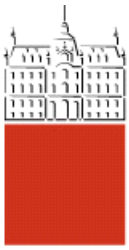
Univerza
v Ljubljani

Fakulteta za
gradbeništvo in
geodezijo

Jamova 2
1000 Ljubljana, Slovenija
telefon (01) 47 68 500
faks (01) 42 50 681
fgg@fgg.uni-lj.si



UNIVERZITETNI ŠTUDIJSKI PROGRAM PRVE STOPNJE GEODEZIJA IN GEOINFORMATIKA

Kandidat:

BLAŽ LIPUŠ

SEGMENTACIJA OBLAKA TOČK Z GAUSSOVO SFERO

Diplomska naloga št.: 131/GIG

POINT CLOUD SEGMENTATION USING GAUSSIAN SPHERE

Graduation thesis No.: 131/GIG

Mentorica:
doc. dr. Mojca Kosmatin Fras

Somentor:
asist. Tilen Urbančič

Ljubljana, 22. 09. 2016

STRAN ZA POPRAVKE, ERRATA

Stran z napako	Vrstica z napako	Namesto	Naj bo

»Ta stran je namenoma prazna«

Spodaj podpisani/-a študent/-ka Blaž Lipuš, vpisna številka 26203506, avtor/-ica pisnega zaključnega dela študija z naslovom: Segmentacija oblaka točk z Gaussovo sfero.

IZJAVLJAM

1. *Obkrožite eno od variant a) ali b)*

a) da je pisno zaključno delo študija rezultat mojega samostojnega dela;

b) da je pisno zaključno delo študija rezultat lastnega dela več kandidatov in izpolnjuje pogoje, ki jih Statut UL določa za skupna zaključna dela študija ter je v zahtevanem deležu rezultat mojega samostojnega dela;

2. da je tiskana oblika pisnega zaključnega dela študija istovetna elektronski obliki pisnega zaključnega dela študija;

3. da sem pridobil/-a vsa potrebna dovoljenja za uporabo podatkov in avtorskih del v pisnem zaključnem delu študija in jih v pisnem zaključnem delu študija jasno označil/-a;

4. da sem pri pripravi pisnega zaključnega dela študija ravnal/-a v skladu z etičnimi načeli in, kjer je to potrebno, za raziskavo pridobil/-a soglasje etične komisije;

5. soglašam, da se elektronska oblika pisnega zaključnega dela študija uporabi za preverjanje podobnosti vsebine z drugimi deli s programsko opremo za preverjanje podobnosti vsebine, ki je povezana s študijskim informacijskim sistemom članice;

6. da na UL neodplačno, neizključno, prostorsko in časovno neomejeno prenašam pravico shranitve avtorskega dela v elektronski obliki, pravico reproduciranja ter pravico dajanja pisnega zaključnega dela študija na voljo javnosti na svetovnem spletu preko Repozitorija UL;

7. da dovoljujem objavo svojih osebnih podatkov, ki so navedeni v pisnem zaključnem delu študija in tej izjavi, skupaj z objavo pisnega zaključnega dela študija.

V/Na: Ljubljana

Datum: 16.9.2016

Podpis študenta/-ke:

_____

»Ta stran je namenoma prazna«

BIBLIOGRAFSKO-DOKUMENTACIJSKA STRAN IN IZVLEČEK

UDK:	**528.8(043.2)**
Avtor:	**Blaž Lipuš**
Mentor:	**doc. dr. Mojca Kosmatin Fras**
Somentor:	**asist. Tilen Urbančič, univ. dipl. geod.**
	asist. dr. Dejan Grigillo
Naslov:	**Segmentacija oblaka točk z Gaussovo sfero**
Tip dokumenta:	**Diplomska naloga – univerzitetni študij**
Obseg in oprema:	**29 str., 1 pregl., 17 slik.**
Ključne besede:	**oblak točk, segmentacija, Gaussova sfera, Houghova transformacija**

Izvleček

Pridobivanje prostorskih podatkov z laserskim skeniranjem se v zadnjih letih hitro razvija. Oblaki točk, ki so primarni rezultat obdelave opazovanj laserskega skeniranja, so uporabni pri modeliranju skeniranih objektov ali zemeljskega površja. Zaradi številnih področij uporabe podatkov in velike količine podatkov se pojavlja potreba po avtomatskem ločevanju ustreznih točk iz oblaka točk.

