SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1912

POSTUPCI PODUDARANJA TRODIMENZIJSKIH UZORKOVANIH PODATAKA

Davor Bokun

Zagreb, rujan 2012.

Umjesto ove stranice umetnite izvornik Vašeg rada.
Da bi ste uklonili ovu stranicu obrišite naredbu `\izvornik`.

SADRŽAJ

POPIS SLIKA

POPIS TABLICA

1. Uvod

Digitalizacija informacije o prostornom položaju i obliku objekata iz stvarnog svijeta je postupak kojim se, pomoću odgovarajućih mjernih uređaja, ta analogna informacija pretvara u digitalni oblik pogodan za pohranu i obradu. Motivi za takav postupak su nebrojeni i nalaze primjenu u najrazličitijim granama ljudske djelatnosti. Od zdravstva (protetika, MRI/CT/PET analiza, ...), industrije (obrnuto inženjerstvo, automatizacija tehnoloških procesa, verifikacija i kontrola kvalitete, ...), preko edukacije i znanosti (arhiviranje, analiza, rekonstrukcija i klasifikacija predmeta u arheologiji, biologiji, geologiji, umjetnosti, ...), do putovanja u svemir (automatizirana navigacija, izbjegavanje prepreka, ...). Područja gdje se ovaj postupak već primjenjuje su mnoga, a nove primjene se nalaze praktički svakodnevno. S time u vidu, jasno je zašto bi se trebala posvetiti posebna pažnja poboljšanju njegovih mogućnosti i kvalitete.

1.1. Pregled postupaka uzorkovanja i primjene

Uređaji pomoću kojih se obavlja digitalizacija prostornih objekata nazivaju se 3D skeneri (engl. 3D scanner) ili skraćeno skeneri. Zadaća 3D skenera je točno izmjeriti položaj mjerene točke u stvarnom svijetu u odnosu na referentnu točku uređaja, i taj položaj proslijediti u digitalnom obliku na daljnju obradu. Poželjno je također i da uređaj može obavljati seriju takvih mjerenja za redom, te da su podaci dobiveni takvom serijom mjerenja međusobno korelirani. To jest da se nalaze u istom referentnom koordinatnom sustavu (što će biti detaljnije pojašnjeno u nastavku). Brzina uzorkovanja koju takav uređaj postiže je također važan čimbenik za primjenu, ali se za ovu raspravu može zanemariti.

Tehnološke izvedbe ovih uređaja su razne, ali općenito ih dijelimo na kontaktne i beskontaktne. Kontaktni uglavnom nalaze primjenu u industriji za kontrolu kvalitete, ili određivanje nulte točke, položaja i oblika sirovine pri automatskoj obradi materijala. Princip rada zasniva se na ostvarivanju fizičkog kontakta mjerne sonde (s poznatim položajem u referentnom koordinatnom sustavu) i predmeta (sirovine) čiji se

položaj određuje. Na taj način se dobiva prostorni položaj točke koja se nalazi na površini mjerenog objekta. Još jedan primjer kontaktnog uzorkovanja je i digitalizacija puteva koristeći mjerno vozilo i osobu s GPS uređajem. U tom slučaju je to vozilo (ili osoba) u ulozi sonde s poznatim položajem. Kontaktni pristup svakako ima svoje prednosti, prvenstveno jer je jednostavan i pouzdan. Nedostatak mu je ograničena primjena zbog nužnog kontakta s mjerenim objektom, ali i male brzine uzorkovanja ograničene mehaničkim svojstvima uređaja i sonde. Zbog toga su rezultati uređaja s kontaktnim pristupom rijetko kada veliki skupovi podataka.

Među beskontaktnim skenerima razlikujemo aktivne, koji emitiraju neku vrstu signala u prostor i zatim analiziraju povratni signal, i pasivne, koji se oslanjaju na signal što ga objekti već sami po sebi emitiraju ili reflektiraju od nekog drugog postojećeg izvora.

Tip signala koju aktivni skeneri emitiraju u prostor može biti bilo koji. Koriste se na primjer zvuk, svijetlost, radio valovi, ili x-zrake. Oni zatim analiziraju povratni signal kako bi odredili položaj promatrane točke objekta, a postoje razni pristupi kojima se to postiže.

Jedan takav pristup naziva se „vrijeme leta“ (engl. time-of-flight), a zasniva se na mjerenju vremena potrebnog da zraka svijetlosti emitirana laserom prijeđe put od uređaja do promatranog predmeta i nazad. Preciznost takvih uređaja je reda veličine pikosekunde, pa je preciznost izmjerene udaljenosti reda veličine milimetra. Serija mjerenja takvim uređajem dobiva se najčešće pomoću sustava rotirajućih ogledala kojima se laserska zraka usmjerava i time dobiva sferna slika udaljenosti predmeta od uređaja. Uobičajena brzina uzorkovanja je od 10 do 100 tisuća točaka u sekundi. U novije vrijeme postoje i uređaji s dvodimenzijskom konfiguracijom takvih senzora, koji mogu trenutno dati sliku udaljenosti u predefiniranoj razlučivosti (slično kao kamera).

Još jedan čest pristup je triangulacija. Jedna točka ili jedna linija projiciraju se na objekt, a zatim se ta projekcija snima s kamerom pod predefiniranim kutem u odnosu na projekciju. Triangulacijom se dobiva položaj točke (ili cijele linije) u odnosu na referentni sustav uređaja. Prednost ovog pristupa je u niskoj cijeni i jednostavnosti izvedbe.

Među novijim pristupima je i korištenje strukturiranog svijetla za prostorno uzorkovanje objekata. Osnova ove metode je također triangulacija i može se reći da je ona proširenje prethodnog pristupa, ali s razlikom da se ne projiciraju točke ili linije, već složeni uzorci pomoću kojih je moguće uzorkovati cijelo vidno polje kamere odjednom. Tako se postižu i vrlo velike brzine uzorkovanja (do 120 okvira u sekundi) uz široko dostupnu i relativno jeftinu računalnu opremu (digitalna kamera i LCD projek-

tor). Neki od poznatih kontrolnih uređaja iz sfere računalnih igara koriste ovaj pristup. Kao na primjer uređaj Kinekt.

Vrlo važan i često korišten pristup je tomografija. Koristi se u medicini i industriji (MRI, CT, PET, ...). Temelji se na tomografskoj rekonstrukciji prostornog odziva objekta na signal koristeći niz mjerenja, bilo prigušenja signala na putu kroz objekt (na primjer CT), releksije signala na putu kroz objekt (ultrazvuk), ili rezonantnog odziva promatranog objekta na pobudni signal (MRI). Na taj se način dobije volumna slika promatranog objekta, a ne samo površina, pa se ovaj postupak naziva i volumetrijskim.

Osim navedenih aktivnih, postoje još i pasivni postupci uzorkovanja prostornih podataka, koji se oslanjaju na signal što ga objekti već sami po sebi emitiraju u prostor ili reflektiraju od nekog postojećeg izvora. U većini primjena, takav se skener sastoji samo od jedne ili više digitalnih kamera. Niz se dobivenih digitalnih fotografija zatim analizira kako bi se pronašle podudarne točke i triangulirao njihov međusobni položaj, kao i položaj u odnosu na uređaj. Među pristupe tog tipa pripadaju stereometrija (ili stereoskopski vid), stereo fotogrametrija, i druge... Stereo fotogrametrija je posebno popularna u poslijednje vrijeme jer omogućava vizualno efektne rekonstrukcije objekata iz postojećih fotografija.

Rezultat uzorkovanja je skup prostornih uzoraka $\boldsymbol{x}_i$ smješten u trodimenzijskom prostoru, to jest $\boldsymbol{x}_i \in \mathbb{R}^3$, uz $i = 1, \ldots, N$. Gdje je N broj uzoraka (uzorkovanih točaka) u skupu. Osim u slučaju tomografije gdje je rezultat volumna slika, što je zapravo prostorna skalarna funkcija $\rho : \mathbb{R}^3 \mapsto \mathbb{R}$. Iz volumnih slika je raznim metodama moguće dobiti rekonstrukciju površine objekta, ali one same neće biti razmatrane. Uz prostorne informacije dobivene uzorkovanjem, često se kao rezultat uzorkovanja dobije i na primjer boja predmeta u promatranoj točki.

Ovdje svakako nisu pobrojani svi danas poznati postupci i pristupi, ali je dan pregled najraširenijih kao uvid u moguće primjene trodimenzijskih uzrokovanih podataka, pa se daje naslutiti u kakvom obliku, koje kvalitete, i u kojim količinama bismo željeli te podatke imati kako bi oni bili svrsishodni. Svaki od navedenih pristupa ima svoje mane, unosi pogrešku u mjerenja, ili ima ograničenja zbog kojih podaci dobiveni uzorkovanjem ponekad nisu zadovoljavajući. To dovodi do potrebe za kombiniranjem različitih postupaka ili višestrukim ponavljanjem uzorkovanja kako bi se smanjila ukupna pogreška. Rezultate takvih ponovljenih uzorkovanja istog promatranog objekta potrebno je na neki način dovesti u isti kontekst. Odnosno, uspostaviti korelaciju između dva ili više skupova uzoraka kako bi rezultat njihova spajanja davao točniju ili potpuniju informaciju o originalnom objektu. Ta važna karika u lancu, koji sačinjava postupak digitalizacije prostornih objekata, je upravo predmet ovog rada.

2. Problem registracije oblaka točaka

Uspostaviti korelaciju između dva skupa uzoraka, to jest dovesti ih u isti prostorni kontekst, podrazumijeva pronaći transformaciju (funkciju ili pravilo) koja referentni sustav jednog skupa preslikava u referentni sustav drugoga, na način da se objekt, kojega svaki od tih skupova predstavljaju u svom vlastitom referentnom sustavu, nakon transformiranja nalazi na istom mjestu i u istom položaju. Indukcijom se ova definicija proširuje i na više od dva skupa, ali će se u nastavku, bez smanjenja općenitosti, govoriti o koreliranju dvaju skupova uzoraka.

Ovisno o postupku i tehnologiji uzorkovanja, navedena transformacija ponekad može biti i poznata. U tom slučaju su skupovi već korelirani te se mogu uzeti kao dva mjerenja iste vrijednosti (u svrhu povećanja preciznosti), ili se njihova unija može smatrati jednim skupom uzoraka kako bi se dobila potpunija slika promatranog objekta nego što je to svaki skup zasebno.

Ukoliko je pak ta korelacija izgubljena u postupku uzorkovanja (na primjer ručnim pomicanjem predmeta kako bi se on snimio s druge strane), ili ukoliko je objekt snimljen različitim (nekoreliranim) metodama (na primjer zgrada snimljena sa zemlje i iz zraka, i slično...), ili ako je snimanje namjerno učinjeno s vremenskim razmakom (kako bi se proučavale promjene tijekom vremena na primjer), tada se ta korelacija mora pronaći naknadno.

Ovisno o načinu na koji je korelacija izgubljena, može biti izgubljena manja ili veća količina informacije. Na primjer, ukoliko je objekt unutar uređaja zarotiran na postolju koje se može zakretati samo oko jedne fiksne osi, tada je transformacija, koja prvi skup uzoraka dovodi u kontekst drugoga, rotacija oko te fiksne osi u tri dimenzije parametrizirana traženim kutem. U tom slučaju govorimo o transformaciji s jednim stupnjem slobode. Kako bi se odredio taj nepoznati kut, dovoljno je naći jedan par podudarnih točaka. U ovom primjeru je transformacija koja korelira skupove jednostavna, ali u nekom drugom slučaju ona može biti i složenija. Na primjer, ako je uzorkovani objekt skulptura uzorkovana triangulacijom iz dva različita kuta i s različite udaljenosti. U tom slučaju je potpuno nepoznat odnos pozicije i orijentacije referent-

nih sustava. Dakle, nepoznata je pozicija odnosno translacija u tri dimenzije, te rotacija oko barem dvije osi. U tom slučaju tražimo transformaciju s pet stupnjeva slobode. Za rješenje ovog problema, potrebno je pronaći barem tri para podudarnih točaka.

Navedena dva primjera opisuju linearne transformacije (uz translaciju). U općenitom slučaju, tražena transformacija može biti bilo koja trodimenzijska vektorska funkcija $\boldsymbol{f} : \mathbb{R}^3 \mapsto \mathbb{R}^3$, takva da preslikava skup uzoraka iz jednog referentnog sustava u ciljani referentni sustav, a u smislu prethodno navedenog koreliranja skupova. Uz tako općenitu definiciju tražene funkcije, bilo bi nemoguće naći jednoznačno rješenje koristeći se konačnim skupom uzoraka. Zato je potrebno odrediti više detalja, to jest postaviti pretpostavke koje diktiraju oblik takve transformacije. Neke takve pretpostavke su napravljene i u prethodna dva primjera analizom samog postupka uzorkovanja. A neke su i napravljene implicitno, podrazumijevajući na primjer, da prostor uzoraka nije iskrivljen. To jest, da odgovara intuitivnom poimanju prostora u kojem se nalaze promatrani stvarni predmeti. Detaljno o prepostavkama i ograničenjima ulaznih podataka biti će rečeno kasnije.

2.1. Notacija

Koordinate točaka u tri dimenzije mogu biti zapisane nehomogenim koordinatama $\boldsymbol{x} = (x, y, z) \in \mathbb{R}^3$, pri čemu notacija $(x_1, x_2, \dots)$ označava stupčani vektor. To jest, vektor $\boldsymbol{x}$ pišemo,

$$\boldsymbol{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \tag{2.1}$$

Točke u tri dimenzije se također mogu zapisati homogenim koordinatama, $\tilde{\boldsymbol{x}} = (\tilde{x}, \tilde{y}, \tilde{z}, \tilde{w}) \in \mathbb{P}^3$, gdje se vektori koji se razlikuju samo u u konstantnom množitelju smatraju ekvivalentnima. $\mathbb{P}^3$ se naziva trodimenzijski projektivni prostor.

Vektor prikazan homogenim koordinatama se može prevesti u nehomogene koordinate tako da se podijeli sa svojim zadnjim članom $\tilde{w}$. To jest,

$$\tilde{\boldsymbol{x}} = (\tilde{x}, \tilde{y}, \tilde{z}, \tilde{w}) = \tilde{w}(x, y, z, 1) = \tilde{w}\bar{\boldsymbol{x}}, \tag{2.2}$$

Gdje se $\bar{\boldsymbol{x}} = (x, y, z, 1)$ naziva prošireni vektor. Homogene točke kojima je poslijednji član $\tilde{w} = 0$ zovu se idealne točke ili točke u beskonačnosti, i nemaju svoju ekvivalentnu nehomogenu reprezentaciju.

2.2. Transformacija podudaranja

Kako bismo mogli odrediti transformaciju podudaranja koristeći ograničen skup uzoraka koji nam je na raspolaganju, funkciju preslikavanja moramo parametrizirati konačnim brojem parametara. Uz vektor parametara transformacije $\boldsymbol{p}$ možemo pisati,

$$\boldsymbol{x'} = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{p}). \tag{2.3}$$

Parametrizacije funkcije transformacije mogu biti razne, ali ovdje će se razmatrati samo linearne transformacije u projektivnom prostoru $\mathbb{P}^3$. Pa će to biti i prvo ograničenje koje će se podrazumijevati u daljem tekstu. To jest, da se funkcija $\boldsymbol{f}$ može napisati kao

$$\boldsymbol{f}(\tilde{\boldsymbol{x}}; \boldsymbol{p}) = \tilde{\boldsymbol{x}}' = \boldsymbol{T} \cdot \tilde{\boldsymbol{x}}, \tag{2.4}$$

gdje je $\tilde{\boldsymbol{x}}$ trodimenzijski uzorak kojega preslikavamo, $\boldsymbol{T}$ matrica linearne transformacije, a $\tilde{\boldsymbol{x}}'$ slika točke $\tilde{\boldsymbol{x}}$. Funkcija opisana jednadžbom 2.4 može se zapisati u matričnom obliku kao

$$\begin{bmatrix} \tilde{x}' \\ \tilde{y}' \\ \tilde{z}' \\ \tilde{w}' \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} \\ T_{21} & T_{22} & T_{23} & T_{24} \\ T_{31} & T_{32} & T_{33} & T_{34} \\ T_{31} & T_{32} & T_{33} & T_{44} \end{bmatrix} \cdot \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ \tilde{w} \end{bmatrix}. \tag{2.5}$$

Gotovo sve probleme u praksi je moguće, ili svrstati u ovu kategoriju, ili skupove uzoraka prethodno transformirati tako da problem podudaranja postane rješiv linearnom transformacijom. Ovime smo (uz razumne pretpostavke) problem koreliranja dvaju skupova uzorkovanih točaka sveli na pronalaženje parametara matrice transformacije $\boldsymbol{T}$. Kako ona sadrži šesnaest nezavisnih parametara, tako je i broj stupnjeva slobode ove transformacije šesnaest. Lako se može pokazati da je za rješenje jednadžbe 2.5 dovoljno pronaći četiri podudarne točke. To jest, četiri točke polaznog skupa $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_4 \in X$, i četiri podudarne točke iz ciljanog skupa uzoraka $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_4 \in Y$. U idealnom slučaju možemo pretpostaviti da podudarne točke ciljanog skupa $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_4$ točno odgovaraju slikama točaka polaznog skupa $\boldsymbol{x'}_1, \ldots, \boldsymbol{x'}_4$. Uvrštavanjem u 2.5 dobivamo sustav jednadžbi kojega možemo napisati u matričnom obliku i riješiti. Te tako dobivamo izraz za matricu transformacije

$$\boldsymbol{T} = \left[\tilde{\boldsymbol{y}}_1 \mid \tilde{\boldsymbol{y}}_2 \mid \tilde{\boldsymbol{y}}_3 \mid \tilde{\boldsymbol{y}}_4\right] \cdot \left[\tilde{\boldsymbol{x}}_1 \mid \tilde{\boldsymbol{x}}_2 \mid \tilde{\boldsymbol{x}}_3 \mid \tilde{\boldsymbol{x}}_4\right]^{-1}. \tag{2.6}$$

Kako bi rješenje ove jednadžbe postojalo, podudarne točke moraju biti prikladno odabrane. Tako da niti jedna trojka točaka ne bude kolinearna, te da četvorka ne bude koplanarna.

Četiri para podudarnih točaka na prvi pogled izgleda kao lak zadatak, no ne treba smetnuti s uma pogreške postupka uzorkovanja, kao niti pogreške prilikom pronalaženja podudarnih parova. Drugim riječima, u praksi se podudarne točke ciljanog skupa $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_4$ nikada sasvim ne podudaraju sa slikama točaka polaznog skupa $\boldsymbol{x'}_1, \ldots, \boldsymbol{x'}_4$. O takvim će pogreškama više govora biti kasnije, ali dovoljno je reći da je svako eventualno dodatno ograničenje postavljeno na oblik transformacije koju tražimo, dobrodošlo u traženju optimalne transformacije. Takva ograničenja logično proizlaze iz postupka uzorkovanja kao što je već bilo navedeno u primjerima.

Translacija. Translacija u dvije dimenzije može se zapisati u matričnom obliku formulom,

$$\boldsymbol{x'} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \bar{\boldsymbol{x}} \tag{2.7}$$

Ovdje se koristi skraćeni oblik koristeći matricu veličine 2×3, ali se lako može pokazati da ona još uvijek spada u linearne transformacije oblika prikazanog formulom 2.5. Samo što smo se ovdje ograničili na samo dva stupnja slobode. Točnije translaciju u dvije osi. Vektor $\boldsymbol{p}$ je u ovom slučaju (t_x, t_y). Translacija u dvije dimenzije ima logično i dva stupnja slobode (što je vidljivo iz vektora parametara).

U tri dimenzije ona izgleda ovako,

$$\boldsymbol{x'} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \end{bmatrix} \bar{\boldsymbol{x}}. \tag{2.8}$$

Parametri ove transformacije su $\boldsymbol{p} = (t_x, t_y, t_z)$.

Karakteristika translacije je da čuva oblik (kuteve), udaljenosti, i orijentaciju objekata koji se transformiraju. Praktični primjeri gdje je samo translacija nepoznanica nisu česti. Stoga se nećemo baviti rješenjima koja su specifična samo za ovaj tip problema, već će se na njega moći primijeniti neki od općenitijih pristupa.

Euklidska transformacija. Translacija uz rotaciju, poznata i kao kretanje krutog tijela ili Euklidska transformacija. Nazvana tako jer se Euklidske udaljenosti između točaka prostora očuvaju primjenom takve transformacije. Jedna je od dvije najzanim-

ljivije transformacije za ovaj rad jer su rotacija i translacija krutog tijela upravo ono što želimo rekonstruirati.

U dvije dimenzije ona izgleda ovako,

$$\boldsymbol{x'} = \begin{bmatrix} c_\theta & -s_\theta & t_x \\ s_\theta & c_\theta & t_y \end{bmatrix} \bar{\boldsymbol{x}}, \tag{2.9}$$

uz parametre (t_x, t_y, θ) i tri stupnja slobode. Značenje konstanti c_θ i s_θ je,

$$c_\theta = cos(\theta), s_\theta = sin(\theta). \tag{2.10}$$

A u tri dimenzije ovako,

$$\boldsymbol{x'} = \begin{bmatrix} c_\theta c_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi & t_x \\ c_\theta s_\psi & c_\phi c_\psi + s_\phi s_\theta s_\psi & -s_\phi c_\psi + c_\phi s_\theta s_\psi & t_y \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta & t_z \end{bmatrix} \bar{\boldsymbol{x}}, \tag{2.11}$$

uz šest parametara, $t_x, t_y, t_z, \phi, \theta$, i ψ. Značenje sinusa i kosinusa je analogno gore navedenom za dvodimenzijski slučaj.

