Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Roman Komac

Analiza izbranih algoritmov za pridobivanje strukture iz gibanja kamere

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: izr. prof. dr. Matej Kristan

Ljubljana, 2017

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

V nalogi obravnavajte problem ocenjevanja strukture iz gibanja (SFM) kamere. Osredotičite se na algoritme, ki predpostavljajo statično sceno, po kateri se premika zgolj kamera. Opišite in razložite bistvene korake standardnega cevovoda algoritmov SFM. Za vsak korak navedite popularne metode, ki se uporabljajo v praksi. Izvedite analizo vpliva izbranih metod v posameznih korakih na kvaliteto rekonstrukcije.

Zahvaljujem se mentorju, dr. Mateju Kristanu, za pomoč in zvrhano mero potrpežljivosti pri oblikovanju diplomske naloge. Zahvalil bi se tudi staršem za podporo tekom študija ter moji Lučki za podporo in motivacijo pri izdelavi diplome.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
RANSAC	Random Sample Consensus	soglasje naključnega vzorca
SIFT	Scale-invariant Feature Transform	velikostno invariantna transformacija značilnic
SURF	Speeded Up Robust Features	pohitrene robustne značilnice
SfM	Structue from Motion	struktura iz gibanja
SVD	Singular Value Decomposition	singularni razcep
DLT	Direct Linear Transformation	direktna linearna transformacija
DoF	Degrees of Freedom	prostostne stopnje
LoG	Laplacian of Gaussian	Laplacov operator z Gaussovim filtrom
DoG	Difference of Gaussian	razlika Gaussovih filtrov

Povzetek

Naslov: Analiza izbranih algoritmov za pridobivanje strukture iz gibanja kamere

Avtor: Roman Komac

Poustvarjanje replike objekta iz delno prekrivajočih se fotografij je cilj ocenjevanja strukture iz gibanja. Za izvedbo popolne rekonstrukcije, je potrebno implementirati algoritme posamezne stopnje cevovoda. V iskanju učinkovitih in zanesljivih tehnik, smo zbrali raznolike algoritme na področjih ekstrakcije značilnic, opisa značilnic, ujemanja značilnic, ocenjevanja pozicije in rotacije, ekstrakcije oblaka točk ter 3D rekonstrukcije. Algoritme posamezne stopnje cevovoda smo nato primerjali po učinkovitosti, hitrosti ter zanesljivosti. Metode predstavljene v diplomskem delu so testirane na umerjenem setu rekonstruiranih 3D objektov. Podobnost med končno rekonstrukcijo ter zajetim 3D objektom je izmerjena z naborom orodij MeshLab [10]. Kot najboljša se je izkazala kombinacija algoritmov: SURF [25] za zaznavo značilnic, Približno iskanje bližnjih sosedov [32] in PROSAC [8] za iskanje pravilnih korespondenc ter ocenjevanje fundamentalne matrike, Inkrementalna struktura iz gibanja [21] za ocenjevanje pozicij in rotacij kamer ter Gosto stereo ujemanje [46] za triangulacijo oblaka točk.

Ključne besede: struktura iz gibanja, prilagajanje snopa, RANSAC, rekonstrukcija, cevovod.

Abstract

Title: Analysis of selected camera-based structure from motion algorithms

Author: Roman Komac

Estimating a 3D representation of a real-world object using partially overlapping images is the main goal of a photogrammetry technique known as Structure from Motion. Whereas the structure from motion pipeline has been well established, the algorithms in practice vary significantly in the implementation of the different stages. In search of effective and reliable techniques we have described several algorithms for feature detection, feature description, feature matching, pose estimation, point cloud extraction and 3D reconstruction. Each algorithm has been evaluated in terms of efficiency, speed and robustness. Methods presented in this thesis are tested on a dataset consisting of reconstructed 3D models and consecutive images of objects. For each model similarity between the reconstructed mesh and the original 3D model has been measured using the MeshLab [10] toolbox. In summary the best performing methods in the reconstruction pipeline proved to be: SURF [25] for feature detection, Fast approximate nearest-neighbour search [32] for feature matching, PROSAC [8] for fundamental matrix estimation, Incremental Structure from Motion [21] for camera pose estimation and Dense stero matching [46] for point cloud reconstruction.

Keywords: structure from motion, bundle adjustment, RANSAC, reconstruction, pipeline.

Poglavje 1

Uvod

Pridobivanje strukture iz gibanja je tehnika, ki vključuje področji fotogrametrije in računalniškega vida. Preko geometrije večih pogledov lahko poustvari repliko sveta, ki ga opisujejo fotografije. Večinoma se implementacije strukture iz gibanja osredotočajo na rekonstrukcijo notranjosti in zunanjosti zgradb [40] ali opis topografije s pomočjo fotografij zajetih iz zraka [26]. Pridobivanje strukture iz gibanja (angl., structure from motion, SfM) je tehnika zajema tridimenzionalne strukture iz fotografij. SfM in sorodne fotogrametrične tehnike podprte z računalniškim vidom so praktičnega značaja saj so aplikativne na mnogo področjih, tudi izven umetnega zaznvanja. Tako pristopi kot je direktna monokularna simultana lokalizacija in mapiranje (angl., simultaneous localization and mapping, SLAM) [11] [34] omogočajo lokalizacijo objekta zajema in modeliranje fizičnega sveta v realnem času ter se jih uporablja tako v aplikacijah z obogateno resničnostjo [38], kot tudi v sistemu za prostorsko lokalizacijo [4]. Pri pridobivanju strukture iz gibanja se je uveljavil standardni cevovod, ki je sestavljen iz več korakov. Ker je to področje še vedno raziskovalno aktivno za vsak korak cevovoda obstaja več možnosti implementacije.

V diplomski nalogi smo opisali različne metode (in njihove modifikacije), ki so skupne vsem standardnim postopkom pridobivanje strukture iz gibanja. S pomočjo splošne teoretične osnove cevovoda rekonstrukcije smo nato

ustvarili modularno implementacijo opisanih metod ter jih testirali.

1.1 Motivacija

Namen implementacije in preučevanja različnih algoritmov je razumljiva ter prenosljiva modularna koda. Prav tako je glavni cilj kode učinkovita in preprosta rešitev za zajem manjših 3D objektov z dovolj podrobnostmi za nadaljnjo uporabo v modelirnih programih.

1.2 Sorodna dela

Sorodna dela smo razvrstili v dva sklopa. Prvi sklop obsega dela avtorjev katerih prispevki so temelj za algoritme cevovoda rekonstrukcije. Celoten sklop je opisan v Poglavju 1.2.1. V drugem sklopu so navedene odprtokodne knjižnice namenjene ocenjevanju strukture iz gibanja. Pregledali smo največkrat citirane ali omenjene knjižnice ter jih primerjali med seboj. Primerjave se nahajajo v Poglavju 1.2.2.

1.2.1 Algoritmi standardnega SfM

Zaznava in opis značilnic

Opis značilnic obsega zajem regije pridobljene z ekstrakcijo značilnice iz rastrske slike. Med najbolj znane detektorje/deskriptorje spadajo: SIFT [25], SURF [5] in ORB [37]. Algoritma SIFT in SURF temeljita na zaznavi homogenih regij, ki se kontrastno razlikujejo od okolice. Za zaznavo regij uporabljata približke Gaussovega filtra z Laplaceovim operatorjem (angl., Laplacian of Gaussian, LoG). SIFT detekcijo regij opravlja z razliko Gaussovih filtrov (angl., difference of Gaussian, DoG), SURF pa s preprostim škatlastim filtrom, ki ga aplicira na integralno sliko. Algoritem ORB, ki za zaznavo implementira detektor FAST [44], pa uporablja preprost test okolišnjih intenzitet s katerim na sliki zaznava kote.

Ujemanje opisnikov

Iz deskripcij regij moramo nato med pogledi najti ujemanja. Najbolj preprost postopek je, da za vsak par pogledov primerjamo med seboj vse opisnike. Pri večjem številu pogledov zato izčrpno ujemanje predstavlja ozko grlo cevovoda. Približno iskanje ujemanj deskripcij reši to težavo, saj vzpostavi kompromis med hitrostjo izvajanja in deležem pravilno najdenih ujemanj. Večina algoritmov približnega iskanja bližnjih sosedov opisnikov deluje na osnovi K-dreves. Pri teh algoritmih ločimo ujemanje glede na tip opisnika. Binarne opisnike se primerja s pomočjo naključno generiranega K-drevesa [31]. Pri izgradnji se rekurzivno v vozlišča drevesa postavijo naključno izbrani opisniki iz prve slike. Iskanje nato poteka rekurzivno. Vsak opisnik druge slike se primerja v vozlišči. Primerjavo nadaljuje v podvozlišču, ki vsebuje najbolj podoben opisnik. Vektorske opisnike lahko primerjamo na podoben način [32]. Izgradnja tega drevesa poteka z naključno izbiro K opisnikov. Za razvejitveni pogoj se v vsakem vozlišču naključno izbere ena od dimenzij opisnikov z največjo varianco.

Približnemu iskanju parov opisnikov je namenjena knjižnica FLANN [33]. Slednja implementira vse naštete metode približnega ujemanja. Poleg implementacij poskrbi tudi za samodejno ocenitev najboljših parametrov izvajanja iz vhodnih podatkov.

Robustno iskanje korespondenc

Nekatera ujemanja opisnikov so lažna, ne glede na algoritem, ki ga uporabimo pri iskanju parov. Raziskovalci so naslovili to z iskanjem množice, ki ustreza geometrijskim zahtevam problema. Preslikavo točk v stereo sistemu kamer opisuje fundamentalna matrika. Slednjo podrobno opišemo in pojasnimo v Poglavju 2.3.

Za robustno iskanje ujemanj je na voljo več različnih metod. Najpogosteje so uporabljene metode, ki temeljijo na osnovnem algoritmu RANSAC [12], saj omogočajo robustno estimiranje parametrov modela po principu *generiraj in preveri*. Obstaja mnogo nadgradenj osnovnega algoritma. PROSAC [8] uvaja

neenotno zajemanje korespondenc za ocenitev modela. Lokalno-optimiziran RANSAC [9] vsebuje dodatno notranjo zanko v kateri poskuša izboljšati trenutno predpostavljen model. Preemptive RANSAC [35] je časovno omejena različica namenjena uporabi v aplikacijah, ki se izvajajo v realnem času. Test $T_{d,d}$ [30] pa se lahko uporablja v povezavi z večino različic osnovnega algoritma. Njegov namen je zavreči potencialno slabo hipotezo pred overitvijo na preostalih točkah.

Struktura iz gibanja

V zadnjih letih je bil narejen velik napredek v področju ugotavljanja pozicij in orientacij pogledov. Globalne metode SfM niso podvržene odstopanju, ki lahko nastane pri inkrementalnih metodah zaradi postopnega dodajanja novih pogledov. Prav tako so hitrejše zaradi možnosti paralelizacije in potrebe po samo enem prilagajanju snopa. Metoda 1DSFM [45] omogoča skalabilno in hitro ocenitev pozicij ter rotacij vseh pogledov. Problemi translacije se rešujejo lokalno. Pozicije pogledov se estimirajo ter izboljšajo v vsaki dimenziji posebej. Globalna linearna metoda [20] pa minimizira geometrično napako s tem ko uveljavlja trikotna razmerja pri trojici kamer. Prednost te metodo je robustno ocenjevanje pozicij kamer tudi pri kolinearnem gibanju.

Triangulacija oblaka točk

Po pravilni oceni pozicij in rotacij pogledov smo korak bližje končni rekonstrukciji. Da lahko poustvarimo prvoten objekt, moramo najprej najti korespondence med vhodnimi slikami s pomočjo katerih bomo lahko ustvarili oblak točk. Na področju stereo ujemanja obstaja mnogo različnih metod. Za izgradnjo globinske mape se pogosto uporabljajo metode izpeljane iz iskanja maksimalnega pretoka grafa [22]. Za nekoreliran stereo sistem kamer pa so na voljo algoritmi postopnega ujemanja, ki med pogledoma vzpostavijo redka ujemanja regij in s pomočjo gradienta okolice postopoma dodajajo nova pravilna ujemanja [46].

Rekonstrukcija

Algoritem marširajočih kock [24] ter Poissonova rekonstrukcija [17] sta najbolj uporabljena algoritma pri rekonstrukciji tridimenzionalne oblike iz oblaka točk. Algoritem marširajočih kock, se izvaja nad oblakom enakomerno razporejenih točk. Razvit je bil za učinkovit prikaz rezultatov računalniške tomografije. S sprehodom skozi polje točk ustvari poligonsko obliko na delih, ki opisujejo zunanjo mejo oblaka. Navkljub modernejšim metodam se še vedno uporablja zaradi preproste implementacije ter hitrosti izvajanja. Prednost slednjega je tudi, da za pridobitev 3D oblike ni potrebno poznati orientacij točk. Poissonova rekonstrukcija je najbolj učinkovita za probleme kjer poznamo orientacije točk. Oblak točk porazdeli s pomočjo adaptivnega osmiškega drevesa. V vsakem podprostoru s pomočjo orientacij točk izračuna vektorsko polje. Na vektorsko polje nato aplicira Poissonovo enačbo s katero izračuna obliko.

