Obrada informacija: Registracija i postupak iterativne najbliže točke

T. Petković

Sveučilište u Zagrebu

prosinac 2021.

Uvod

Ovisno o uređaju za 3D skeniranje kao rezultat skeniranja možemo dobiti različite vrste izlaznih podataka.

Ako govorimo o **profilometriji** onda je rezultat skeniranja **profil**/površina predmeta kojeg skeniramo.

Uređaji za profilometriju su na primjer 3D skeneri koji koriste strukturirano svjetlo (engl. *structured light 3D scanner*) ili laserski 3D skeneri (engl. *lidar*, složenica od *light* i *radar*).

Površina predmeta snimljena takvim uređajima je najčešće predstavljena kao nestrukutirani skup točaka kojeg zovemo oblak točaka (eng. *point cloud*).

Oblak točaka

Oblak od N točaka sadrži koordinate tih točaka, npr. (x_i, y_i, z_i), $i = 1, 2, \ldots N$, za pravokutni ili Kartezijev koordinatni sustav.

Koordinate točaka su obično izražene u koordinatnom sustavu mjernog uređaja.

Za oblak kažemo da je **nestrukturiran** ako ne postoji unaprijed određen redoslijed točaka.

Neki oblaci točaka mogu imati parcijalni uređaj, npr. za strukturirano svjetlo možemo prostorno poredati točke sukladno njihovom susjedstvu u 2D slikovnoj ravnini. Dostupnost takvih dodatnih informacija (ponekad) olakšava obradu oblaka točaka.

Boja i normala točke

Svakoj točci u oblaku točaka osim njenih koordinata mogu biti pridružene razne značajke površine u toj točci kao što su boja/albedo i površinska normala.

Boju obično predstavljamo kao trojku (r, g, b) gdje je r crvena (eng. *red*), g zelena (eng. *green*), i b plava (eng. *blue*).

Površinsku normalu predstavljamo kao normirani vektor (n_x, n_y, n_z) za kojeg vrijedi $n_x^2 + n_y^2 + n_z^2 = 1$.

Pohrana oblaka točaka

Neki od formata za pohranu oblaka točaka su `PLY`, `PCD`, `E57`, `XYZ`.

`PLY`, od engl. *polygon file format*, je jednostavan format podataka koji omogućava pohranu isključivo nestrukturiranih oblaka točaka u ASCII i u binarnom formatu, vidi `http://paulbourke.net/dataformats/ply/`.

`PCD`, od engl. *point cloud data*, je jednostavan format podataka koji omogućava pohranu oblaka točaka koji dozvoljavaju spremanje 2D strukture odnosno informacije kako su 3D točke posložene u slikovnu ravninu, vidi `https://pcl.readthedocs.io/projects/tutorials/en/latest/pcd_file_format.html`.

`E57` je standardni format za pohranu lidarski podataka, a `XYZ` je obična tekstualna datoteka u kojoj svaki redak sadrži tri koordinate pojedine točke.

Primjer PLY formata

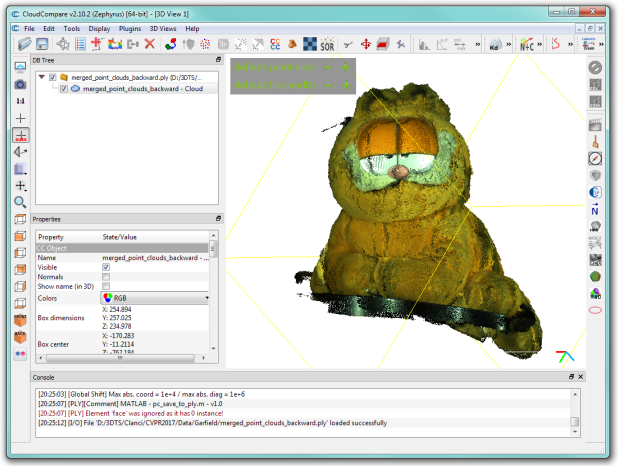
PLY datoteka započinje zaglavljem nakon kojeg slijede podaci bilo u ASCII bilo u binarnom formatu. Radi smanjivanja veličine datoteke preferirani način pohrane je binarni. Primjer:

```
ply
format binary_little_endian 1.0
comment MATLAB - pc_save_to_ply.m - v1.0
element vertex 498519
property float x
property float y
property float z
property float nx
property float ny
property float nz
property uchar red
property uchar green
property uchar blue
element face 0
property list uchar int vertex_indices
end_header
<PODACI U BINARNOM FORMATU>
```

Programi i biblioteke za rad s oblacima točaka

Od interesa su dva programa te jedna biblioteka.

Programi su MeshLab i CloudCompare.



Biblioteka je Point Cloud Library (PCL).

Meshlab

MeshLab je započeo kao projekt Visual Computing Lab-a pri CNR-ISTI-u (*Consiglio Nazionale delle Ricerche*, *Istituto di Scienza e Tecnologie dell'Informazione*).