V nalogi smo obravnavali segmentacijo (ločevanje) objektov v oblaku točk z Gaussovo sfero. Metodo smo uporabili na konkretnem primeru testnih podatkov terestričnega laserskega skeniranja. Opisali smo vse korake od skeniranja testnih podatkov do rezultatov segmentacije. Osredotočili smo se na tri geometrijske oblike objektov, in sicer ravnino, valj in stožec. Opisali smo osnovne lastnosti Houghove transformacije, binarizacije in pretvorbe podatkov v diskretno obliko ter jih vključili v postopek segmentacije. Za izvedbo praktičnega dela naloge smo napisali program za segmentacijo v programskem jeziku Python. Končni rezultat so izrisani oblaki točk treh geometrijskih oblik, ki smo komentirali in ovrednotili. Končne ugotovitve o segmentaciji z Gaussovo sfero so podane v zaključku.

»Ta stran je namenoma prazna«

BIBLIOGRAPHIC-DOCUMENTALISTIC INFORMATION AND ABSTRACT

UDC:	**528.8(043.2)**
Author:	**Blaž Lipuš**
Supervisor:	**Assist. Prof. Mojca Kosmatic Fras, Ph.D.**
Co-advisors:	**Assist. Tilen Urbančič, B.Sc. of Geodesy**
	Assist. Dejan Grigillo, Ph.D.
Title:	**Point cloud segmentation using Gaussian sphere**
Document type:	**Graduation Thesis – University Studies**
Scope and tools:	**29 p., 1 tab., 17 fig.**
Keyword:	**point cloud, segmentation, Gaussian sphere, Hough transform**

Abstract

Collecting spatial data using laser scanning is developing quickly. Point clouds, which are primary result of laser scanning, are useful for 3D modelling of scanned objects and Earth's surfaces. Because point clouds are useful in numerous fields and provide a large amount of data, there is a need to automatically extract certain features from point clouds.

This thesis describes point cloud segmentation using Gaussian sphere. We used this method on actual terrestrial scanned test data. We described the process in steps from obtaining test data to final results of segmentation. We focused on three geometrical objects, that are plane, cylinder and cone. We also described basic theory about the Hough transform, binarization and changing test data in discrete form, that were used in the process of segmentation. For processing the test data, we wrote a program in Python. With Python and its add-on libraries we plotted intermediate and final results, which we commented and evaluated. Final thoughts about the described segmentation method are given in conclusion.

»Ta stran je namenoma prazna«

ZAHVALA

Zahvaljujem se vsem profesorjem, ki so skozi leta študija pripomogli moji izobrazbi. Posebej pa se zahvaljujem tudi mentorici in obema somentorjema, kateri so me odlično vodili skozi postopek izdelovanja diplomske naloge.

Za dosežen uspeh se posebej zahvaljujem moji družini, ki me je moralno in finančno podpirala pri študiju, kljub nekaterim vmesnim neuspehom.

»Ta stran je namenoma prazna«

KAZALO VSEBINE

»Ta stran je namenoma prazna«

KAZALO SLIK

»Ta stran je namenoma prazna«

KAZALO PREGLEDNIC

»Ta stran je namenoma prazna«

1 UVOD

Tehnologija laserskega skeniranja omogoča pridobivanje ogromne količine podatkov o površju. Iz oblakov točk je mogoče opredeliti ploskve, ki so bile zajete, vendar neobdelan oblak točk pove bolj malo. Za branje in izrisovanje želenih informacij iz oblaka so potrebni računalniški algoritmi, saj je ne avtomatsko delo s takšno količino podatkov nesmiselno. V večini primerov posamezne točke ne dajejo informativne vrednosti. Zanimajo nas določeni deli oblaka ter kako jih modelirati v ploskve.

V nalogi bomo opisali enega od postopkov segmentacije oblaka točk. S segmentiranjem oz. razvrščanjem določimo željene dele oblaka. Obravnavana metoda temelji na prikazu na Gaussovi sferi, ki z analiziranjem smeri normalnih vektorjev v točkah oblaka, omogoča segmentacijo glede na sferne koordinate.

V drugem poglavju je na kratko opisana teorija o segmentaciji in oblaku točk. V tretjem poglavju opišemo izločevanje premic s Houghovo transformacijo, v četrtem poglavju predstavimo Gaussovo sfero ter kako z njo prepoznati ploskve. V petem poglavju podrobneje obravnavamo praktični primer, kjer opišemo postopek skeniranja oblaka točk ter postopek segmentacije z analizo normalnih vektorjev na Gaussovi sferi. Uporabljeno metodo segmentacije preizkusimo z iskanjem točk ravnine, stožca ter valja.