Transformacija sličnosti. Prethodno opisana Euklidska transformacija odgovarala bi u većini slučajeva, kada bi skup uzoraka bio zapisan u poznatom mjerilu u odnosu na uzorkovani predmet. To jest kada bi udaljenost u jednom skupu bila jednaka udaljenosti u drugom skupu. Ali taj omjer udaljenosti uglavnom nije poznat, pa je zato najčešća transformacija koju je potrebno naći kako bi se riješio problem podudaranja skupova, transformacija sličnosti. Karakteristično za nju je da čuva kuteve među linijama, no nije pretpostavljeno da mora čuvati i udaljenosti, kao što je to slučaj kod Euklidske.

U dvodimenzijskom slučaju je,

$$\boldsymbol{x'} = \begin{bmatrix} 1+\mu & -\alpha & t_x \\ \alpha & 1+\mu & t_y \end{bmatrix} \bar{\boldsymbol{x}}, \tag{2.12}$$

uz $\boldsymbol{p} = (t_x, t_y, \alpha, \mu)$, to jest četiri stupnja slobode. A u trodimenzijskom,

$$\boldsymbol{x'} = \begin{bmatrix} 1+\mu & \alpha_3 & -\alpha_2 & t_x \\ -\alpha_3 & 1+\mu & \alpha_1 & t_y \\ \alpha_2 & -\alpha_1 & 1+\mu & t_z \end{bmatrix} \bar{\boldsymbol{x}} \tag{2.13}$$

uz $\boldsymbol{p} = (t_x, t_y, t_z, \alpha_1, \alpha_2, \alpha_3, \mu)$, ili sedam stupnjeva slobode.

Ovo je najčešći slučaj pa će se većina postupaka dalje odnositi na transformaciju sličnosti.

Afina transformacija. Afina i perspektivna transformacija nisu zanimljive za ovaj slučaj jer ćemo pretpostaviti da je prostor uzoraka homogen, te da se nije dogodilo iskrivljenje u postupku uzorkovanja kojim bi se narušili kutovi ili izmijenio oblik uzorkovanog predmeta. Za većinu navedenih primjena je to i razumna pretpostavka. No zbog potpunosti će i one biti ukratko navedene.

Afina transformacija ne čuva oblike jer ne čuva kutove. Njome ostaju očuvane samo paralelne linije. U divje dimenzije ona ima šest stupnjeva slobode,

$$\boldsymbol{x'} = \begin{bmatrix} 1+a_{11} & a_{12} & t_x \\ a_{21} & 1+a_{22} & t_y \end{bmatrix} \bar{\boldsymbol{x}}, \tag{2.14}$$

uz $\boldsymbol{p} = (t_x, t_y, a_{11}, a_{12}, a_{21}, a_{22})$, a u trodimenzijskom obliku ima dvanaest,

$$\boldsymbol{x'} = \begin{bmatrix} 1+a_{11} & a_{12} & a_{13} & t_x \\ a_{21} & 1+a_{22} & a_{23} & t_y \\ a_{31} & a_{32} & 1+a_{33} & t_z \end{bmatrix} \bar{\boldsymbol{x}}, \tag{2.15}$$

uz $\boldsymbol{p} = (t_x, t_y, t_z, a_{11}, \ldots, a_{33})$.

Projektivna transformacija. Ova transformacija djeluje nad homogenim koordinatama i naziva se još i perspektivna transformacija ili homeografija. Projektivnom transformacijom se čuvaju samo ravne linije, ali ne i duljine, kutovi, ili orijentacija.

U dvije dimenzije ima osam stupnjeva slobode,

$$\tilde{\boldsymbol{x}}' = \begin{bmatrix} 1+h_{11} & h_{12} & h_{13} \\ h_{21} & 1+h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \tilde{\boldsymbol{x}}, \tag{2.16}$$

uz $\boldsymbol{p} = (h_{11}, \ldots, h_{32})$. A u tri dimenzije petnaest i ima oblik,

$$\tilde{\boldsymbol{x}}' = \begin{bmatrix} 1+h_{11} & h_{12} & h_{13} & h_{14} \\ h_{21} & 1+h_{22} & h_{23} & h_{24} \\ h_{31} & h_{32} & 1+h_{33} & h_{34} \\ h_{41} & h_{42} & h_{43} & 1 \end{bmatrix} \tilde{\boldsymbol{x}}, \tag{2.17}$$

uz $\boldsymbol{p} = (h_{11}, \ldots, h_{43})$.

Oblik ili vrsta tražene transformacije dakle, ovise o zadanom problemu i postupku uzorkovanja. Općenito linearne transformacije oblika prikazanog jednadžbom 2.4, možemo svrstati prema broju stupnjeva slobode kao u tablici 2.1.

Vrsta transformacije	Matrica	Stupnjeva slobode	Očuvano
Translacija u $\mathbb{R}^2$	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	2	orijentacija
Euklidska u $\mathbb{R}^2$	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	3	duljine
Sličnost u $\mathbb{R}^2$	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	4	kutovi
Afina u $\mathbb{R}^2$	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times 3}$	6	paralelnost
Projekcija u $\mathbb{R}^2$	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times 3}$	8	ravne linije
Translacija u $\mathbb{R}^3$	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{3\times 4}$	3	orijentacija
Euklidska u $\mathbb{R}^3$	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{3\times 4}$	6	duljine
Sličnost u $\mathbb{R}^3$	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{3\times 4}$	7	kutovi
Afina u $\mathbb{R}^3$	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{3\times 4}$	12	paralelnost
Projekcija u $\mathbb{R}^3$	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{4\times 4}$	15	ravne linije

Tablica 2.1: Tipovi transformacija prema domeni i broju stupnjeva slobode

Ukoliko postavimo takva ograničenja na funkciju transformacije, jasno je da neće svaka slučajno odabrana četvorka parova točaka dati ispravnu transformaciju. To jest, jednadžba 2.4 će dati ispravan rezultat samo u slučaju da su odabrani točno oni parovi točaka koji koreliraju, te da slika točaka polaznog skupa točno odgovara podudarnim točkama ciljanog skupa kao što je već rečeno. Kako to općenito nije slučaj, potrebno je primijeniti neki od postupaka optimiranja kako bi se našli optimalni parametri matrice transformacije. Upravo iz tog razloga je poželjno imati čim manji broj parametara (to jest stupnjeva slobode), i razumno velik broj podudarnih parova kako bi postupak optimiranja konvergirao do prihvatljivog rješenja.

2.3. Pristupi određivanju transformacije podudaranja

Prvi korak postupka određivanja transformacije podudaranja je priprema ulaznih skupova uzoraka. Ovisno o obliku ulaznih podataka, potrebno je te podatke prilagoditi postupku kojim se dalje traže podudarne točke skupova, ili postupku traženja značajki. Najčešće se to odnosi na uklanjanje pogrešaka uzorkovanja, na primjer šuma, izoliranih točaka, grubih pogrešaka uzorkovanja, ili korekcije iskrivljenja. Te na pripremnu

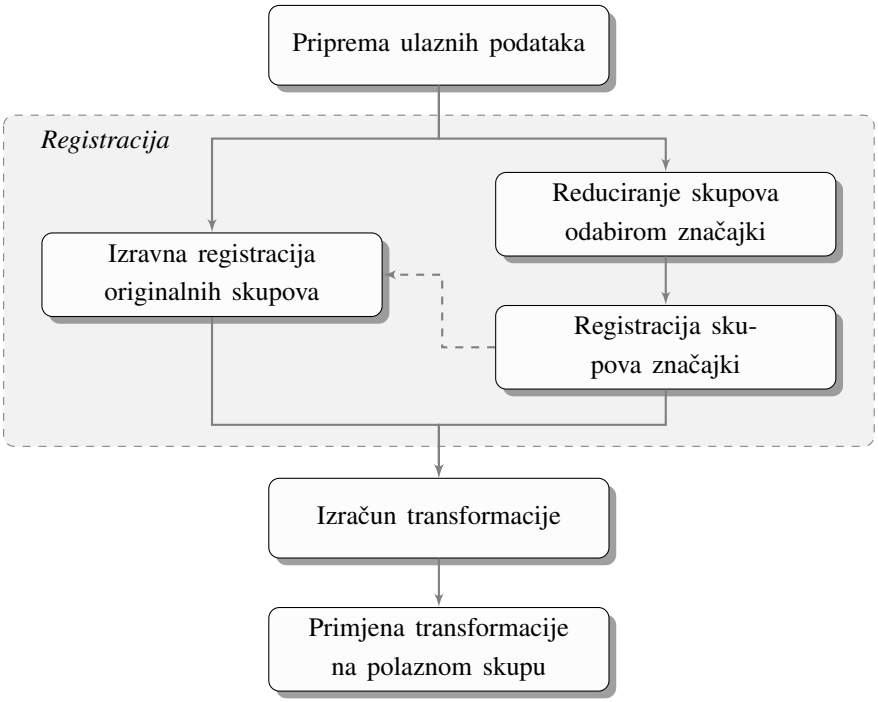
obradu uzoraka u smislu procjene zakrivljenosti i vektora normale uzorkovane plohe u točki uzorka (ukoliko oni nisu mjereni u postupku uzorkovanja), zatim rekonstrukcija mreže poligona, ujednačavanje gustoće uzoraka selekcijom, i tako dalje...

Slijedeći korak je registracija trodimenzijskih skupova uzoraka. Pojam registracije u užem smislu obuhvaća pronalaženje podudarnih parova točaka, što je preduvjet za izračun tražene transformacije preslikavanja. U širem smislu možemo reći da je to postupak koreliranja informacije sadržane u polaznom skupu s informacijom sadržanom u ciljnom skupu uzoraka. Ovdje se pod informacijom smatra prostorna informacija o površini uzorkovanog objekta. To jest, dio entropije površine objekta koji je postupkom uzorkovanja očuvan u podacima skupova uzorkovanih točaka. Registracija je u tom smislu pronalaženje transinformacije koja opisuje vezu između dva skupa uzoraka.

Dva su glavna pristupa u izvedbi registracije skupova uzorkovanih podataka. Jedan se zasniva izravnom traženju optimalne transformacije koristeći cijele skupove uzoraka, a uobičajeni pristup je koristeći neki od postupaka nelinearnog optimiranja. Prvo se definira skalarna funkcija cilja $F : \mathbb{R}^n \mapsto \mathbb{R}$ kojom se kvalitativno ocjenjuje transformacija preslikavanja. n označava broj stupnjeva slobode tražene transformacije. Funkcija cilja predočava odstupanje trenutnog rješenja od zadanih kriterija. Zatim se odabere početno rješenje koje može biti odabrano slučajno, manualno na osnovi iskustva, ili na osnovi nekog proračunskog predviđanja. Nakon toga se iterativno podešavaju parametri transformacije tako da se to odstupanje smanjuje. To jest, minimizira se vrijednost funkcije cilja $F(\boldsymbol{A})$. Postupci koji se pri tome koriste su, postupak minimiziranja srednje kvadratne pogreške (engl. Minimum mean square error ili MMSE), ICP postupak (engl. Iterated closest point), i drugi...

Ovim se pristupom dobivaju vrlo precizni rezultati upravo zbog toga što se zasniva na minimiziranju ukupne ili srednje pogreške cijelog skupa podataka. Prednost je i relativno jednostavna izvedba. Vrijeme izvršavanja postupka ovisi o broju ulaznih podataka, i o broju iteracija potrebnih za konvergiranje. Broj uzoraka se po potrebi može smanjiti tijekom pripreme ulaznih podataka ukoliko je potrebno, no veći problem je broj iteracija. On naravno ovisi o primjenjenom postupku, ali u velikoj mjeri i o odabranom početnom rješenju. To je upravo najveći nedostatak izravnog pristupa registraciji skupova. Ukoliko je početno rješenje loše odabrano, biti će potreban velik broj iteracija kako bi se došlo do traženog rješenja. Ili, u najgorem slučaju, postupak niti neće konvergirati.

Drugi se pristup sastoji iz dva koraka. Iz uzorkovanih skupova podataka se prvo izdvajaju samo one točke koje su značajne za uspješnu registraciju. Zatim se među tim odabranim točkama pronalaze podudarni parovi pomoću kojih se izračunava transfor-



Slika 2.1: Pristupi registraciji trodimenzijskih skupova uzoraka

macija. Ne moraju nužno izdvojene biti samo točke skupa, moguće je izdvajati podskupove ili regije koje predstavljaju reprezentativno prostorno obilježje uzorkovanog objekta, koje se pak može u čim većoj mjeri korelirati s podudarnim obilježjem ciljnog skupa uzoraka. Zato govorimo o značajkama općenito (engl. features), a ne striktno o značajnim točkama skupa. Jedna takva značajka je predstavljena s minimalno jednom točkom u prostoru, ali ukoliko je to moguće, može biti sa više. Najčešće je to jedna točka i vektor normale na uzorkovanu površinu. Na taj način su dovoljna dva para podudarnih značajki kako bi se jednoznačno odredila matrica transformacije.

Glavni zadatak kod odabira značajki je pri tome sačuvati čim je moguće veću količinu one entropije uzorkovane površine objekta, koja je potrebna (i dovoljna) za uspješnu registraciju skupova. To jest, u postupku odabira očuvati dovoljnu količinu transinformacije među skupovima, kako bi uspješna registracija bila moguća.

Izdvajanjem značajki se u velikoj mjeri smanjuju skupovi u kojima se traže podudarni parovi. Osim toga, ukoliko je postupak izdvajanja značajki uspješan prema gore postavljenom zadatku, pretpostavlja se da je u izdvojenim značajkama sadržana samo potrebna informacija. To jest da je izostavljena ona informacija, koja bi eventualno vodila ka pogrešnom registriranju skupova. Smanjeni skupovi pretraživanja podudarnih parova omogućuju također i pretraživanje većeg broja mogućih rješenja transformacije podudaranja.

Prednost ovog pristupa je svakako veća robusnost budući da nije potrebno pret-

postaviti početno rješenje o kojemu kasnije u velikoj mjeri ovisi ishod postupka. U većini slučajeva moguće je pretražiti sve kombinacije kako bi se došlo do rezultata. S druge strane, rješenje dobiveno na ovaj način nije pouzdano precizno, i ovisi o kvaliteti izdvojenih značajki. No pod pretpostavkom da su značajke dobro odabrane, postupak registracije će dati ispravno korelirane skupove. A posljedično i ispravnu matricu preslikavanja.

Tako dobivena matrica transformacije je točna, ali ne nužno i precizna. To je posljedica odbacivanja velikog broja uzoraka. Zbog toga se ponekad tako dobivena matrica transformacije može koristiti kao inicijalno rješenje za izravni postupak registracije. Time se osigurava brzo i uspješno konvergiranje izravnog postupka ka preciznom rezultatu.

Fokus ovog rada je na ovom drugom neizravnom pristupu registraciji skupova trodimenzijskih uzorkovanih podataka i postupcima koji se u tu svrhu koriste.

3. Pretpostavke i ograničenja ulaznih podataka

3.1. Karakteristike uzorkovanih objekata

U daljem tekstu se podrazumijevaju neke karakteristike uzorkovanih objekata koje same po sebi intuitivno logične, ali su ovdje navedene radi potpunosti.

Uzorkovana ploha je orijentabilna. Ovo je nužna pretpostavka za puno postupaka koji se oslanjaju na postojanje normalnog vektora u uzorkovanim točkama. U praksi svaki uzorkovani objekt ima jasno definirano što je unutar objekta, a što izvan njega, pa je time i njegova površina orijentabilna.

Uzorkovana ploha je neprekinuta. Kao i u prethodnoj pretpostavci, tako je i ovdje jasno da je površina uzorkovanog objekta zatvorena i neprekinuta. Iako objekt može imati oštre bridove koji bi, gledajući samo skup uzoraka, mogli izgledati kao prekid kontinuiteta plohe, u stvarnom slučaju možemo uvijek naći dovoljno mali polumjer zakrivljenosti na tom mjestu kako bismo plohu opisali neprekinutom funkcijom. To je važna pretpostavka kod određivanja granice skupa uzoraka i rubnih točaka.

Vremenska nepromjenjivost plohe. Pod vremenskom nepromjenjivošću ovdje smatramo da se uzorkovani objekt nije promijenio tijekom jedne serije uzorkovanja. To jest prilikom uzorkovanja jednog skupa točaka. Ne odnosi se na projene između dvije serije, koje mogu biti i namjerne. Ova pretpostavka često nije zadovoljena u praktičnim primjerima. Pogotovo ukoliko su uzorkovani objekti živa bića. Ali će se u ostatku teksta svejedno smatrati da je ona točna, te će se promjene na objektima koje se događaju tijekom jedne serije uzorkovanja, smatrati pogreškama u mjerenju (na primjer treptanje ili disanje kod uzorkovanja lica).

Gustoća uzorkovanja. Pretpostavljamo da se uzorkovanje vrši približno jednolikom gustoćom uzorkovanja u sve tri prostorne dimenzije. Ili barem da ta pretpostavka vrijedi lokalno u susjedstvu svakog uzorka. Uzorci koji odstupaju od ovog pravila preko određene granice tolerancije, smatraju se neispravnima, to jest grubim pogreškama mjerenja.

Euklidski prostor uzoraka. Podrazumijevamo da su uzorci zapisani Kartezijevim koordinatama u euklidskom prostoru.

3.2. Izvori pogrešaka ulaznih podataka

Šum. Ovisno o postupku uzorkovanja i korištenom uređaju, uvijek postoji određena pogreška u mjerenju. Ta se pogreška može odnositi na prostornu pogrešku određivanja položaja točke, ali i na određivanje smjera vektora normale, ili na mjerenje boje objekta. Podrazumijevamo normalnu (Gaussovu) raspodjelu pogrešaka.

Sjena. Svaki uređaj ima svoja mehanička ili tehnološka ograničenja zbog kojih ne može uzorkovati cijelu površinu objekta u jednoj seriji uzorkovanja. Područje površine koje nije bilo uzorkovano nazivamo područjem u sjeni u analogiji sa sjenom koju baca izvor signala korišten od strane uređaja. Sjena od izvora signala je najčešće i sam uzrok sjene u skupu uzoraka, ali uzrok može biti i drugog porijekla. Na primjer dijelovi objekta koji se ne vide iz perspektive prijemnika signala jer su zaklonjeni ostatkom objekta, ili artefakti koji nastaju postupkom tomografije. Pod sjenom ne smatramo prostor između uzoraka koji nastaje diskretizacijom na područjima koji su uređaju uredno dostupni.

Gustoća uzorkovanja. Prostorna ili ravninska gustoća uzoraka može varirati. Ona ovisi o kvaliteti i tipu uređaja, a često i o udaljenosti promatranog predmeta od uređaja.

Iskrivljenje. Da bismo dobili uzorke u Kartezijevim koordinatama, uređaji (ili naknadno obradom sirovih podataka s uređaja) primjenjuju transformacije kako bi uzorke preveli iz prostora mjerenja u euklidski prostor. U tom procesu je važno da su uređaj i primjenjena transformacija ispravno kalibrirani. Unatoč tome, u praksi su moguća veća ili manja iskrivljenja u prostoru uzorkovanja. Ta iskrivljenja se također smatraju pogreškom mjerenja.

3.3. Dodatne informacije o uzorcima

Osim podatka o prostornom položaju, uzorak može uz sebe imati vezanu i dodatnu informaciju o promatranoj točki na objektu.

Vektor normale. Vektor okomit na površinu originalnog objekta u točki uzorkovanja, ili vektor normalne, je ponekad dobiven izravno postupkom uzorkovanja, ali postoje postupci kojima se on može naknadno procijeniti. U daljem tekstu se podrazumijeva da je vektor normale poznat (ili procijenjen) za svaku točku skupa uzoraka.

Zakrivljenost. Zakrivljenost se obično ne mjeri uređajem za uzorkovanje, već se ona dobiva proračunski naknadno. U dvodimenzijskom prostoru površine objekta općenito imamo dvije glavne zakrivljenosti. No kako parametrizacija te plohe nije poznata, tako ne možemo niti govoriti o glavnim zakrivljenostima, već se zakrivljenost kvalitativno opisuje na razne načine. Jedan pristup je da se procjenjuju minimalna i maksimalna zakrivljenost u promatranoj točki. Drugi pristup procjenjuje zakrivljenost plohe samo jednim skalarom. Rezultat je uvijek jedan ili dva skalara koji kvantitativno opisuju svojstva plohe lokalno oko promatrane točke.

Boja. Boja je opcionalna informacija u kontekstu ovog rada, ali se svi postupci koje ćemo opisati na na primjeru zakrivljenosti mogu primjeniti i na boju. Ili se ti podaci mogu koristiti simultano kako bi se povećala uspješnost postupka. Boja se obično dobiva postupkom uzorkovanja ukoliko to uređaj podržava.

4. Priprema podataka

4.1. k-Susjedstvo

Pod k-susjedstvom (engl. k-neighborhood) se smatra k točaka najbližih promatranom uzorku, s time da se udaljenost među točkama mjeri kao najkraći put, od jedne točke do druge, po površini uzorkovanog objekta. To jest, k-susjedstvo predstavlja jedan neprekinuti dio uzorkovane plohe u okolici promatrane točke.