1.2.2 Knjižnice SfM

SFM-Toy-Library [39] je preprosta odprtokodna knjižnica spisana v programskem jeziku C++ s pomočjo platforme OpenCV [18]. Namenjena je predvsem kot osnova ali pomoč pri implementaciji lastnega cevovoda rekonstrukcije. Najbolj se od večine knjižnic za rekonstrukcijo razlikuje v načinu ugotavljanja pozicij kamer. Inkrementalni algoritem za strukturo iz gibanja implementiran v tej knjižnici iz para kamer vzpostavi začetno relacijo, nato pa naknadno v vsaki iteraciji pripne nov pogled s pomočjo algoritma perspektiva-in-točka [13].

OpenSfM [27] sloni na platformi OpenCV [18]. Spisana je v programskih jezikih Python in C++. Omogoča uporabo mnogo več pogledov, podpira uporabo GPS podatkov za rekonstrukcijo v realni velikosti ter ekstrakcijo parametrov kamere iz vhodnih slik, ki to omogočajo.

Theia SFM [42] implementira najsodobnejše metode na vseh stopnjah cevovoda. V rekonstrukcijo običajno uspe uvrstiti več kot 90% vseh pogle-

dov [41]. Za manjše scene povprečna odstopanja ne presegajo nekaj milimetrov. Spisana je v jeziku C++. Tako kot preostali navedeni knjižnici za nelinearno optimizacijo pri prilagajanju snopa uporablja knjižnico Ceres [3].

1.3 Struktura naloge

V Poglavju 2 je predstavljena potrebna teoretična podlaga. Podane so tudi predpostavke, ki omogočajo potek izgradnje strukture iz gibanja. Stopnje cevovoda so pojasnjene v Poglavju 3, kjer so podane tudi osnovne zamisli posameznega algoritma trenutne stopnje. Poleg vsakega opisa so pripisane prednosti in slabosti dotičnega algoritma ter opazke avtorjev ali drugih raziskovalcev. V Poglavju 4 se bomo spoznali z implementacijo cevovoda in pripadajočih metod. Poleg večine postopkov podajamo psevdokodo. V Poglavju 5 so prikazane meritve zanesljivosti in hitrosti delovanja omenjenih algoritmov. Meritve so opravljene za detekcijo in opis značilnic, ujemanje značilnic, iskanje fundamentalne matrike in rekonstrukcijo. Evalvacija je narejena nad prostodostopnim setom podatkov [19]. Poglavje 6, povzame in interpretira ugotovitve evalvacije. Navedene so tudi smernice nadaljnjega razvoja.

Poglavje 2

Teoretični del

SfM je združitev več problemov na področju računalniškega vida. Da razumemo potek celotnega cevovoda rekonstrukcije moramo poznati osnovne teoretične gradnike. Teoretične osnove opisane v naslednjih poglavjih se nanašajo na družino metod standardnega cevovoda rekonstrukcije.

2.1 Predpostavke

V tem poglavju so navedene predpostavke, ki omogočajo delovanje metod opisanih v diplomskem delu. Poglavje 2.1.1 vsebuje predpostavke, ki se nanašajo na slike zajete z kamero in notranje parametre take kamere. Poglavje 2.1.2 pa se osredotoča na pravila nanašajoča se na geometrijo zajete scene.

2.1.1 Kamera

Za zajete slike se predpostavi model kamere s točkovno odprtino, kjer so preslikave linearne. Zaradi uporabe leč v modernih fotoaparatih pa linearnost preslikav ni zagotovljena. Uporaba leče namreč vpelje radialno popačenje. V primeru, da ne poznamo parametrov, ki opisujejo popačenje ga ne moremo odpraviti zanesljivo. Brez predhodnih podatkov o kameri moramo predpostaviti, da izkrivljanje nima prevelikega vpliva na zajeti sliki, drugače iz parov slik ne moremo izpeljati fundamentalne matrike. Za pravilno delovanje mora

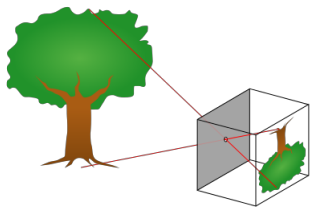
tudi veljati konsistenca notranjih parametrov kamere skozi vse zajete poglede. Če se goriščna razdalja močno spreminja skozi poglede, to zelo oteži iskanje pozicij kamer ter hkratno optimizacijo parametrov vseh kamer, bolje poznano kot prilagajanje snopa.

2.1.2 Scena

Tako kot za kamero tudi za zajeto sceno veljajo določena pravila. Telesa v zajeti sceni ne smejo biti deformabilna. V vseh pogledih morajo zavzemati isto tridimenzionalno obliko. Prav tako se ne smejo premikati v prostoru. Edini objekt, ki se sme premikati je kamera. Premikanje in deformiranje teles je dovoljeno samo za predmete, ki zavzemajo zanemarljiv del zajetega pogleda. Dodatna predpostavka glede scene je, da vsebuje teksturirane regije ter geometrične oblike. Brez slednjih v zajetih slikah ne moremo poiskati značilnic.

2.2 Kamera s točkovno odprtino

Teoretični model kamere s točkovno odprtino temelji na nekoč uporabljeni napravi *Camera obscura*. Tak model predpostavlja, da vsi svetlobni žarki potujejo skozi gorišče kamere, zato se da opazovan prostor zapisati v homogenem koordinatnem sistemu.



Slika 2.1: Skica modela kamere s točkovno odprtino. Slika povzeta po [2].

2.2.1 Homogen koordinatni sistem

Homogen koordinatni sistem omogoča zapis linearnih preslikav z matrikami. Poleg kartezičnih koordinat vsebuje še dodatno vrednost, ki se uporablja za normalizacijo vektorja. Tako lahko točko $\mathbf{x_E} = \begin{bmatrix} x \\ y \end{bmatrix}$ v dvodimenzionalnem kartezičnem prostoru zapišemo z neskončno mnogo točkami v homogenem prostoru $\boldsymbol{x} = \begin{bmatrix} wx \\ wy \\ w \end{bmatrix}$, kjer $w \neq 0$. Kartezično točko lahko dobimo, tako da vektor delimo z vrednostjo w. Če je $w = 0$, vektor opisuje poseben primer, kjer se točka nahaja v neskončnosti. Po definiciji se v tej točki stikajo vse premice, ki so vzporedne z vektorjem $\mathbf{x_E}$.

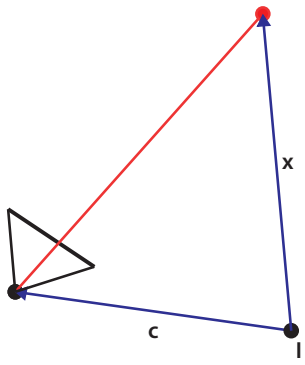
2.2.2 Kamera z točkovno odprtino

S pomočjo novo definiranih koordinat lahko sedaj lažje zapišemo preslikavo prizorišča v slikovno ravnino. Pri kameri s točkovno odprtino sicer poznamo dve preslikavi. Prva preslika tridimenzionalne točke iz prostora prizorišča v tridimenzionalen prostor kamere.

Preslikava točke $\mathbf{x}$ iz svetovnih koordinat v točko $\mathbf{x_c}$, ki se nahaja v koordinatah kamere je pogojena z enačbo

$$\mathbf{x_c} = \mathbf{R}(\mathbf{x} - \mathbf{c}), \tag{2.1}$$

kjer je $\mathbf{R}$ rotacijska matrika kamere, $\mathbf{c}$ pa izhodišče koordinatnega sistema kamere v svetovnih koordinatah.



Slika 2.2: Preslikava točke v koordinatni sistem kamere.

Slika 2.2 prikazuje preslikavo poljubne točke v koordinatni sistem kamere. Z **I** je označeno koordinatno izhodišče sveta, $\boldsymbol{x}$ predstavlja vektor od koordinatnega izhodišča do točke, $\boldsymbol{c}$ pa vektor od izhodišča do gorišča kamere. Preslikavo točke lahko povzamemo z enačbo

$$\mathbf{x_c} = \Big[\mathbf{R}| - \mathbf{Rc}\Big]\,\mathbf{x} = \Big[\mathbf{R}|\mathbf{t}\Big]\,\mathbf{x}. \tag{2.2}$$

Vektor $\mathbf{t}$ v Enačbi (2.2) označuje vektor premika med svetovnim koordinatnim izhodiščem in izhodiščem koordinatnega sistema kamere.

Druga preslikava pretvori točko iz tridimenzionalnega koordinatnega sistema kamere v slikovno ravnino kamere. Preslikavo predstavlja enačba

$$\mathbf{x_s} = \mathbf{K}\mathbf{x_c}, \tag{2.3}$$

kjer je $\mathbf{K}$ kalibracijska matrika kamere, t.j.

$$\mathbf{K} = \begin{bmatrix} a_x & s & x_0 \\ 0 & a_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.4}$$

V Enačbi (2.4) preslikata parametra $a_x = fM_x/S_x$ in $a_y = fM_y/S_y$ koordinate točke v pripadajoč piksel. Izračuna se ju s pomočjo goriščne razdalje

kamere (f) in števila pikslov na meter v obeh dimenzijah. Parametra x_0 in y_0 pa opisujeta zamik, ki koordinatno izhodišče slike postavi v zgornji levi kot. Parameter s je uporabljen v primeru, da so vrstice senzorja zamaknjene. Končno projekcijsko matriko $\mathbf{P}$ sestavljajo matriki $\mathbf{K}$, $\mathbf{R}$ in vektor t, t.j.

$$\mathbf{P} = \mathbf{K}\left[\mathbf{R}|\mathbf{t}\right]. \tag{2.5}$$

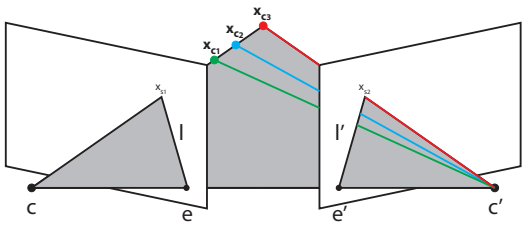
Teoretična kamera s točkovno odprtino ima tako 9 prostostnih stopenj(DoF). Tri prostostne stopnje zavzema rotacija, tri translacija. Z dvema opišemo pozicijo v slikovni ravnini kjer slednjo preseka goriščna os. Zadnja prostostna stopnja je namenjena goriščni razdalji. Ker so digitalne kamere le približek teoretičnega modela, pa za opis fotografij potrebujemo še tri DoF. Z njimi je opisana relacija med slikama, ki jih tvorita teoretični model in digitalna kamera. Tako digitalna kamera uvaja skalirni faktor, radialno popačenje in asimetrijo senzorja.

2.3 Epipolarna geometrija

Tako kot ljudje se tudi metode umetnega zaznavanja poslužujejo uporabe binokularne paralakse za razbiranje strukture prostora. Za razliko od naše percepcije pa zaznavanje prostora pri umetnem zaznavanju omogoča več DoF. Epipolarna geometrija opisuje stereo sistem kamer, ki sta poljubno orientirani in oddaljeni med seboj.

Slika 2.3 predstavlja stereo sistem kamer. Z $\mathbf{c}$ in $\mathbf{c}'$ sta označeni izhodišči koordinatnih sistemov kamer. Presečišče vektorjev $\mathbf{x_c}$ in $\mathbf{x'_c}$ predstavlja triangulirano točko vidno v obeh pogledih. Z $\mathbf{e}$ in $\mathbf{e}'$ sta označeni epipolni točki. S sivo barvo je obarvan trikotnik imenovan epipolna ravnina, saj povezuje triangulirano točko z izhodiščema kamer kjer prebada slikovno ravnino pa so določene epipolarne linije in epipoli. Na liniji $\mathbf{l}'$ se nahajajo možne preslikave točke $\mathbf{x_s}$.

Relacijo med točko $\mathbf{x_s}$ v prvi kameri in točko $\mathbf{x'_s}$ v drugi definira esencialna



Slika 2.3: Geometrija več pogledov.

matrika (angl., essential matrix) $\mathbf{E}$, z enačbo

$$\mathbf{E} = [\mathbf{t}]_{\mathbf{x}}\mathbf{R}. \tag{2.6}$$

V Enačbi (2.6) $\mathbf{R}$ označuje rotacijsko matriko, $[\mathbf{t}]_{\mathbf{x}}$ pa označuje matrični zapis vektorja razdalje med izhodiščema kamer. Ta omogoča matrični zapis vektorskega produkta med vektorjem $\mathbf{t}$ in matriko $\mathbf{R}$ [16].