Internetska stranica:
`https://www.meshlab.net/`

GitHub:
`https://github.com/cnr-isti-vclab/meshlab`

Najavni rad:
P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, G. Ranzuglia. "MeshLab: an Open-Source Mesh Processing Tool." Sixth Eurographics Italian Chapter Conference, pp. 129-136, 2008.

CloudCompare

CloudCompare je započeo kao projekt između Télécom ParisTech-a i Électricité de France.

Internetska stranica:
`https://www.cloudcompare.org/`

GitHub:
`https://github.com/cloudcompare/cloudcompare`

Point Cloud Library (PCL)

Point Cloud Library (PCL) je započeo kao projekt danas ugašene tvrtke WillowGarage.

Internetska stranica:
`https://pointclouds.org/`

GitHub:
`https://github.com/PointCloudLibrary/pcl`

Najavni rad:
R. B. Rusu, S. Cousins. "3D is here: Point Cloud Library (PCL)." IEEE ICRA, Shanghai, 2011.

Registracijski problem

Optički 3D skener zbog zaklanjanja ili okluzije ne može vidjeti/skenirati predmet sa svih strana.

Zbog toga:

1. skener pomičemo oko predmeta (tipično za velike predmete), ili
2. predmet okrećemo ispred skenera (tipično za male predmete za koje koristimo okretni motorizirani stol).

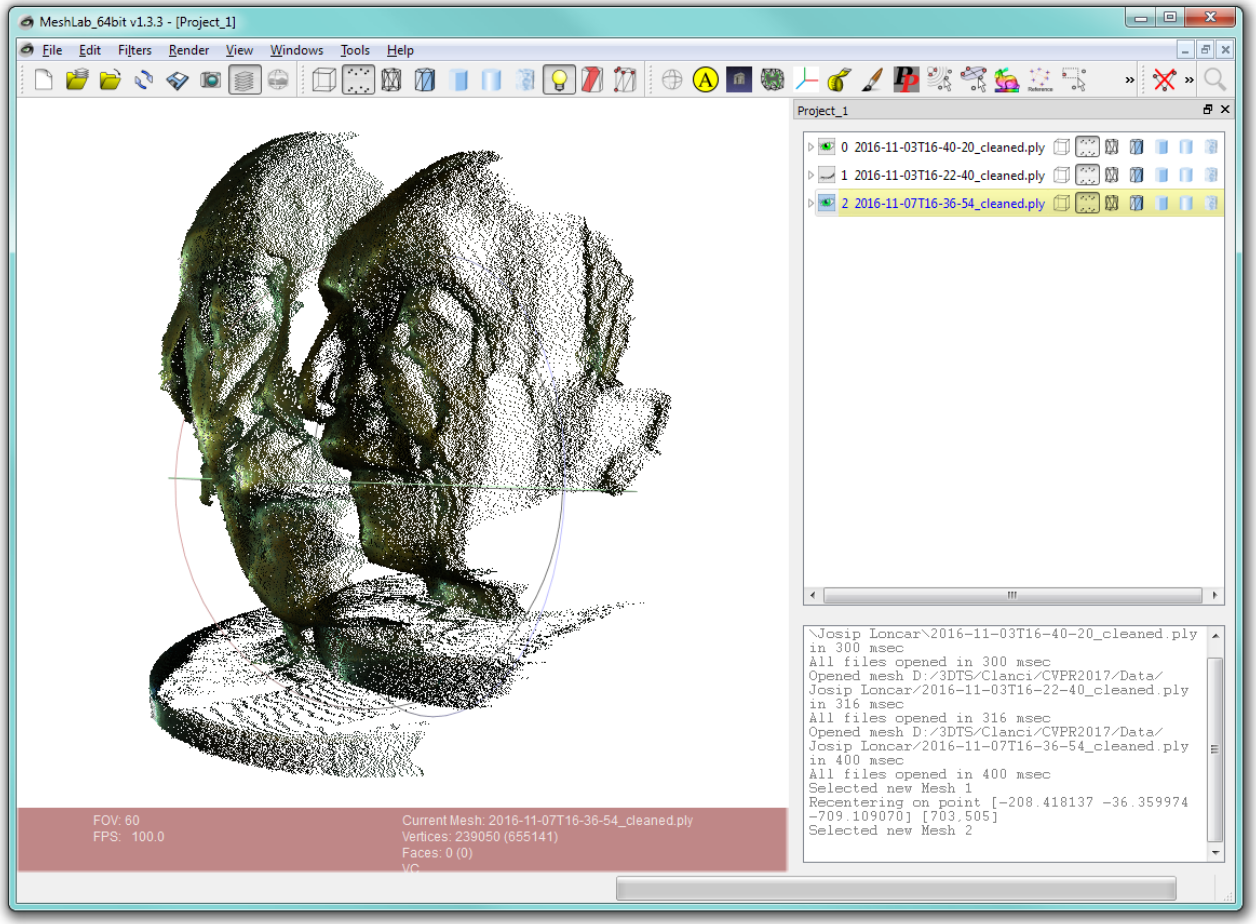
Svaki od oblaka točaka kojeg dobivamo sadrži koordinate u koordinatom sustavu skenera.