Kako je originalni oblik površine objekta izgubljen diskretizacijom, i nije poznata parametrizacija prostora te površine, u zadanom skupu uzoraka nije moguće egzaktno odrediti k-susjedstvo po gore navedenoj definiciji. No moguće je približno odrediti koje su to točke uz pretpostavku da je gustoća uzorkovanja dovoljno velika. U tom slučaju možemo pretpostaviti da se neprekinuti dio uzorkovane plohe objekta kojega opisuje k-susjedstvo, može aproksimirati osno-simetričnom plohom oko vektora normale u toj točki. Tada se najbliži susjedi mogu određivati i prostornom euklidskom udaljenošću točaka. Također, moramo pretpostaviti da je gustoća uzoraka dovoljno velika i da se unutar euklidske udaljenosti najdalje točke k-susjedstva ne nalazi neka točka koja pripada drugom objektu ili dijelu iste plohe, koji ne ulazi u k-susjedstvo po gornjoj definiciji.

U praksi su ovakve striktne pretpostvake rijetko zadovoljene, ali se aproksimacija susjedstva koristeći euklidsku prostornu udaljenost svejedno koristi jer daje dovoljno dobre rezultate tamo gdje se primjenjuje. No važno je znati da kasniji postupci podrazumijevaju k-susjedstvo na površini objekta, a da je euklidsko prostorno k-susjedstvo samo aproksimacija te da potencijalno unosi pogreške u računanje.

4.2. Normale

Postupci koji će dalje biti navedeni podrazumijevaju da je vektor normale na uzorkovanu plohu, u svakoj od uzorkovanih točaka, poznat. Neke metode uzorkovanja kao

rezultat mjerenja daju i vektor normale na uzorkovanu površinu u promatranoj točki. No kako to nije općeniti slučaj, vektor normale može biti i naknadno procijenjen na temelju informacije o k-susjedstvu promatrane točke.

Aproksimacija polinomijalnom plohom (engl. jet surface fitting) preko k susjeda promatrane točke je jedna od metoda. (jet_estimate_normals funkcija paketa CGAL)

Nešto jednostavnija i brža metoda je aproksimacija ravninom koristeći linearnu metodu najmanjih kvadrata. (pca_estimate_normals funkcija paketa CGAL)

Navedene metode dobro pogađaju prostorni smjer vektora normale, ali je predznak smjera određen slučajem. To jest nije jednoznačno koja je strana plohe unutar objekta, a koja van njega. Zato je potrebno naknadno korigirati orijentaciju što se može učiniti na primjer postupkom kojega predlažu Hoppe et al. (Surface reconstruction from unorganized points). Osnovna ideja je da se orijentacija normale propagira preko minimalnog razapinjućeg stabla. (funkcija mst_orient_normals paketa CGAL)

4.3. Zakrivljenost

Mjera zakrivljenosti ili kraće zakrivljenost je važan podatak za kasnije određivanje značajki u skupovima točaka. Pošto taj podatak ne dobivamo postupkom uzorkovanja, niti ga je moguće naknadno egzaktno utvrditi, on mora biti procijenjen na osnovu postojećih podataka. Mogući su razni pristupi tom problemu. Ovdje će biti navedeni samo neki, a njihova učinkovitost kasnije i ispitana.

Aproksimacija stošcem. Ideja je da se k-susjedstvo točke aproksimira stošcem čija je osnovna os kolinearna s normalom u promatranoj točki. Mjera zakrivljenosti se onda definira kao kosinus kuta između vektora normale i plašta stošca. Ekvivalent kosinusa kuta je visina takvog stošca čija je duljina izvodnice normirana.

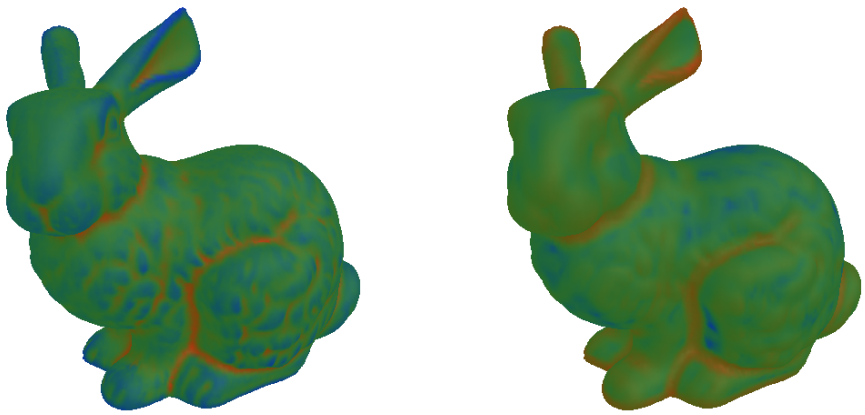
$$c_{konus}(\boldsymbol{x}_i) = \frac{1}{k} \sum_{j=1}^{k} \boldsymbol{n}_i^T \frac{\boldsymbol{r}_j}{\|\boldsymbol{r}_j\|} \tag{4.1}$$

Ova mjera daje vrijednosti u intervalu $[-1, 1]$, gdje su pozitivne vrijednosti u slučaju konkavne površine oko vektora normale, a negativne u slučaju konveksne površine s normalom na vrhu.

Srednji skalarni produkt normala. Kao mjera zakrivljenosti se uzima srednji skalarni produkt vektora normale promatrane točke, i vektora normale svakog od k-susjeda.

$$c_{dot}(\boldsymbol{x}_i) = \frac{1}{k}\sum_{j=1}^{k} \boldsymbol{n}_i^T \boldsymbol{n}_j \tag{4.2}$$

Vrijednosti se zakrivljenosti dobivene ovim postupkom također nalaze u intervalu $[-1, 1]$, ali s velikom gustoćom u blizini gornje granice intervala.



(a) **(b)**

Slika 4.1: Zakrivljenost: (a) Aproksimacija stošcem, (b) Srednji skalarni produkt normala.

4.4. Površina objekta (poligonalna mreža)

Kao i normale, tako se nekim postupcima uzorkovanja mogu dobiti i poligonalne mreže površine objekta. Posebno u postupcima gdje se točke uzorkuju slijedno, red po red. Iako poligonalne mreže nisu nužne za samu registraciju, poželjno je imati ih na raspolaganju iz nekoliko praktičnih razloga. Prvi je lakša vizualizacija skupova uzoraka i njihovih karakteristika. Zatim lakše određivanje susjednih uzoraka, te uzoraka na rubu zasjenjenog područja gdje su češće pogreške mjerenja i manjkavosti postupaka određivanja značajki.

Poligonalna se mreža može i naknadno rekonstruirati. Taj postupak nazivamo triangulacijom (engl. triangulation). Jedan brzi postupak pohlepnim algoritam triangulacije plohe je implementiran u paketu PCL (Point Cloud Library) (Marton et al., 2009), zatim PowerCrust (Amneta et al., 2001), Poissonova rekonstrukcija, i drugi...

5. Postupci određivanja značajki

5.1. Opis problema i ciljevi

Smisao izdvajanja značajnih točaka iz početnih skupova podataka je u smanjenju prostora domene pri kasnijem postupku traženja transformacije podudaranja. Time se općenito skraćuje vrijeme potrebno za taj postupak. Osim toga, povećava se i vjerovatnost pronalaska ispravnog rješenja. Izdvajanjem značajnih točaka eliminiraju se one čije bi prisustvo postupak odvelo u krivom smjeru, pa traženje transformacije podudaranja ne bi nužno konvergiralo prema ispravnom rješenju.

Izdvojene značajke iz svakog od skupova ulaznih podataka trebaju biti takve da, ako se podskupovi ulaznih podataka koji su opisani tim značajkama podudaraju, onda se moraju podudarati i same značajke. Budući da izbor značajki ovisi o primjenjenom postupku izdvajanja, tako je za očekivati da će se na svakome od ulaznih skupova primijeniti isti postupak. Općenito to ne mora biti slučaj, ali u praksi gotovo uvijek je. Uz uvjete koji su postavljeni nad ulaznim podacima, taj izbor je ispravan pa će se nadalje u opisu postupaka određivanja značajki promatrati jedan skup ulaznih podataka podrazumijevajući da će se isti postupak primijeniti i na drugi skup.

Prvi zahtjev na postupak izdvajanja značajki iz ulaznog skupa je da se tim postupkom očuva ona informacija sadržana u ulaznom skupu koja je nužna za uspješno pronalaženje tražene podudarnosti. Pri tome je poželjno i da se očuva *samo* nužna informacija jer se tako eliminiraju smetnje i povećava robusnost cjelokupnog postupka.

Koja informacija treba biti očuvana, a koja odbačena, nagoviješteno je i samom definicijom problema, tj. definicijom tražene transformacije podudaranja. Na primjer, budući da je pretpostavka da su apsolutni položaj i rotacija skupova nepoznati, nije ispravno tu informaciju koristiti pri odabiru značajki skupa. U navedenom se slučaju problem može činiti trivijalan, ali u drugima (skaliranje, afine transformacije, iskrivljenja uzorkovanja, projekcija, šum, i sl...) ne mora biti. Općenito to može odlučivati o primjenjivosti pojedinog postupka na zadani problem. S time u vidu, jedan od kriterija za usporedbu treba biti i osjetljivost postupaka na nepoznanice definirane zadatkom.

Jedan od ključnih čimbenika primjenjivosti postupka određivanja značajki je i odabir prikladne domene. Domena ulaznog skupa uzoraka sadrži trodimenzijske prostorne informacije kao što su pozicija i orijentacija uzorka, zatim pozicija i orijentacija u dvodimenzijskom prostoru uzorkovane površine, kao i neprostorne informacije, na primjer boja, zakrivljenost, gustoća uzorkovanja, i sl. Očito je da nisu svi atributi potrebni za određivanje značajki, niti su svi primjenjivi u svakom postupku. Potrebno je, dakle, domenu ulaznih podataka reducirati kako bi bila prikladna za odabrani postupak.

5.2. Primjena dvodimenzijskih postupaka

Neki od postupaka, koji se koriste za odabir značajki pri registraciji slika u dvije dimenzije, mogu se primijeniti i na problem značajki na trodimenzijskim objektima. U osnovi bi se to trebalo podrazumijevati jer je površina uzorkovanog objekta dvodimenzijski prostor, dakle postoji jednoznačno preslikavanje $\boldsymbol{\varphi} : \mathbb{R}^2 \to \mathbb{R}^2$ te plohe na ravninu. Po tome bi se svaki postupak, koji se može primijeniti na ravninu, mogao primijeniti i na plohu uzorkovanog objekta. Ali općenito, to preslikavanje podrazumijeva proizvoljne parametre (na primjer ishodište i orijentacija) koji u velikoj mjeri utječu na geometriju rezultirajuće slike, a time i na rezultat postupka odabira značajki. Takve parametre je u konkretnom slučaju, na odvojenim skupovima uzoraka, teško odrediti ispravno bez poznavanja međusobnog odnosa tih skupova. Taj je odnos po definiciji nepoznanica pa je problem preslikavanja dvodimenzijskog prostora plohe uzorkovanog objekta, uz očuvanje podudarnih značajki međusobno odvojenih skupova uzoraka, na ravninu netrivijalan.

Ipak, veliki je broj postupaka moguće prilagoditi za ovaj konkretan slučaj. Na primjer, kod postupaka koji značajke traže analizirajući užu okolinu svake promatrane točke $\boldsymbol{x}_i$, problem ishodišta se može zanemariti ukoliko se svaka takva okolina preslikava zasebno $\boldsymbol{\varphi}_i : \mathbb{R}^2 \to \mathbb{R}^2$. Time se uz određene pretpostavke ili aproksimacije može u zadovoljavajućoj mjeri očuvati geometrija slike okoline uzorkovane točke. Slično tome, kod onih koji koriste kružno simetrični kernel pri izračunavanju nije bitna niti orijentacija. Ukoliko je pak orijentacija bitna, ponekad je moguće koristiti gradijent $\vec{\nabla} c(\boldsymbol{x}_i)$ kao osnovni smjer takve slike okoline, ukoliko se pri tome čuva podudarnost među skupovima uzoraka.

S druge strane, postupke koji se zasnivaju na frekvencijsko-faznoj analizi uglavnom nije moguće prilagoditi. Svakako ovisi od postupka do postupka pa će stoga način prilagodbe u tom smislu za svaki postupak biti razmatran zasebno.

5.3. Aproksimacija okoline uzorka ravninom

Kako bi bilo moguće primijeniti dvodimenzijske postupke određivanja značajki, prvo je potrebno projicirati okolinu promatrane točke $\boldsymbol{x}_i$ na ravninu. Ravnina neka bude okomita na vektor normale u toj točki, to jest neka je ravnina, u koju projiciramo okolinu, upravo tangencijalna ravnina na uzorkovanu plohu u točki $\boldsymbol{x}_i$. Orijentacija koordinatnog sustava ravnine mora biti odabrana tako da se ne gubi prijelazom u drugi referentni sustav. Neka to bude smijer najvećeg gradijenta skalarne funkcije zakrivljenosti $c(\boldsymbol{x})$, to jest $\vec{\nabla} c(\boldsymbol{x}_i)$. To je moguće postići na slijedeći način:

1. Izračunati vektor u smjeru najvećeg rasta funkcije $c(\boldsymbol{x})$ u trodimenzijskom prostoru oko $\boldsymbol{x}_i$. To se postiže tako da se funkcija zakrivljenosti oko točke $\boldsymbol{x}_i$ aproksimira lineranom skalarnom funkcijom

$$c(\boldsymbol{x}) = \boldsymbol{g}(\boldsymbol{x}_i)^T \boldsymbol{x} + c(\boldsymbol{x}_i). \tag{5.1}$$

   Na taj način, uz definiciju funkcije cilja kao

$$E_{LLS} = \frac{1}{k} \sum_{j=1}^{k} \left\| \Delta_{ji} c(\boldsymbol{x}) - \Delta c_{ji} \right\|^2, \tag{5.2}$$

   možemo linearnom metodom najmanjih kvadrata dobiti vektor $\boldsymbol{g}(\boldsymbol{x}_i)$ iz jednadžbe

$$\boldsymbol{g}(\boldsymbol{x}_i) = \frac{1}{a_i} \boldsymbol{b}_i, \tag{5.3}$$

   gdje su

$$\boldsymbol{b}_i = \sum_{j=1}^{k} \Delta \boldsymbol{x}_{ji} \Delta c_{ji},$$

$$a_i = \sum_{j=1}^{k} \Delta \boldsymbol{x}_{ji}^T \Delta \boldsymbol{x}_{ji}.$$

   Skalarna se funkcija gradijenta u $\boldsymbol{x}_i$, ukoliko bi ona bila zanimljiva, tada može dobiti kao duljina tog vektora,

$$c_{grad}(\boldsymbol{x}_i) = \left\| \boldsymbol{g}(\boldsymbol{x}_i) \right\|. \tag{5.4}$$

2. Preko vektora prostornog gradijenta izračunati osnovnu os tangencijalne ravnine oko točke $\boldsymbol{x}_i$,

$$\boldsymbol{u}_p(\boldsymbol{x}_i) = \boldsymbol{g}(\boldsymbol{x}_i) - (\boldsymbol{g}(\boldsymbol{x}_i)^T \boldsymbol{n}_i)\boldsymbol{n}_i \tag{5.5}$$

3. Normalizirati vektor osnovne osi,

$$\boldsymbol{u}_p(\boldsymbol{x}_i) \leftarrow \frac{\boldsymbol{u}_p(\boldsymbol{x}_i)}{\|\boldsymbol{u}_p(\boldsymbol{x}_i)\|} \tag{5.6}$$

4. Izračunati sekundarnu os normalne ravnine oko $\boldsymbol{x}_i$,

$$\boldsymbol{v}_p(\boldsymbol{x}_i) = \boldsymbol{n}_i \times \boldsymbol{u}_p(\boldsymbol{x}_i) \tag{5.7}$$

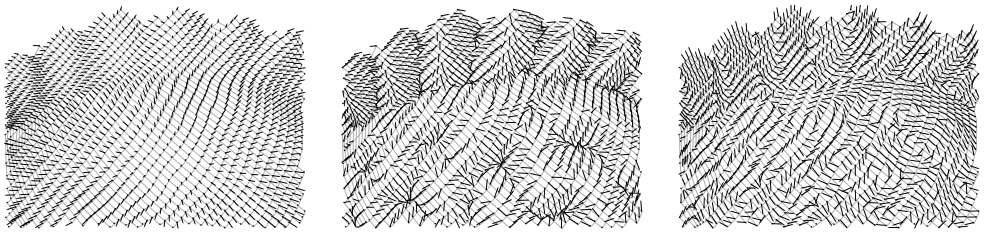
Vektori $\boldsymbol{u}_p(\boldsymbol{x})$ i $\boldsymbol{v}_p(\boldsymbol{x})$ su ilustrirani primjerom na slici 5.1.

Matrica projekcije točaka iz $\mathbb{R}^3$ na normalnu ravninu oko $\boldsymbol{x}_i$, razapetu jediničnim vektorima $\boldsymbol{u}_p(\boldsymbol{x}_i)$ i $\boldsymbol{v}_p(\boldsymbol{x}_i)$ je tada

$$\boldsymbol{A}(\boldsymbol{x}_i)_{2\times 3} = \Big[\boldsymbol{u}_p(\boldsymbol{x}_i) \mid \boldsymbol{v}_p(\boldsymbol{x}_i)\Big]^T \tag{5.8}$$

A funkcija transformacije kojom dobivamo koordinate točke $\boldsymbol{x}$ u normalnoj ravnini oko točke $\boldsymbol{x}_i$,

$$f_i(\boldsymbol{x}) = \boldsymbol{A}(\boldsymbol{x}_i)(\boldsymbol{x} - \boldsymbol{x}_i). \tag{5.9}$$



(a) **(b)** **(c)**

Slika 5.1: Aproksimacija okoline uzorka: (a) Poligonalna mreža s točkama i pripadajućim vektorima normale, (b) osnovnim vektorima ravnine u smjeru gradijenta $\boldsymbol{u}_p(\boldsymbol{x}_i)$, (c) i sekundarnim vektorima ravnine $\boldsymbol{v}_p(\boldsymbol{x}_i)$.

Orijentacija koordinatnih osi tangencijalne ravnine, ovako određena smjerom gradijenta, nije definirana u svakoj točki uzorkovane površine objekta. Točnije, tamo gdje je gradijent jednak nuli, nije moguće odrediti smjer gradijenta. To se događa upravo u

točkama koje su lokalni minimumi ili lokalni maksimumi skalarne funkcije zakrivljenosti $c(\boldsymbol{x})$. Na to treba obratiti pažnju prilikom definiranja deskriptora značajki.

5.4. Postupci

5.4.1. Osnovni algoritam određivanja značajki

Prilikom postupaka registracije dvodimenzijskih slika, postavlja se pitanje što čini neku točku slike boljim ili lošijim izborom kada je u pitanju traženje korelacije s odgovarajućom točkom iz druge slike. Intuitivno bismo odabrali točke koje u svojoj okolini imaju velike, kontrastne promjene. Također je poželjno da su te promjene izražene u dva neovisna smjera kako bi bilo moguće jednoznačno lokalizirati takvu točku. Ove intuitivne zaključke je moguće formalizirati najjednostavnijim kriterijem za koreliranje dvaju dijelova dvodimenzijske slike, to jest zbrojem kvadrata njihove razlike, pomnoženim težinskom funkcijom (engl. weighted summed square difference),

$$E_{WSSD}(\boldsymbol{u}) = \sum_i w(\boldsymbol{x}_i) \left[I_1(\boldsymbol{x}_i + \boldsymbol{u}) - I_0(\boldsymbol{x}_i)\right]^2 \tag{5.10}$$

gdje su I_0 i I_1 dvodimenzijske slike koje uspoređujemo, $\boldsymbol{u}$ je vektor pomaka jedne slike u odnosu na drugu u dvije dimenzije, a $w(\boldsymbol{x}_i)$ je težinska funkcija kojom je promatranje lokalizirano oko točke $\boldsymbol{u}$. Od značajki se zahtjeva da budu nedvosmislene u svojoj bližoj okolici kako bi se mogle uspješno kasnije poravnati. Zato možemo odabrati značajke tako da se mali pomak u bilo kojem smjeru očitava kao čim veća promjena zbroja kvadrata razlike. To se naziva auto-korelacijska funkcija ili ploha, a računa se kao

$$E_{AC}(\Delta\boldsymbol{u}) = \sum_i w(\boldsymbol{x}_i) \left[I_0(\boldsymbol{x}_i + \delta\boldsymbol{u}) - I_0(\boldsymbol{x}_i)\right]^2. \tag{5.11}$$

Točke koje na auto-korelacijskoj plohi imaju jasno izražen minimum, mogu se vrlo uspješno lokalizirati jer pomakom u bilo kojem smjeru pogreška jasno raste.

Koristeći razvoj auto-korelacijske funkcije Taylorovim redom (Shi i Tomasi, 1994), dobivamo aproksimaciju te funkcije u obliku

$$E_{AC}(\Delta\boldsymbol{u}) = \Delta\boldsymbol{u}^T \boldsymbol{A} \Delta\boldsymbol{u}, \tag{5.12}$$

gdje je

$$\boldsymbol{A} = w \cdot \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}. \tag{5.13}$$

I_x i I_y označavaju parcijalne derivacije funkcije $I(\boldsymbol{x})$ u smjeru komponenti dvodimenzijskog vektora $\boldsymbol{x}$ x i y.