Za opis sistema kamer v merskih enotah trirazsežnega prostora je potrebno poznati notranje parametre kamere. V primeru, da ne poznamo kalibracijskih matrik kamer, lahko zastavimo problem kot ujemanje v pikslih namesto v metrih. To povezavo opisuje fundamentalna matrika $\mathbf{F}$ (angl., fundamental matrix)

$$\mathbf{F} = \mathbf{K}^{-\mathbf{T}}\mathbf{E}\mathbf{K}^{-\mathbf{1}}. \tag{2.7}$$

S pomočjo slednje poljubno točko $\mathbf{x_s}$ prvega pogleda lahko preslikamo v pripadajočo epipolarno linijo $\mathbf{l}'$ v drugem pogledu

$$\mathbf{l}' = \mathbf{F}\mathbf{x_s}. \tag{2.8}$$

2.3.1 Ocenjevanje fundamentalne matrike

Fundamentalno matriko lahko ocenimo s pomočjo točk, ki se ujemajo med slikovnima ravninama. Poljubna točka prvega pogleda preslika v epipolarno

linijo drugega pogleda. Za točko drugega pogleda (x'_s), ki leži nekje na tej liniji lahko zapišemo

$$\mathbf{x_s'^T l'} = \mathbf{x_s'^T F x_s} = 0. \tag{2.9}$$

Za izračun fundamentalne matrike se zelo pogosto uporablja preprost 8-točkovni algoritem [16]. Iz slik se izbere 8 parov točk. Za normalizacijo točk se najprej izračunata njihova centra v levem in desnem pogledu. Nato se uniformno skalirajo točke okoli pripadajočih centrov, tako da je njihova povprečna oddaljenost od centra $\sqrt{2}$. To transformacijo točke $\mathbf{x_s}$ iz homogenega koordinatnega sistema v normaliziran sistem $\tilde{\mathbf{x}}_\mathbf{s}$ zapišemo kot množenje z matriko $\mathbf{T}$

$$\tilde{\mathbf{x}}_\mathbf{s} = \mathbf{T x_s}. \tag{2.10}$$

Nato sestavimo enačbo fundamentalne matrike $\tilde{\mathbf{F}}$, ki povezuje med seboj normalizirane točke

$$\tilde{\mathbf{x}}_\mathbf{s}'^\mathbf{T} \tilde{\mathbf{F}} \tilde{\mathbf{x}}_\mathbf{s} = 0. \tag{2.11}$$

Enačbo (2.11) lahko predrugačimo in zapišemo kot

$$\begin{bmatrix} uu' & uv' & u & vu' & vv' & v & u' & v' & 1 \end{bmatrix} \begin{bmatrix} \tilde{F}_{11} \\ \tilde{F}_{12} \\ \tilde{F}_{13} \\ \tilde{F}_{21} \\ \tilde{F}_{22} \\ \tilde{F}_{23} \\ \tilde{F}_{31} \\ \tilde{F}_{32} \\ \tilde{F}_{33} \end{bmatrix}, \tag{2.12}$$

kjer so u, v, u' in v' vrednosti vektorjev

$$\tilde{\mathbf{x}}_\mathbf{s} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tilde{\mathbf{x}}'_\mathbf{s} = \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix}. \tag{2.13}$$

Spremenljivke $\tilde{F}_{rc}$ pa v Enačbi (2.12) označujejo vrednosti matrike $\tilde{\mathbf{F}}$ na pozicijah r in c. Z r je označena številka vrstice matrike z c pa številka

stolpca. Na ta način lahko za osem normaliziranih točk zapišemo sistem osmih enačb z osmimi neznankami

$$\mathbf{A}\tilde{\mathbf{f}} = \begin{bmatrix} u_1u'_1 & u_1v'_1 & u_1 & v_1u'_1 & v_1v'_1 & v_1 & u'_1 & v'_1 & 1 \\ u_2u'_2 & u_2v'_2 & u_2 & v_2u'_2 & v_2v'_2 & v_2 & u'_2 & v'_2 & 1 \\ u_3u'_3 & u_3v'_3 & u_3 & v_3u'_3 & v_3v'_3 & v_3 & u'_3 & v'_3 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_8u'_8 & u_8v'_8 & u_8 & v_8u'_8 & v_8v'_8 & v_8 & u'_8 & v'_8 & 1 \end{bmatrix} \begin{bmatrix} \tilde{F}_{11} \\ \tilde{F}_{12} \\ \tilde{F}_{13} \\ \tilde{F}_{21} \\ \tilde{F}_{22} \\ \tilde{F}_{23} \\ \tilde{F}_{31} \\ \tilde{F}_{32} \\ \tilde{F}_{33} \end{bmatrix} = 0. \quad (2.14)$$

Z izračunom SVD dekompozicije matrike $\mathbf{A}$ poiščemo lastne vrednosti in lastne vektorje. Izmed lastnih vektorjev vzamemo tistega ($\mathbf{v}$) kateremu pripada najmanjša lastna vrednost in ga pretvorimo v matriko

$$\tilde{\mathbf{F}} = \begin{bmatrix} v_1 & v_2 & v_3 \\ v_4 & v_5 & v_6 \\ v_7 & v_8 & v_9 \end{bmatrix}. \quad (2.15)$$

Izračunano matriko $\tilde{\mathbf{F}}$ razstavimo z SVD. Najmanjšo lastno vrednost nastavimo na 0 in zmnožimo nazaj z matrikama singularnih vektorjev, tako zagotovimo, da ima končna matrika rang 2. Novodobljena fundamentalna matrika opisuje relacijo med normaliziranimi točkami. Če jo množimo z matriko normalizacije točk levega pogleda $\mathbf{T}$ in matriko normalizacije točk desnega pogleda $\mathbf{T}'$

$$\mathbf{F} = \mathbf{T}'^{\mathbf{T}}\tilde{\mathbf{F}}\mathbf{T}, \quad (2.16)$$

dobimo fundamentalno matriko za točke slikovne ravnine.

2.4 Struktura iz gibanja in prilagajanje snopa

Zaenkrat smo opisali geometrijo le enega in dveh pogledov. Geometrija večih pogledov se ne razlikuje dosti od sistema opisanega v Poglavju 2.3, saj lahko

problem večih pogledov opišemo kot N sistemov epipolarne geometrije. Iz esencialne matrike, ki smo jo opisali v prejšni sekciji, moramo dobiti pozicijo in orientacijo, oziroma pozo pogledov. Ker vemo, da esencialna matrika vsebuje tako translacijo kot rotacijo(glej Poglavje 2.3), jo lahko razstavimo nazaj na ti dve komponenti. SVD razcep matrike E [16] da dve različni rešitvi za rotacijo in dve za translacijo. Možne končne pozicije in rotacije kamer so zato štiri. Opisane so z enačbami

$$\begin{aligned} \mathbf{R} &= \begin{pmatrix} \mathbf{UWV^T} \\ \mathbf{UW^TV^T} \end{pmatrix}, \\ \mathbf{t} &= \begin{pmatrix} \mathbf{u_3} \\ \mathbf{-u_3} \end{pmatrix}. \end{aligned} \tag{2.17}$$

Unitarni matriki $\mathbf{U}$ in $\mathbf{V^T}$ omenjeni v Enačbi (2.17) dobimo s SVD razcepom esencialne matrike

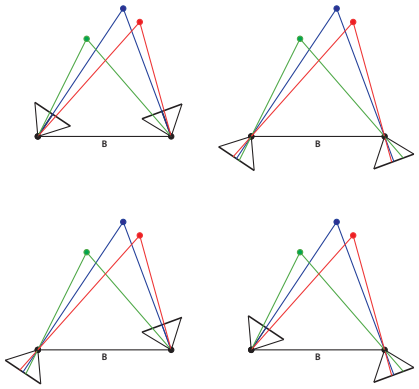
$$SVD(\mathbf{E}) = \mathbf{U\Sigma V^T}. \tag{2.18}$$

$\mathbf{W}$ je poševnosimetrična matrika, vektor $\mathbf{u_3}$ pa je tretji stolpec matrike $\mathbf{U}$, torej

$$\begin{aligned} \mathbf{W} &= \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \\ \mathbf{U} &= [\mathbf{u_1}|\mathbf{u_2}|\mathbf{u_3}]. \end{aligned} \tag{2.19}$$

Med štirimi možnimi orientacijami kamer nato izberemo tisto, ki opisuje sistem z geometrijo pred obema pogledoma. To naredimo s triangulacijo ene od točk. Če triangulirana točka leži pred obema kamerama, kot je prikazano na Sliki 2.4, smo našli pravilno konfiguracijo. Dobljene poze vseh pogledov nato povežemo skupaj.

Natančnost rekonstrukcije je močno odvisna od pravilnosti estimirane fundamentalne in esencialne matrike. Če so vhodne slike šumne ali vsebujejo



Slika 2.4: Možni pari po razcepu matrike E.

radialno popačenje potem ni nujno, da za vsako točko $\mathbf{x_c}$ velja, da se preslika v $\mathbf{x_s}$. Kot zadnji postopek cevovoda rekonstrukcije je zato predlagana prilagoditev snopa [16]. Obstajata dva načina prilagajanja, inkrementalni [21] in globalni [21]. Inkrementalna struktura iz gibanja v prvi iteraciji triangulira točke iz dveh pogledov. V vsaki nadaljnji iteraciji doda nov pogled, triangulira točke med njim in ostalimi pogledi v sceni ter naredi prilagoditev snopa. Globalne metode strukture iz gibanja pa poskušajo za vse kamere popraviti začetne parametre. Prilagoditev snopa izvedejo nad vsemi pogledi hkrati.

Namen prilagajanja snopa je minimizirati reprojekcijsko napako med resnično pozicijo točke $\mathbf{x_s}$ na slikovni ravnini in napovedano pozicijo $\tilde{\mathbf{x}}_\mathbf{s}$ te točke

$$\tilde{\mathbf{x}}_\mathbf{s} = \mathbf{P}\mathbf{x_c}. \tag{2.20}$$

Najbolj uporabljena metoda je Levenberg–Marquardt [23] [28]. Zasnovana je kot združitev gradientnega spusta in Gauss-Newtonove metode [21] s čimer doseže konvergenco tudi v primeru močnega odstopanja vhodnih podatkov. Gauss-Newtnova metoda izvira iz enostavnejše Newtnove metode [21]. Slednja s pomočjo rekurenčne formule v vsakem koraku poišče boljšo aproksimacijo ničli funkcije.

Predpostavimo, da imamo poljubno funkcijo $f(x_n)$, ki je definirana nad

realnimi števili. S pomočjo trenutne aproksimacije x_n želimo poiskati ničlo te funkcije. Iskanje ničle lahko izvajamo z Newtnovo metodo, kjer s pomočjo trenutne aproksimacije v naslednji iteraciji najdemo boljšo

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}. \tag{2.21}$$

V Enačbi (2.21) x_n predstavlja trenutno aproksimacijo, x_{n+1} pa vrednost naslednje aproksimacije. Funkcija $f(x_n)$ je poljubna, mora pa biti definirana nad realnimi števili, $f'(x_n)$ pa je njen odvod.