Sve te oblake je potrebno spojiti u jedan tako da se ne vide spojevi.

Taj problem spajanja oblaka točaka zovemo **registracijski** problem.

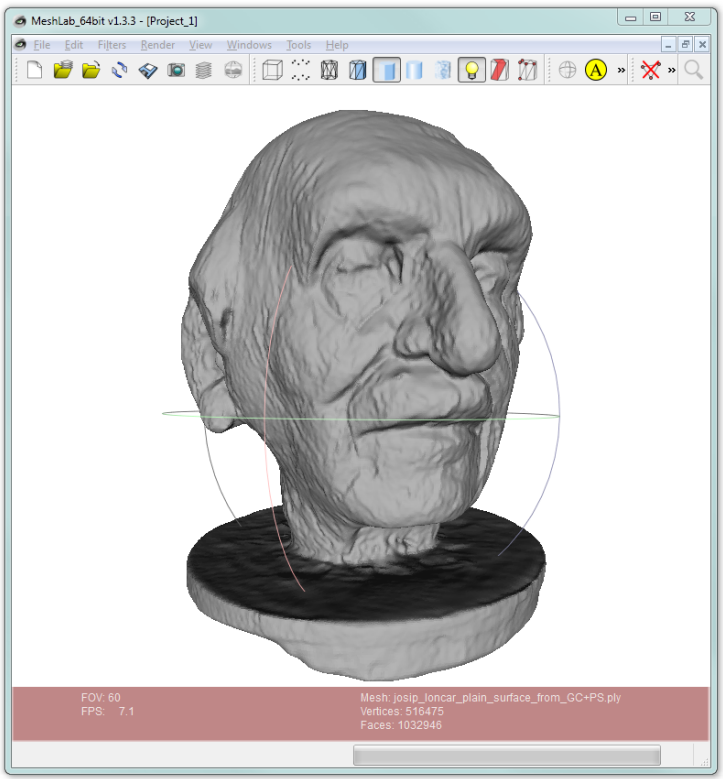
Svaki oblak ima svoj koordinatni sustav

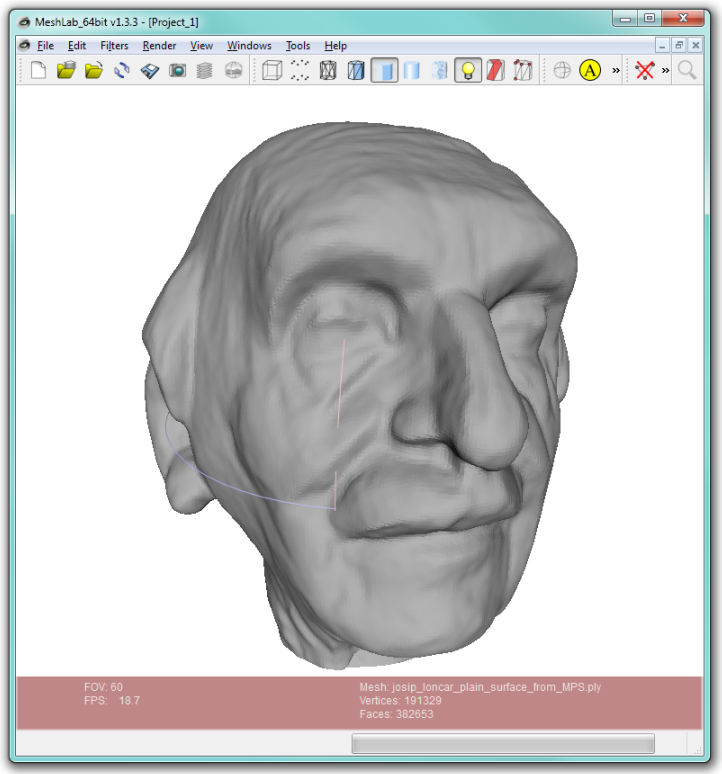
Dva oblaka prikazana u koordinatnom sustavu skenera:



3D rekonstrukcija iz više oblaka

Primjer konačnih rekonstrukcija, lijevo je rekonstrukcija za MPS kodiranje strukturiranog svjetla, a desno za GC+PS kodiranje:





Transformacije oblaka točaka

Ako oblak točaka predstavlja točke na površini čvrstog tijela onda su transformacije od interesa rotacija i translacija u 3D prostoru.

Rotaciju možemo prikazati preko rotacijske matrice R dimenzija 3×3.

Translaciju možemo prikazati preko translacijskog vektora t dimezija 3×1.

Točka (x, y, z) se transformira u točku (x', y', z') kao:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}}_{\mathrm{R}} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \underbrace{\begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}}_{\mathrm{t}} .$$

Reprezentacija 3D rotacije

Koliko stupnjeva slobode ima 3D rotacija?

