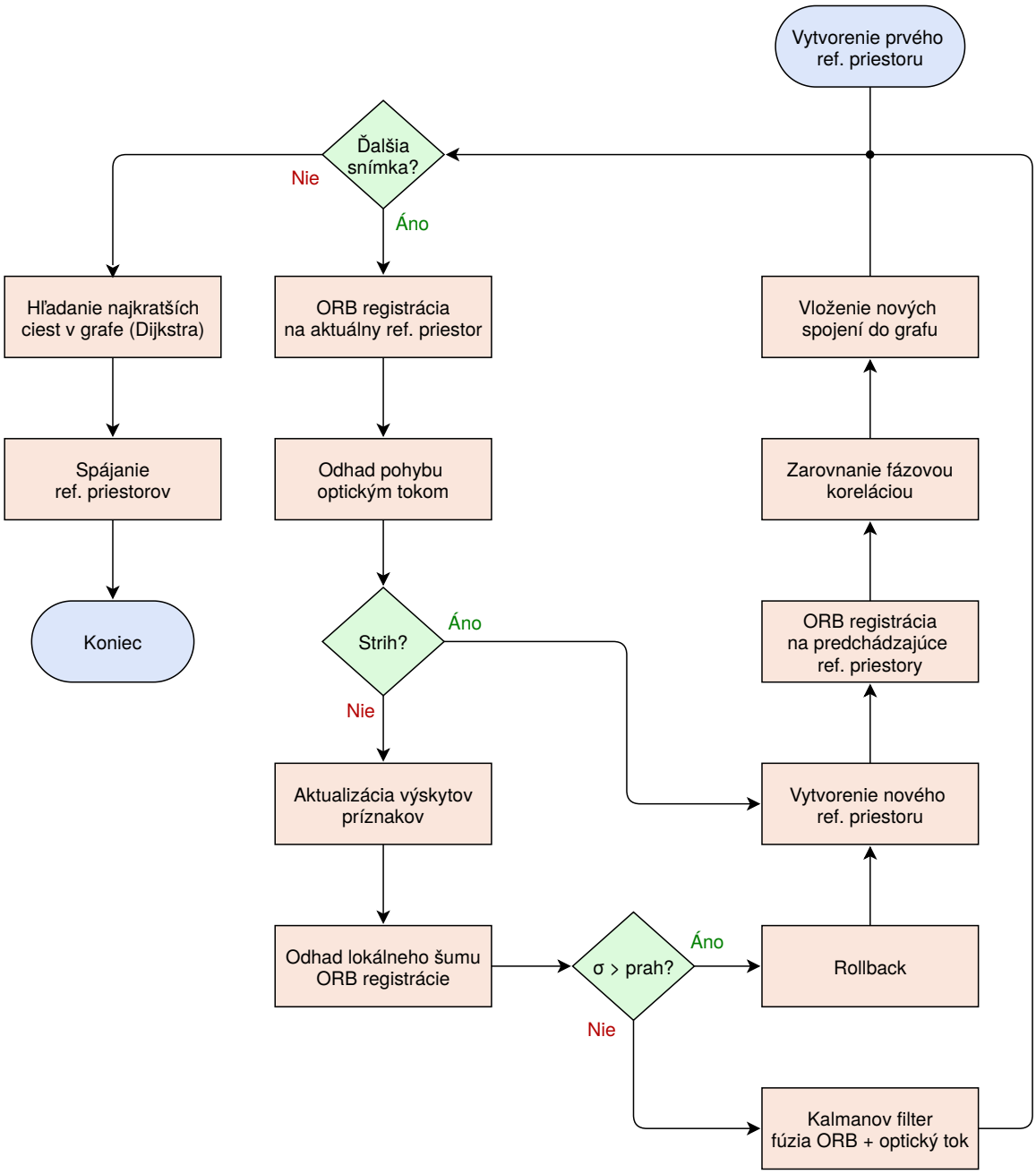
Obrázek 6.11: Histogram pomerov úspešnosti optického toku

Ak je detekovaný strih, vytvorí sa v tomto bode nový referenčný priestor, rollback sa ale nevykoná. Nový uzol tiež pochopiteľne nemusí byť spojený s jeho priamym predchodcom v čase. Nemusí teda existovať cesta od medzi prvým a posledným referenčným priestorom.