Z sledečimi enačbami smo zapisali formulacijo splošnega primera prilagoditve snopa z uporabo Newtnove metode. Predpostavimo, da imamo problem z m trianguliranimi točkami ter množico β, ki vsebuje n parametrov kamer, ki jih želimo prilagoditi. Parametri, ki jih ocenjujemo pri prilagajanju snopa so rotacija in translacija pogledov. Vrednost, ki jo želimo minimizirati je označena s S in predstavlja vsoto kvadratov reprojekcijskih napak

$$S = \sum_{i=1}^{m} r_i = \sum_{i=1}^{m} d(\mathbf{x_{si}}, \tilde{\mathbf{x}}_{\mathbf{si}}). \tag{2.22}$$

V Enačbi (2.22) je z $d(\mathbf{x_s}, \tilde{\mathbf{x}}_{\mathbf{s}})$ oziroma r_i označena razdalja med pravo pozicijo točke i na slikovni ravnini ter njeno napovedano pozicijo. Minimum spremenljivke S je dosežen, ko je skupni gradient enak ničli. Ker imamo n parametrov je zato tudi n gradientov

$$\frac{\delta S}{\delta \beta_j} = 2 \sum_i r_i \frac{\delta r_i}{\delta \beta_j} = 0. \tag{2.23}$$

Enačba (2.23) opisuje gradient S pri parametru β_j, kjer je $j = (1, 2, \ldots, n)$. V rekurenčni Enačbi (2.24) je prikazan izračun vrednosti parametrov v naslednji iteraciji. Oznaka $\mathbf{H}$ predstavlja Hessovo matriko parcialnih odvodov, oznaka $\mathbf{g}$ pa vektor gradienta

$$\beta_{n+1} = \beta_n - \mathbf{H^{-1}g}. \tag{2.24}$$

Gauss-Newtnova metoda aproksimira $\mathbf{H}$ in $\mathbf{g}$ s pomočjo Jacobijeve matrike parcialnih odvodov preostankov($\mathbf{J_r}$). Vrstica r in stolpec c matrike $\mathbf{J_r}$ vsebujeta vrednosti $\frac{\delta r_i}{\delta \beta_j}$. Aproksimaciji $\mathbf{H} \approx 2\mathbf{J_r^T J_r}$ in $\mathbf{g} = 2\mathbf{J_r^T} r$ nato vstavimo v

osnovno rekurenčno Enačbo (2.24), da dobimo Gauss-Newtnovo rekurenčno enačbo

$$\beta_{n+1} = \beta_n - (J_r^T J_r)^{-1} J_r^T r. \tag{2.25}$$

2.5 Stereo z več pogledi

Stereo z več pogledi opisuje sistem, ki v svoji osnovi uporablja zakone epipolarne geometrije. Paroma med kamerami lahko rekonstruiramo del scene. Vse delne rekonstrukcije nato združimo v enotno predstavitev. Obstajajo metode, ki omogočajo direktno triangulacijo s pomočjo vhodnih slik. V primeru, da želimo triangulirati s pomočjo globinske slike, pa moramo pred izvedbo algoritma popraviti vhodne slike. Ta postopek opišemo v Poglavju 2.5.1.

2.5.1 Popravek pogledov

V primeru, da se poslužujemo triangulacije s pomočjo globinske slike, morata pogleda biti vzporedna. Verjetnost da bosta poljubni dve kameri bili vzporedni je zelo majhna, zato je treba slike predhodno popraviti(angl. stereo rectification). Posebna lastnost stereo konfiguracije z vzporednimi pogledi je, da so epipolarne linije vodoravne v obeh pogledih. Iskanje ujemajočih točk tako poteka le vzdolž dimenzije x na slikovni ploskvi. Kot smo omenili v Poglavju 2.2.1, homogene koordinate omogočajo zapis točke v neskončnosti. Če slikovno ravnino enega od pogledov stereo sistema poravnamo z linijo, ki povezuje njuni gorišči se epipol premakne v neskončnost, epipolarne linije pa se poravnajo. Prav ta lastnost homogenega koordinatnega sistema je izkoriščena pri poravnavi dveh pogledov v postopku, ki ga opisujeta Hartley in Zisserman [16]. Za desni pogled para kamer poskušamo najti tako matriko $\mathbf{H}'$, ki bo epipolarno točko tega pogleda preslikala v neskončnost. Na Sliki 2.5 je ilustriran popravek pogledov, z Enačbo (2.27) pa podajamo računsko opredelitev.

Če epipol drugega pogleda opisuje točko $\mathbf{e}'$, ki se nahaja v

$$\mathbf{e}' = \begin{bmatrix} h_x \\ h_y \\ 1 \end{bmatrix}, \tag{2.26}$$

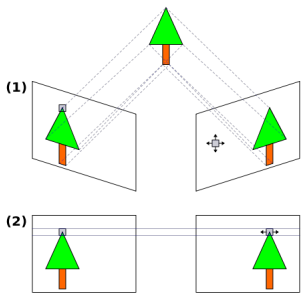
potem jo množenje z ustrezno matriko $\mathbf{H}'$

$$\mathbf{H}'\mathbf{e}' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{-1}{h_x+h_y} & \frac{-1}{h_x+h_y} & 1 \end{bmatrix} \begin{bmatrix} h_x \\ h_y \\ 1 \end{bmatrix} = \begin{bmatrix} h_x \\ h_y \\ 0 \end{bmatrix}, \tag{2.27}$$

preslika v vektor katerega zadnja komponenta ima vrednost 0. Preslikana točka se zato nahaja v neskončnosti, epipolarne linije pa postanejo vzporedne(glej Poglavje 2.2.1). Ko obrnemo drugo kamero v položaj kjer so epipolarne linije vzporedne moramo poravnati tudi prvi pogled. V tem primeru je dovolj, da minimiziramo

$$S = \sum_{i}^{n} d(\mathbf{H}\mathbf{x}_{\mathbf{si}}, \mathbf{H}'\mathbf{x}'_{\mathbf{si}})^2. \tag{2.28}$$

Spremenljivka S v Enačbi (2.28) predstavlja seštevek kvadratov razdalj, $d(,)^2$, med n točkami dveh slikovnih ravnin. Po izračunu matriki $\mathbf{H}$ in $\mathbf{H}'$ apliciramo na točke slikovne ravnine. Preslikane točke imajo enako koordinato y v poravnanih pogledih.



Slika 2.5: Ilustracija, ki prikazuje kako popravek pogledov poenostavi iskanje korespondenc pri stereo ujemanju. Slika povzeta po [43].

Poglavje 3

Metode

V tem poglavju predstavljamo različne metode posamezne stopnje cevovoda rekonstrukcije. Opisane metode so tudi implementirane in ovrednotene. Evalvacija le-teh je na voljo v Poglavju 5.

3.1 Zaznava in opis značilnic

Najti dobre značilnice je najpomembnejši del cevovoda. Brez dovolj dobrih značilnic in opisnikov ne moremo vzpostaviti povezave med pogledi. V tem poglavju navajamo delovanje treh detektorjev. V Poglavju 3.1.1 je opisan detektor SIFT [25]. Poglavje 3.1.2 vsebuje opis detektorja SURF [5]. Zadnje Poglavje 3.1.3 pa opisuje detektor ORB [37]. Poleg vsakega opisa detektorja podamo tudi splošne prednosti in slabosti.

3.1.1 Detektor SIFT [25]

SIFT [25] je detektor in vektorski opisnik. Zaznava značilnic poteka z konvolucijo filtrov nad sivinsko sliko. Filter, ki se uporablja za zaznavo značilnic, aproksimira LoG, Laplacov operator apliciran na sliko, ki jo predhodno zameglimo z Gaussovim filtrom. Prednost aproksimacije LoG z razliko dveh različno velikih Gaussovih filtrov (DoG) je separabilnost jedra filtra, kar omogoča hitrejšo izvajanje. Iskanje se izvaja nad piramido različnih veliko-

sti slike. S tem algoritem zagotovi invariantnost na velikost. Po konvoluciji se regije, kjer se nahajajo lokalni ekstremi, privzamejo kot možne lokacije značilnic. V okolici značilnice se nato izračunata moč gradienta ter smer. Ustvari se histogram orientacij. Vse vrednosti, ki presegajo 80% vrednosti najvišjega stolpca, se tudi upoštevajo pri računanju orientacije. Po normalizaciji regije opisnik vzame okolico značilnice velikosti 16×16. Razdeli jo v šestnajst kvadratov velikosti 4×4. Za vsak kvadrat se nato izračuna histogram 8-ih orientacij.

Prednost opisanega algoritma je, da je invarianten na spremembo v velikosti in orientaciji značilnice. Z uporabo histograma orientacij zagotovi tudi zaznavo perspektivno popačenih regij. Invarianten je tudi na spremembo osvetlitve, saj se ne zanaša na sivinsko predstavitev značilnice temveč na gradient. Slabost algoritma, v primerjavi z novejšimi detektorji, pa je izvajalni čas. Prav tako je uporaba algoritma za komercialne namene plačljiva.

3.1.2 Detektor SURF [5]

Algoritem SURF [25] poskuša izboljšati največjo pomanjkljivost SIFT-a, torej čas izvajanja. Kjer SIFT aproksimira LoG z razliko različno velikih Gaussovih filtrov, SURF za aproksimacijo uporabi škatlast filter. Pred začetkom izvajanja algoritma se iz vhodne slike izračuna integralna slika. Ta omogoča hitro računanje z pravokotnimi filtri, saj za izračun vrednosti celotne regije zadošča poznati vrednosti integralne slike v ogliščih pravokotnega filtra. Iz zaznanih regij opisnik nato ustvari opisnik. Okolico vsake regije razdeli v 16 pod-regij. Za vsako pod-regijo se s pomočjo integralne slike izračunajo valovni odzivi v vodoravni in navpični smeri. Seštevki teh se nato uporabijo pri sestavi opisnega vektorja.

Prednost opisanega algoritma je hitrost izvajanja. Z uporabo integralne slike in aproksimiranjem LoG s pomočjo škatlastega filtra doseže mnogo krajši izvajalni čas. Slabost je slabša invarianca na večje spremembe v osvetlitvi. Prav tako kot pri SIFT-u je uporaba algoritma za komercialne namene plačljiva.

3.1.3 Detektor ORB [37]

Detektor ORB [37] združuje algoritma detektorja FAST [44] in opisnika BRIEF [7]. Tako kot algoritma SIFT in SURF tudi ORB uporablja piramido slik za detekcijo značilk v več razsežnostih. Z uporabo algoritma FAST zazna močno teksturirane regije. Algoritem FAST potrdi zaznavo, če se v njegovi okolici nahaja n stikajočih se pikslov, ki so vsi svetlejši ali temnejši od trenutnega piksla. Dobljene značilke se filtrirajo s pomočjo detektorja ogljišč Harris [15]. ORB tudi izračuna intenzitetno težišče. Smer med središčem zaznave in težiščem predstavlja orientacijo značilnice. Za opis značilnic ORB uporablja modificirano verzijo algoritma BRIEF, saj originalna implementacija slednjega ni invariantna na rotacijo. Za vsak set n binarnih testov na lokacijah x_i, y_i se vzpostavi matrika velikosti $2*n$ v kateri so pozicije teh lokacij. S pomočjo prej izračunane orientacije se izračunajo nove pozicije točk in s tem opisnik postane rotacijsko invarianten.

Prednost detektorja ORB je, da izvajanje lahko poteka v realnem času. Največja slabost, poleg neinvariantnosti na osvetlitev, je način detekcije značilnic. Detektorja SIFT in SURF zaznavata homogene regije, detektor ORB pa regije, ki ustrezajo kotom. Pogosto pomanjkanje takih regij na vhodnih sivinskih slikah povzroči, da detektor najde mnogo manj značilnic kot prej omenjena algoritma.

3.2 Ujemanje značilnic

Z detekcijo značilnic v sivinskih slikah najdemo regije za katere je najbolj verjetno, da bodo prisotne v več pogledih. Samo zaznavanje teh regij pa nam ne koristi preveč, saj bi radi vedeli katere izmed zaznanih regij so prisotne v več kot enem pogledu. V Poglavju 3.2.1 so opisani postopki za računanje ujemanj. Poglavje 3.2.2 vsebuje opis enostavnega algoritma za iskanje ujemanj značilnic. Poglavje 3.2.3 pa vsebuje opise algoritmov približnega iskanja ujemanj.

3.2.1 Razdalje

Da lahko določimo ujemanje dveh značilnic moramo definirati relacije, s katerimi bomo lahko opisali podobnost opisnikov. Našteli smo tri postopke, s katerimi se računajo ujemanja.

Hammingova razdalja

Binarni opisniki opisujejo okolico značilnice z binarnim zaporedjem določene velikosti. Ker so deskriptorji binarna zaporedja enakih dolžin lahko med njimi računamo Hammingovo razdaljo. Ta predstavlja število mest v katerih se razlikujeta zaporedji

$$Hamming\left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}\right) = \sum_{k=1}^{n} (x_k \neq y_k). \tag{3.1}$$

Z x_1 do x_n so označeni biti prvega zaporedja. Bite drugega zaporedja označujejo biti y_1 do y_n. Dobljeno Hammingovo razdaljo delimo z dolžino binarnega zaporedja. Končna razdalja je vrednost v intervalu $[0, 1]$.

L_1 norma

L_1 norma se uporablja pri deskriptorjih predstavljenih z vektorjem realnih števil. Med vsemi vrednostmi, ki sestavljajo deskriptor se izračuna absolutna razdalja. Seštevek absolutnih razdalj predstavlja razdaljo med opisnikoma

$$L_1\left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}\right) = \sum_{k=1}^{n} |x_k - y_k|. \tag{3.2}$$

Vektorja sestavljajo realna števila označena z x_1 do x_n ter y_1 do y_n.