Iako matrica rotacije ima devet članova, oni nisu nezavisni.

Matrica rotacije mora zadovoljiti dodatna svojstva:

1. matrica rotacije je ortogonalna matrica,

$$\mathrm{R}^T\mathrm{R} = \mathrm{R}\mathrm{R}^T = \mathrm{I},$$

2. njena determinanta je jedan,

$$\det \mathrm{R} = 1.$$

Uzimajući u obzir ta svojstva možemo pokazati da matrica 3D rotacije ima samo tri stupnja slobode, odnosno njeni elementi ovise samo o tri parametra.

Reprezentacija 3D rotacije

Kako najelegantnije zapisujemo 3D rotaciju?

Postoji više mogućnosti:

1. rotacijske matrice (grupa rotacija $SO(3)$),
2. kvaternioni,
3. os i kut rotacije,
4. Eulerovi i Tait-Bryanovi kutovi,
5. Rodriguesova reprezentacija,
6. Cayleyeva reprezentacija,
7. eksponencijalna mapa (Lijeva algebra $\mathfrak{so}(3)$),
8. i mnoge druge reprezentacije.

Što koristiti?
Koje su prednosti i nedostatci pojedinih reprezentacija?

Prekinutost reprezentacije rotacija

Svaka od reprezentacija ima određene parametre iz kojih možemo izračunati rotacijsku matricu.

Mi želimo pronaći rotaciju i translaciju koji registriraju dva dva oblaka točaka.

Kvalitetu te registracije ćemo mjeriti preko funkcije greške koju želimo minimizirati obzirom na rotaciju i translaciju.

U tom smislu nam je bitno da reprezentacija rotacije nema **prekida** u svojoj parametrizaciji, odnosno želimo da **mala** promjena parametara reprezentacije uzrokuje **malu** promjenu parametara rotacijske matrice.

Prekinutost reprezentacije rotacija

Eulerovi i Tait-Bryanovi kutovi su prekinuta reprezentacija jer nakon rotacijskog kuta od $+\pi$ radijana slijedi kut od $-\pi$ radijana što znači da mala promjena u vrijednostima rotacijske matrice može značiti veliku promjenu u rotacijskim kutovima.

Neke od lokalno neprekinutih reprezentacija rotacije su između ostalog rotacijske matrice, kvaternioni te os i kut rotacije.

U pravilu želimo neprekinutu reprezentaciju sa što manjim brojem parametara (idealno tri) što znači da rotacijske matrice sa svojih devet parametara nisu dobar izbor.

Često se koriste kvaternioni koji imaju četiri parametra, te os i kut rotacije s isto četiri parametra.

Os i kut rotacije

Rotaciju predstavljamo sa četiri parametra od kojih tri opisuju **os** rotacije, a četvrti **kut** rotacije oko te osi.

Os rotacije je opisana vektorom $\vec{e} = (e_x, e_y, e_z)$.

Kut rotacije θ je obično zadan u radijanima. Primijetite da promjena kuta rotacije za višekratnik od 2π ne mijenja rotaciju i da se lokalno uvijek možemo pomaknuti za mali kut.

Promjena duljine vektora $\vec{e}$ ne mijena os rotacije tako da je vektor uobičajeno normiran na jediničnu duljinu, odnosno vrijedi $e_x^2 + e_y^2 + e_z^2 = 1$ te vektor označavamo kao $\hat{e}$.

Matricu rotacije za zadane os i kut rotacije računamo pomoću Rodriguesove formule.

Os i kut rotacije

Reprezentacija preko četiri parametra od kojih tri definiraju os rotacije a četvrti kut rotacije se može koristiti u optimizacijskim postupcima jer je lokalno neprekinuta.

Glavni problem reprezentacije preko osi u kuta rotacije jest kompozicija više uzastopnih rotacija koja u praksi zahtjeva prijelaz u reprezentaciju koja se lagano ulančava kao što su rotacijske matrice gdje uzastopne rotacije postaju obično matrično množenje.

Reprezentacija se može sažeti u jedan vektor $\theta\hat{\mathbf{e}}$ gdje smjer vektora definira os rotacije i gdje duljina vektora definira kut rotacije. Takav sažeti zapis ponekad nije prikladan za optimizacijske postupke jer za $\theta = 0$ gubimo informaciju o smjeru osi rotacije.

Kvaternioni

Kvaternioni ili hiperkompleksni brojevi su proširenje kompleksnih brojeva.

Kvaternion zapisujemo kao uređenu četvorku ili u algebarskom obliku,

$$q = (q_0, q_1, q_2, q_3) \quad \text{ili} \quad q = q_0 + q_1\mathrm{i} + q_2\mathrm{j} + q_3\mathrm{k}.$$

Oznake i, j i k u algebarskom zapisu su kvaternionske imaginarne jedinice.