Navrhol som nasledovné riešenie. Na začiatku je vytvorený zoradený zoznam spojených referenčných priestorov M a do tohto zoznamu je vložená dvojica $(R_0, \emptyset)$, kde R_0 značí prvý referenčný priestor. Nasleduje rekurzívny algoritmus. Referenčný priestor z poslednej dvojice v zozname M je nastavený ako aktuálny uzol R_i. Nájdu sa najkratšie cesty do aktuálneho uzlu. Ak sa podarí nájsť cestu do daného uzlu, je daná cesta skombinovaná do výslednej transformácie, ktorá je pridaná do množiny priradenej k R_i. Inak, ak ide o prvé zlyhanie na aktuálnej úrovni, je do M pridaná nová dvojica $(R_i, \emptyset)$ a vykoná sa rekurzia.

Výstupom metódy teda môže byť viac referenčných priestorov, ak nebolo možné všetky referenčné priestory spojiť do jedného.

6.8 Bloková schéma



Kapitola 7

Testovanie

7.1 Testovanie na generovaných videách

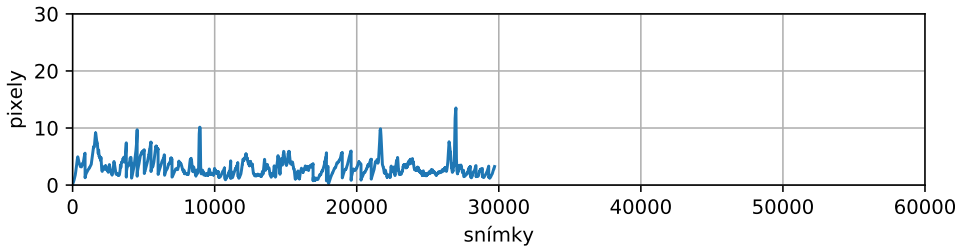
Na grafoch 7.4 je časový vývoj chyby stabilizácie generovaných ground-truth videí, ktoré som detailnejšie popisoval v sekcii 5.2. Na odhad chyby som použil projekčnú mriežkou 400 bodov, ktorú som opísal v sekcii 6.1. Videá mali rôznu dĺžku a preto je časová os grafov zarovnaná podľa najdlhšieho videa na 60 000 snímkov.

Okrem grafu vývoja chyby som určil aj skálarnu metriku chyby a to celkovú štandartnú odchýlku σ, spočítanú ako odmocninu priemeru štvorcov chýb jednotlivých snímok voči ground-truth. Na grafoch je vidieť, že chybu sa na všetkých troch videách darilo väčšinu času udržať pod 10 pixelov. Hodnotil som priemernú chybu v rámci celej mriežky, takže na okrajoch bola chyba o niečo väčšia ako táto hodnota, v strede obrazu bola zase blízka nule. Vizuálne som však usúdil, že priemerná chyba do 10 pixelov v rámci tejto mriežky je považovateľná za vysoko kvalitný výsledok. V extrémne náročných úsekoch došlo k miernej kumulácii chyby spôsobenej väčším počtom kontrolných bodov v najkratších cestách a tiež pohybom kamery po kružnici. Aj tu sa však maximálna chyba pohybovala v hodnotách len okolo 20 pixelov.

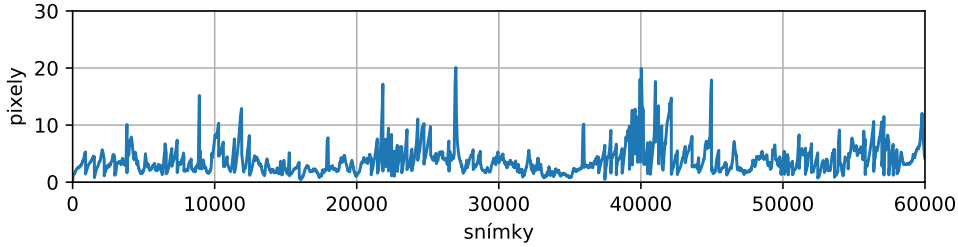
- Na grafe 7.1a je vývoj chyby videa so statickým osvetlením. Kvalita výstupu je podobná ako pri klasickej kontinuálnej stabilizácii algoritmom ORB. Hlavným rozdielom je minimalizácia šumu, resp. lokálnej chyby. Je ale prítomný akýsi kompromis, vysoká stabilita výstupu je dosiahnutá za cenu o niečo väčšej celkovej chyby. V cieľových aplikáciách je však na stabilitu výstupu kladený veľký dôraz a tak je tento kompromis v konečnom dôsledku pozitívny.