Analiza rezultatov kaže, da je segmentacija z Gaussovo sfero izvedljiva za izrazite objekte v oblaku točk. Predvidevamo, da je metoda za velike oblake točk manj primerna, saj objekti niso tako izraziti zaradi šuma ter majhnega deleža točk, ki pripadajo iskanim objektom.

2 O OBLAKU TOČK

Če želimo obdelovati oblak točk, moramo najprej vedeti, kaj to sploh je. Oblak točk je množica prostorskih točk v izbranem koordinatnem sistemu. Koordinatni sistemi so večinoma lokalni, lahko so tudi vezani na državne ali globalne koordinatne sisteme.

Točke v oblaku točk so podane s kartezičnimi koordinatami x, y in z. Poleg samih koordinat točk lahko oblak vsebuje tudi druge spremenljivke, kot je na primer intenzivnost odboja v vsaki točki.

V večini primerov je oblak točk pridobljen z laserskimi skenerji. Oblak točk lahko uvozimo v določeno programsko opremo, ki nato z vgrajenimi algoritmi modelira ploskve, izmerjene z laserskim skenerjem, ali diskretne točke oblaka poveže v mrežo (angl. mesh).

2.1 O segmentaciji oblaka točk

Segmentacija oblaka točk je velikokrat prva stopnja pridobivanja informacij iz 3D oblaka točk. Algoritmi za segmentacijo združujejo točke iz oblaka na podlagi določenih kriterijev. Najbolj pogosto segmentacije delujejo po principu združevanja točk, ki pripadajo določeni ploskvi, najsi bo ravnina, valj, stožec ali pa objekt zahtevnejše geometrije. Če gledamo na segmentacijo s tem razmišljanjem, bi lahko rekli, da je segmentacija dejansko prepoznavanje enostavnih ploskev v oblaku točk (Shan, Toth, 2009).

Deli oblaka točk, pridobljeni s segmentacijo, se lahko uporabljajo za veliko stvari. Segmenti ravnin so zelo pomembni pri pridobivanju ravnin za modeliranje stavb. Segmenti valja in stožca pa so lahko uporabni pri modeliranju industrijskih obratov in linij, medtem ko so segmenti gladkih ploskev zelo pomembni pri modeliranju terena (Shan, Toth, 2009).

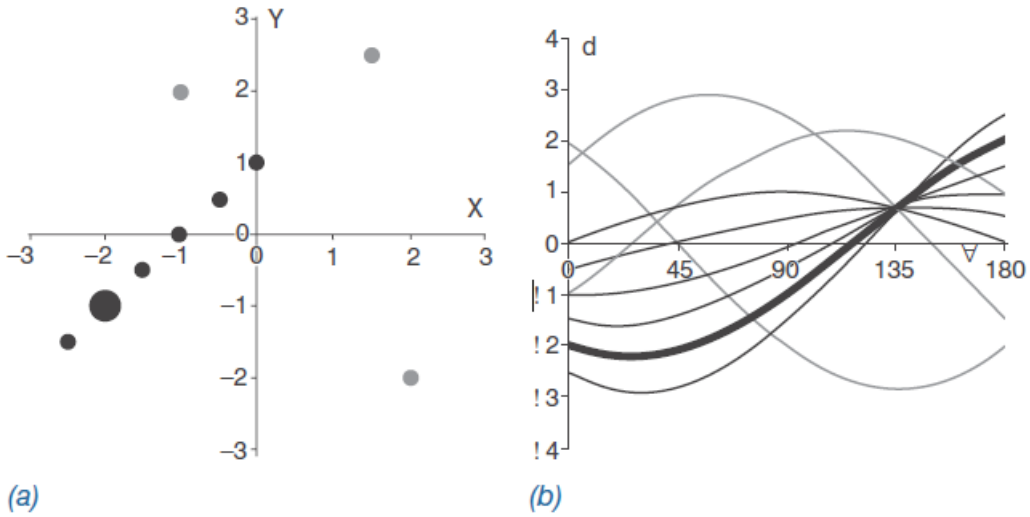
3 IZLOČEVANJE LINIJ

V primeru modeliranja objektov so linije in ravnine posebno zanimive. Izločevanje linij je v oblakih točk zahtevno, saj so točke naključno porazdeljene po ploskvah. Glede na gostoto točk v oblaku, so lahko robovi objektov začrtani bolj ali manj natančno. Izločevanje 3D ali 2D linij je lahko pomembno pri določanju žičnih modelov, tlorisov stavb ali pa napenjanju rastrov na 3D modele, pridobljene iz oblakov točk.