Realni dio kvaterniona q_0 se naziva *skalarnim* dijelom kvaterniona dok je ostatak $q_1\mathrm{i} + q_2\mathrm{j} + q_3\mathrm{k}$ *vektorski* dio.

Kvaternioni

Zbrajanje i množenje kvaterniona u algebarskom obliku poštuje normalna algebarska pravila uz dodatna svojstva množenja kvaternionskih imaginarnih jedinica:

$$\begin{aligned} -1 = \mathrm{i}^2 = \mathrm{j}^2 = \mathrm{k}^2 = \mathrm{ijk} \\ \mathrm{i} = \mathrm{jk} = -\mathrm{kj} \\ \mathrm{j} = \mathrm{ki} = -\mathrm{ik} \\ \mathrm{k} = \mathrm{ij} = -\mathrm{ji} \end{aligned}$$

Primijetite da je redoslijed množenja bitan, odnosno kvaternionske imaginarne jedinice su antikomutativne.

Kvaternioni

Kada koristimo kvaternione za reprezentaciju rotacije obično se ograničavamo na jedinične kvaternione za koje vrijedi $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$.

Vektorski dio kvaterniona $q_1\mathrm{i} + q_2\mathrm{j} + q_3\mathrm{k}$ se koristi za zapis osi rotacije ê u smislu da se smjer vektorskog dijela poklapa sa smjerom od ê. Kut rotacije θ oko osi ê je razdvojen na način da je $q_0 = \cos(\theta/2)$ te da je $(q_1, q_2, q_3) = \sin(\theta/2)(e_x, e_y, e_z)$.

Rotacijsku matricu R računamo iz pripadnog kvaterniona q kao:

$$\mathrm{R}(q) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 + q_2^2 - q_1^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 + q_3^2 - q_1^2 - q_2^2 \end{bmatrix}$$

Kvaternioni

Reprezentacija rotacije preko kvaterniona se može koristiti u optimizacijskim postupcima jer je lokalno neprekinuta, no nakon svakog koraka je potrebno renormalizirati dobiveni kvaternion.

Osim toga formalno algebarsko množenje kvaterniona odgovara kompoziciji rotacija.

Za razliku od reprezentacije preko kuta i osi rotacije nije potrebno računati rotacijske matrice ako želimo ulančati više uzastopnih rotacija.

Postupak iterativne najbliže točke

Postupak iterativne najbliže točke ili ICP algoritam (engl. *iterative closest point*) omogućava relativno jednostavnu registraciju dva oblaka točaka.

Literatura:

1. K. S. Arun, T. S. Huang, S. D. Blostein. "Least-Squares Fitting of Two 3-D Point Sets." IEEE TPAMI, vol. 9 no. 5, pp. 698-700, 1987. https://doi.org/10.1109/TPAMI.1987.4767965
2. Y. Chen, G. Medioni. "Object modelling by registration of multiple range images." Image and Vision Computing, vol. 10, no. 3, pp. 145-155, 1992. https://doi.org/10.1016/0262-8856(92)90066-C
3. P. J. Besl, N. D. McKay. "A method for registration of 3-D shapes." IEEE TPAMI, vol. 14, no. 2, pp. 239-256, 1992. https://doi.org/10.1109/34.121791

Matematičke osnove

Sukladno svom imenu ICP algoritam u svakoj iteraciji (koraku) traži najbliže točke.

Znate li formule za računanje sljedećih udaljenosti u 3D prostoru:

1. udaljenost točke od točke,
2. udaljenost točke od segmenta pravca, i
3. udaljenost točke od trokuta?

Udaljenost točka-točka

Neka je $\mathrm{x} = (x_1, x_2, x_3)$ prva točka te neka je $\mathrm{y} = (y_1, y_2, y_3)$ druga točka.

Udaljenost između točaka x i y je

$$d(\mathrm{x}, \mathrm{y}) = \sqrt{\sum_{i=1}^{3} (x_i - y_i)^2} = ||\mathrm{x} - \mathrm{y}||_2$$

Ako imamo više točaka y_j, $j = 1, 2, \ldots, N$, tada za zadanu točku x tražimo najbližu točku y_k pronalaženjem minimalne udaljenosti:

$$k = \arg\min_j d_P(\mathrm{x}, \mathrm{y}_j) = \arg\min_j ||\mathrm{x} - \mathrm{y}_j||_2$$

Složenost računanja k je $O(N)$ jer moramo izračunati udaljenost za sve točke y_j.

Udaljenost točka-pravac

Neka je $\mathrm{x} = (x_1, x_2, x_3)$ točka te neka je L linija koja spaja točke $\mathrm{a} = (a_1, a_2, a_3)$ i $\mathrm{b} = (b_1, b_2, b_3)$.