- Na grafe 7.1b je vývoj chyby videa s intenzívnymi zmenami počasia. Kvalita stabilizácie je porovnateľná s videom so statickým osvetlením, čo je obrovský úspech, pretože kontinuálna stabilizácia vo väčšine tohto videa absolútne zlyhala.

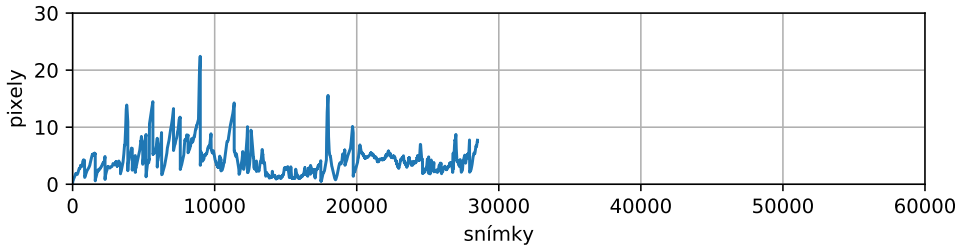
- Na grafe 7.1c je vývoj chyby videa s prechodmi dňa do noci a naspäť. V náročnej nočnej časti došlo k najväčšej chybe, stále sa však podarilo pomerne dobre video stabilizovať. Kontinuálna stabilizácia opäť vo väčšine tohto videa absolútne zlyhala.



(a) Video so statickým osvetlením, celková $\sigma \approx 3.44$



(b) Video s intenzívnymi zmenami počasia, celková $\sigma \approx 5.16$



(c) Video s prechodmi deň – noc, celková $\sigma \approx 4.49$

Obrázek 7.1: Chyba na generovanej dátovej sade z GTA V

7.2 Testovanie na reálnych videách

Z dátovej sady reálnych videí z praxe som na validáciu pripravil 3 náročné videá. Z týchto som 2 manuálne anotoval pomocou aplikácie Google Earth a vykonal georegistráciu jednoduchým nástrojom, ktorý som implementoval. Pre prehľadnosť uvádzam tabuľku 7.1. Na všetkých troch videách kontinuálna stabilizácia zlyhávala, navrhnutá metóda metóda však dokázala bez problémov tieto videá kvalitne registrovať do spoločného priestoru, vďaka čomu je možné každé z nich georegistrovať pomocou jedinej ľubovoľne zvolenej snímky.

Dátová sada videí z praxe však postrádala objektívnu ground-truth informáciu a preto ňou vývoj metódy nebol príliš ovplyvnenený. Účelom navrhnutej metódy je ale použitie na reálnych videách a tak som ju na týchto videách validoval vizuálne. Na hodnotenie som použil vizuálnu pomôcku – načrtol som v programe Inkscape kontúry križovatky na georegistračnej snímke. Na obrázkoch 7.4 sú momentky z dvoch georegistrovaných videí. Prvé (7.4a) malo trvanie 20 minút, druhé (7.4b) 30 minút.

č.	plynulé zmeny	skokové zmeny	nespojiteľné zmeny	georegistrácia
1	áno	áno	nie	áno
2	áno	áno	nie	áno
3	áno	áno	áno	nie

Tabulka 7.1: Validačná sada

Testovanie detekcie strihu

Detekciu strihu a následné zotavenie z chyby som testoval na videách z praxe, kde prirodzene došlo ku strihu pri výmene dronov. Všetky sa podarilo registrovať do jediného spoločného referenčného priestoru. Na videách, ktoré som mal k dispozícii, sa teda vo výslednom grafe nevyskytla oddelená časť a vždy boli nájdené všetky cesty. Moja metóda je však schopná sa s týmto javom vysporiadať. Mnou navrhnutý algoritmus a jeho implementáciu som testoval na videách s diametrálne odlišnými scénami (obrázok 7.2), ktoré som umelo spojil do jedného videa nástrojom ffmpeg. Výstupom boli dva dobre stabilizované referenčné priestory.