L_2 norma

L_2 norma, oziroma evklidska razdalja, se prav tako lahko računa pri deskriptorjih opisanih z realnimi števili. Računa se kot koren seštevkov kvadratov razdalj

$$L_2\left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}\right) = \sqrt{\sum_{k=1}^{n}(x_k - y_k)^2}. \tag{3.3}$$

Vektorja sestavljajo realna števila označena z x_1 do x_n ter y_1 do y_n.

3.2.2 Izčrpno ujemanje

Izčrpno ujemanje primerja vsak deskriptor iz prvega pogleda z vsakim deskriptorjem drugega pogleda. Ujemata se deskriptorja med katerima je najmanjša razdalja.

Izčrpno ujemanje vedno najde najboljša ujemanja. Zaradi primerjanja vsakega deskriptorja prvega pogleda z vsakim deksriptorjem drugega pa je algoritem časovno potraten. Primeren je le za rekonstrukcije z malo pogledi, saj ne potrebuje dodatnega časa za inicializacijo.

3.2.3 Približno iskanje bližnjih sosedov

Za večje rekonstrukcije je ujemanje deskriptorjev ozko grlo, tako da v primeru veliko vhodnih slik lahko sklenemo kompromis med natančnostjo ujemanja ter hitrostjo izvajanja. Algoritmi približnega iskanja lahko pospešijo izvajanje za več redov velikosti in le v majhnem delu primerov vrnejo napačno ujemanje. V Poglavju 3.2.3 predstavimo približno iskanje binarnih in vektorskih opisnikov na osnovi naključno generiranih dreves.

Iskanje binarnih deskriptorjev

Približno iskanje bližnjih sosedov binarnih deskriptorjev [31] deluje na osnovi naključno generiranih dreves. Algoritem iz vhodnih binarnih zaporedji izbere

K naključnih opisnikov, kjer je K faktor razvejitve drevesa. Izbrani opisniki predstavljajo vrednost vozlišča. Preostala binarna zaporedja so razporejena v vozlišča, glede na Hammingovo razdaljo do opisnika, ki predstavlja vozlišče. Razvejitev se ponavlja rekurzivno, dokler vozlišče vsebuje več kot L opisnikov. V listih drevesa se ujemanje opisnikov preverja linearno.

Iskanje po drevesih poteka vzporedno. Najprej se pregledajo vsa drevesa. V vsaki iteraciji iskanja po drevesu se poišče najbližje vozlišče. Neraziskana vozlišča se dodajo v prioritetno vrsto. Ko iskanje doseže list, preveri razdaljo do vseh opisnikov, ki se nahajajo v njem. Po prvotnem pregledu, se za posamezno drevo iz prioritetne vrste izbere najvišje ocenjeno vozlišče iz katerega se nato nadaljuje iskanje.

Točno število in konfiguracija dreves sta pogojena z več faktorji. Za preciznost višjo od 85% se priporoča uporaba 4-8 naključno generiranih dreves. Maksimalna velikost lista naj ne presega 150 deskriptorjev in razvejitveni faktor naj ne presega 16 [31].

Prednost približnega iskanja binarnih ujemanj je, da je izvajanje lahko pospešeno za več velikostnih redov. Kot lahko sklepamo že iz imena iskanje ne vrne vedno istih deskriptorjev. Nekatera ujemanja, ki jih vrne so zato lahko napačna.

Iskanje vektorskih deskriptorjev

Za iskanje vektorskih opisnikov se uporabljata dve različno generirani drevesni strukturi [32]. Najbolj zanesljiva je metoda prioritetnega preiskovanja drevesa K-srednjih vrednosti. Pri izgradnji drevesa se vhodni podatki v vsakem vozlišču rekurzivno razvejijo v K različnih skupin. V vsako skupino uvrstimo med seboj najbolj podobne vektorje. Podobnost vektorjev se računa z evklidsko razdaljo. Po končani razvejitvi se začne preiskovanje. Ob vstopu v vozlišče se primerja razdalja trenutnega opisnika z preostalimi K opisniki. Neraziskane veje se dodajo v prioritetno vrsto. Ko preiskovanje doseže list, se nadaljuje preiskovanje v še ne-raziskani veji z najmanjšo evklidsko razdaljo.

KD drevesa [6], z naključno izbiro, se gradijo podobno kot navadna KD

drevesa. Glavna razlika je razvejitveni pogoj. Pri navadnih KD drevesih se posamezno vozlišče razveji glede na dimenzijo vektorja, ki ima največjo varianco. Pri drevesih z naključno izbiro pa se naključno izbere ena izmed N dimenzij z največjo varianco. Pri preiskovanju dreves se vzpostavi samo ena prioritetna vrsta, v katero se dodajajo neraziskane vozlišča iz vseh dreves. Vozlišča znotraj prioritetne vrste so urejena najprej po oddaljenosti do najbližjega lista ter nato po razdalji do vektorja s katerim preiskujemo. Stopnja aproksimacije je pogojena s številom listov preko vseh dreves, ki jih algoritem obišče.

Prednost približnega iskanja vektorskih ujemanj je, da je izvajanje lahko pospešeno za več velikostnih redov. Slabost približnega iskanja je, da iskanje ne vrne vedno istih deskriptorjev. Nekatera ujemanja, ki jih vrne so zato lahko napačna.

3.3 RANSAC

Metoda soglasja naključnega vzorca [12] omogoča ocenjevanje parametrov modela iz močno odstopajočih podatkov. Deluje po principu *generiraj in preveri*. Vzorec velikosti n(najmanjša potrebna velikost za opredelitev iskanega modela) je izbran naključno iz seta točk. Iz vzorca se nato generira hipoteza, ki se overi na preostalih N-n točkah.

Algoritem se izvaja dokler ni dosežen ustavitveni pogoj. Standardni ustavitveni pogoj izvajanja je pogojen z verjetnostjo p, ki predstavlja verjetnost pravilnosti končnega modela. Parameter p tudi določa po koliko iteracijah naj se ustavi izvajanje algoritma. Če je znan delež točk w, ki pripadajo pravemu modelu potem je verjetnost izbire vzorca brez odstopajočih točk w^n. Verjetnost, da naključno izbran vzorec vsebuje vsaj eno točko, ki ne pripada modelu pa je $(1-w^n)$. Da lahko z gotovostjo p zagotovimo čistost vsaj enega izmed tvorjenih vzorcev, mora $1-(1-w^n)^k$ preseči p. Število iteracij je tako pogojeno z enačbo

$$k = \left\lceil \frac{log(i-p)}{log(1-w^n)} \right\rceil. \tag{3.4}$$

Ker delež pripadajočih točk ponavadi ni znan vnaprej, se število iteracij posodablja med izvajanjem algoritma. Izračuna se iz deleža točk, ki pripadajo hipotezi z največjo podporo. Zaradi dela z realnimi podatki, ki so pogosto šumni, pa se pri overitveni fazi uporabi dodaten parameter t. Slednji predstavlja maksimalno odstopanje pozicije i-te točke od pričakovane. Če je razlika med pozicijama manjša od t, lahko i-to točko uvrstimo med pripadajoče.

3.3.1 Test $T_{d,d}$ [30]

Test $T_{d,d}$ [30] je bil zasnovan z namenom pred-evalvacije. Omogoča hitro filtriranje slabih hipotez. Iz N točk sta naključno izbrana vzorca velikost n in d. Iz seta n točk generiramo hipotezo in jo nato overimo na ostalih d točkah. V primeru, da vse točke iz seta velikosti d potrdijo hipotezo, se ta overi še na preostalih $N - (n + d)$ točkah. Za optimalno delovanje avtor algoritma priporoča vrednost parametra d in sicer $d = 1$ [30].

Test $T_{d,d}$ pripomore k pohitritvi izvajanja, saj ni potrebno preveriti odstopanja celega seta točk v vsaki iteraciji. Zaradi naključne izbire setov obstaja verjetnost, da v pred-evalvacijski fazi zavržemo pravilno hipotezo. Verjetnost, da zavržemo pravilno hipotezo, narašča z deležom odstopajočih točk. Test $T_{d,d}$ zato pogosto generira več hipotez kot običajni RANSAC.

3.3.2 Locally Optimized RANSAC [9]

Lokalno optimizirano soglasje naključnega vzorca [9] poskuša skrajšati čas izvajanja in najti večje število pripadajočih točk z notranjo zanko. V primeru, da je najdena hipoteza, ki bolje opisuje model kot prejšnja, se izvede notranja zanka. V tej zanki se poskuša izboljšati trenutno najboljšo hipotezo samo z uporabo točk, ki ji pripadajo.

V notranji zanki se I-krat izvede metoda RANSAC. Ker vse točke nad katerimi deluje notranji RANSAC pripadajo isti hipotezi, za računanje novih parametrov lahko vzamemo več kot minimalno število točk potrebnih za opis

modela. Avtorji predlagajo vrednost parametra $I == 20$ [36].

Prednost notranje zanke je v tem, da v primeru, ko najboljša hipoteza opisuje pravilni model jo notranja zanka izboljša in s tem najde točke, ki jih prvotna hipoteza morda ni. Lokalno optimiziran RANSAC tako ponavadi najde več neodstopajočih točk in hitreje zmanjša število potrebnih iteracij. Slabost dotičnega algoritma je, da v primeru večkratnega vzorčenja nepripadajočih točk z večjo podporo kot jo ima trenutna hipoteza notranja zanka znatno prispeva k času izvajanja.

3.3.3 PROSAC [8]

Večina različic originalnega algoritma predpostavi, da o točkah ni znanih nobenih predhodnih informacij, kar ne velja pri problemih kot je iskanje fundamentalne matrike. Pri iskanju ujemanja značilnic je na voljo funkcija s pomočjo katere se oceni podobnost opisnikov. Algoritem progresivnega vzorčenja [8] deluje po predpostavki, da ujemanja z večjo podobnostjo bolj verjetno tvorijo čiste vzorce. PROSAC torej vzorči v drugačnem zaporedju kot prej omenjene metode.

Naj bo T_N število vzorcev, ki jih tvori običajni RANSAC in n minimalno število točk potrebnih za opis modela. Verjetnost, da bo vzorec velikosti s vseboval vseh n točk, ki so izbrane iz seta velikosti N je:

$$\frac{\binom{s}{n}}{\binom{N}{n}}. \tag{3.5}$$

Število iteracij v katerih bo izbral točke samo iz vzorca je zato

$$T_s = T_N \frac{\binom{s}{n}}{\binom{N}{n}}. \tag{3.6}$$

Naslednika T_s je mogoče definirati z rekurenčno enačbo $T_{s+1} = \frac{s+1}{s+1-m} T_s$. Ker ta enačba ne tvori nujno celoštevilskih rešitev, je predlagana celoštevilska rekurenčna enačba za število iteracij $T'_{s+1} = T'_s + \lceil T_{s+1} - T_s \rceil$. Ko v setu dosežemo število iteracij T'_s, vanj dodamo naslednjo najboljše ujemanje ter povečamo število iteracij na T'_{s+1}.

Prednost algoritma PROSAC je, da dosega velike pohitritve zaradi progresivnega vzorčenja s pomočjo ocenitvene funkcije. Pomankljivost algoritma je, da je izvajanje odvisno od zanesljivosti ocenitvene funkcije. Prav tako je način naključne izbire podvržen napakam. V poročilu komparativne analize različic RANSAC-a [36] je navedeno, da višje ocenjena ujemanja v veliko primerih ležijo na isti prostorski strukturi(v primeru ocenjevanja fundamentalne matrike je to ploskev), kar lahko povzroči nastanek hipoteze, ki opisuje degenerirano konfiguracijo.

3.3.4 Preemptive RANSAC [35]

Slabost ostalih opisanih metod je nepredvidljiv čas izvajanja, saj je odvisen od naključnega generiranja hipotez. Algoritem [35] je bil zasnovan primarno za aplikacije, ki naj bi se izvajale v realnem času. S fiksnim številom tvorjenih hipotez in fiksnim številom točk nad katerimi se testirajo te hipoteze zagotavlja omejen čas izvajanja.

Najprej je generiranih M hipotez. Vsaka se nato vzporedno overi na podmnožici ujemanj velikosti B. Hipoteze nato uredimo po deležu pripadajočih ujemanj ter obdržimo le del teh. Izvajanje se zaključi, ko je vrednost funkcije $f(i) \leq 1$. Funkcija $f(i)$ določa koliko hipotez lahko obdržimo po evalvaciji vseh hipotez na i-tem ujemanju v podmnožici B

$$f(i) = \lfloor M 2^{-\lfloor \frac{i}{B} \rfloor} \rfloor. \tag{3.7}$$

Funkcijo lahko poenostavimo in jo predstavimo s številom iteracij. Poenostavljena različica $f'(it)$ originalne funkcije $f(i)$ opisane v [35] pa določa koliko hipotez naj obdržimo po zaključku trenutne iteracije k

$$f'(k) = \lfloor M 2^{-k} \rfloor. \tag{3.8}$$

Algoritem [35] uvaja omejen čas izvajanja, odvisen od parametrov M in B. Ker je časovno omejen je primeren za izvajanje v realnem času. Pomanjkljivost opisanega postopka je, da primeru majhnega deleža odstopajočih ujemanj evalvira preveč hipotez ter porabi več časa kot običajen RANSAC.