Mnogi se postupci određivanja značajki u dvije dimenzije zasnivaju na auto-korelacijskoj funkciji. Među njima i klasični Harrisov detektor (Harris i Stephens, 1988) te detektor razlike Gaussiana (engl. difference of Gaussian - DoG). Osnovni algoritam tih postupaka po Szeliskom (Szeliski, 2010) je slijedeći:

1. Izračunati horizontalne i vertikalne derivacije slike I_x i I_y konvolucijom originalne slike s derivacijama Gaussa;
2. Izračunati tri slike, koje odgovaraju umnošcima tih derivacija. Tako da one čine komponente matrice korelacije $\boldsymbol{A}$;
3. Konvolvirati svaku od tih slika s većim Gaussom;
4. Izračunati skalarnu mjeru značaja;
5. Pronaći lokalne maksimume iznad određene razine i prijaviti ih kao značajke.

5.4.2. Neovisnost o gustoći uzorkovanja

Vrlo bitan cilj kod određivanja značajki je postići da su one neovisne o skaliranju skupa uzoraka, ili promjeni gustoće uzorkovanja. Jedno često korišteno rješenje uključuje korištenje piramide različitih rezolucija slike prilikom određivanja značajki. Na taj se način mogu izdvojiti one značajke koje su manje osjetljive na skaliranje.

Dio problema koji se odnosi na različitu gustoću uzorkovanja, ali uz poznato skaliranje, riješava se tako da se parovi značajki traže na istim razinama piramide rezolucija. Ukoliko na jednoj od slika postoji pomanjkanje značajki na određenoj razini (zbog manje gustoće uzorkovanja, na primjer), tada se podudarne značajke mogu naći na nekoj od ostalih razina piramide.

Općenitije rješenje je tražiti značajke koje su ujedno i stabilne po položaju, i neosjetljive na skaliranje. Postoje razni postupci koji nude ovo rješenje, među njima Laplace Gaussiana (engl. Laplacian of Gaussian - LoG) (Lindeberg, 1993), zatim suboktavna razlika Gaussiana (engl. sub-octave Difference of Gaussian) (Lowe, 2003), i drugi...

5.4.3. Neovisnost značajki o rotaciji

Kod analize dvodimenzijskih slika je često važno da se odabrane značajke mogu uspoređivati i po orijentaciji, a ne samo po položaju. Jedan način je da se odabiru samo značajke koje su osno-simetrične u odnosu na vektor normale u središtu značajke. Na taj način je spriječeno da se značajke ne registriraju ispravno u slučaju da je druga slika zarotirana u odnosu na prvu (Schmid i Mohr, 1997). No takve značajke ne nose puno informacije o svom obliku pa često se registriraju i značajke koje nisu podudarne. Što je opet drugi ekstrem.

Bolje rješenje je procijeniti dominantnu orijentaciju značajke (u dvodimenzijskom prostoru slike), i na taj način imati mogućnost registriranja i značajki koje nisu osno-simetrične oko vektora normale.

Iako je ovaj problem vrlo izražen u postupcima registracije dvodimenzijskih slika, u slučaju trodimenzijskih uzoraka je to manje izraženo. Naime, osim informacije o zakrivljenosti u dvodimenzijskom prostoru parametrizirane površine objekta, u trodimenzijskom slučaju svaka značajka sadrži i informaciju o vektoru normale na površinu objekta. U ovom radu će se koristiti osno-simetrične značajke, ali je proširenje na orijentirane značajke, analogno spomenutim postupcima, moguće.

Postupak	Izvor	Godina
Winston and Lerman	Winston et al.	1972
Difference of Gaussian (DoG)	Marr et al.	1979
Kitchen and Rosenfeld	Kitchen et al.	1980
Fortsner and Gulch	Förstner i Gülch	1987
The Harris detector	Harris i Stephens	1988
The LoG detector	Blostein i Ahuja	1989
Tomasi and Kanade	Tomasi i Kanade	1991
Kohlmann	Kohlmann	1996
Slika spina	Johnson	1997
Rohr	Rohr	1999
Ando	Ando	2000
Loy i Zelinski	Loy i Zelinsky	2002
Multi-scale oriented patches (MOPS)	Brown et al.	2004
Goshtasby	Goshtasby	2005
Scale invariant feature transform (SIFT)	Lindeberg	2012

Tablica 5.1: Deterministički postupci odabira značajki dvodimenzijskih slika

Postupak	Izvor	Godina
Simulated annealing (SA)	Kirkpatrick et al.	1983
Quantum annealing		1989
Graduated optimization	Blake i Zisserman	1987
Ant colony optimization (ACO)	Colorni et al.	1991
Particle swarm optimization	Kennedy i Eberhart	1995
Stochastic gradient descent	Bottou	1998
Genetic algorithms	Falkenauer	1998
The cross-entropy method (CE)	Boer et al.	2002
Stochastic optimization	Spall	2003
Parallel tempering	Earl i Deem	2005
Reactive search optimization	Battiti et al.	2008
Harmony search	Geem	2009
Intelligent Water Drops (IWD)	Shah-Hosseini	2009

Tablica 5.2: Stohastički postupci odabira značajki dvodimenzijskih slika

5.4.4. Pregled dvodimenzijskih postupaka

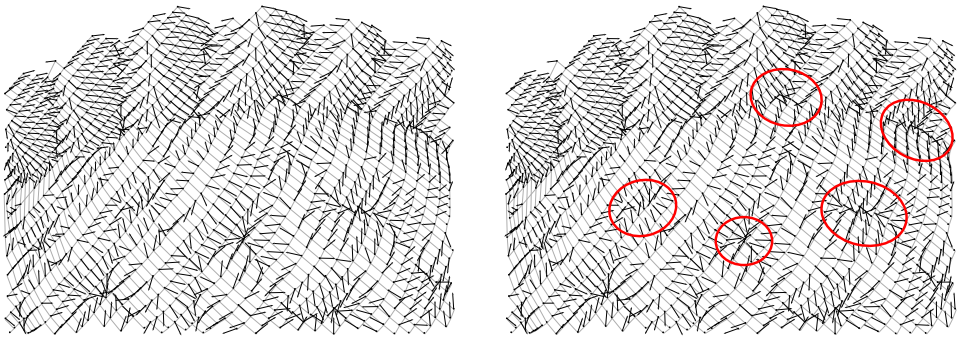
Postupci odabira značajki u dvije dimenzije su vrlo opširno istraženi tijekom posljednjih nekoliko desetljeća. Neki od najpoznatijih su navedeni u tablicama 5.1 i 5.2 poredani po godini objave.

Ugrubo ih možemo podijeliti na determinističke, koji analiziraju skupove uzoraka u cijelosti i daju kao rezultat sve značajke koje su prepoznate odabranim postupkom. Te na stohastičke, kojima se lokalne značajke nalaze u okolini unaprijed (ili slučajno) odabranih točaka. Stohastički postupci ne daju nužno sve značajke koje je moguće prepoznati u skupu uzoraka, ali su vrlo prikladne za analiziranje velikih skupova gdje bi deterministički pristup bio nepraktičan.

5.5. Odabir značajki u trodimenzijskom prostoru

U idealnom bi slučaju, po uzoru na gore navedeni osnovni algoritam odabira značajki u dvodimenzijskim slikama, bilo poželjno i u trodimenzijskom slučaju analizirati auto-korelaciju na površini objekta. U postupku navedenom u nastavku će to biti izostavljeno, a traženje značajki će se zasnivati isključivo na lokalnim minimumima i maksimumima. Ali uz pravilnu orijentaciju značajki, i projekciju okoline pomoću funkcije 5.9, postupak je otvoren za poboljšanja.

Iz slike 5.2 na prvi pogled intuitivno očekujemo da su točke na mjestima gdje vektori kovergiraju, upravo značajke, te da one sadrže dovoljno informacije za uspješnu



(a) **(b)**

Slika 5.2: Odabir značajki: (a) Vektorsko polje gradijenta $\boldsymbol{u}_p(\boldsymbol{x}_i)$, (b) intuitivno očekivane značajke.

registraciju. Ovaj intuitivni zaključak moguće je formalizirati u obliku slijedećeg algoritma:

1. Iz skupa uzoraka izdvojiti one točke $\boldsymbol{x}_i$ koje zadovoljavaju da je, skalarni produkt radij vektora $\boldsymbol{r}_{ij} = \boldsymbol{x}_j - \boldsymbol{x}_i$ prema svakoj od susjednih točaka poligonalne mreže, i pripadajućeg vektora gradijenta $\boldsymbol{u}_p(\boldsymbol{x}_j)$, manji od nule. To jest, da vektori gradijenta konvergiraju prema točki $\boldsymbol{x}_i$.

   Kako bi $\boldsymbol{u}_p(\boldsymbol{x})$ bio simetričan s obzirom na susjedne točke, uzima se aritmetička sredina $\boldsymbol{u}_p(\boldsymbol{x}_i)$ i $\boldsymbol{u}_p(\boldsymbol{x}_j)$. U protivnom su česte pojave značajki koje su pronađene u više susjednih točaka, a ne samo u jednoj. Tako dobivamo prvi uvjet za odabir značajki,

$$(\boldsymbol{x}_j - \boldsymbol{x}_i)^T \cdot (\boldsymbol{u}_p(\boldsymbol{x}_i) + \boldsymbol{u}_p(\boldsymbol{x}_j)) < 0. \tag{5.14}$$

2. Proširiti prsten susjednih točaka po poligonalnoj mreži oko svake značajke pronađene u prethodnom koraku. I prebrojati u kojem omjeru te točke zadovoljavaju

$$\boldsymbol{r}_i j^T \cdot \boldsymbol{u}_p(\boldsymbol{x}_j) < 0. \tag{5.15}$$

3. Ukoliko je postotak točaka iz prethodnog koraka koje zadovoljavaju tvrdnju, veći od definiranog praga (npr. 95%), ponoviti korak 2.

4. Stopiti značajke koje dijele značajan broj točaka, obuhvaćenih proširivanjem u prethodna dva koraka, u jednu.

5. Položaj značajke se određuje kao aritmetička sredina svih točaka obuhvaćenih

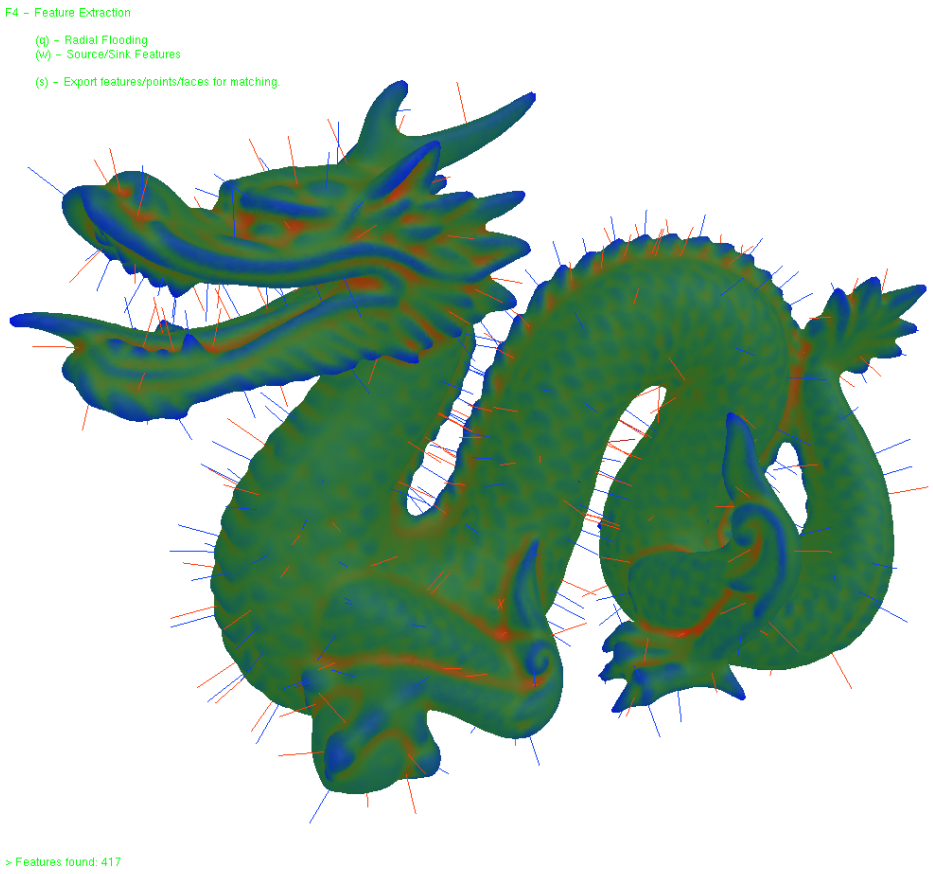
proširivanjem iz drugog i trećeg koraka. Uz položaj se određuje i vektor normale na isti način.

Broj proširivanja prstena oko značajke je također informacija koja je određuje. Taj ćemo broj dalje nazivati polumjerom značajke. Pri tome važno uzeti u obzir da taj broj ovisi o gustoći uzorkovanja te se ne može izravno koristiti za registraciju značajki, ali se omjeri tih polumjera mogu.

Kao što se određuju mjesta konvergencije, analogno se određuju i mjesta divergencije.

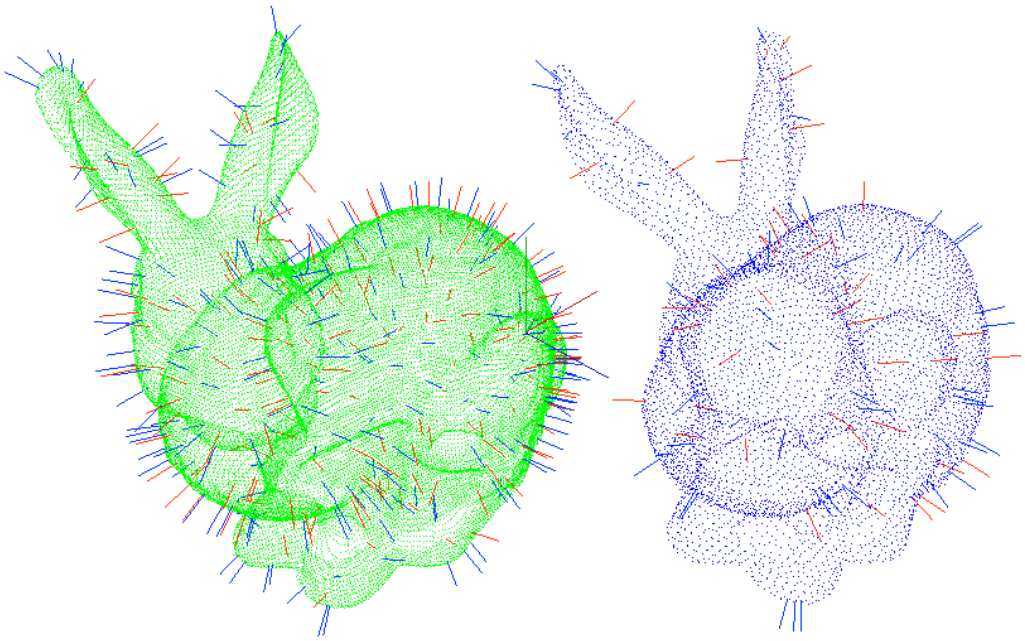
Prag kojim određujemo koliko će se prsten oko značajke širiti, također određuje i koliko se šuma tolerira u skupu uzoraka. Niži prag označava i veću toleranciju na šum.

Značajke, koje pokrivaju veće područje širenja, su pouzdanije s obzirom na šum ili različitu gustoću uzorkovanja. Ukoliko je broj nađenih značajki prevelik, i potrebno je odbaciti dio značajki, bolje je sačuvati one značajke koje imaju veći polumjer.



Slika 5.3: Rezultat odabira značajki.

6. Određivanje podudarnih značajki



Slika 6.1: Skupovi uzoraka različite gustoće uzorkovanja s pripadajućim značajkama, spremni za postupak registracije.

6.1. Opis problema i ciljevi

Nakon što su iz ulaznih skupova odabrane značajke ili značajne točke, njih je potrebno upariti kako bi se pomoću parova uparenih točaka mogla izračunati tražena transformacija.

Problem pronalaska parova značajki među skupovima uzoraka nije toliko geometrijske prirode kao što je to bilo u slučaju određivanja značajki, koliko spada u domenu računalne inteligencije i analize podataka (engl. data mining). Cilj je između dvaju skupova značajki, na osnovi informacije koju one nose, pronaći parove međusobno podudarnih značajki. Pri tome je vrlo važno ne oslanjati se na činjenice koje su promjenjive s obzirom na odabrani tip funkcije preslikavanja.

Budući da smo se ograničili na transformaciju sličnosti, kutovi među vektorima su očuvani. To jest, kutovi u jednom skupu odgovaraju kutovima u drugom. Isto se odnosi i na skalarne produkte jediničnih vektora, to jest kosinuse kutova.

Zakrivljenost se ne mijenja rotacijom, niti translacijom, ali se mijenja skaliranjem. Polumjer zakrivljenosti je veličina koja ovisi o jediničnoj dužini u referentnom sustavu. Također, budući da se aproksimacija zakrivljenosti računa s obzirom na k susjednih točaka, ona ovisi i o gustoći uzorkovanja na tom mjestu. Zakrivljenost se, dakle, ne može koristiti u svrhu registracije u svom apsolutnom iznosu. No omjer zakrivljenosti između dvije značajke nije promjenjiv s obzirom na transformaciju skaliranja. A pošto smo pretpostavili da je gustoća uzorkovanja jedne serije uzoraka približno konstantna, omjer je zakrivljenosti stabilan i u tom slučaju, te ga je ispravno koristiti u svrhu registracije.

Slično kao zakrivljenost, tako i polumjeri značajki u svom apsolutnom iznosu ovise o skaliranju. Ali je, po analogiji sa zakrivljenošću, ispravno koristiti njihove omjere.

Svojstava, koja na ovaj način ne ovise o zadanoj transformaciji, može biti mnogo. Navedimo samo kao primjer boju u točki uzorkovanja, ili projekcijsku sliku okoline značajke na tangencijalnu ravninu. Iz ove se poslijednje može izvesti još velik broj svojstava ukoliko tu ravninu orijentiramo, prema glavnoj osi zakrivljenosti na primjer. I tako dalje...

Odabrana se svojstva značajki, koja će se koristiti u postupku registracije, u numeričkom obliku mogu složiti u uređenu n-torku, to jest vektor. Ukoliko su ta svojstva pravilno odabrana, ona ne ovise o traženoj nepoznatoj transformaciji. Vektore tih svojstava nazivamo deskriptorima (engl. descriptor) (Szeliski, 2010), a prostor kojega razapinju svojstva deskriptora nazivamo prostorom deskriptora. Budući da je referentni sustav deskriptora isti za oba skupa uzoraka, njih je moguće izravno uspoređivati i jednostavno registrirati parove. Na primjer, po euklidskoj udaljenosti.

6.2. Deskriptori

Značenje deskriptora je vrlo intuitivno jer ljudski um određuje takve deskriptore posve prirodno. Na primjer, promatrajući lice čovjeka, jednom kada je ljudski um određenom segmentu lica dodijelio pojam „lijevo oko“, tada je taj pojam odlično definirani deskriptor koji više ne ovisi o prostornim informacijama. Ukoliko bi se na drugom licu našao segment koji se također naziva “lijevo oko“, implicitno bi bilo jasno da ta dva segmenta na dva različita lica predstavljaju istu značajku, to jest da koreliraju. Ljudskom umu je na raspolaganju ogroman kontekst prilikom analize značajki i povezivanja svojstava u općenitije pojmove, zbog kojega problem opisivanja segmenata površine, ili značajki, izgleda jednostavan. Bez ispravnog konteksta je ponekad i čovjeku teško ispravno opisati značajke. Na primjer, jedan segment travnjaka je teško razlikovati od drugog, iako dva različita segmenta nikada nisu ista. Zato je teško uočiti i kada to je jedan te isti segment u dva različita prikaza.

Kontekst definiran u sklopu problema je vrlo bitan za određivanje deskriptora. Budući da su na početku ovog rada postavljene vrlo općenite pretpostavke, izbor je svojstava, nepromjenivih s obzirom na funkciju preslikavanja, relativno ograničen. U nekom specifičnijem slučaju, bi na rapolaganju bio veći izbor.

Svojstva značajki odabranih postupkom iz prethodnog poglavlja su: položaj u prostoru, vektor normale, zakrivljenost, polumjer značajke, i binarno svojstvo koje označava konvergira li gradijent oko značajke, ili divergira.

Od svih ovih svojstava, jedino je oznaka konvergencije nepromjenjiva s obzirom na transformaciju sličnosti koju želimo pronaći. Kako samo jedno binarno svojstvo nije dovoljno za razlučivanje velikog broja značajki, potrebno je dublje analizirati međusobne odnose više značajki.

Par značajki u tom smislu ima slijedeća svojstva: međusobnu udaljenost, kutove između vektora normali i radij vektora razlike položaja dviju značajki, par binarnih oznaka konvergencije, te omjere skalarnih vrijednosti zakrivljenosti i polumjera značajki.

Više od dvije značajke već mogu tvoriti i uzorke (engl. pattern) koji bi mogli dati više prostorno neovisne informacije, no tada broj permutacija raste značajno pa se postavlja pitanje isplativosti. U slučaju para značajki, taj je broj

$$N_D = N_F(N_F - 1) \tag{6.1}$$

gdje je N_F broj značajki pronađenih u skupu uzoraka, a N_D broj deskriptora.