Obrázek 7.2: Validačné video č. 3 – nespojiteľné scény

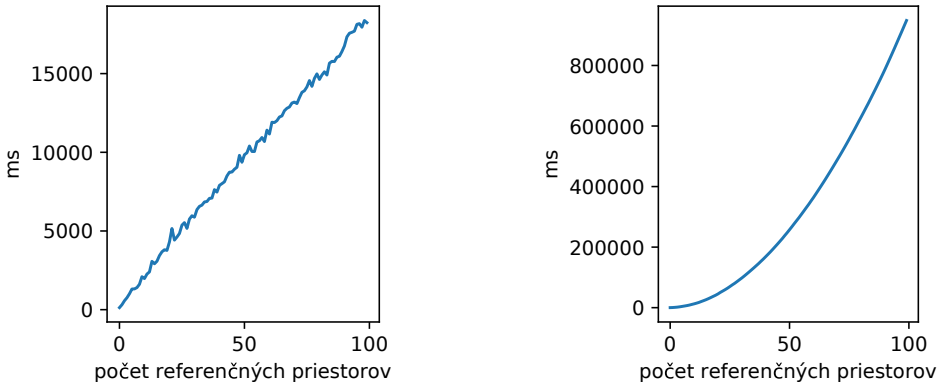
7.3 Časová náročnosť navrhnutého algoritmu

Podľa časovej náročnosti sa mnou navrhnutý stabilizačný algoritmus dá rozdeliť na dve časti. Časť kontinuálnej stabilizácie v rámci referenčného priestoru a časť zotavovania sa z chyby a vytvárenia nového referenčného priestoru.

Časť kontinuálnej stabilizácie má teoreticky lineárnu časovú náročnosť priamo úmernú počtu snímok. Na reálnych videách, ktoré bolo možné registrovať priamo do jedného referenčného priestoru, som testoval metriku počet stabilizovaných snímkok za sekundu, resp. FPS (Frames Per Second). Na referenčnom stroji s procesorom Intel Core i7 a grafickým procesorom GeForce GTX 1060 algoritmus dosahoval priemernú rýchlosť stabilizácie 18.3 FPS. Na výkonnejšom stroji by teda dosiahnutie výpočtu v reálnom čase nemal byť veľký problém. Na slabšom stroji je toto realizovateľné tiež s malou stratou kvality vďaka použitou Kalmanovho filtru. Je možné registráciu algoritmom ORB periodicky vynechávať a interpolácia chýbajúcich matíc homografie je realizovaná predikciou pohybu na základe fyzikálneho modelu, korigovanou optickým tokom. Pri vykonaní algoritmu ORB len na každej druhej snímke rýchlosť stabilizácie na referenčnom stroji dosiahla miestami až 30.6 FPS.