Udaljenost između točke x i linije L je

$$d(\mathrm{x}, L) = \min_{\substack{\alpha+\beta=1 \\ \alpha,\beta\in[0,1]}} ||\mathrm{x} - \alpha\mathrm{a} - \beta\mathrm{b}||_2$$

Ako imamo više linija L_j, $j = 1, 2, \ldots, N$, tada za zadanu točku x tražimo najbližu liniju L_k između točaka a_k i b_k pronalaženjem minimalne udaljenosti:

$$k = \arg\min_j d(\mathrm{x}, L_j) = \arg\min_j \left(\min_{\substack{\alpha+\beta=1 \\ \alpha,\beta\in[0,1]}} ||\mathrm{x} - \alpha\mathrm{a}_j - \beta\mathrm{b}_j||_2 \right)$$

Udaljenost točka-trokut

Neka je $\mathrm{x} = (x_1, x_2, x_3)$ točka te neka je T trokut određen vrhovima $\mathrm{a} = (a_1, a_2, a_3)$, $\mathrm{b} = (b_1, b_2, b_3)$ i $\mathrm{c} = (c_1, c_2, c_3)$.

Udaljenost između točke x i trokuta T je

$$d(\mathrm{x}, T) = \min_{\substack{\alpha+\beta+\gamma=1 \\ \alpha,\beta,\gamma\in[0,1]}} ||\mathrm{x} - \alpha\mathrm{a} - \beta\mathrm{b} - \gamma\mathrm{c}||_2$$

Ako imamo više trokuta T_j, $j = 1, 2, \ldots, N$, tada za zadanu točku x tražimo najbliži trokut T_k pronalaženjem minimalne udaljenosti:

$$k = \arg\min_j d(\mathrm{x}, T_j) = \arg\min_j \left(\min_{\substack{\alpha+\beta+\gamma=1 \\ \alpha,\beta,\gamma\in[0,1]}} ||\mathrm{x} - \alpha\mathrm{a}_j - \beta\mathrm{b}_j - \gamma\mathrm{c}||_2 \right)$$

Napomena: Udaljenost između točke i svih točaka neke linije ili nekog trokuta u navedenim izrazima računamo korištenjem baricentričnog koordinatnog sustava.

Što registriramo?

Sada kada smo se prisjetili kako se računaju udaljenosti razmotrimo što registriramo.

Dva uobičajena problema su:

1. registracija dva oblaka točaka, i
2. registracija oblaka točaka i modela sastavljenog od linija i trokuta.

Detaljno ćemo opisati samo registraciju dva oblaka točaka.

Registracija oblaka točaka i modela sastavljenog od linija i trokuta se dobiva zamjenom odgovarajućih funkcija udaljenosti.

Oznake

Raspolažemo s dva oblaka točaka, $\mathcal{X}$ i $\mathcal{Y}$.

Neka su $\mathrm{x}_i = (x_{1i}, x_{2i}, x_{3i})$ točke u $\mathcal{X}$ te neka ih ima N_X.

Isto tako neka su $\mathrm{y}_j = (y_{1j}, y_{2j}, y_{3j})$ točke u $\mathcal{Y}$ te neka ih ima N_Y.

Jedan od oblaka je nepomičan, neka to bude $\mathcal{Y}$. Drugi oblak transfomiramo tako da ga rotiramo i translatiramo, to je $\mathcal{X}$.

Prema tome tražimo rotaciju R i translaciju t koje transfomiraju oblak $\mathcal{X}$ tako da bude što sličniji oblaku $\mathcal{Y}$.

Svaka točka $\mathrm{x}_i \in \mathcal{X}$ se transformira u x' prema:

$$\mathrm{x_i}' = \mathrm{R} \cdot \mathrm{x}_i + \mathrm{t}.$$

Funkcija cilja (pogreške)

Da bi odredili pogrešku registracije moramo znati parove točaka iz oblaka $\mathcal{X}$ i $\mathcal{Y}$ koje su međusobno korespondentne.

Neka korespondentna točka od x_i bude y_k, odnosno par indeksa (i, k) definira korespondentni par točaka $(\mathrm{x}_i, \mathrm{y}_k)$.

Definiramo funkciju pogreške registracije

$$e = \frac{1}{N} \sum_{\text{po svim parovima } (i,k)} \left|\left| \mathrm{y}_k - (\mathrm{R}\mathrm{x}_i + \mathrm{t}) \right|\right|_2^2,$$

u kojoj zbroj ide po $N \leq N_x, N_y$ korespondentnih parova (i, k).

Cilj je **minimizacija** funkcije pogreške registracije.

Dva podproblema

Postupak iterativne najbliže točke razdvaja problem registracije u dva podproblema koji se naizmjenice (iterativno) rješavaju:

1. određivanje korespondencija, i
2. određivanje optimalne transformacije trenutne m-te iteracije koja se sastoji od rotacije R_m i translacije t_m.

Nakon što iteriranje završi konačna transformacija koja preklapa oblak točaka $\mathcal{X}$ preko obklaka $\mathcal{Y}$ je kompozicija svih nađenih rotacija R_m i translacija t_m.