Od navedenih svojstava parova značajki, koristiti ćemo slijedeće:

1. Kosinus kuta između vektora normali

$$d_1 = \boldsymbol{n}_{F1}^T \cdot \boldsymbol{n}_{F2}; \tag{6.2}$$

2. Kosinus kuta što ga zatvara vektor normale prve značajke, i radij-vektor od prve značajke prema drugoj

$$d_2 = \frac{\boldsymbol{n}_{F1}^T \cdot (\boldsymbol{x}_{F2} - \boldsymbol{x}_{F1})}{\|\boldsymbol{x}_{F2} - \boldsymbol{x}_{F1}\|}; \tag{6.3}$$

3. Kosinus kuta što ga zatvara vektor normale druge značajke, i radij-vektor od druge značajke prema prvoj

$$d_3 = \frac{\boldsymbol{n}_{F2}^T \cdot (\boldsymbol{x}_{F1} - \boldsymbol{x}_{F2})}{\|\boldsymbol{x}_{F1} - \boldsymbol{x}_{F2}\|}; \tag{6.4}$$

4. Omjer zakrivljenosti u točkama značajki, ali transformiran na interval $[-1, 1]$ na slijedeći način

$$d_4 = \begin{cases} c_{F1}/c_{F2} & \text{ako je } c_{F1}/c_{F2} \leqslant 1 \\ -c_{F2}/c_{F1} & \text{ako je } c_{F1}/c_{F2} > 1 \end{cases}; \tag{6.5}$$

5. Omjer polumjera značajki, također transformiran na pomenuti način

$$d_5 = \begin{cases} r_{F1}/r_{F2} & \text{ako je } r_{F1}/r_{F2} \leqslant 1 \\ -r_{F2}/r_{F1} & \text{ako je } r_{F1}/r_{F2} > 1 \end{cases}. \tag{6.6}$$

Ova svojstva čine vektor deskriptora

$$\boldsymbol{d} = (d_1, d_2, d_3, d_4, d_5), \tag{6.7}$$

gdje su elementi vektora $d_i \in [-1, 1]$. Prostor deskriptora u ovom slučaju je petdimenzijski prostor omeđen granicama $[-1, 1]$ u svakoj od dimenzija.

6.3. Određivanje podudarnih deskriptora

Uz dobro odabrane deskriptore, traženje podudarnih je trivijalno. Dovoljno je naći najbliže po euklidskoj udaljenosti u prostoru deskriptora.

No u praksi ne postoje postoje univerzalni deskriptori pomoću kojih je moguće diferencirati svaki mogući slučaj. U analogiji s gore spomenutim primjerom, protuprimjer bi bila lopta koja ima naslikane oči (preciznije, lijevo oko) svuda po površini. U tom se slučaju ne može jednoznačno registrirati deskriptore čak ni kada su oni ispravno opisali svaki segment kao „lijevo oko“. To jest, svaki se od tih deskriptora nalazi na istom mjestu u prostoru deskriptora, iako ne označavaju podudarne segmente.

Za navedeni problem višeznačnosti deskriptora nije moguće naći općenito rješenje. Postoje bolja ili lošija rješenja u ovisnosti o granicama primjene svakog pojedinog slučaja. Odabir pravog u velikoj mjeri ovisi o empirijskim spoznajama, i tema je za sebe. U ovom će se radu, stoga koristiti jednostavna euklidska udaljenost. A potencijalno podudarnim deskriptorom će se uzeti jedan koji je najbliži.

Jednom kada su podudarni deskriptori uspješno registrirani, time su registrirane i značajke kojima su oni opisani. Budući da su kosinusi kutova u vektoru deskriptora poredani, znači da su i značajke deskriptora uređeni par. Kako su deskriptori podudarni, onda su podudarne i značajke, prva značajka prvog deskriptora s prvom značajkom drugoga, i druga značajka prvog deskriptora s drugom značajkom drugog deskriptora.

Ta dva para podudarnih značajki su dovoljna za izračun matrice funkcije preslikavanja, jer svaka značajka sadrži poziciju i vektor normale, što čini četiri podudarne točke. Važno je primijetiti da jedinična duljina vektora nije očuvana transformacijom, pa se vektori normalepodudaraju po orijentaciji, ali ne nužno i po duljini. No taj problem se lako riješava skaliranjem normale na udaljenost između značajki, koja mora biti podudarna. Na taj način dobivamo skup točaka $\boldsymbol{X}$ jednog deskriptora,

$$\boldsymbol{X} = \left[(\boldsymbol{x}_{F1} + D \cdot \boldsymbol{n}_{F1}) \mid \boldsymbol{x}_{F1} \mid \boldsymbol{x}_{F2} \mid (\boldsymbol{x}_{F2} + D \cdot \boldsymbol{n}_{F2}) \right], \tag{6.8}$$

gdje je

$$D = \|\boldsymbol{x}_{F2} - \boldsymbol{x}_{F1}\| \tag{6.9}$$

udaljenost između značajki, a kojega preslikavamo na skup točaka drugog deskriptora $\boldsymbol{Y}$, definiranog na isti način.

6.4. Ocjena podudarnosti

Uparivanjem deskriptora najbližim susjedom, dobivamo veliki broj parova deskriptora koji vrlo vjerovatno nisu svi ispravno registrirani. Jasno je da deskriptori, koji su međusobno bliži u prostoru deskriptora, imaju veću vjerovatnos da su ispravno registrirani

od onih koji su dalje. To je jedan od mogućih kriterija selekcije parova deskriptora, i biti će korišten u implementaciji u sklopu ovog rada za početnu selekciju parova.

Bolji način ocjenjivanja podudarnosti parova je izračunati transformaciju, i zatim analizirati rezultat dobiven primjenom te transformacije. Bilo da se radi o skupovima značajki, ili o originalnim skupovima uzoraka. Takvu ocjenu je moguće dobiti na razne načine. A najčešće se zasniva na kvadratnoj pogrešci, gdje manja ocjena po iznosu označava bolju podudarnost. Taj pristup je i ovdje korišten.

Dobiveni par deskriptora daje jedno od mogućih rješenja tražene transformacije, pa se ova ocjena zapravo odnosi na to rješenje. Ocjena zato treba biti takva da najvjerovatnije rješenje bude i najbolje ocijenjeno.

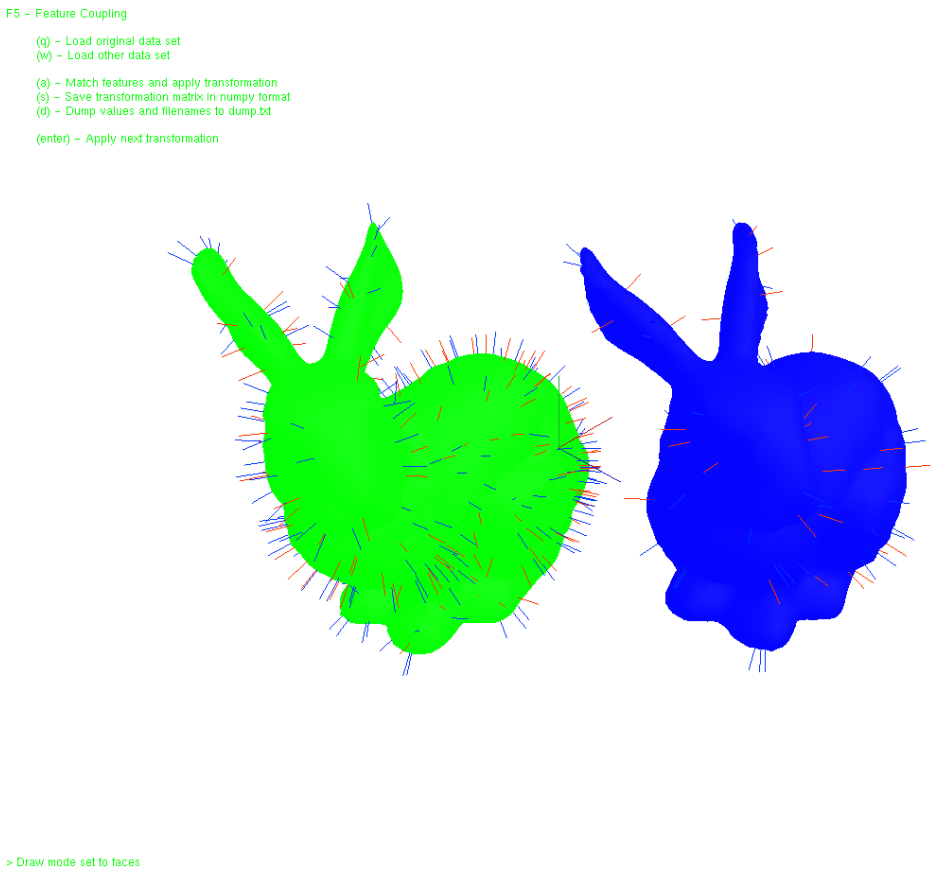
6.5. Grupiranje

Jedan od načina izdvajanja boljih parova deskriptora, to jest boljih rješenja funkcije preslikavanja, je grupiranje. Pod pretpostavkom da će ispravno registrirani parovi svi davati kao rezultat istu ili sličnu transformaciju, a neispravno registrirani svaki svoju neovisnu transformaciju, slijedi da grupa vrlo sličnih transformacija ima veću vjerojatnost biti ispravna od onih koje su izolirane.

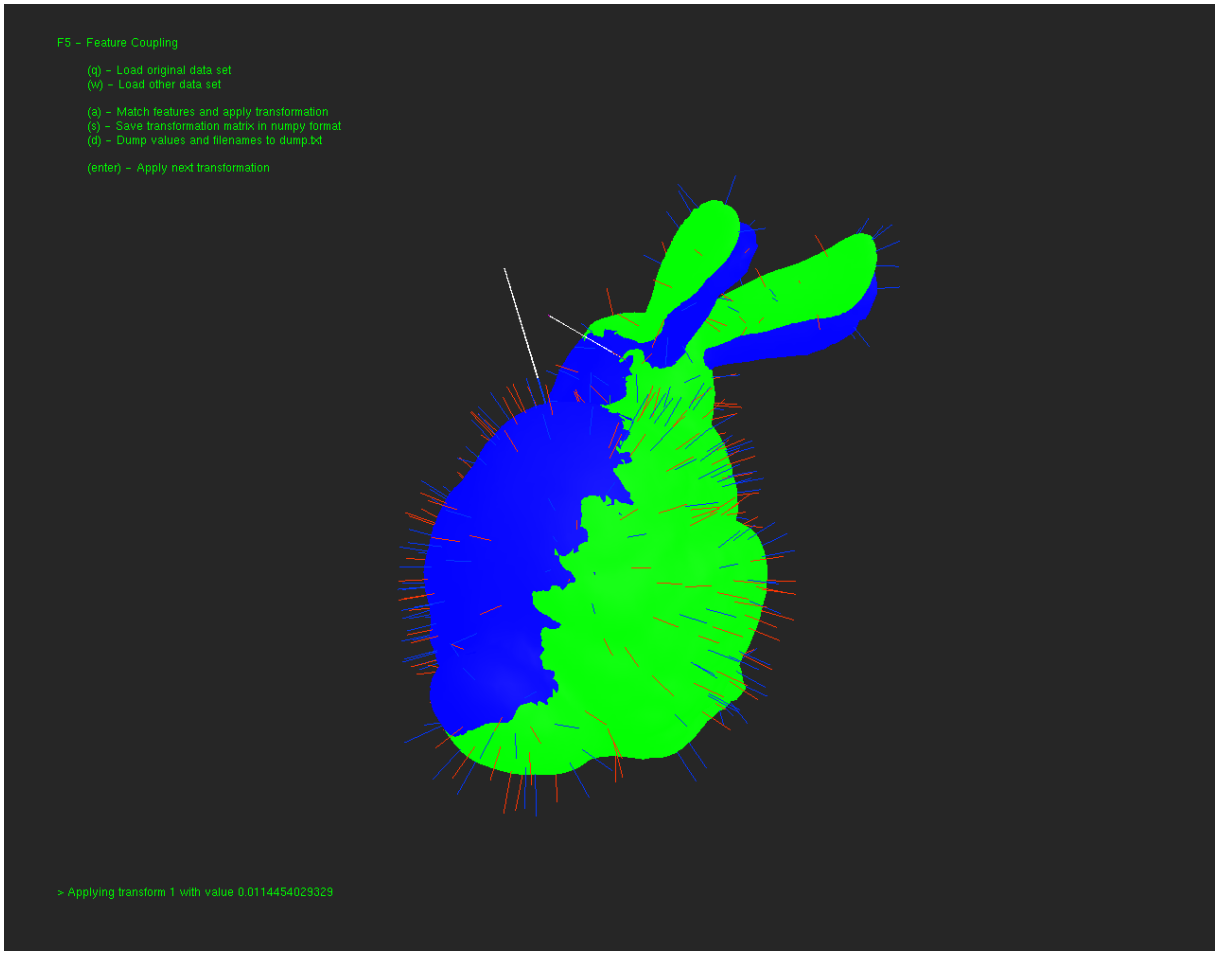
U tu svrhu se mogu koristiti metode računalne inteligencije i analize podataka. Konkretno, transformacije se grupiraju metodom srednjih pomaka (engl. mean shift clustering) (Comaniciu i Meer, 2002). Ideja ove metode je da s unaprijed određenog broja početnih pozicija iterativno pomiče središta grupa prema većoj gustoći uzoraka. Jednom kada se središta ustabile na lokalnim maksimumima gustoće, tada se širenjem iz središta grupe određuju svi uzorci koji pripadaju grupi.

Primjenom ove metode grupiranja dobivamo grupe uparenih deskriptora, to jest grupe uparenih značajki. Vrijednost rješenja dobivenog minimiziranjem kvadratne pogreške na cijelom skupu uparenih značajki, određuje se analogno kao što se određuje i vrijednost jednog para deskriptora.

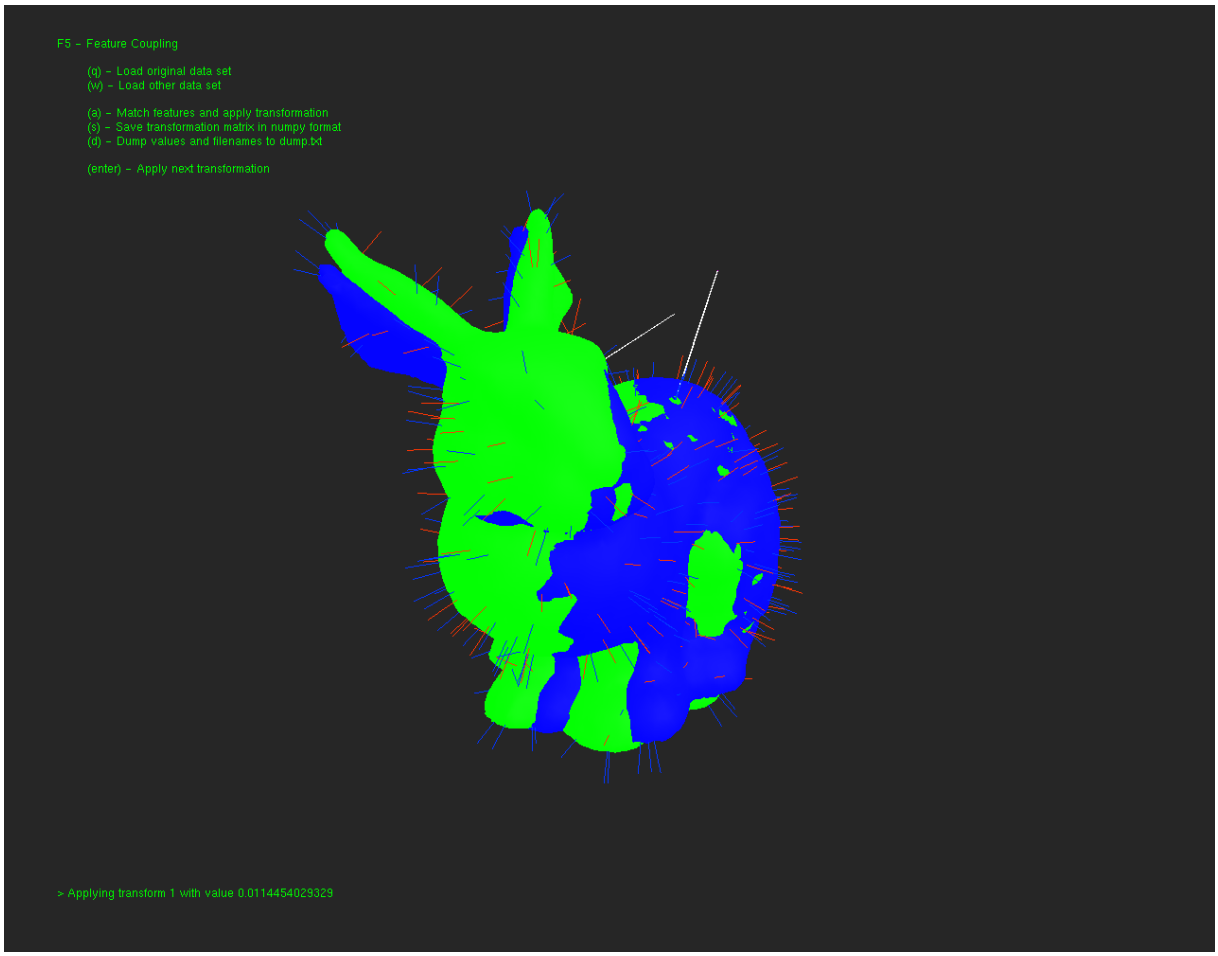
Time je postupak registracije završen, a kvaliteta nađene funkcije preslikavanja ovisi o cijelom postupku od pripreme ulaznih podataka, do ocjene dobivenih transformacija.



Slika 6.2: Poligonalne mreže s pripadajućim značajkama.



Slika 6.3: Uparene poligonalne mreže s pripadajućim značajkama, pogled straga. (Registrirane značajke su označene bijelim linijama.)



Slika 6.4: Uparene poligonalne mreže s pripadajućim značajkama, pogled sprijeda. (Registrirane značajke su označene bijelim linijama.)

7. Izračun transformacije

7.1. Postupci

Uz zadani skup uparenih značajki $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}$ i linearnu parametriziranu transformaciju oblika

$$\boldsymbol{x'} = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{p}), \tag{7.1}$$

tražimo najbolju procjenu parametara transformacije $\boldsymbol{p}$ koja preslikava sve značajke polaznog skupa $\boldsymbol{x}_i$ u odgovarajuće značajke ciljanog skupa $\boldsymbol{y}_i$. Uobičajeni pristup problemu je minimiziranje zbroja kvadrata pogreške,

$$E_{LS} = \sum_i \|\boldsymbol{r}_i\|^2 = \sum_i \|\boldsymbol{f}(\boldsymbol{x}; \boldsymbol{p}) - \boldsymbol{y}_i\|^2, \tag{7.2}$$

gdje je pogreška

$$\boldsymbol{r}_i = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{p}) - \boldsymbol{y}_i \tag{7.3}$$

udaljenost između položaja ciljane značajke $\boldsymbol{y}_i$ i položaja predviđenog transformacijom s trenutnim parametrima $\boldsymbol{x'} = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{p})$.

7.1.1. Linearna metoda najmanjih kvadrata

Za transformacije poput translacije, transformacije sličnosti, i afine transformacije, koje imaju linearan odnos između pomaka učinjenog transformacijom $\Delta\boldsymbol{x} = \boldsymbol{x'} - \boldsymbol{x}$ i parametara transformacije $\boldsymbol{p}$, vrijedi

$$\Delta\boldsymbol{x} = \boldsymbol{x'} - \boldsymbol{x} = \boldsymbol{J}(\boldsymbol{x})\boldsymbol{p}, \tag{7.4}$$

gdje je $\boldsymbol{J} = \partial\boldsymbol{f}/\partial\boldsymbol{p}$ Jakobijeva matrica transformacije $\boldsymbol{f}$ s obzirom na parametre $\boldsymbol{p}$. Jakobijeve matrice za neke od transformacija dane su u tablici 7.1. U ovom se

Transformacija	Parametri	Jakobijeva matrica
Translacija u $\mathbb{R}^2$	(t_x, t_y)	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Euklidska u $\mathbb{R}^2$	(t_x, t_y, θ)	$\begin{bmatrix} 1 & 0 & -s_\theta - c_\theta \\ 0 & 1 & c_\theta - s_\theta \end{bmatrix}$
Sličnost u $\mathbb{R}^2$	(t_x, t_y, α, μ)	$\begin{bmatrix} 1 & 0 & x & -y \\ 0 & 1 & y & x \end{bmatrix}$
Afina u $\mathbb{R}^2$	$(t_x, t_y, a_{11}, \ldots, a_{22})$	$\begin{bmatrix} 1 & 0 & x & y & 0 & 0 \\ 0 & 1 & 0 & 0x & y & \end{bmatrix}$
Projekcija u $\mathbb{R}^2$	$(h_{11}, \ldots, h_{32})$	$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -x'x & -x'y \\ 0 & 0 & 0 & x & y & 1 & -y'x & -y'y \end{bmatrix}$
Translacija u $\mathbb{R}^3$	(t_x, t_y, t_z)	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
Sličnost u $\mathbb{R}^3$	$(t_x, t_y, t_z, \alpha_1, \alpha_2, \alpha_3, \mu)$	$\begin{bmatrix} 1 & 0 & 0 & 0 & -z & y & x \\ 0 & 1 & 0 & z & 0 & -x & y \\ 0 & 0 & 1 & -y & x & 0 & z \end{bmatrix}$

Tablica 7.1: Jakobijeve matrice nekih transformacija

linearnom slučaju zbroj kvadrata pogreške može pisati kao

$$E_{LLS} = \sum_i \|\boldsymbol{J}(\boldsymbol{x}_i)\boldsymbol{p} - (\boldsymbol{y}_i - \boldsymbol{x}_i)\|^2 \tag{7.5}$$

$$= \boldsymbol{p}^T \left[\sum_i \boldsymbol{J}^T\boldsymbol{J}\right] \boldsymbol{p} - 2\boldsymbol{p}^T \left[\sum_i \boldsymbol{J}^T \cdot (\boldsymbol{y}_i - \boldsymbol{x}_i)\right] + \sum_i \|(\boldsymbol{y}_i - \boldsymbol{x}_i)\|^2 \tag{7.6}$$

$$= \boldsymbol{p}^T\boldsymbol{A}\boldsymbol{p} - 2\boldsymbol{p}^T\boldsymbol{b} + c. \tag{7.7}$$

Minimum se u tom slučaju može naći rješavanjem jednadžbe (to jest sustava jednadžbi)

$$\boldsymbol{A}\boldsymbol{p} = \boldsymbol{b}, \tag{7.8}$$

gdje je

$$\boldsymbol{A} = \sum_i \boldsymbol{J}^T(\boldsymbol{x}_i)\boldsymbol{J}(\boldsymbol{x}_i) \tag{7.9}$$

Hesseova matrica, a

$$\boldsymbol{b} = \sum_i \boldsymbol{J}^T(\boldsymbol{x}_i)(\boldsymbol{y}_i - \boldsymbol{x}_i). \tag{7.10}$$

Ovu metodu nazivamo linearnom metodom najmanjih kvadrata (engl. linear least squares), a dobiveno rješenje je optimalno za slučaj da su značajke ispravno uparene. Kako bi se dobila veća robusnost na krivo registrirane značajke, uvodi se dodatni težinski faktor kojim je moguće označiti koji su parovi značajki bolje kvalitete, a koji lošije.