V primeru večjega deleža nepripadajočih ujemanj pa ni zagotovila, da katera od vnaprej generiranih hipotez zares opisuje pravilen model.

3.4 Stereo z več pogledi

Ko z ocenimo esencialno matriko in posledično pozicije ter rotacije kamer(glej Poglavje 2.4) lahko izvedemo gosto rekonstrukcijo. Kot smo omenili v Poglavju 2.5 se za triangulacijo oblaka točk v sistemu poravnanih kamer lahko uporablja globinska slika. V Poglavju 3.4.1 opišemo ta postopek. Za iskanje korespondenc med slikami v splošnem sistemu pa obstaja tudi način, ki deluje neposredno na vhodnih sivinskih slikah brez potrebe po predhodni transformaciji. Slednji postopek je opisan v Poglavju 3.4.2.

3.4.1 Globinska slika

Lastnost specifičnega sistema v katerem sta pogleda vzporedna je, da so epipolarne črte vodoravne ali navpične. To zelo poenostavi triangulacijo, saj da najdemo ujemanje rabimo primerjati točke le vzdolž dimenzije kateri pripada epipolarna črta. V tem poglavju predstavimo izčrpno ujemanje in naprednejšo metodo za generiranje globinskih slik.

Izčrpno ujemanje

Za vsako območje $\mathbf{P}$ prve slike velikosti $o \times o$ okoli neke točke primerjamo območja $\mathbf{P}'$, ki ležijo na pripadajoči epipolarni črti v drugem pogledu. Izmed vseh potencialnih ujemanj nato izberemo tisto, katere okolica najbolj ustreza okolici točke v prvem pogledu. Kot funkcija ujemanja se pogosto uporablja vsota kvadratov razlik

$$S = \sum_{i}^{o} \sum_{j}^{o} (\mathbf{P}_{ij} - \mathbf{P'}_{ij})^2, \tag{3.9}$$

saj je primerna za stereo ujemanje kjer se intenziteta svetlobe ne spreminja veliko med pogledi. Enačba (3.9) označuje seštevek kvadratov razlik elemen-

tov območji, predstavljenimi kot matriki $\mathbf{P}$ in $\mathbf{P}'$.

3.4.2 Gosto stereo ujemanje [46]

Algoritem gostega stereo ujemanja deluje neposredno na vhodnih slikah. Pri inicializaciji predpostavi, da smo predhodno med slikami poiskali korespondence. Pred začetkom izvajanja jih doda v prioritetno vrsto. Formula s katero oceni moč korespondence je srednje-vrednostno normalizirana križna korelacija(angl. Zero-Mean Normalized Cross-Correlation)

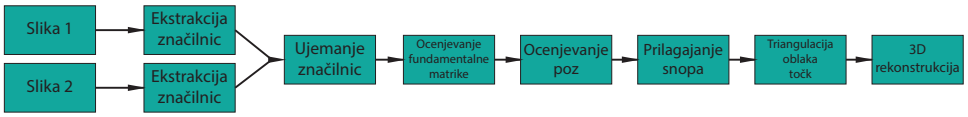
$$\begin{aligned} \overline{P} &= \frac{\sum_i^o \sum_j^o \mathbf{P}_{ij}}{\frac{1}{o^2}} \\ \rho(\mathbf{P}, \mathbf{P}') &= \sum_i^o \sum_j^o (\mathbf{P}_{ij} - \overline{P})(\mathbf{P'}_{ij} - \overline{P'}) \\ \sigma(\mathbf{P}) &= \sqrt{\frac{1}{o^2}\overline{P}} \\ ZNCC &= \frac{\frac{1}{o^2}\rho(\mathbf{P}, \mathbf{P}')}{\sigma(\mathbf{P})\sigma(\mathbf{P}')}. \end{aligned} \tag{3.10}$$

Enačba (3.10) prikazuje izračun srednje-vrednostno normalizirane križne korelacije(ZNCC). Spremenljivka o označuje velikost območji $\mathbf{P}$ in $\mathbf{P}'$. Po izračunu začetnih moči korespondenc iz prioritetne vrste vzame najvišje ocenjeno. Iz nje poskuša interpolirati na vse sosedne točke. Ujemanja novih točk preveri z ZNCC ter najvišja doda v prioritetno vrsto. Algoritem se zaključi, ko se vrsta izprazni.

Poglavje 4

Implementacija cevovoda

Implementacija cevovoda rekonstrukcije je potekala v programskem jeziku C++. Celotna implementacija zavzema ekstrakcijo značilnic, iskanje ujemanj značilnic, ocenitev fundamentalne matrike, ocenitev poz pogledov, prilagoditev snopa, triangulacijo oblaka točk in rekonstrukcijo. Diagram poteka opisanega cevovoda je prikazan na Sliki 4.1.



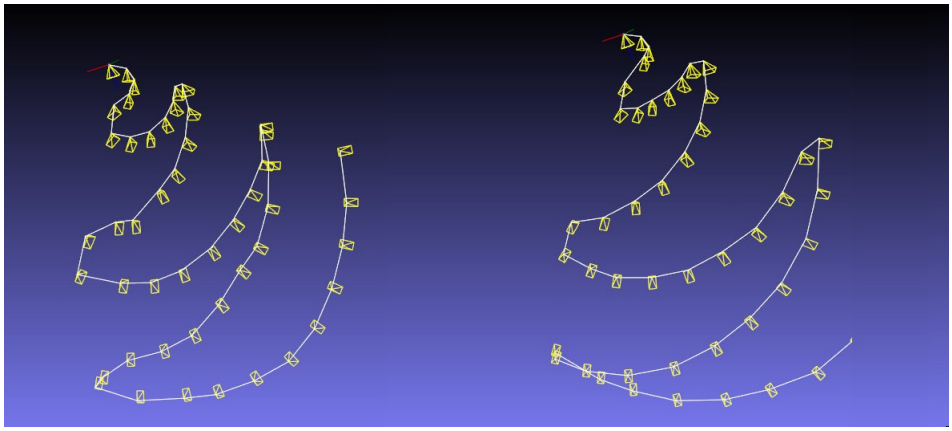
Slika 4.1: Diagram poteka implementiranega cevovoda rekonstrukcije.

Kot pomožne knjižnice smo uporabili Ceres solver [3], OpenCV [18] in Eigen [14]. Knjižnica FLANN [33] je implementirana v platformi OpenCV. Pri iskanju ujemanj med značilnicami smo uporabili dodaten parameter, s katerim smo lahko odstranili veliko lažnih ujemanj. Če je ocena najboljšega ujemanja u_n

$$u_n > u_d p, \tag{4.1}$$

kjer u_d opisuje oceno drugega najboljšega ujemanja, p pa nastavljiv parameter, potem to ujemanje zavržemo. Po priporočilu raziskovalca in izumitelja detektorja SIFT [25] smo vrednost parametra p nastavili na 0.8. S tem smo se znebili mnogo nepravilnih ujemanj.

Pri implementaciji algoritmov družine RANSAC [12] smo si pomagali s publikacijo [36]. Za boljšo ocenitev preslikav smo na ocenjenih fundamentalnih matrikah uporabili že implementirano metodo LMedS [29], kar je izboljšalo natančnost ocenjenih pozicij in rotacij kamer ter omogočilo konvergenco pri prilagajanju snopa. Pot gibanja kamere, prikazano na Sliki 4.2, smo primerjali pred in po uporabi omenjenega algoritma.



Slika 4.2: Optimizacija fundamentalnih matrik. Leva slika prikazuje ocenjeno trajektorijo kamere pri zajemu slik enega od setov iz podatkovne zbirke [19]. Popravljena pot v drugem pogledu bolje opisuje realne poze kamer. Skupno odstopanje sistema je tako manjše, kar nam omogoči uporabo več kamer pri končni rekonstrukciji.

Za ocenitev zunanjih parametrov kamer smo uporabili inkrementalno metodo SfM. V prvi iteraciji smo za par zanesljivih pogledov triangulirali točke in nato naredili prilagoditev snopa. V vsaki naslednji iteraciji smo dodali novo kamero, triangulirali točke in naredili prilagoditev snopa. Kamere pri katerih prilagoditev snopa ni uspela smo spustili iz končne rekonstrukcije.

Implementirali smo tudi metode za gosto rekonstrukcijo. Na začetku smo projicirali točke v prostor s pomočjo globinskih map. Ker smo večkrat zasledili omenjene metode gostega ujemanja smo se odločili implementirati tudi eno od njih [46].

Poglavje 5

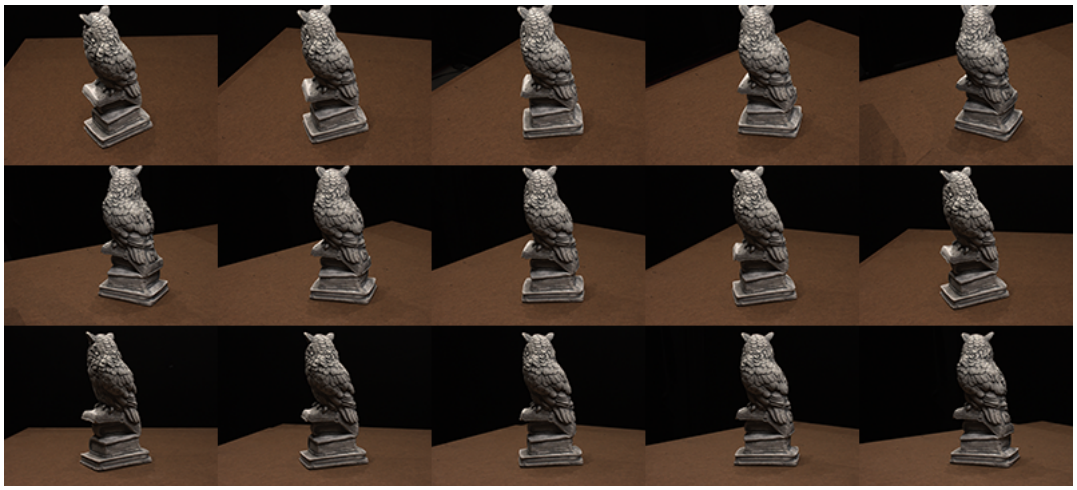
Eksperimentalna evalvacija

V tem poglavju je najprej predstavljena podatkovna zbirka na kateri smo evalvirali implementacijo strukture iz gibanja. Sledi evalvacija detektorjev katere namen je poiskati najbolj robusten detektor, ta je prikazana v Poglavju 5.2. Nato je v Poglavju 5.3 izvedena analiza metod ujemanja deskriptorjev. Sledita analiza različic metode RANSAC v Poglavju 5.4 in primerjava rekonstruiranih 3D modelov ter 3D modelov iz podatkovne zbirke. Slednja je opisana v Poglavju 5.5.

5.1 Podatkovna zbirka

Evalvacija naše implementacije cevovoda rekonstrukcije je bila narejena na podatkovni zbirki [19]. Zbirka je sestavljena iz 128 setov. Vsak set vsebuje od 49 fotografij resolucije 1600×1200. Pozicije kamer in notranji parametri kamer so bili ocenjeni s pomočjo kalibracijskega nabora orodji Matlab [1]. Scene so bile zajete pod šestimi različnimi osvetlitvenimi pogoji. Zbirka za vsak set vsebuje tudi 49 slik na katerih so modeli povsod enakomerno osvetljeni. Pri nekaterih scenah je na voljo tudi 15 fotografij enakomerno osvetljenega predmeta, slikanega z večje razdalje. To zaporedje vsebuje tri intervale po pet fotografij. Višina objektiva se med zajemom intervala ne spreminja. Kamera se v posameznem intervalu zasuka okoli opazovanega predmeta za okoli

90°. Primer ene od teh sekvenc, na kateri smo tudi evalvirali metode, je na Sliki 5.1.



Slika 5.1: Primer sekvence fotografij 124-ega seta. Zaporedje vsebuje tri intervale po pet slik. Slike posameznega intervala so zajete na isti višini. Celotno sekvenco uporabljamo pri končni rekonstrukciji.