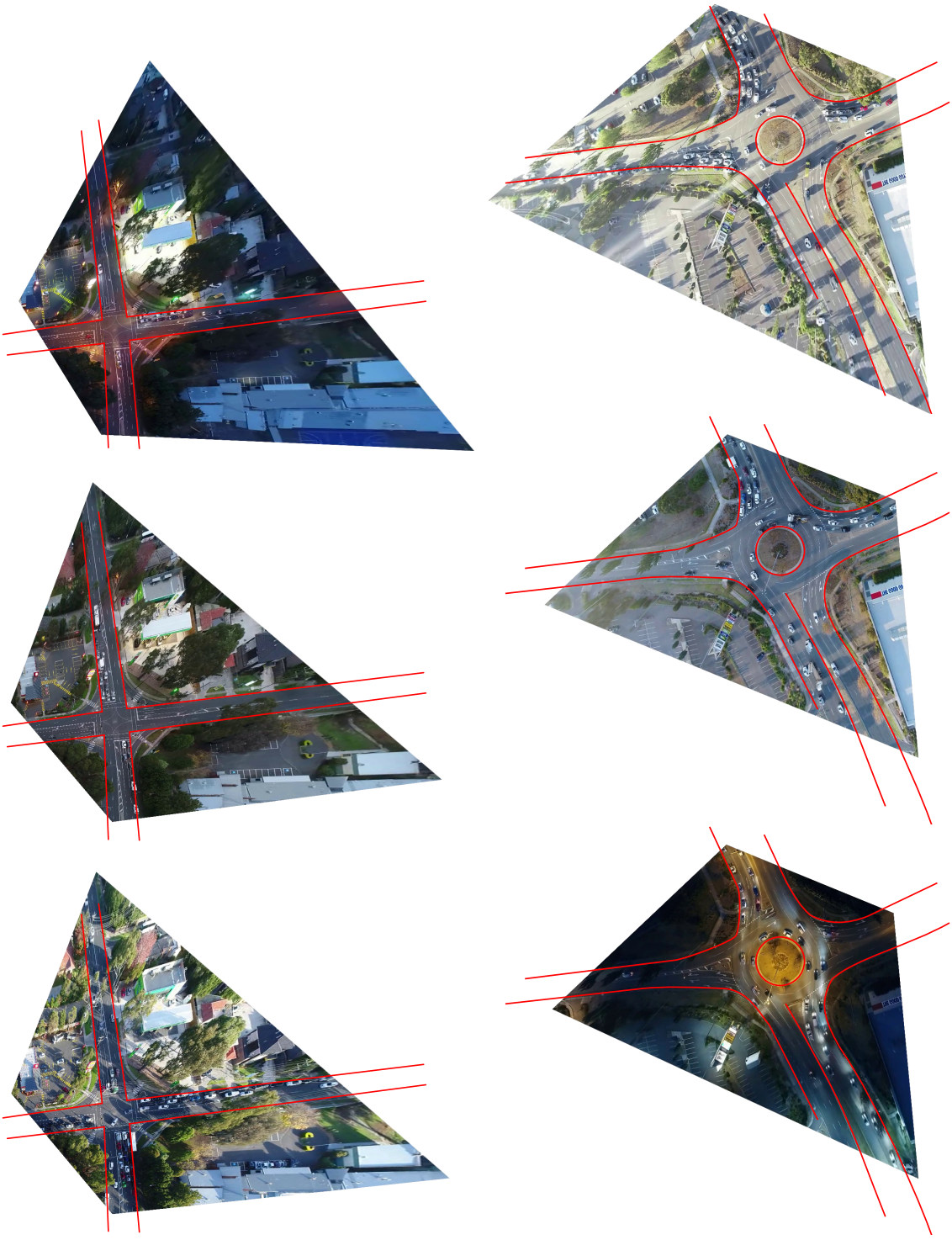
Časová náročnosť procesu zotavenia z chyby rastie priamo úmerne z počtom vytvorených kontrolných bodov, ako je vidieť na grafe 7.3a. Celková kumulatívna časová zložitosť tohto procesu má vzhľadom na počet kontrolných bodov kvadradický priebeh, ako zobrazuje graf 7.3b. Priemerné FPS stabilizácie je teda závislé od náročnosti videa, t.j. ako často je potrebné vytvárať nové referenčné priestory. Veľmi dlhé a náročné videá teda môžu byť

z časového hľadiska problematické. Aj pri náročných videách ale ide relatívne o zriedkavý jav a napríklad stabilizácia 40 minútového ground-truth videa so zmenami počasia aj po vytvorení 81 referenčných priestorov stále dosahovala priemernú rýchlosť až okolo 8 FPS.