7.1.2. Iterativni postupci

Metoda najmanjih kvadrata za linearni slučaj je najjednostavnija metoda za procjenu parametara transformacije. Međutim, ukoliko ne postoji linearna veza između pomaka uzrokovanog transformacijom i njenih parametara, rješenje se mora tražiti iterativno. To se naziva nelinearnim problemom najmanjih kvadrata (engl. non-linear least squares), ili nelinearnom regresijom (engl. non-linear regression).

Na primjeru dvodimenzijske euklidske transformacije parametrizirane translacijom (t_x, t_y) i kutom rotacije θ, Jakobijeva matrica ovisi o trenutnom iznosu kuta θ. Te se zbog toga ne može izravno doći do rješenja kao u linearnom slučaju. Kako bi se minimizirao nelinearni problem najmanjih kvadrata, potrebno je iterativno računati $\Delta\boldsymbol{p}$ u odnosu na trenutno procijenjeni vektor parametara $\boldsymbol{p}$, minimizirajući

$$\begin{aligned}
E_{NLS}(\Delta\boldsymbol{p}) &= \sum_i \|\boldsymbol{f}(\boldsymbol{x}; \boldsymbol{p} - \Delta\boldsymbol{p}) - \boldsymbol{y}_i\|^2 && (7.11)\\
&\approx \sum_i \|\boldsymbol{J}(\boldsymbol{x}; \boldsymbol{p})\Delta\boldsymbol{p} - \boldsymbol{r}_i\|^2 && (7.12)\\
&= \Delta\boldsymbol{p}^T \left[\sum_i \boldsymbol{J}^T\boldsymbol{J}\right] \Delta\boldsymbol{p} - 2\Delta\boldsymbol{p}^T \left[\sum_i \boldsymbol{J}^T\boldsymbol{r}_i\right] + \sum_i \|\boldsymbol{r}_i\|^2 && (7.13)\\
&= \Delta\boldsymbol{p}^T\boldsymbol{A}\Delta\boldsymbol{p} - 2\Delta\boldsymbol{p}^T\boldsymbol{b} + c, && (7.14)
\end{aligned}$$

gdje je Hesseova matrica $\boldsymbol{A}$ ista kao u jednadžbi 7.9, a vektor $\boldsymbol{b}$ dan kao

$$\boldsymbol{b} = \sum_i \boldsymbol{J}^T(\boldsymbol{x}_i)\boldsymbol{r}_i \tag{7.15}$$

je suma vektora pogreške pomnožena Jakobijevom matricom kao težinskim faktorom. To za poslijedicu ima da su u svakom koraku parametri korigirani u smjeru

vektora pogreške, i to skalirano s Jakobijanom.

Kada su $\boldsymbol{A}$ i $\boldsymbol{b}$ izračunati, $\Delta\boldsymbol{p}$ dobivamo iz jednadžbe

$$(\boldsymbol{A} + \lambda diag(\boldsymbol{A}))\Delta\boldsymbol{p} = \boldsymbol{b}, \tag{7.16}$$

te pomoću njega korigiramo vektor parametara $\boldsymbol{p} \leftarrow \boldsymbol{p} + \Delta\boldsymbol{p}$ za slijedeću iteraciju postupka. Uveden je i dodatni parametar prigušenja λ, koji je u nekim slučajevima potreban kako bi postupak učinio korak prema minimumu funkcije kvadratne pogreške. U većini se slučajeva međutim, on može postaviti na 0, a da postupak ipak uspješno konvergira.

Ove metode najmanjih kvadrata su uobičajene i često se koriste kada prostorne pogreške uparenih značajki predstavljaju šum normalne (Gaussove) raspodjele. No one nisu robusne na pogrešno uparene značajke, kao ni na grube prostorne pogreške. U tim je slučajevima bolje koristiti M-estimatore (Huber 1981; Hampel, Ronchetti, Rousseeuw et al. 1986; Black and Rangarajan 1996; Stewart 1999) koji implementiraju robusnu skalarnu funkciju pogreške umjesto kvadratne i time eliminiraju utjecaj značajki čija je pogreška izrazito velika.

M-estimatori smanjuju osjetljivost na grube pogreške i loše uparene značajke, ali ne rješavaju problem početne procjene vektora parametara. Ukoliko početna procjena značajno odstupa od traženog rješenja, svi će parovi značajki izgledati kao pogrešni, pa postupak neće konvergirati. U tom slučaju je bolje prvo potražiti procjenu dominantne transformacije među parovima značajki.

Dva često korištena pristupa tom problemu su RANSAC, to jest konsenzus slučajnih uzoraka (engl. RANdom SAmple Consensus) (Fischler and Bolles 1981), ili LMS, to jest najmanji medijan kvadrata (engl. least median of squares) (Rouseeuw 1984).

8. Rezultati

8.1. Programska implementacija

Implementacija opisanog postupka registracije trodimenzijskih uzorkovanih podataka, podržana je na dvije 32-bitne platforme:

- Linux (ispitano na Ubuntu 12.04 LTS), i
- Windows XP.

Priprema ulaznih podataka, kao i operacije na velikim skupovima originalnih uzoraka, i poligonalnoj mreži, implementirani su u programskom jeziku C++ u obliku dinamičke biblioteke funkcija. Osim standardnih biblioteka, korišteni su

- CGAL 4.0.1 (Computational Geometry Algorithms Library) (http://www.cgal.org/) Za rekonstrukciju vektora normale, rekonstrukciju poligonalne mreže, te k-susjedstvo.
- RPly library (Diego Nehab, IMPA) (http://www.impa.br/ diego/software/rply) Čitanje .ply datoteka s uzorkovanim podacima.
- OpenGL, GLUT, pthread, i druge...

Korisničko sučelje, analiza značajki, deskriptora, registracija, i izračun transformacija, implementirani su u Python programskom jeziku. Verzija Python interpretera je 2.6. Alatom PyInstaller verzije 2.0 (http://www.pyinstaller.org/), dobivena je krajnja izvršna datoteka. Ostale korištene Python biblioteke su:

- scikit-learn 0.12 (http://scikit-learn.org)
- NumPy (http://numpy.scipy.org/)
- PyGTK (http://pygtk.org/)
- PyOpenGL, GLU, GLUT (http://pyopengl.sourceforge.net/)

8.1.1. Instalacija

Budući da ovaj program ne postavlja vlastite varijable okoline, niti mu je potrebna posebna integracija sa sustavom, dovoljno je raspakirati arhivu u zaseban folder (direktorij), i pokrenuti izvršnu datoteku „point_registration“.

Jedina konfiguracijska datoteka je „.go_fullscreen“. Ukoliko ona postoji u radnom direktoriju, program će se pokrenuti preko cijelog ekrana. U protivnom će se pokrenuti u prozoru.

8.1.2. Opis sučelja

Pokretanjem programa („point_registration“ ili „point_registration.exe“ na Windows operacijskom sustavu), dobije se početni ekran na kojemu su:

- Upute za korištenje kratica (u gornjem lijevom kutu),
- Statusna linija (u donjem lijevom kutu), i
- Kursor u tri dimenzije (u sredini).

Funkcijskim tipkama (F1, . . . , F5) korisnik odabire mod rada.
Brojevima 1, . . . , 5 se mijenja način prikaza podataka.
Tipkama ’y’, ’x’, i ’c’ postavlja se osnovna os navigacije.
Navigacija po prostoru se obavlja mišem, i to tako da

- povlačenje miša uz pritisnutu lijevu tipku miša, rotira pogled oko kursora;
- povlačenje miša uz pritisnutu desnu tipku pomiče kursor kroz trodimenzijski prostor i to po dvije trenutno sporedne osi;
- pomicanje kursora uzduž glavne osi, obavlja se istovremenim pritiskom na desnu tipku miša, razmaknicu, i povlačenjem miša naprijed ili nazad;
- povlačenjem miša uz pritisnutu srednju tipku, približava ili udaljava pogled u odnosu na kursor;

8.1.3. Određivanje značajki

Tipičan slijed radnji za određivanje značajki je slijedeći:

1. Pokrenuti program,
2. Pritisnuti tipku ’l’ i odabrati model (.ply),
3. Tipkom ’F2’ prijeći u način rada za izračunavanje zakrivljenosti,

4. Pritisnuti ’q’ za procjenu normala (provjeriti u pogledu ’2’),
5. Pritisnuti ’a’ za izračun zakrivljenosti (provjeriti na ’1’ ili ’3’),
6. Pritisnuti ’g’ za određivanje vektora tangencijalne plohe (provjeriti na ’4’ i ’5’),
7. Opcionalno: Ukoliko ulazni skup nema vlastitu triangulaciju,
   (a) Tipkom ’F3’ prijeći u način rada za triangulaciju poligona,
   (b) Pritisnuti ’q’ za Amenta postupak
   (c) Provjeriti rezultat tipkom ’3’
8. Tipkom ’F4’ prijeći u način rada za traženje značajki,
9. Pritisnuti ’w’ za traženje lokalnih minimuma i maksimuma funkcije zakrivljenosti,
10. Snimiti rezultat tipkom ’s’ i odabirom izlazne datoteke,
11. Zatvoriti program.

8.1.4. Registracija skupova

Tipičan slijed radnji za registraciju skupova:

1. Pokrenuti program,
2. Tipkom ’F5’ prijeći u način rada za registraciju skupova,
3. Pritisnuti tipku ’q’ i odabrati ciljani skup podataka sa značajkama (.features),
4. Pritisnuti tipku ’w’ i odabrati radni skup podataka sa značajkama (.features),
5. Pritisnuti tipku ’a’ za pokretanje postupka uparivanja značajki,
6. Tipkom ’Return’ iterirati kroz deset najbolje rangiranih transformacija,
7. Tipkama ’s’ i ’d’ spremiti rezultate.
8. Zatvoriti program.

U načinu rada za registraciju skupova, moguće je pomicati radni skup podataka uobičajenom navigacijom mišem, uz pritisnutu tipku ’TAB’.

8.2. Ispitni skupovi

Korišteni su ispitni skupovi dostupni na stranicama sveučilišta Stanford u Kaliforniji (http://graphics.stanford.edu/data/3Dscanrep/). Korišteni modeli su:

Stanford Bunny

Izvor: Stanford University Computer Graphics Laboratory
Skener: Cyberware 3030 MS
Veličina rekonstrukcije: 35 947 točaka, 69 451 trokuta

Dragon

Source: Stanford University Computer Graphics Laboratory
Skener: Cyberware 3030 MS + spacetime analysis
Veličina rekonstrukcije: 566 098 točaka, 1 132 830 trokuta

Oba modela imaju nekoliko rekonstruiranih oblaka točaka s različitom gustoćom uzoraka, i nekoliko sirovih uzorkovanih podataka prije rekonstrukcije.

8.3. Ocjena postupaka

Kvantitativna ocjena uspješnosti postupka registracije može se dobiti prebrojavanjem ispravnih i neispravnih podudarnih parova, koristeći slijedeće definicije (Fawcett, 2006):

- TP: ispravni parovi (engl. true positives), ili broj ispravno uparenih značajki;
- FN: neispravno izostavljeni parovi (engl. false negatives);
- FP: neispravni parovi (engl. false positives);
- TN: ispravno izostavljeni parovi (engl. true negatives);

Primjer gore definiranih brojeva naveden je u tablici 8.1. Iz tih brojeva možemo izračunati učestalosti slijedećim izrazima (Fawcett, 2006),

- učestalost ispravnih parova (engl. true positive rate),

$$TPR = \frac{TP}{TP + FN} = \frac{TP}{P}; \tag{8.1}$$

- učestalost neispravnih parova (engl. false positive rate),

$$FPR = \frac{FP}{FP + TN} = \frac{FP}{N}; \tag{8.2}$$

– ispravnost uparenih značajki (engl. positive predictive value),

$$PPV = \frac{TP}{TP + FP} = \frac{TP}{P'}; \tag{8.3}$$

– preciznost (engl. accuracy),

$$ACC = \frac{TP + TN}{P + N}. \tag{8.4}$$

	Podudarni parovi	Bez para		
Uparene značajke	TP=18	FP=4	P'=2	PPV=0,82
Izostavljene značajke	FN=2	TN=76	N'=78	
	P=20	N=80	Total=100	
	TPR=0,90	FPR=0,05		ACC=0,94

Tablica 8.1: Primjer matrice konfuzije koja pokazuje odnos ispravno i neispravno uparenih značajki i ocjenu preciznosti postupka uzimajući u obzir ispravne parove (TP), neispravno izostavljene parove (FN), neispravne parove (FP), ispravno izostavljene parove (TN)

Ovako definirana ocjena postupaka je moguća samo za ispitne podatke u kojima je unaprijed poznato ispravno rješenje podudarnih parova značajki.

Na realnim ispitnim podacima kao što su „Stanford Bunny“ ili „Dragon“ moguća je samo naknadna ocjena ispravno uparenih značajki, kao što će to biti učinjeno kasnije. Ali točanu statistiku o izostavljenim parovima nije moguće dobiti na ovaj način.

8.4. Rezultati registracije realnih skupova podataka

U tablicama u prilogu A, analizirani su rezultati registracije realnih skupova uzoraka. Datoteke s nastavcima „_zipper“ i „_vrip“ su rekonstrukcije koje dolaze s ispravno trianguliranom mrežom poligona. Skupovima iz ostalih datoteka, poligonalna je mreža rekonstruirana Amenta postupkom.

„Vrijednost“ zapisana u tablicama se odnosi na proračunsku ocjenu podudarnosti transformacije. Manje vrijednosti označavaju bolje transformacije, a veće lošije. transformacije označene od 1 do 10 su poredane po rastućoj ocjeni, to jest po padajućoj podudarnosti, od najbolje prema najlošijoj.

Broj deskriptora označava s koliko je uparenih deskriptora određena transformacija. Brojevi veći od 1 znače da je ta transformacija rezultat grupiranja parova des-

kriptora. Inače, ako je broj deskriptora 1, transformacija je izračunata i ocijenjena na temelju samo jednog para deskriptora.

Dalje slijedi tablica u kojoj je subjektivno ocijenjeno zadovoljava li transformacija navedeni kriterij u dovoljnoj mjeri da bi se smatrala ispravnom. „Solid“ označava da je dobivena transformacija, transformacija sličnosti. To jest da ne deformira objekte. Iako je kao uvjet postavljena transformacija sličnosti sa svojim parametrima, moguće je da ukoliko točke pomoću kojih se transformacija računa nisu ispravno odabrani, linearna metoda najmanjih kvadrata ne može dati izpravan rezultat. U tom slučaju se polje „Solid“ ostavlja prazno.

Polje „Konvergira“ označava subjektivnu procjenu je li transformirani skup dovoljno blizu da bi neki od postupaka iterativnog smanjivanja pogreške konvergirao. Idealno bi bilo svaki upareni skup točaka još proslijediti takvom algoritmu, al ovdje smo se radi jednostavnosti ograničili na subjektivnu ocjenu na temelju vizualne procjene.

Ukupna ocjena svake transformacije je dana kao

$$ukupno = solid \cdot [velicina + rotacija + pozicija + 2 \cdot konvergira]. \tag{8.5}$$

Težinski prosjek je prosječna ocjena usrednjena Gaussovom krivuljom oko najbolje ocijenjene transformacije.

Podudarne značajke su označen na skupovima s bijelim linijama spojenim rozom bojom, a njihov broj je dva puta veći od broja uparenih deskriptora. Na temelju vizualne inspekcije je određen broj onih koji su ispravno uparene. Pa se iz tog omjera računa ispravnost uparenih značajki (engl. positive predictive value), to jest PPV.

Težinski prosjek ispravnosti se računa analogno težinskom prosjeku ocjene.

8.5. Zapažanja

Skupovi na kojima nije bilo potrebno rekonstruirati poligonalnu mrežu Amenta postupkom, bili su uspješnije registrirani od onih na kojima to je.

To se objašnjava postojanjem praznina u rekonstruiranoj mreži koje se tumače kao granica područja uzorkovanja pa se zbog toga na tim skupovima ne nalaze one najvrijednije, odnosno najveće značajke.

Model „Stanford Bunny“ je općenito bolje registriran od modela „Dragon“. To je za očekivati jer model „Dragon“ ima ponavljajuće uzorke na sebi, poput ljuski, i

zavoja na tijelu koji se ponavljaju. To čini registraciju težom od samog početka jer informacija sadržana u obliku objekta nije posve jednoznačna.

9. Zaključak

Problem obrađen u ovom radu je neizostavna stepenica na putu ka potpunoj automatizaciji registracije uzorkovanih skupova točaka. Neki od pristupa rješavanju tog problema su ovdje obrađeni, a drugi su navedeni.

Opisani se postupci zasnivaju na deterministčkom pretraživanju skupa uzoraka za lokalne minimume i maksimume skalarne funkcije zakrivljenosti. Zatim na deskriptoru značajki, koji se sastoji od kuteva između vektora, i omjera skalaranih svojstava značajki. Uparivanju deskriptora na temelju euklidske udaljenosti u prostoru deskriptora. I na kraju, na grupiranju dobivenih transformacija po međusobnoj sličnosti.

Na temelju rezultata dobivenih primjenom postupka na testnim uzorcima, dobivena su zapažanja na temelju kojih se mogu zaključiti ograničenja primjene takvog pristupa, kao i prilike za eventualna poboljšanja.

Skupovi uzoraka različite gustoće uzorkovanja, ali potpuni u smislu da je cijela površina objekta ravnomjerno pokrivena uzorcima, imali su vrlo visoku uspješnost registracije. Uspješnost opada s povećanjem razlike u gustoći uzorkovanja, kao što je i za očekivati, no rezultati zadovoljavaju čak i kada je gustoća jednog skupa nekoliko puta veća od gustoće drugoga.

Postupak je pokazao značajnu osjetljivost na kvalitetu rekonstruirane poligonalne mreže. Skupovi koji su imali loše rekonstruiranu poligonalnu mrežu, što to se posebno odnosi na praznine unutar granica uzorkovanja, a manje na praznine na rubnom području, imali su i iznimno loše rezultate u postupku registracije. To je najvećim dijelom zbog toga što se značajke koje graniče s takvim prazninama odbacuju, pa postupak odabire samo male značajke koje su ujedno i najosjetljivije na pogreške uzorkovanja.

Ograničenja postavljena na svojstva primijenjena u deskriptorima, vrlo su općenita. Sužavanjem primjene postupka na određene uređaje ili situacije, moguće je dobiti daleko više prostorno neovisnih svojstava nego što je korišteno u opisanom postupku. Time bi se dobili kvalitetniji deskriptori i povećala uspješnost uparivanja.

Deskriptori su ovom postupku uparivani sa svojim najbližim susjedom u prostoru svojstava. Ovaj postupak je najjednostavniji za implementaciju i daje relativno dobre

rezultate, ali moguće je i proširenje koje bi vrlo vjerovatno povećalo robusnost registracije parova, bez velikog povećanja složenosti algoritma. Najveći dio vremena u ovom koraku otpada na izgradnju kd-stabla, pa proširenje pretraživanja s jednog susjeda na njih više, ne bi značajnije utjecalo na vrijeme izvođenja.

Brzina izvođenja nije bila detaljno analizirana u ovom radu, pa je razumljivo da su moguća mnoga poboljšanja na svim razinama. Za početak, na razini implementacije, korištenjem nižih programskih jezika za vemenski kritične dijelove. Radi lakše analize postupaka su neki koraci računanja međusobno razdvojeni i izvršavani slijedno, iako bi u realnom slučaju bili implementirani paralelno. Na primjer, jednom izgrađeno kd-stablo nad određenim skupom bi se trebalo koristiti tijekom cijelog postupka, bez da se ponovo gradi u slijedećem koraku. Zatim, neke je korake moguće paralelizirati unutar istog procesa. A većinu je moguće i paralelizirati u svrhu višeprocesorskog računanja. To nije učinjeno u ovoj implementaciji, već se posebna radna dretva koristila samo radi razdvajanja korisničkog sučelja od vremenski dugotrajnih operacija računanja.