5.2 Analiza detektorjev

Robustnost detektorjev smo testirali na zadnjih petnajstih slikah petih setov. Za vsak vhodni set smo barvne slike pred evalvacijo pretvorili v sivinske slike. Med zaporednimi sivinskimi slikami smo nato tudi opravljali detekcijo. Ujemanja med zaporednimi pari pogledov smo poiskali z algoritmom izčrpnega ujemanja. Obdržali smo tisa, ki pripadajo fundamentalni matriki z največ podpore. Slednjo smo ocenili s pomočjo metode RANSAC [12]. Parametru p, s katerim opišemo verjetnost, da smo našli pravilno fundamentalno matriko, smo nastavili vrednost $p = 0.99$. Ocenjevali smo število detekcij značilnic in število pravih ujemanj. Prava ujemanja smo definirali kot pare, ki pripadajo pravilni fundamentalni matriki. Pohitritev izvajanja posameznega algoritma smo primerjali s časom izvajanja detektorja SIFT.

SET SLIK	SIFT	SURF	ORB
	714/840	888/1315	40/319
0.91s	1	2.06	27.07
	566/628	594/816	84/176
0.86s	1	2.72	29.12
	975/1074	992/1307	71/319
0.87s	1	2.64	30.31
	307/346	320/511	92/180
0.88s	1	2.69	29.90
	846/1153	1064/1560	81/146
1.09s	1	2.45	34.12

Slika 5.2: Primerjava detektorjev. Vrstice celic vsebujejo povprečne vrednosti, t.j. kolikšen je bil povprečen čas izvajanja za posamezno sliko v setu in koliko značilnic je bilo v povprečju najdenih na vsaki sliki. Prva vrstica vsebuje število pravih ujemanj/število vseh ujemanj, druga vrstica pa pohitritev v primerjavi z algoritmom SIFT. Čas izvajanja algoritma SIFT je zapisan pod sliko.

Največ detekcij je pri vsakem setu zaznal algoritem SURF, vendar pa je veliko teh tvorilo lažna ujemanja. Po času izvajanja je prehitel metodo SIFT. Čeprav dosega izjemne pohitritve, pa je po številu zaznav bil najslabši algoritem ORB. Razlog za mnogo manjše število zaznanih regij pri slednjem je, da na vhodnih slikah ni veliko regij, ki bi ustrezale kotom.

5.3 Analiza ujemanja opisnikov

Naredili smo tudi primerjavo metod ujemanja opisnikov. Primerjali smo algoritme po hitrosti ter zanesljivosti. Hitrost in zanesljivost metod približnega iskanja smo primerjali z metodo izčrpnega ujemanja. Izbiro parametrov smo prepustili knjižnici [33], saj je namenjena samodejni optimizaciji

iskanja. Primerjavo ujemanja vektorskih opisnikov smo izvedli na opisnikih metode SIFT. Primerjavo ujemanja binarnih opisnikov pa s pomočjo metode ORB, saj edina izmed treh opisanih metod tvori binarne opisnike.

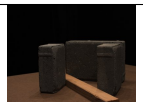
SET SLIK	IZČRPNO(BIN)	FLANN(BIN)	IZČRPNO(VEC)	FLANN(VEC)
	319	**381**	3689	**3896**
	40	39	**2748**	2633
0.15s/21.43s	**1.00**	0.25	1.00	**2.89**
	176	**229**	2721	**2860**
	84	83	**2174**	2097
0.17s/8.7s	**1.00**	0.28	1.00	**1.69**
	319	**379**	5360	**5479**
	71	69	**4376**	4254
0.18s/17.4s	**1.00**	0.30	1.00	**3.31**
	180	**231**	1426	**1495**
	92	90	**1025**	1001
0.18s/3.37s	**1.00**	0.30	1.00	**1.26**
	146	**193**	5934	**6577**
	81	**81**	**3425**	3385
0.16s/100.95s	**1.00**	0.27	1.00	**5.86**

Tabela 5.1: Primerjava algoritmov ujemanja opisnikov. Pod vsako sliko je napisan čas izvajanja izčrpnega ujemanja(binarni opisniki/vektorski opisniki). Vrstice celic vsebujejo število najdenih ujemanj, število pravih ujemanj in pohitritev v primerjavi z izčrpnim ujemanjem.

Približno iskanje najbližjega vektorja dosega večkratne pohitritve pri vektorskih opisnikih, ne pa pri binarnih. Najverjetneje je razlog za to manjše število opisnikov, ki jih tvori algoritem ORB. Pri ponovnem testiranju smo detekcijo značilnic namesto z algoritmom FAST, ki ga implementira ORB, opravili z detektorjem SURF. Zaradi večjega števila generiranih značilnic je metoda iskanja za ujemanje binarnih opisnikov dosegala pohitritve pri testih,

ki so presegali ≈ 1000 opisnikov. Razlog, da je izčrpno ujemanje učinkovitejše pri manjših primerih je verjetno način izvajanja. FLANN potrebuje dodatni čas, da zgradi KD drevo s katerim primerja opisnike. V času poteka izgradnje lahko izčrpno ujemanje primerja že več sto opisnikov. Metode knjižnice FLANN so zato primernejše, če imamo veliko pogledov ter s tem tudi veliko opisnikov.

5.4 Analiza različic algoritma RANSAC

Različice algoritma RANSAC smo evalvirali na značilnicah, zaznanih in opisanih z algoritmom SURF. Prvotna evalvacija metod, izvedena na značilnicah pridobljenih z metodo SIFT, ni bila primerna zaradi prevelikega ujemanja značilnic. Število iteracij in čas izvajanja sta zato bila slabo primerljiva. V prvem stolpcu so prikazane slike. Poleg vsake slike je napisano število ujemanj. V preostalih stolpcih so prikazane naslednje tri vrednosti: število iteracij, pohitritev(v primerjavi z osnovnim RANSAC-om) in število pripadajočih ujemanj.

Parametri posameznih algoritmov uporabljenih v evalvaciji so bili sledeči: Število notranjih iteracij pri algoritmu LO-RANSAC: ($k = 10$). Število d pri pred-verifikacijskem testu $T_{d,d}$: ($d = 1$). Parametra M in B pri metodi PE-RANSAC: ($M = 500$), ($B = 100$), število niti = 2.

Kot je razvidno iz Tabele 5.2, najmanj hipotez večinoma tvori PROSAC. Prav tako uspe v večini primerov najti največ točk, ki podpirajo model. PROSAC z pred-verifikacijskim testom $T_{d,d}$ pa dosega največje pohitritve izmed vseh metod z neomejenim časom izvajanja. Preemptive RANSAC je najhitrejši le v zadnjih dveh primerih, kjer se ostale metode zaradi večjega deleža odstopajočih točk upočasnijo.

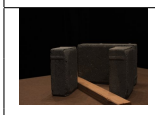
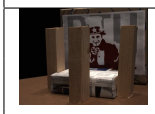
SET SLIK	RANSAC	PROSAC	LO-RANSAC	RANSAC $T_{d,d}$	PROSAC $T_{d,d}$	PE RANSAC
	214	143	185	252	**135**	993
	1.00	2.60	0.93	1.17	**5.20**	1.17
1632	1035	1081	1052	1020	**1088**	1048
	181	**106**	148	215	110	993
	1.00	2.80	1.03	1.03	**3.94**	0.86
764	497	**522**	505	491	520	509
	143	**75**	125	169	83	993
	1.00	3.38	0.92	1.13	**5.01**	0.79
1315	878	**932**	890	868	924	907
	552	**512**	514	702	615	993
	1.00	1.32	1.07	1.13	1.61	**2.64**
639	361	364	**365**	352	357	346
	468	**402**	421	576	443	993
	1.00	2.09	1.05	1.18	3.13	**4.24**
1555	897	**911**	908	880	904	873

Tabela 5.2: Primerjava različic algoritma RANSAC. Pod vsako sliko je napisano število vseh ujemanj pridobljenih z izčrpnim algoritmom. Vrstice celic vsebujejo število iteracij, pohitritev(v primerjavi z osnovnim RANSAC-OM) in število pravih ujemanj.

5.5 Analiza rekonstrukcije

Zbirka podatkov poleg slik posameznega modela vsebuje tudi pravilno rekonstrukcijo končnega modela. V primeru naše primerjave je slednja zlati standard. Pred primerjavo smo model poravnali z zlatim standardom. Na obeh smo definirali ključne točke s pomočjo katerih je MeshLab [10] izvedel poravnavo. Za pretvorbo oblaka točk v poligonsko mrežo smo uporabili Poissonovo rekonstrukcijo [17]. Med poligonskimi predstavitvami smo nato ocenili Hausdorffovo razdaljo. Hausdorffova razdalja podaja razliko dveh 3D modelov. Razliko med modeloma izračuna kot razdaljo med njunimi poligoni

in oglišči. V vsaki prvi vrstici celice v Tabeli 5.5 je napisan odstotek. Slednji nam pove za koliko povprečno odstopata modela. Kot absolutno razdaljo odstopanja MeshLab privzame dolžino diagonale škatle, ki obdaja zlati standard (angl., bounding-box). Spodnja vrstica vsake celice pa podaja število oglišč končnega modela.

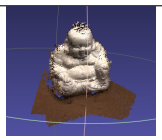
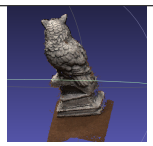
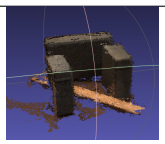
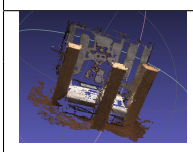
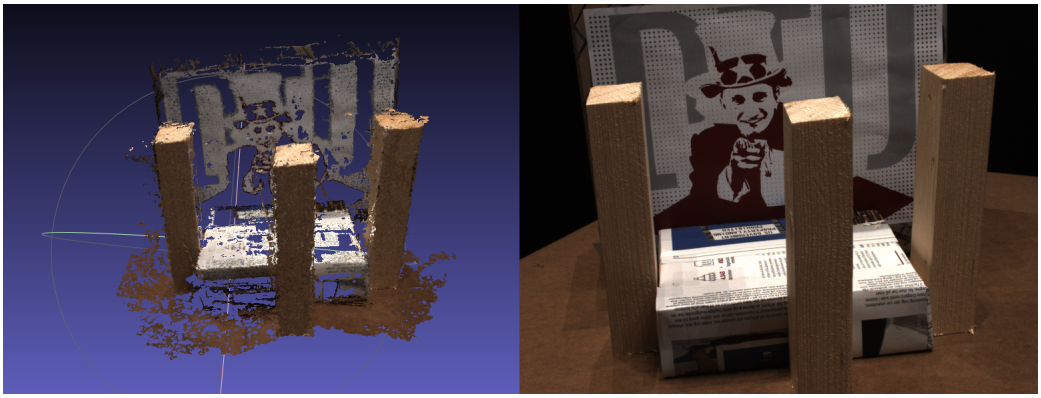
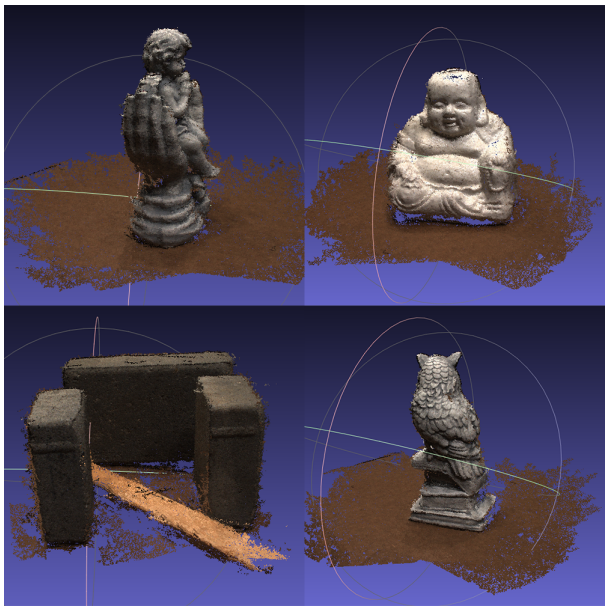
RECON.	Globinska slika	Gosto ujemanje
	3,411% 142279	2,543% 157718
	3,429% 84570	3,262% 91725
	1,857% 94328	1,642% 91967
	1,021% 136724	1,009% 146221
	17,510% 90872	15,224% 112399

Tabela 5.3: Primerjava rekonstrukcij z 3D modeli v zbirki podatkov. Prvi stolpec prikazuje najboljšo rekonstrukcijo seta, ki jo ustvari naš cevovod. Vsaka celica drugega in tretjega stolpca je sestavljena iz vrstic, ki vsebujeta povprečno Hausdorffovo razdaljo med ustvarjenim modelom in zlatim standardom ter število oglišč v rekonstrukciji. Drugi stolpec predstavlja rekonstrukcijo z globinsko sliko, tretji pa z metodo gostega ujemanja.