(a) O (n) (b) Kumulovaná O $\left(n^2\right)$

Obrázek 7.3: Časová náročnosť zotavenia z chyby



(a) Validačné video č. 1

(b) Validačné video č. 2

Obrázek 7.4: Videá z validačnej sady po stabilizácii a georegistrácii

Kapitola 8

Záver

Cieľom tejto práce bol vývoj novej metódy, ktorá by čo najviac automatizovala proces georegistrácie videa z malého bezpilotného lietadla (napr. dronu) za účelom lokalizácie pozemných cieľov. Na úvod som opísal základnú teóriu týkajúcu sa hľadaného geometrického modelu medzi obrazom videa a geografickými súradnicami, spomenul algoritmy potrebné na zostavenie tohto modelu z bodových korešpondencii a tiež vhodný, aj keď voliteľný preprocesing odstránenia distorzie v obraze. Opísal som všeobecne registráciu obrazu a spomenul tie najdôležitejšie algoritmy, ktoré sú v súčasnosti považované za state-of-the-art v oblasti nízkoúrovňového spracovania obrazu. Na spomenutých algoritmoch (s výnimkou algoritmu SIFT) som postavil navrhnutú metódu. V súčasných prístupoch ku georegistrácii v cieľovej oblasti sa video automaticky registruje do spoločného referenčného priestoru. Výsledná sekvencia sa georegistruje pomocou jednej vhodne zvolenej snímky manuálnou anotáciou medzi zvolenou snímkou a satelitnou snímkou danej lokality.

Dôvodom manuálneho prístupu je prítomnosť perspektívy vo videu a potenciálne extrémny vizuálny rozdiel medzi týmito snímkami. V mojom výskume som teda vychádzal z tohto postupu a mnou navrhnutá metóda mala za cieľ automatizovať proces registrácie videa do spoločného priestoru. Tento proces som v mojej práci striedavo a skrátene označoval aj ako stabilizáciu videa, nešlo však o stabilizáciou v bežnom slova zmysle odstránenia prudkých pohybov, ale o precíznu registráciu do spoločného priestoru. Súčasné prístupy v danej aplikácii pozostávajú zvyčajne z pokusu o kontinuálnu registráciu. Tento postup je v praxi často postačujúci, je však veľmi obmedzujúci. Zlyháva na videách, kde dochádza k veľkým zmenám scény, presnejšie zmenám svetelných podmienok či geometrie pohľadu kamery. Pohyblivé a generické objekty sú tiež častým problémom. Pretože k reálnym videám nieje dostupná ground-truth informácia, vývoj som postavil na generovanej dátovej sade z hry GTA V, ktorá simulovala tieto extrémne podmienky.

Navrhnutá robustná metóda rieši problém zlyhania algoritmov príznakovej registrácie obrazu včasnou detekciou blížiaceho sa zlyhania a vytvorením nového referenčného priestoru. Z týchto priestorov je budovaný graf, ktorého hrany sú ohodnotené na základe váh príznakov zo silných zhôd medzi obrazmi. Váhy sú počítané na základe intenzity výskytu týchto príznakov v čase v rámci jedného referenčného priestoru. Na konci sú priestory spájané hľadaním najkratších ciest. Na subpixelové vylepšenie transformácii spolu s elimináciou častí scény s objektami, ktoré nespadajú do hľadaného modelu (vysoké budovy, stĺpy, bilboardy atď.), je použitá lokálna fázová korelácia. Minimalizácia lokálnej chyby je realizovaná integráciou odhadu rýchlosti pohybu optickým tokom cez Kalmanov filter. Opticky tok je použitý aj na odhad strihu vo videu.

Na generovanej aj reálnej dátovej sade výsledná implementácia podala veľmi kvalitné výsledky a ukázala vysokú mieru robustnosti voči tým najextrémenjším zmenám scény, aké môžu v praxi nastať. Bez problémov či viditeľného dopadu na kvalitu výstupu sa podarilo stabilizovať všetky videá, na ktorých kontinuálna stabilizácia zlyhala. Metódu je možné využiť napríklad pri analýze dopravy, z výslednej kvalitne georegistrovanej scény je možné precízne odhadovať skutočné rýchlosti či zrýchlenia objektov zachytených na scéne.

Možným pokračovaním tejto práce je vylepšenie empiricky určených prahov a revízia metriky hodnotenia hrán budovaného grafu. Tiež je možná úprava implementácie na výpočet v reálnom čase. V real-time implementácii je potrebné modifikovať odhad a zotavenie sa z chyby príznakovej registrácie. Prvá možnosť je použiť centrované okno, rollback je však nutné vynechať. Pri vytváraní nového referenčného priestoru tak dôjde k výpadku, ktorý je ale možné vyriešiť predikciou pohybu pomocou optického toku a Kalmanovho filtru. Druhá možnosť je použiť na odhad šumu kauzálne okno, odhadujúce rozptyl len na základe minulosti. Prvý prístup je robustnejší, je však prítomné opozdenie voči reálnemu času. Druhý prístup je zase bez opozdenia. Voľba teda závisí od požiadaviek cieľovej aplikácie. Po dosiahnutí požadovanej rýchlosti na danom stroji je pochopiteľne potrebné implementovať synchronizačné mechanizmy.