Kompozicija rotacija i translacija

Jedan korak kompozicije rotacije i translacije je

$$R_2(R_1 x + t_1) + t_2 = R_2 R_1 x + R_2 t_1 + t_2.$$

Općenito vrijedi

$$R = \prod_{\substack{m=M \\ \text{unazad}}}^{1} R_m \quad \text{i} \quad t_m = \sum_{m=1}^{M} \Big(\prod_{\substack{l=M \\ \text{unazad}}}^{m+1} R_l \Big) t_m,$$

gdje pri računanju oba produkta rotacijskih matrica morate paziti na redoslijed jer idemo unazad (indeksi se smanjuju s lijeva na desno).

Određivanje korespondencija

Za određivanje korespondencija koristimo funkciju udaljenosti.

Za svaku točku $x_i \in \mathcal{X}$ iz oblaka točaka kojeg transformiramo tražimo korespondentnu točku y_k kao najbližu točku u oblaku $\mathcal{Y}$:

$$k = \arg\min_j ||x_i - y_j||_2.$$

Kako k ovisi o odabranom x_i da to naglasimo pišemo $k(i)$. Primijetite da ovakvo definiranje korespondencija nije uvijek smisleno jer više različitih x_i može imati istu korespondnentnu točku y_k.

Kako za svaku točku x_i moramo proći kroz sve točke y_j složenost traženja korespondencija je $O(N_x N_y)$.

Određivanje optimalne transformacije

Jednom kada znamo korespondenciju $k(i)$ tražimo optimalne R i t koji minimiziraju

$$e = \frac{1}{N_x} \sum_{i=1}^{N_x} ||\mathrm{y}_{k(i)} - (\mathrm{R}\mathrm{x}_i + \mathrm{t})||_2^2.$$

Rotacijsku matricu R predstavljamo kvaternionom q koji ima 4 parametra i 3 stupnja slobode jer mora biti $||q|| = 1$. Translacija t ima 3 parametra i 3 stupnja slobode.

Do rješenja možemo doći gradijentnim spustom koji započinje u $q = (1, 0, 0, 0)$ (što odgovara $\mathrm{R} = \mathrm{I}$) i $\mathrm{t} = 0$ (dakle nema transformacije) te se zatim spušta prema traženom rješenju.

No osim korištenja numeričkog postupka zadani problem je i analitički rješiv.

Analitičko rješenje (I)

Analitičko određivanje optimalne transformacije prvo centrira oba oblaka točaka tako da pripadni centri mase budu u ishodištu, a tek zatim se traži optimalna rotacija.

Neka su centri oblaka

$$\boldsymbol{\mu}_{\mathcal{X}} = \frac{1}{N_X}\sum_{i=1}^{N_X} \mathrm{x}_i \quad \text{i} \quad \boldsymbol{\mu}_{\mathcal{Y}} = \frac{1}{N_Y}\sum_{j=1}^{N_Y} \mathrm{y}_j$$

Nakon centriranja oblaka računamo kros-korelacijsku matricu centriranih oblaka (koja je identična kros-kovarijancijskoj matrici polaznih oblaka):

$$\boldsymbol{\Sigma}_{XY} = \frac{1}{N_X}\sum_{i=1}^{N_X}(\mathrm{x}_i - \boldsymbol{\mu}_{\mathcal{X}})(\mathrm{y}_{k(i)} - \boldsymbol{\mu}_{\mathcal{Y}})^T = \frac{1}{N_X}\sum_{i=1}^{N_X}\mathrm{x}_i\mathrm{y}_{k(i)}^T - \boldsymbol{\mu}_{\mathcal{X}}\boldsymbol{\mu}_{\mathcal{Y}}$$

Analitičko rješenje (II)

Označimo elemente matrice $\mathbf{\Sigma}_{XY}$ kako slijedi

$$\mathbf{\Sigma}_{XY} = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{bmatrix}.$$

Zatim formirajmo stupčani vektor

$$\mathbf{\Delta} = \begin{bmatrix} \sigma_{23} - \sigma_{32} \\ \sigma_{31} - \sigma_{13} \\ \sigma_{12} - \sigma_{21} \end{bmatrix}.$$

Analitičko rješenje (III)

Naposlijetu formiramo 4×4 matricu

$$\mathrm{Q} = \begin{bmatrix} \mathrm{tr}(\boldsymbol{\Sigma}_{XY}) & \boldsymbol{\Delta}^T \\ \boldsymbol{\Delta} & \boldsymbol{\Sigma}_{XY} + \boldsymbol{\Sigma}_{XY}^T - \mathrm{tr}(\boldsymbol{\Sigma}_{XY})\mathrm{I}_3 \end{bmatrix}$$

$$= \begin{bmatrix} \sigma_{11} + \sigma_{22} + \sigma_{33} & \sigma_{23} - \sigma_{32} & -\sigma_{13} + \sigma_{31} & \sigma_{12} - \sigma_{21} \\ -\sigma_{32} + \sigma_{23} & \sigma_{11} - \sigma_{22} - \sigma_{33} & \sigma_{12} + \sigma_{21} & \sigma_{13} + \sigma_{31} \\ \sigma_{31} - \sigma_{13} & \sigma_{21} + \sigma_{12} & \sigma_{22} - \sigma_{11} - \sigma_{33} & \sigma_{23} + \sigma_{32} \\ -\sigma_{21} + \sigma_{12} & \sigma_{31} + \sigma_{13} & \sigma_{32} + \sigma_{23} & \sigma_{33} - \sigma_{11} - \sigma_{22} \end{bmatrix}$$