Algoritmi su mahom zasnovani na linearnoj algebri i aritmetici s pomičnim zarezom, što se vrlo dobro može implementirati koristeći i grafički procesor koji je daleko brži. Posebno za korake postupka koji se daju paralelizirati.

Osim ubrzanja u smislu implementacije, moguća su i ubrzanja korištenjem naprednijih algoritama. Na primjer, ukoliko je poznato da su uzorkovani podaci uvijek uzorkovani s lica ljudi, moguće je naslutiti ili empirijski odrediti optimalni broj značajki toga skupa. U tom je slučaje bolje primijeniti neku stohastičku metodu traženja značajki, kojom ne pretražujemo cijeli skup uzoraka, pa time i postupak kraće traje.

Za specifične se primjene mogu primijeniti specifični postupci koji su robusniji i daju daleko bolje rezultate od nekog općenitog, kao što je postupak obrađen u ovom radu. Postupak registracije skupova uzoraka je samo u određenoj mjeri egzaktna znanost, a veliki dio se zasniva na empirijskom znanju i vještini prilagodbe konkretnom zadatku.

Uz navedena poboljšanja, užu definiciju područja primjene, te uz naknadnu doradu rezultata primjenom iterativnog postupka za precizno poravnanje, navedeni postupak sasvim sigurno može naći svoju primjenu u znanosti i industriji.

LITERATURA

Nina Amneta, Sunghee Choi, i Ravi Kolluri. *Proceedings: sixth ACM Symposium on Solid Modeling and Applications ; Sheraton Inn, Ann Arbor, Michigan, June 6-8*, poglavlje The power crust, stranice 249–260. ACM Press, 2001. ISBN 9781581133660. URL http://books.google.hr/books?id=gJZRAAAAMAAJ.

Shigeru Ando. Image field categorization and edge/corner detection from gradient covariance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(2):179–190, Veljača 2000. ISSN 0162-8828. doi: 10.1109/34.825756. URL http://dx.doi.org/10.1109/34.825756.

Roberto Battiti, Mauro Brunato, i Franco Mascia. *Reactive Search and Intelligent Optimization*. Springer Publishing Company, Incorporated, 1st izdanju, 2008. ISBN 038709623X, 9780387096230.

Andrew Blake i Andrew Zisserman. Visual reconstruction, 1987.

D. Blostein i N. Ahuja. *Computer Vision, Graphics, and Image Processing*, svezak 45, poglavlje A multiscale region detector, stranice 22–41. Academic Press, 1989. URL http://books.google.hr/books?id=EnszAAAAIAAJ.

P. T. De Boer, D.P. Kroese, S. Mannor, i R.Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134, 2002.

Léon Bottou. Online algorithms and stochastic approximations. U David Saad, urednik, *Online Learning and Neural Networks*. Cambridge University Press, Cambridge, UK, 1998. URL http://leon.bottou.org/papers/bottou-98x.

Matthew Brown, R Szeliski, i S Winder. Multi-scale oriented patches. Technical Report MSR-TR-2004-133, December 2004. URL http://opus.bath.ac.uk/26125/.

Alberto Colorni, Marco Dorigo, i Vittorio Maniezzo. Distributed Optimization by Ant Colonies. U *European Conference on Artificial Life*, stranice 134–142, 1991.

Dorin Comaniciu i Peter Meer. Mean shift: A robust approach toward feature space analysis. U *In PAMI*, stranice 603–619, 2002.

David J. Earl i Michael W. Deem. Parallel tempering: Theory, applications, and new perspectives. *Phys. Chem. Chem. Phys.*, 7:3910–3916, 2005. doi: 10.1039/B509983H. URL http://dx.doi.org/10.1039/B509983H.

Emanuel Falkenauer. *Genetic algorithms and grouping problems*. Wiley, 1998. ISBN 9780471971504. URL http://books.google.hr/books?id=FacrAAAAYAAJ.

Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861 – 874, 2006. ISSN 0167-8655. doi: 10.1016/j.patrec.2005.10.010. URL http://www.sciencedirect.com/science/article/pii/S016786550500303X. <ce:title>ROC Analysis in Pattern Recognition</ce:title>.

W. Förstner i E. Gülch. *Fast Processing of Photogrammetric Data: Proceedings Intercommission Conference, Interlaken, June 2-4, 1987*, poglavlje A fast operator for detection and precise location of distinct points, corners and centers of circular features, stranice 281–305. IGP, ETH Zürich., 1987. URL http://books.google.hr/books?id=9o1UewAACAAJ.

Zong Woo Geem. *Music-Inspired Harmony Search Algorithm: Theory and Applications*. Springer Publishing Company, Incorporated, 1st izdanju, 2009. ISBN 364200184X, 9783642001840.

A. Ardeshir Goshtasby. *2-D and 3-D Image Registration: for Medical, Remote Sensing, and Industrial Applications*. Wiley-Interscience, 2005. ISBN 0471649546.

Chris Harris i Mike Stephens. A combined corner and edge detector. U *In Proc. of Fourth Alvey Vision Conference*, stranice 147–151, 1988.

A. Johnson. *Spin-images: a representation for 3-d surface matching*. Doktorska disertacija, Carnegie Mellon University, USA, 1997.

J. Kennedy i R. Eberhart. Particle swarm optimization. U *Neural Networks, 1995. Proceedings., IEEE International Conference on*, svezak 4, stranice 1942–1948 vol.4.

IEEE, Studeni 1995. ISBN 0-7803-2768-3. doi: 10.1109/ICNN.1995.488968. URL http://dx.doi.org/10.1109/ICNN.1995.488968.

S. Kirkpatrick, C. D. Gelatt, i M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. doi: 10.1126/science.220.4598.671. URL http://www.sciencemag.org/content/220/4598/671.abstract.

L. Kitchen, A. Rosenfeld, MARYLAND UNIV COLLEGE PARK COMPUTER VISION LAB., i College Park. Computer Science Center University of Maryland. *Gray-Level Corner Detection*. Report (University of Maryland, College Park. Computer Vision Laboratory). Defense Technical Information Center, 1980. URL http://books.google.hr/books?id=oklYNwAACAAJ.

Klaus Kohlmann. Corner detection in natural images based on the 2-d hilbert transform. *Signal Process.*, 48(3):225–234, Veljača 1996. ISSN 0165-1684. doi: 10.1016/0165-1684(95)00138-7. URL http://dx.doi.org/10.1016/0165-1684(95)00138-7.

T. Lindeberg. Scale invariant feature transform. *Scholarpedia*, 7(5):10491, 2012.

Tony Lindeberg. Detecting salient blob-like image structures with a scale-space primal sketch: A method for focus-of-attention. *INT. J. COMP. VISION*, 11(3), 1993.

David G. Lowe. Distinctive image features from scale-invariant keypoints, 2003.

Gareth Loy i Alexander Zelinsky. A fast radial symmetry transform for detecting points of interest. U *Proceedings of the 7th European Conference on Computer Vision-Part I*, ECCV '02, stranice 358–368, London, UK, UK, 2002. Springer-Verlag. ISBN 3-540-43745-2. URL http://dl.acm.org/citation.cfm?id=645315.757460.

D. Marr, E. Hildreth, i MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB. *Theory of Edge Detection*. AI memo. Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 1979. URL http://books.google.hr/books?id=EvFOHAAACAAJ.

Zoltan Csaba Marton, Radu Bogdan Rusu, i Michael Beetz. On Fast Surface Reconstruction Methods for Large and Noisy Datasets. U *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, May 12-17 2009.

K. Rohr. *Extraction of 3d Anatomical Point Landmarks Based on Invariance Principles*. 1999. URL http://books.google.hr/books?id=ZJKtPgAACAAJ.

Cordelia Schmid i Roger Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:530–535, 1997.

Hamed Shah-Hosseini. The intelligent water drops algorithm, a nature-inspired swarm-based optimization algorithm. *Int. J. Bio-Inspired Comput.*, 1(1/2):71–79, Siječanj 2009. ISSN 1758-0366. doi: 10.1504/IJBIC.2009.022775. URL http://dx.doi.org/10.1504/IJBIC.2009.022775.

Jianbo Shi i Carlo Tomasi. Good features to track. stranice 593–600, 1994.

James C. Spall. Introduction to Stochastic Search and Optimization. 2003.

R. Szeliski. *Computer Vision: Algorithms and Applications*. Texts in Computer Science. Springer, 2010. ISBN 9781848829343. URL http://books.google.hr/books?id=bXzAlkODwa8C.

Carlo Tomasi i Takeo Kanade. shape and motion from image streams: a factorization method. Technical report, International Journal of Computer Vision, 1991.

P.H. Winston, J.B. Lerman, i Massachusetts Institute of Technology. Artificial Intelligence Laboratory. Robotics Section. *Circular Scan*. Vision Flash. MIT Artificial Intelligence Laboratory, 1972. URL http://books.google.hr/books?id=cdrtSAAACAAJ.

10. Prilog A - Izvorni rezultati registracije ispitnih skupova

Datoteke	Vrijednost	# Deskriptora									
bun_zipper											
bun_zipper_res2											
			Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
1	0.012948	46	1	1	1	1	1	5	92	89	0.97
2	0.021198	1	1	1	1	1	1	5	2	2	1
3	0.024779	1	1	1	1	1	1	5	2	2	1
4	0.025828	1	1	1	1	1	1	5	2	2	1
5	0.027552	1	1	1	1	1	1	5	2	2	1
6	0.027752	1	1	1	1	1	1	5	2	2	1
7	0.028504	1	1	1	1	1	1	5	2	2	1
8	0.028686	1	1	1	1	1	1	5	2	2	1
9	0.032249	1	1	1	1	1	1	5	2	2	1
10	0.033773	1	1	1	1	1	1	5	2	2	1
							Težinski prosjek:	5.00		Težinski prosjek:	99.10%

Datoteke	Vrijednost	# Deskriptora									
bun_zipper											
bun_zipper_res3											
			Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
1	0.051205	1	1	1	1	1		3	2	2	1
2	0.079464	1	1	1	1	1		3	2	1	0.5
3	0.084009	1	1		1	1		2	2	2	1
4	0.106325	1	1	1				1	2	0	0
5	0.113441	1	1					0	2	0	0
6	0.122206	6						0	12	0	0
7	0.129917	1	1	1		1		2	2	0	0
8	0.131129	1	1	1				1	2	0	0
9	0.131204	1	1	1				1	2	0	0
10	0.135845	1	1	1				1	2	0	0
							Težinski prosjek:	2.10		Težinski prosjek:	57.75%

Datoteke	Vrijednost	# Deskriptora									
bun_zipper											
bun_zipper_res4											
			Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
1	0.067171	1	1	1	1	1		3	2	2	1
2	0.126332	1	1					0	2	0	0
3	0.126891	1						0	2	0	0
4	0.127461	1	1					0	2	0	0
5	0.128662	1	1					0	2	0	0
6	0.137630	1	1					0	2	0	0
7	0.158397	1	1					0	2	0	0
8	0.162487	1						0	2	0	0
9	0.164183	1						0	2	0	0
10	0.166858	1						0	2	0	0
							Težinski prosjek:	0.82		Težinski prosjek:	27.49%

Datoteke	Vrijednost	# Deskriptora									
bun_zipper_res2											
bun_zipper_res3											
			Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
1	0.043531	1	1	1	1	1	1	5	2	2	1
2	0.056799	1	1	1		1		2	2	1	0.5
3	0.071631	1	1	1	1	1		3	2	2	1
4	0.071697	3	1	1	1	1		3	6	4	0.67
5	0.086592	1	1	1	1	1	1	5	2	2	1
6	0.109461	4	1		1			1	8	2	0.25
7	0.111219	1	1			1		1	2	2	1
8	0.112805	1	1			1		1	2	1	0.5
9	0.113165	1	1	1		1		2	2	0	0
10	0.113299	1	1		1			1	2	1	0.5
							Težinski prosjek:	3.30		Težinski prosjek:	79.07%

Datoteke	Vrijednost	# Deskriptora									
bun_zipper_res2											
bun_zipper_res4											
			Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
1	0.075324	1	1	1	1	1	1	5	2	2	1
2	0.097933	1	1	1	1	1	1	5	2	2	1
3	0.118581	1	1	1	1	1		3	2	1	0.5
4	0.122898	1	1					0	2	0	0
5	0.127453	1	1	1		1		2	2	0	0
6	0.130251	1		1		1		0	2	0	0
7	0.148283	1	1	1		1		2	2	0	0
8	0.148758	1	1	1	1	1		3	2	1	0.5
9	0.161145	1	1	1				1	2	0	0
10	0.164429	1						0	2	0	0
							Težinski prosjek:	3.37		Težinski prosjek:	60.96%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
bun_zipper_res3											
bun_zipper_res4											
1	0.054514	1	1	1	1	1	1	5	2	2	1
2	0.061958	1	1			1		1	2	1	0.5
3	0.073875	1	1	1	1			2	2	0	0
4	0.102116	1	1	1	1	1	1	5	2	1	0.5
5	0.107284	1	1	1	1	1	1	5	2	1	0.5
6	0.119746	1	1	1	1		1	4	2	1	0.5
7	0.127813	1						0	2	0	0
8	0.186986	1	1		1			1	2	1	0.5
9	0.188350	1	1	1				1	2	0	0
10	0.190439	1	1					0	2	1	0.5
							Težinski prosjek:	3.25		Težinski prosjek:	53.03%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
bun000											
bun045											
1	0.028252	82	1	1	1	1	1	5	164	120	0.73
2	0.031861	117	1					0	234	10	0.04
3	0.032142	44	1		1			1	88	30	0.34
4	0.033005	73	1		1			1	146	50	0.34
5	0.036436	1	1					0	2	0	0
6	0.037198	30	1					0	60	0	0
7	0.037572	32	1			1		1	64	20	0.31
8	0.039002	18	1			1		1	36	15	0.42
9	0.039786	24	1					0	48	0	0
10	0.041215	27	1					0	54	0	0
							Težinski prosjek:	1.73		Težinski prosjek:	33.13%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
bun045											
bun090											
1	0.022308	94	1					0	188	0	0
2	0.024825	36	1		1			1	72	0	0
3	0.026020	35	1					0	70	0	0
4	0.027329	42	1					0	84	0	0
5	0.027740	52	1			1		1	104	0	0
6	0.038518	1	1					0	2	0	0
7	0.038547	17	1					0	34	5	0.15
8	0.040436	1						0	2	0	0
9	0.041085	12	1					0	24	0	0
10	0.041562	1						0	2	0	0
							Težinski prosjek:	0.32		Težinski prosjek:	0.37%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
bun_zipper											
bun000											
1	0.010111	72	1	1	1	1	1	5	144	125	0.87
2	0.021264	1	1	1	1	1	1	5	2	2	1
3	0.021606	1	1	1	1	1	1	5	2	2	1
4	0.022750	1	1	1	1	1	1	5	2	2	1
5	0.023680	1	1	1	1	1	1	5	2	2	1
6	0.024738	1	1	1	1	1	1	5	2	2	1
7	0.024749	136	1					0	272	0	0
8	0.026074	1	1	1	1	1	1	5	2	2	1
9	0.026533	47	1					0	94	0	0
10	0.026759	1	1	1	1	1	1	5	2	2	1
							Težinski prosjek:	4.85		Težinski prosjek:	93.37%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
bun_zipper											
bun045											
1	0.028448	78	1	1	1	1	1	5	156	110	0.71
2	0.032943	112	1			1		1	224	0	0
3	0.036876	7	1					0	14	0	0
4	0.037514	32						0	64	0	0
5	0.042183	19						0	38	0	0
6	0.042661	15						0	30	0	0
7	0.044292	29	1		1			1	58	10	0.17
8	0.045570	19	1			1		1	38	20	0.53
9	0.045622	24						0	48	0	0
10	0.046404	13						0	26	0	0
							Težinski prosjek:	1.65		Težinski prosjek:	20.44%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
bun_zipper											
bun090											
1	0.019960	45	1					0	90	0	0
2	0.023201	24	1					0	48	0	0
3	0.024865	27						0	54	0	0
4	0.025457	1						0	2	0	0
5	0.026977	1						0	2	0	0
6	0.027445	11						0	22	0	0
7	0.028479	1						0	2	0	0
8	0.029013	1						0	2	0	0
9	0.029390	17						0	34	0	0
10	0.030795	31						0	62	0	0
							Težinski prosjek:	0.00		Težinski prosjek:	0.00%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
dragon_vrip_res2											
dragon_vrip_res3											
1	0.031930	12	1	1	1	1	1	5	24	20	0.83
2	0.032907	1	1					0	2	0	0
3	0.033455	11	1	1	1	1	1	5	22	19	0.86
4	0.037105	1	1		1	1		2	2	2	1
5	0.038642	1	1		1	1		2	2	2	1
6	0.040977	1	1	1		1		2	2	2	1
7	0.042305	1	1	1		1		2	2	1	0.5
8	0.042777	1	1		1	1		2	2	1	0.5
9	0.043647	8	1		1	1		2	16	7	0.44
10	0.044067	65	1					0	130	10	0.08
							Težinski prosjek:	2.90		Težinski prosjek:	66.94%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
dragon_vrip_res2											
dragon_vrip_res4											
1	0.042756	99	1					0	198	10	0.05
2	0.056563	1	1	1				1	2	0	0
3	0.058482	1	1	1				1	2	0	0
4	0.063906	1	1					0	2	0	0
5	0.064276	1	1	1				1	2	0	0
6	0.065287	21	1		1			1	42	0	0
7	0.067887	7	1		1	1		2	14	6	0.43
8	0.068178	1	1					0	2	0	0
9	0.068458	17	1		1	1		2	34	2	0.06
10	0.072620	10	1		1	1		2	20	4	0.2
							Težinski prosjek:	0.62		Težinski prosjek:	2.53%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
dragon_vrip_res3											
dragon_vrip_res4											
1	0.018489	1	1	1	1	1	1	5	2	2	1
2	0.041477	1	1					0	2	0	0
3	0.043876	1	1	1	1	1	1	5	2	1	0.5
4	0.045448	72	1					0	144	0	0
5	0.047024	1	1					0	2	0	0
6	0.048287	1	1					0	2	0	0
7	0.048504	1	1	1		1		2	2	1	0.5
8	0.049293	1	1					0	2	0	0
9	0.050831	1	1		1	1		2	2	0	0
10	0.052980	9	1		1	1		2	18	2	0.11
							Težinski prosjek:	2.36		Težinski prosjek:	37.98%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
dragonStandRight_0											
dragonStandRight_24											
1	0.025171	111	1	1	1	1	1	5	222	190	0.86
2	0.030823	1	1	1	1	1	1	5	2	2	1
3	0.032248	112	1		1	1		2	224	90	0.40
4	0.034646	1	1					0	2	0	0
5	0.034756	110	1					0	220	1	0.00
6	0.038011	1	1					0	2	0	0
7	0.039899	1	1		1	1		2	2	1	0.5
8	0.040292	1	1			1		1	2	1	0.5
9	0.041455	30	1					0	60	1	0.02
10	0.041477	1	1	1		1		2	2	1	0.5
							Težinski prosjek:	2.99		Težinski prosjek:	56.58%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
dragonStandRight_24											
dragonStandRight_48											
1	0.030404	1	1					0	2	0	0
2	0.037503	1	1					0	2	0	0
3	0.040818	77	1	1	1	1	1	5	154	120	0.78
4	0.047348	1	1					0	2	0	0
5	0.047783	1	1					0	2	0	0
6	0.048640	1	1					0	2	0	0
7	0.048853	1	1					0	2	0	0
8	0.050289	1	1					0	2	0	0
9	0.051162	1	1					0	2	0	0
10	0.053036	1	1					0	2	0	0
							Težinski prosjek:	0.92		Težinski prosjek:	14.36%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
dragonStandRight_48											
dragonStandRight_72											
1	0.029376	44	1					0	88	0	0
2	0.032035	30						0	60	0	0
3	0.032226	11	1					0	22	0	0
4	0.034100	18	1					0	36	0	0
5	0.037111	1	1					0	2	0	0
6	0.039881	1	1					0	2	0	0
7	0.043096	1	1					0	2	0	0
8	0.043105	1	1					0	2	0	0
9	0.045002	1	1					0	2	0	0
10	0.045189	1	1					0	2	0	0
							Težinski prosjek:	0.00		Težinski prosjek:	0.00%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
dragon_vrip_res2											
dragonStandRight_0											
1	0.040512	48	1		1			1	96	4	0.042
2	0.041254	36	1					0	72	0	0
3	0.041328	81	1					0	162	0	0
4	0.046357	83	1					0	166	0	0
5	0.047842	1	1					0	2	0	0
6	0.050908	40	1					0	80	0	0
7	0.053109	1	1	1	1	1		3	2	1	0.5
8	0.053171	56	1					0	112	0	0
9	0.053966	46	1					0	92	0	0
10	0.056160	27	1					0	54	0	0
							Težinski prosjek:	0.35		Težinski prosjek:	2.39%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
dragon_vrip_res3											
dragonStandRight_0											
1	0.029211	1	1					0	2	0	0
2	0.033158	191	1		1			1	382	1	0.003
3	0.038861	1	1					0	2	0	0
4	0.042765	54	1					0	108	0	0
5	0.047211	7	1		1	1		2	14	8	0.571
6	0.048934	19	1					0	38	0	0
7	0.049353	12	1					0	24	0	0
8	0.050752	1	1	1				1	2	0	0
9	0.051012	9	1		1			1	18	2	0.111
10	0.053992	36	1					0	72	0	0
							Težinski prosjek:	0.42		Težinski prosjek:	4.85%