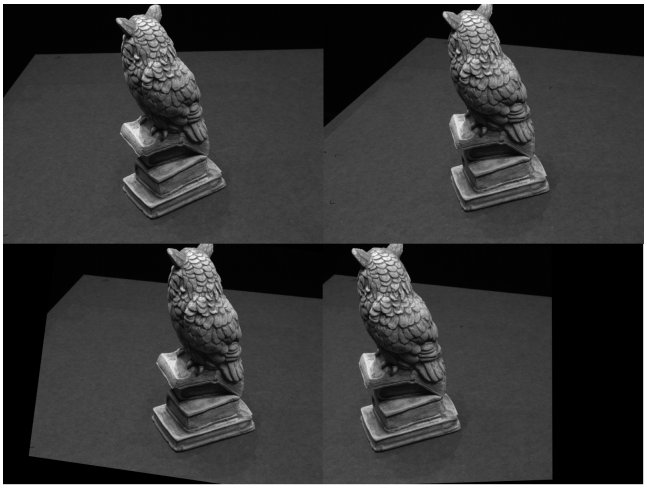
Slika 5.3: Primer slabe rekonstrukcije iz seta slik zadnje evalvacije. Homogenih območij nismo mogli triangulirati, zato končna rekonstrukcija vsebuje mnogo lukenj. Slika na desni strani prikazuje resničen izgled scene.



Slika 5.4: Dobre triangulacije oblakov točk prvih štirih evalviranih setov.

Končni tridimenzionalni model, ki smo ga primerjali z zlatim standardom, je bil zgrajen z najbolj zanesljivimi in efektivnimi metodami, ki so implementirane v našem cevovodu. Za detekcijo značilnic smo uporabili metodo SURF [5]. Ujemanje med značilnicami smo poiskali s pomočjo me-

tode približnega iskanje bližnjih sosedov [32]. Za ocenjevanje fundamentalne matrike smo uporabili algoritem PROSAC [8]. Inkrementalna struktura iz gibanja [21] je bila uporabljena za ocenjevanje pozicij in rotacij kamer ter Gosto stereo ujemanje [46] za triangulacijo oblaka točk. Potek rekonstrukcije smo testirali tudi z drugimi metodami posameznega modula. Uporaba detektorja ORB namesto SURF ni omogočala poteka celotnega cevovoda. Zaradi premajhnega števila korespondenc nismo mogli izvesti prilagoditve snopa. Detektor SIFT je našel največ pravilnih ujemanj a ga nismo uporabili v končnem cevovodu zaradi časa izvajanja. Testirali smo tudi različice algoritma RANSAC [12]. Zaradi visokega deleža ujemanj in naknadne uporabe algoritma LMedS (glej Poglavje 4) je ocenjena fundamentalna matrika vedno bila ustrezna, zato smo se odločili za algoritem PROSAC, ki ima najkrajši čas izvajanja.



Slika 5.5: Površina na kateri lahko opravimo triangulacijo z metodo gostega ujemanja je prikazana na zgornjem paru slik. Spodnji par prikazuje sliki, ki imata poravnane vrstice. Ta popravek omogoča rekonstrukcijo z globinsko sliko. Ker smo morali sliki preoblikovati pa smo zgubili dele, ki so na voljo metodi gostega ujemanja.

Kot je razvidno iz Tabele 5.5 boljše rezultate daje metoda gostega ujemanja. Nekatere izmed rekonstrukcij so prikazane na Sliki 5.4. Uporaba

globinske slike bi bila uporabna le v primeru, da imamo implementiran algoritem, ki zna izračunati ali pa pravilno interpolirati vrednosti v globinski sliki, kjer se nahajajo homogene regije. Zadnji primer v Tabeli 5.5 vsebuje mnogo takih območij. Posledično je končna rekonstrukcija slabša (Slika 5.3) in Hausdorffova razdalja kaže večje odstopanje. Razlog za manjše število oglišč pri večini rekonstrukcij z globinsko sliko je verjetno izguba površine primerne za ujemanje. Preden lahko med pogledi izračunamo globinsko sliko moramo popraviti poglede (glej Poglavje 2.5.1). S tem nastanejo regije v katerih ne moremo izračunati globinske slike. Primer slednjega je prikazan na Sliki 5.5.

Poglavje 6

Sklepne ugotovitve

V diplomskem delu smo obdelali teoretično podlago standardnega cevovoda rekonstrukcije in probleme, ki nastanejo pri vsaki od stopenj. S pomočjo osvojenega teoretičnega znanja smo se nato lotili implementacije naše lastne knjižnice za rekonstrukcijo. Za testiranje učinkovitosti in zanesljivosti cevovoda smo uporabili prosto dostopno zbirko podatkov [19], na kateri smo poganjali algoritme za zaznavo značilnic, ujemanje značilnic, robustno estimacijo fundamentalne matrike in gosto rekonstrukcijo oblaka točk. Natančnost končnih rekonstrukcij smo ovrednotili z uporabo nabora orodij MeshLab [10].

Najbolj zanesljiv in robusten cevovod rekonstrukcije so sestavljale sledeče metode. Za zaznavo in ujemanje značilnic se je najbolje izkazal detektor SIFT [25]. Drugi najboljši je bil detektor SURF [5]. Za zaznavo značilnic smo se raje poslužili slednjega, saj so razlike v točnosti detekcij zanemarljive, pohitritve pa znatne. Za iskanje korespondenc smo uporabili Približno iskanje ujemanj, saj predstavlja dober kompromis med časom izvajanja in pravilnostjo najdenih ujemanj. Fundamentalno matriko smo nato ocenili s pomočjo metode PROSAC [8]. Razlog za uporabo te različice je naveden v Poglavju 5.4. Nad dobljenimi pozicijami in orientacijami kamer smo nato izvedli Iterativno prilagajanje snopa. Za triangulacijo točk v prostoru pa smo uporabili gosto stereo ujemanje, ki omogoča natančnejšo rekonstrukcijo.

6.1 Nadaljnje delo

Implementacija ponuja mnogo možnosti izboljšav. Največjo težavo trenutno predstavlja prilagajanje snopa, saj v celotno rekonstrukcijo ne uspe uvrstiti več kot 13 izmed 15-ih pogledov vsakega seta na katerem smo testirali. Inkrementalna metoda SfM je tudi zelo časovno potratna. Ker nas predvsem zanimata robustnost in skalabilnost knjižnice želimo v prihodnosti implementirati še globalne metode SfM [45]. Trenutna verzija prav tako ne omogoča izvedbe rekonstrukcije brez predhodnih podatkov o kameri. To bi lahko razrešili z uporabo metapodatkov, ki jih na primer vsebujejo slike formata jpg.

Kot smo omenili v Poglavju 1.1 je glavni cilj kode tudi učinkovita in preprosta rešitev za zajem manjših objektov. Z nadgradnjo knjižnice in paralelizacijo nekaterih ključnih procesov, bi kodo lahko poganjali na mobilnih platformah. Trenutna implementacija je prenosljiva, saj se opira le na knjižnice OpenCV [18] in Ceres [3], ki ju je mogoče uporabljati tudi na napravah Android. Izvajanje na mobilnih napravah bi nam zagotovo prineslo mnogo izzivov ter nekaj prednosti. Med slednje spadata merilni napravi IMU in GPS, ki bi pripomogli k robustnosti delovanja.

Literatura

[1] Camera calibration toolbox for matlab. Dosegljivo: `http://www.vision.caltech.edu/bouguetj/calib_doc/`. [Dostopano 1. 9. 2017].

[2] Pinhole camera model. Dosegljivo: `https://commons.wikimedia.org/wiki/File:Pinhole-camera.svg`. [Dostopano 1. 9. 2017].

[3] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. Dosegljivo: `http://ceres-solver.org`. [Dostopano 1. 9. 2017].

[4] Hernán Badino and Takeo Kanade. A head-wearable short-baseline stereo system for the simultaneous estimation of structure and motion. In *MVA*, pages 185–189, 2011.

[5] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. *Computer vision–ECCV 2006*, pages 404–417, 2006.

[6] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[7] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. *Computer Vision–ECCV 2010*, pages 778–792, 2010.

[8] Ondrej Chum and Jiri Matas. Matching with prosac-progressive sample consensus. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 220–226. IEEE, 2005.

[9] Ondřej Chum, Jiří Matas, and Josef Kittler. Locally optimized ransac. *Pattern recognition*, pages 236–243, 2003.

[10] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. MeshLab: an Open-Source Mesh Processing Tool. In Vittorio Scarano, Rosario De Chiara, and Ugo Erra, editors, *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008.

[11] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision*, pages 834–849. Springer, 2014.

[12] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[13] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE transactions on pattern analysis and machine intelligence*, 25(8):930–943, 2003.

[14] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[15] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Manchester, UK, 1988.

[16] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[17] Hugues Hoppe. Poisson surface reconstruction and its applications. In *Proceedings of the 2008 ACM symposium on Solid and physical modeling*, pages 10–10. ACM, 2008.

[18] Itseez. Open source computer vision library. Dosegljivo: `https://github.com/itseez/opencv`, 2017. [Dostopano 1. 9. 2017].

[19] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 406–413. IEEE, 2014.

[20] Nianjuan Jiang, Zhaopeng Cui, and Ping Tan. A global linear method for camera pose registration. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 481–488, 2013.

[21] Kenichi Kanatani, Yasuyuki Sugaya, and Yasushi Kanazawa. *Guide to 3D Vision Computation: geometric analysis and implementation*. Springer, 2016.

[22] Vladimir Kolmogorov and Ramin Zabih. Computing visual correspondence with occlusions using graph cuts. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 508–515. IEEE, 2001.

[23] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168, 1944.

[24] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM siggraph computer graphics*, volume 21, pages 163–169. ACM, 1987.

[25] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[26] Francesco Mancini, Marco Dubbini, Mario Gattelli, Francesco Stecchi, Stefano Fabbri, and Giovanni Gabbianelli. Using unmanned aerial vehicles (uav) for high-resolution reconstruction of topography: The struc-

ture from motion approach on coastal environments. *Remote Sensing*, 5(12):6880–6898, 2013.

[27] Mapillary. Opensfm, open source structure from motion pipeline. Dosegljivo: `https://github.com/mapillary/OpenSfM`. [Dostopano 1. 9. 2017].

[28] Donald W Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

[29] Desire L Massart, Leonard Kaufman, Peter J Rousseeuw, and Annick Leroy. Least median of squares: a robust method for outlier and model error detection in regression and calibration. *Analytica Chimica Acta*, 187:171–179, 1986.

[30] Jiri Matas and Ondrej Chum. Randomized ransac with t d, d test. *Image and vision computing*, 22(10):837–842, 2004.

[31] Marius Muja and David G. Lowe. Fast matching of binary features. In *Computer and Robot Vision (CRV)*, pages 404–410, 2012.

[32] Marius Muja and David G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36, 2014.

[33] Marius Muja and David G. Lowe. Fast library for approximate nearest neighbors. Dosegljivo: `https://github.com/mariusmuja/flann`, 2017. [Dostopano 1. 9. 2017].

[34] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[35] David Nistér. Preemptive ransac for live structure and motion estimation. *Machine Vision and Applications*, 16(5):321–329, 2005.

[36] Rahul Raguram, Jan-Michael Frahm, and Marc Pollefeys. A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. *Computer Vision–ECCV 2008*, pages 500–513, 2008.

[37] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE international conference on*, pages 2564–2571. IEEE, 2011.

[38] Thomas Schops, Jakob Engel, and Daniel Cremers. emi-dense visual odometry for ar on a smartphone. In *IEEE International Symposium on Mixed and Augmented reality*. 2014.

[39] Roy Shilkrot. Sfm-toy-library, a toy library for structure from motion using opencv. Dosegljivo: `https://github.com/royshil/SfM-Toy-Library`. [Dostopano 1. 9. 2017].

[40] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3d. In *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, pages 835–846, New York, NY, USA, 2006. ACM.

[41] Chris Sweeney. Theia multiview geometry library: Performance. Dosegljivo: `http://theia-sfm.org/performance.html`. [Dostopano 1. 9. 2017].

[42] Christopher Sweeney, Tobias Hollerer, and Matthew Turk. Theia: A fast and scalable structure-from-motion library. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 693–696. ACM, 2015.

[43] Bart van Andel. Image rectification. Dosegljivo: `https://commons.wikimedia.org/w/index.php?curid=20655925`. [Dostopano 1. 9. 2017].

[44] Deepak Geetha Viswanathan. Features from accelerated segment test (fast), 2009.

[45] Kyle Wilson and Noah Snavely. Robust global translations with 1dsfm. In *European Conference on Computer Vision*, pages 61–75. Springer, 2014.

[46] Jianxiong Xiao, Jingni Chen, Dit-Yan Yeung, and Long Quan. Learning two-view stereo matching. *Computer Vision–ECCV 2008*, pages 15–27, 2008.