Kvaternion q koji opisuje traženu optimalnu rotaciju jest svojstveni vektor matrice Q koji odgovara najvećoj svojstvenoj vrijednosti.

Matrica optimale rotacije je $\mathrm{R}(q)$.

Vektor optimalne translacije je $\mathrm{t} = \boldsymbol{\mu}_{\mathcal{Y}} - \mathrm{R}(q)\boldsymbol{\mu}_{\mathcal{X}}$.

Koraci ICP algoritma

1. Zadani su $\mathrm{x}_i \in \mathcal{X}$ i $\mathrm{y}_j \in \mathcal{Y}$.
2. Inicijaliziraj: brojač iteracija u $m = 0$, rotaciju u $q[0] = (1, 0, 0, 0)$, translaciju u $\mathrm{t}[0] = 0$, i grešku u $e[0] = +\infty$.
3. Ponavljaj
   - 3.1 Uvećaj m za 1.
   - 3.2 Za svaki x_i odredi korespondentni $\mathrm{y}_{k(i)}$ minimizacijom udaljenosti.
   - 3.3 Odredi optimalne rotaciju q_{opt} i translaciju $\mathrm{t}_{\mathrm{opt}}$ te pripadnu grešku $e[m]$.
   - 3.4 Primjeni $R(q_{\mathrm{opt}})$ i $\mathrm{t}_{\mathrm{opt}}$ na sve $\mathrm{x}_i \in \mathcal{X}$.
   - 3.5 Izračunaj ulančanu rotaciju i translaciju:

$$q[m] = q[m-1]q_{\mathrm{opt}}$$
$$\mathrm{t}[m] = \mathrm{R}(q_{\mathrm{opt}})\mathrm{t}[m-1] + \mathrm{t}_{\mathrm{opt}}$$

sve dok je $e[m-1] - e[m] > \tau$.

Teorem: lokalna konvergencija

Postupak iterativne najbliže točke ili ICP algoritam uvijek monotono konvergira prema lokalnom minimumu određenom funkcijom cilja (koja mjeri srednju kvadratnu udaljenost između korespondentnih točaka).

Dvije osnovne ideje potrebne za dokaz tvrdnje su:

1. računanje R i t za zadane korespondencije tako da minimiziramo e smanjuje prosječnu udaljenost između korespondentnih točaka, i
2. nalaženje novih parova korespondentnih točaka (x_i, y_j) postupkom traženja najbližih točaka smanjuje pojedinačne udaljenosti za svaki korespondentni par.

Diskusija kroz pitanja

1. Konvergiramo li u lokalni ili globalni minimum?
2. Jesmo li (ne)osjetljivi na početnu transformaciju?
3. Postoje li oblaci/podaci koje nije moguće registrirati?
4. Što radimo za različite neodređenosti koordinata?
5. Što ako je broj točaka u oblacima značajno različit?
6. Koliko je minimalno preklapanje oblaka potrebno za registraciju?
7. Što sa štršećim točkama (engl. *outliers*)?
8. Je li postupak simetričan? Što ako zamijenimo oblake $\mathcal{X}$ i $\mathcal{Y}$?
9. Je li brzina algoritma zadovoljavajuća?

Gradivo koje ulazi u ispit

Od tri cjeline

1. "Podaci, signali, sustavi te invarijante i simetrije",
2. "Profilometrija projiciranjem pruga i korištenjem boja" i
3. "Registracija i postupak iterativne najbliže točke"

u ispit ulazi samo zadnja cjelina
"Registracija i postupak iterativne najbliže točke".

Predmet 3D skeniranje

Ako vas zanima predstavljeno i ako ćete nastaviti diplomski studij na FER-u razmislite o upisu predmeta **3D skeniranje** (222434) koji je slobodni izborni predmet.

Stranice predmeta:
`https://www.fer.unizg.hr/predmet/3dske`

Namjera nam je da na svi studenti na predmetu sudjeluju u konstrukcij i izradi replikatora jednostavnih predmeta koji omogućava da skeniramo predmet, pripremimo sken za 3D pritanje, te zatim otprintamo plastičnu repliku.

Daljnje čitanje:

G.Taubin, D. Moreno, D. Lanman. "3D Scanning for Personal 3D Printing: Build Your Own Desktop 3D Scanner" (`http://mesh.brown.edu/desktop3dscan/`).