Datoteke	Vrijednost	# Deskriptora	Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
dragon_vrip_res3											
dragonStandRight_24											
1	0.036550	241	1					0	482	0	0
2	0.057395	57	1					0	114	0	0
3	0.059559	1	1	1				1	2	0	0
4	0.062769	16	1					0	32	0	0
5	0.063064	1	1					0	2	0	0
6	0.063675	1	1	1				1	2	0	0
7	0.064658	36	1					0	72	0	0
8	0.065152	1	1	1				1	2	0	0
9	0.067026	1	1	1				1	2	0	0
10	0.067795	1	1					0	2	0	0
							Težinski prosjek:	0.25		Težinski prosjek:	0.00%

Datoteke	Vrijednost	# Deskriptora									
dragon_vrip_res3											
dragonStandRight_48											
			Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
1	0.056618	1	1					0	2	0	0
2	0.059882	56						0	112	0	0
3	0.074794	1	1	1				1	2	0	0
4	0.075245	20						0	40	0	0
5	0.079472	31						0	62	0	0
6	0.081465	1	1	1				1	2	0	0
7	0.083363	1	1	1				1	2	0	0
8	0.083510	16	1	1				1	32	0	0
9	0.085395	9	1					0	18	0	0
10	0.092381	32						0	64	0	0
							Težinski prosjek:	0.27		Težinski prosjek:	0.00%

Datoteke	Vrijednost	# Deskriptora									
dragon_vrip_res3											
dragonStandRight_72											
			Solid	Veličina	Rotacija	Pozicija	Konvergira	Ocjena	# Značajki	Ispravne	PPV
1	0.063304	1	1	1				1	2	0	0
2	0.067538	1	1	1				1	2	0	0
3	0.082859	11						0	22	0	0
4	0.090569	12						0	24	0	0
5	0.091774	1	1	1				1	2	0	0
6	0.096321	1						0	2	0	0
7	0.097730	14						0	28	0	0
8	0.100399	13						0	26	0	0
9	0.108905	1	1					0	2	0	0
10	0.111470	19						0	38	0	0
							Težinski prosjek:	0.59		Težinski prosjek:	0.00%

11. Prilog B - Detalji programske implementacije

11.1. Python

11.1.1. Označavanje rubnih točaka

```
def mark_endpoints(self):
    for p in self.points:
        open_pts = []
        for f in p.faces:
            for fpt in f.points:
                if fpt is not p:
                    if fpt in open_pts:
                        open_pts.remove(fpt)
                    else:
                        open_pts.append(fpt)
        p.endpoint = len(open_pts)>0
```

11.1.2. Odabir značajki

Inicijalni izbor točaka

```
while len(pointlist):
    p = pointlist.pop()
    p_direction = _source_direction(p)
    if p_direction is not None :
        # Expansion
        expansion = _expand(p, p_direction)
        if expansion is not None:
```

```
            sourcepoint, feature_radius, inside =\
             expansion
            featurelist.append(
             SourceSinkFeature(
             sourcepoint,
             feature_radius,
             p_direction,
             inside))
```

Ekspanzija graničnog kruga

```
def _expand(p, p_direction):
    EXPANSION_COEFFICIENT = 0.95
    inside = [p]
    border = p.neighbors[:]
    expansion_ratio = 1.0
    feature_radius = 1.0
    while expansion_ratio>EXPANSION_COEFFICIENT:
        inside.extend(border)
        new_border = []
        for n in border:
            for nn in n.neighbors:
                if nn not in inside:
                    new_border.append(nn)

        border = new_border
        feature_radius += 1.0

        if any([pn.endpoint for pn in border]) or \
         len(border)==0:
            return None

        directions = []
        for pn in border:
            r = pn.point - p.point
            r_dot_u = float((r.T * pn.u)[0,0])
```

```
            directions.append(1 if r_dot_u>0 else -1)

        expansion_ratio = float(
         directions.count(p_direction))/len(directions)

    sourcepoint = p
    sourcepoint.point = \
     sum([ip.point for ip in inside])/len(inside)
    sourcepoint.normal = \
     sum([ip.normal for ip in inside])/len(inside)

    return sourcepoint, feature_radius, inside
```

Određivanje smijera gradijenta značajke

```
def _source_direction(p):
    p_direction = None # None-unknown, 1-source, -1-sink
    for pn in p.neighbors:
        if pn.endpoint:
            return None
        r = pn.point - p.point
        r_dot_u = float((r.T * (pn.u+p.u) )[0,0])
        if r_dot_u>0: # Could be source
            if p_direction is not None and p_direction<0:
                p_direction = None
                break # inconsistent => not a source
            else:
                p_direction = 1
        elif r_dot_u<0: # Could be sink
            if p_direction is not None and p_direction>0:
                p_direction = None
                break # inconsistent => not a sink
            else:
                p_direction = -1
    return p_direction
```

500 najboljih

```
featurelist.sort(key=lambda f: f.feature_radius, reverse=True)
featurelist = featurelist[:500]
```

Stapanje značajki koje se preklapaju

```
MERGE_COEFFICIENT = 0.6
while len(featurelist)>0:
    f = featurelist.pop()
    similar_feats = []
    inside = f.inside
    for ff in featurelist:
        if float(len(inside.intersection(ff.inside)))/ \
         len(ff.inside)>MERGE_COEFFICIENT:
            similar_feats.append(ff)
            inside.update(ff.inside)
    map(featurelist.remove, similar_feats)
    f.sourcepoint.point = \
     sum([ip.point for ip in inside])/len(inside)
    f.sourcepoint.normal = \
     sum([ip.normal for ip in inside])/len(inside)
    self.features.append(f)
```

11.1.3. Vektor deskriptora

```
self.D = linalg.norm(feat2.position - feat1.position)

if feat1.curvature==0.0 or feat2.curvature==0.0:
    self.curv_ratio = 0.0
else:
    self.curv_ratio = feat1.curvature / feat2.curvature
    self.curv_ratio = \
     self.curv_ratio if \
     self.curv_ratio<1.0 else -1.0/self.curv_ratio

self.radius_ratio = \
```

```
feat1.feature_radius / feat2.feature_radius
self.radius_ratio = \
self.radius_ratio if \
self.radius_ratio<1.0 else -1.0/self.radius_ratio

self.vector = \
matrix([[(feat1.normal.T * feat2.normal)[0,0]],
            [(feat1.normal.T * \
             (feat2.position - feat1.position))[0,0] / \
              self.D],
            [(feat2.normal.T * \
             (feat1.position - feat2.position))[0,0] / \
              self.D],
            [self.curv_ratio],
            [self.radius_ratio]
            ])
```

11.1.4. Odredivanje podudarnih deskriptora

```
def matches(self, descriptor_setA, descriptor_setB):
    MAX_SET_SIZE = 10000

    neigh = NearestNeighbors(12, 1.)
    neigh.fit([d.vector.T.tolist()[0]  \
     for d in descriptor_setB[:MAX_SET_SIZE]])

    def _compatible_by_directions(dpair):
        return (dpair[0].feat1.direction!= \
         dpair[0].feat1.direction and
                dpair[1].feat2.direction!= \
                 dpair[1].feat2.direction or
                dpair[0].feat1.direction!= \
                 dpair[1].feat1.direction)

    for d_A in descriptor_setA[:MAX_SET_SIZE]:
        distances, indices = \
```

```
         neigh.kneighbors(d_A.vector.T, 10)
        distances, indices = \
         distances[0].tolist(), indices[0].tolist()

        d, i = distances.pop(0), indices.pop(0)
        d_B = descriptor_setB[i]
        while not _compatible_by_directions((d_A, d_B)) \
         and len(indices)>0:
            d, i = distances.pop(0), indices.pop(0)
            d_B = descriptor_setB[i]
        if len(indices)>0:
            yield DescriptorMatch(d_A, d_B)
```

11.1.5. Ocjena podudarnosti

Ocjena

```
def transformed_points_grade(self, A):
    # Points matched
    cross_point1 = \
     cross((self.descriptor1.feat2.position- \
     self.descriptor1.feat1.position).T,
               self.descriptor1.feat1.normal.T).T
    cross_point1 = cross_point1 / linalg.norm(cross_point1)
    testpts1 = mconcat((self.descriptor1.feat1.position,
                        self.descriptor1.feat1.position + \
                         self.descriptor1.D * \
                         self.descriptor1.feat1.normal,
                        self.descriptor1.feat1.position + \
                         self.descriptor1.D * \
                         cross_point1,
                        self.descriptor1.feat2.position
                        ), 1)
    cross_point2 = \
     cross((self.descriptor2.feat2.position- \
     self.descriptor2.feat1.position).T,
               self.descriptor2.feat1.normal.T).T
```

```
    cross_point2 = cross_point2 / linalg.norm(cross_point2)
    testpts2 = mconcat((self.descriptor2.feat1.position,
                        self.descriptor2.feat1.position + \
                         self.descriptor2.D * \
                         self.descriptor2.feat1.normal,
                        self.descriptor2.feat1.position + \
                         self.descriptor1.D * \
                         cross_point2,
                        self.descriptor2.feat2.position
                        ), 1)
    testpts2_transformed = A * testpts2

    # Test distances preserved
    D_test1 = \
     linalg.norm(testpts2_transformed[:,1] - \
     testpts2_transformed[:,0])
    D_test2 = \
     linalg.norm(testpts2_transformed[:,2] - \
     testpts2_transformed[:,0])
    D_test3 = \
     linalg.norm(testpts2_transformed[:,3] - \
     testpts2_transformed[:,0])
    D_error = (2*D_test2 - D_test1 - D_test3)**2


    return linalg.norm(testpts2_transformed - testpts1)\
      + D_error
```

Set točkaka za ocjenu transformacije

```
def get_point_sets(self):
    # Make normals scale invariant by
    # scaling them to distance between feats.
    p1f1_point = self.descriptor1.feat1.position
    p1f2_point = self.descriptor1.feat2.position
    p1f1_normal = self.descriptor1.D * \
```

```
        self.descriptor1.feat1.normal
    p1f2_normal = self.descriptor1.D * \
        self.descriptor1.feat2.normal

    X = [p1f1_point, (p1f1_point + p1f1_normal),
         p1f2_point, (p1f2_point + p1f2_normal)]

    p2f1_point = self.descriptor2.feat1.position
    p2f2_point = self.descriptor2.feat2.position
    p2f1_normal = self.descriptor2.D * \
        self.descriptor2.feat1.normal
    p2f2_normal = self.descriptor2.D * \
        self.descriptor2.feat2.normal

    Y = [p2f1_point, (p2f1_point + p2f1_normal),
         p2f2_point, (p2f2_point + p2f2_normal)]

    return X, Y
```

11.1.6. Grupiranje

Mean Shift Clustering metoda

```
def clustering(self):
    test_vectors = \
     matrix('1. 0. 0.; 0. 1. 0.; 0. 0. 1.')
    X = array([(dm.A * test_vectors).T.A1 \
     for dm in self.matchlist])

    bandwidth = 0.3

    # Mean Shift (in 1975 by Fukunaga and Hostetler)
    ms = MeanShift(bandwidth=bandwidth,
        bin_seeding=True,
        cluster_all=False)
    ms.fit(X)
    labels = ms.labels_
```

```
    matchgroups = []
    for k in set(labels)-set([-1.]):
        k_indices = filter(lambda i: labels[i]==k,
            range(len(X)))
        if len(k_indices)>0:
            group = [self.matchlist[dmi] for dmi in k_indices]
            matchgroups.append(DescriptorGroup(group))

    return matchgroups
```

Ocjena grupe

```
self.value = \
sum([dm.transformed_points_grade(self.A) \
for dm in dmatch_group])/ \
(len(dmatch_group)*math.log(len(dmatch_group)))
```

11.1.7. Linearna metoda najmanjih kvadrata

```
def lls_transform(self, X, Y):
    def J(point):
        x, y, z = point.T.tolist()[0]
        return matrix([[1.,0.,0.,0.,-z, y, x],
                       [0.,1.,0., z,0.,-x, y],
                       [0.,0.,1.,-y, x,0., z]])

    b = sum([J(xi).T*(yi-xi) for xi,yi in zip(X,Y)])

    A = sum([J(xi).T*J(xi) for xi in X])

    p = inverse(A) * b

    tx, ty, tz, a1, a2, a3, mu = p.T.tolist()[0]

    T = matrix([[1+mu,  a3, -a2, tx],
                [ -a3,1+mu,  a1, ty],
```

```
                [  a2, -a1,1+mu, tz],
                [  0.,  0.,  0., 1.]])

    self.A = TransformMatrix(T)
```

11.2. C++

11.2.1. Normale

```
// Estimates normals direction.
const int nb_neighbors = 18 + 1;
// K-nearest neighbors = 2 rings + origin

pthread_rwlock_wrlock( &this->__point_normal_data_rwlock);
CGAL::pca_estimate_normals(
   this->point_normal_data.begin(),
   this->point_normal_data.end(),
   CGAL::First_of_pair_property_map<PointVectorPair>(),
   CGAL::Second_of_pair_property_map<PointVectorPair>(),
   nb_neighbors);
pthread_rwlock_unlock( &this->__point_normal_data_rwlock);

// Orients normals.
pthread_rwlock_wrlock( &this->__point_normal_data_rwlock);
std::vector<PointVectorPair>::iterator \
unoriented_points_begin =
    CGAL::mst_orient_normals(
     this->point_normal_data.begin(),
     this->point_normal_data.end(),
CGAL::First_of_pair_property_map<PointVectorPair>(),
CGAL::Second_of_pair_property_map<PointVectorPair>(),
nb_neighbors);
pthread_rwlock_unlock( &this->__point_normal_data_rwlock);
```

11.2.2. Zakrivljenost

Aproksimacija stoščem

```
case CURVATURE_CONE_APPROX: {
Vector s0(cnpit->x(), cnpit->y(), cnpit->z());
Vector n0(cnpit->normal);
double cone_height = 0.0;
cnpit->curvature = 0.0;

for(Neighbor_search::iterator it = search.begin();
it != search.end(); ++it){
Vector s1(it->first.x(),
  it->first.y(),
  it->first.z());
Vector ds = s1 - s0;
if (it->second > 0.0) { // distance > 0
ds = ds / sqrt(it->second);
cone_height = cone_height +
  (ds.x()*n0.x() +
   ds.y()*n0.y() +
   ds.z()*n0.z());
}
}

cnpit->curvature =
cone_height / (double)(nb_neighbors-1);
}
break;
```

Srednji skalarni produkt normala

```
case CURVATURE_NORMFIELD_DOT: {
Vector n0(cnpit->normal);
double dot_average = 0.0;
cnpit->curvature = 0.0;
```

```
for(Neighbor_search::iterator it = search.begin();
it != search.end(); ++it){
Vector n1(static_cast<Point>(it->first).normal);
dot_average += n0.x()*n1.x() +
   n0.y()*n1.y() +
   n0.z()*n1.z();
}

cnpit->curvature =
dot_average/(double)(nb_neighbors);
}
break;
```

Transformiranje intervala vrijednosti zakrivljenosti

```
case CURVATURE_CONE_APPROX:
// Cone approximation gives in range [-1.0, 1.0].
// => Transform to valid range.
cnpit->curvature =
0.5 + atan(div_k*cnpit->curvature) / div_B;
break;
case CURVATURE_NORMFIELD_DOT:
// Dot gives values in range [-1.0, 1.0] but
//  usualy very close to +1.0 because of
// very little divergence in k-neighborhood.
// => Using logarithmic mapping the range to [0,1].
cnpit->curvature =
1.0 - (dot_A-log(1+dot_k-cnpit->curvature)) / dot_B;
break;
```

11.2.3. Aproksimacija okoline uzorka ravninom

```
for (vector<Point>::iterator cnpit=this->cnpoints.begin();
cnpit != this->cnpoints.end(); ++cnpit) {
Neighbor_search search(tree, *cnpit, nb_neighbors);
++progress;
```

```
Vector xi = Vector(cnpit->x(),
   cnpit->y(),
   cnpit->z());
double ci = cnpit->curvature;

Vector gi = Vector();

// LLS parameters
Vector b = Vector();
double a = 0.0;

// Summation inner loop.
for(Neighbor_search::iterator it=search.begin();
it != search.end(); ++it){
Vector xj(it->first.x(),
  it->first.y(),
  it->first.z());
double cj = it->first.curvature;

double delta_c = cj - ci;
Vector delta_x = xj - xi;

a += delta_x.squared_length();
b = b + (delta_c * delta_x);
}

// Optimized gradient vector approximation
gi = (1.0/a) * b;


    cnpit->gradient =
     sqrt((double) gi.squared_length());

cnpit->u_p =
gi - (gi.x()*cnpit->normal.x()+
  gi.y()*cnpit->normal.y()+
```

```
  gi.z()*cnpit->normal.z()) * cnpit->normal;

double norm_u =
1.0 / sqrt((double) cnpit->u_p.squared_length());
cnpit->u_p =
Vector( norm_u*cnpit->u_p.x(),
norm_u*cnpit->u_p.y(),
norm_u*cnpit->u_p.z());

cnpit->v_p =
Vector(cnpit->normal.y()*cnpit->u_p.z() -
cnpit->normal.z()*cnpit->u_p.y(),
   cnpit->normal.z()*cnpit->u_p.x() -
    cnpit->normal.x()*cnpit->u_p.z(),
   cnpit->normal.x()*cnpit->u_p.y() -
    cnpit->normal.y()*cnpit->u_p.x());
}
```

11.2.4. poligonalna mreža

Amenta kora

```
case TRIANGULATION_AMENTACRUST: {

DT3::Vertex::initializing = true;
cout << "Inserting " <<
this->synchronizer->point_data.size() <<
" points into Delaunay...[05%]" << endl;
SyncRWLock pointslock=
this->synchronizer->
lock_point_data(SyncRWLock::READ);
this->delaunay.insert(
this->synchronizer->point_data.begin(),
this->synchronizer->point_data.end());

// Extract Voronoi vertices.
vector<DT3::Point> voronoi_centers;
```

```
cout << "Extracting Voronoi vertices...[15%]" << endl;
for (DT3::Finite_cells_iterator
cit=this->delaunay.finite_cells_begin();
cit!=this->delaunay.finite_cells_end();
++cit) {
voronoi_centers.push_back(
this->delaunay.dual(cit));
}

// Insert Voronoi vertices into Delaunay.
DT3::Vertex::initializing = false;
cout << "Inserting " <<
voronoi_centers.size() <<
" Voronoi vertices into Delaunay...[" <<
15+this->synchronizer->point_data.size()*
80/voronoi_centers.size() << "%]" << endl;

this->delaunay.insert(
voronoi_centers.begin(),
voronoi_centers.end());

// Get all crust faces.
// (Those without Voronoi vertices in them)
cout <<
"Geting all crust faces...[95%]" << endl;
for(DT3::Finite_facets_iterator
fit=this->delaunay.finite_facets_begin();
fit!=this->delaunay.finite_facets_end();
++fit){

if ( fit->first->vertex(
this->delaunay.vertex_triple_index(
fit->second, 0 ) )->is_original
 && fit->first->vertex(
  this->delaunay.vertex_triple_index(
  fit->second, 1 ) )->is_original
```

```
 && fit->first->vertex(
  this->delaunay.vertex_triple_index(
  fit->second, 2 ) )->is_original) {

pthread_rwlock_wrlock(
&this->__crust_faces_rw_lock);
this->crust_faces.push_back(*fit);
pthread_rwlock_unlock(
&this->__crust_faces_rw_lock);
}
}

cout << "Done making crust!" << endl;
}
break;
```

Alpha spheres

```
case TRIANGULATION_ALPHASHAPES: {

DT3::Vertex::initializing = true;
cout << "Inserting " <<
this->synchronizer->point_data.size() <<
" points into Delaunay...[05%]" << endl;
SyncRWLock pointslock=
this->synchronizer->
lock_point_data(SyncRWLock::READ);
this->delaunay.insert(
this->synchronizer->point_data.begin(),
this->synchronizer->point_data.end());

if (this->alphashape)
delete this->alphashape;
this->alphashape =
new Alpha_shape_3(this->delaunay);
```

```
// find optimal alpha values
this->alphashape->set_alpha(
this->synchronizer->n18_distance/10.0);
pthread_rwlock_wrlock(
&this->__crust_faces_rw_lock);
cout << "Getting regular facets...[45%]" << endl;
this->alphashape->get_alpha_shape_facets(
back_inserter(this->crust_faces),
Alpha_shape_3::REGULAR);
pthread_rwlock_unlock(
&this->__crust_faces_rw_lock);

cout << "Done making crust!" << endl;
}
break;
```

POSTUPCI PODUDARANJA TRODIMENZIJSKIH UZORKOVANIH PODATAKA

Sažetak

Pregled osnovnih principa i postupaka za robusnu registraciju trodimenzijskih uzorkovanih podataka. Analiza ulaznih podataka, određivanje značajki, konstrukcija deskriptora značajki, registracija podudarnih deskriptora, odnosno značajki i podudarnih dijelova mreže, izračun transformacije potrebne za podudaranje uzorkovanih skupova podataka.

Ključne riječi: 3d, skener, uzorkovanje, značajke, deskriptori, transformacija, podudaranje, robusno, registracija.

METHODS FOR REGISTERING THREEDIMENSIONAL SCANNED DATA

Abstract

Overview of basic principles and methods for robust registration of threedimensional scanned data. Input data analysis, feature extraction, feature description, registering matching descriptors, features and network segments, computation of transforms needed for matching scanned data sets.

Keywords: 3d, scanner, sampling, features, descriptors, transformation, matching, robust, registration.