Literatura

[1] Calonder, M., Lepetit, V. a Fua, P. BRIEF: Binary Robust Independent Elementary Features. *Electronics Letters*. 2011.

[2] Derpanis, K. G. Overview of the RANSAC Algorithm. In: [online]. Dostupné z: https://pdfs.semanticscholar.org/42fc/990c356cedcac998b9e2cbdd021d25836781.pdf.

[3] Gavin, H. P. The Levenberg-Marquardt algorithm for nonlinear least squares curve-fitting problems. In: [online]. Dostupné z: http://people.duke.edu/~hpgavin/ce281/lm.pdf.

[4] Harris, S. M. A Combined Corner and Edge Detector. In:. Plessey Research Roke Manor, United Kingdom, 1988.

[5] Hartley, R. a Zisserman, A. *Multiple View Geometry in Computer Vision*. 2. vyd. Cambridge University Press, 2004. ISBN 9780521540513.

[6] Heikkila, J. a O., S. A four-step camera calibration procedure with implicit image correction. In:. IEEE, 1997.

[7] Hughes, J. F., Van Dam, A., Mcguire, M., Sklar, D. F., Foley, J. D. et al. *Computer Graphics Principles and Practice*. 3. vyd. Cambridge University Press, 1997. ISBN 9780321399526.

[8] Javaid, A. Understanding Dijkstra Algorithm. *SSRN Electronic Journal*. 2013.

[9] Kim, Y. a Bang, H. Introduction to Kalman Filter and Its Applications. In:. 2018.

[10] Lim, A., Ramesh, B., Yang, Y., Xiang, C., Gao, Z. et al. Real-time optical flow-based video stabilization for unmanned aerial vehicles. *Journal of Real-Time Image Processing*. 2017.

[11] Lowe, D. G. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*. 2004. ISSN 0920-5691.

[12] Lucas, B. D. a Kanade, T. An Iterative Image Registration Technique with an Application to Stereo Vision. In:. Morgan Kaufmann Publishers Inc., 1981.

[13] Malis, E. a Vargas, M. Deeper understanding of the homography decomposition for vision-based control. In:. French Institute for Research in Computer Science and Automation, 2007.

[14] Pazderka, R. *Segmentace obrazových dat pomocí hlubokých neurónových sít*. 2019. Diplomová práce. Vysoké učení technické v Brně.

[15] Rosin, P. L. Measuring Corner Properties. *Comput. Vis. Image Underst.* 1999. ISSN 1077-3142.

[16] Rosten, E. a Drummond, T. Machine Learning for High-Speed Corner Detection. In:. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-33833-8.

[17] Rublee, E., Rabaud, V., Konolige, K. a Bradski, G. ORB: An efficient alternative to SIFT or SURF. In:. IEEE, 2011.

[18] Shi, J. a Tomasi, C. Good features to track. In:. IEEE, 1994. ISSN 1063-6919.

[19] Sundararajan, D. *The Discrete Fourier Transform: Theory, Algorithms and Applications*. 2001. ISBN 9789810245214.

[20] Thunuguntla, S. S. *Object tracking using log-polar transformation*. 2005. Diplomová práce. Louisiana State University.

[21] Yi, Z., Zhiguo, C. a Yang, X. Multi-spectral remote image registration based on SIFT. *Electronics Letters*. 2008. ISSN 0013-5194.

Příloha A

Obsah priloženého pamäťového média

- **videostab/** – zdrojové súbory implementácie robustnej metódy
- **georegister/** – zdrojové súbory jednoduchého nástroja na georegistráciu
- **warp/** – zdrojové súbory nástroja na warping videa (vizualizácia)
- **thesis/** – zdrojové súbory textu bakalárskej práce
- **thesis.pdf** – text bakalárskej práce
- **poster.pdf** – prezentačný plagát
- **video.mp4** – prezentačné video
- **examples/** – validačná sada reálnych videí