Prepoznavanje linij je lahko izvedljivo z že dolgo uveljavljeno Houghovo transformacijo. Premico je možno opisati z naslednjo enačbo (Vosselman, Mass, 2010):

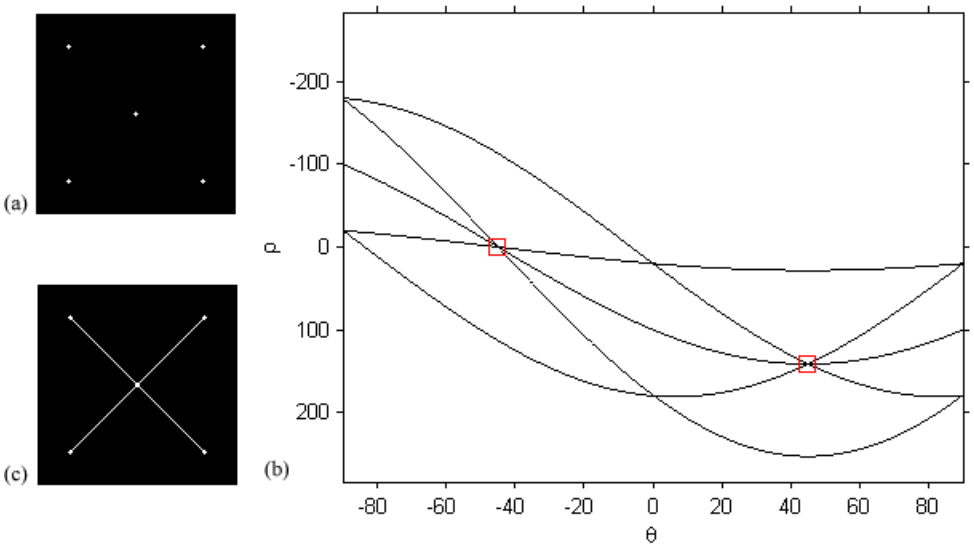
$$d = Xcos\alpha + Ysin\alpha$$

Pri tej enačbi je α kot med premico in Y osjo, d pa predstavlja razdaljo premice od izhodišča koordinatnega sistema. Houghova transformacija za vsak fiksen α in d definira neskončno množico točk (X,Y) na premici. Po drugi strani, vsaka točka (X,Y) podaja neskončno število premic (Vosselman, Mass, 2010).



Slika 1: Houghova transformacija: (a) točke v kartezičnem 2D (objektnem) prostoru, (b) pripadajoče sinusoide v 2D parametričnem prostoru (Vosselman, Mass, 2010).

Vsaka točka poda sinusoido v dvodimenzionalnem parametričnem prostoru z osema α in d. Če naredimo transformacijo vseh točk, dobimo množico sinusoid v parametričnem prostoru (slika 1). Houghova transformacija privede do parametrov premice v objektnem prostoru z določanjem lokacije v parametričnem prostoru, kjer je največje število presekov sinusoid, kot je vidno na sliki 2.



Slika 2: Primer uporabe Houghove transformacije na enostavnem primeru (Grigillo, 2009).

Da bi transformacijo izvedli na računalniku, je potrebno parametričen prostor spremeniti v diskretno obliko. To pomeni, da ga je potrebno razdeliti v zbiralne celice (Grigillo, 2009). Za vsako celico se prešteje število sinusoid, ki gredo skozi njo. Celica z največjim številom sinusoid podaja parametre premice, ki gre skozi največ točk. Na sliki 2 (b) sta v parametričnem prostoru z rdečim kvadratom označeni premici, ki povezujeta 3 točke. Zaradi velikega šuma v koordinatah je mogoče, da se vse sinusoide ne sekajo v eni točki. Zato je pri spreminjanju parametričnega prostora v diskretno obliko, pomembno dobro izbrati ustrezno število celic po obeh oseh parametričnega prostora. Če je ločljivost parametrov α in d premajhna, obstaja možnost slabe natančnost določitve parametrov premice. V primeru previsoke ločljivosti parametrov α in d, pa se v celici ne seka dovolj veliko število sinusoid, kar tudi vodi k slabši natančnosti (Vosselman, Mass, 2010).

Postopek določanja maksimumov se ponavlja, dokler ni več mogoče najti premic z minimalnim predpisanim številom točk. Tako dobimo vse premice iz množice točk.

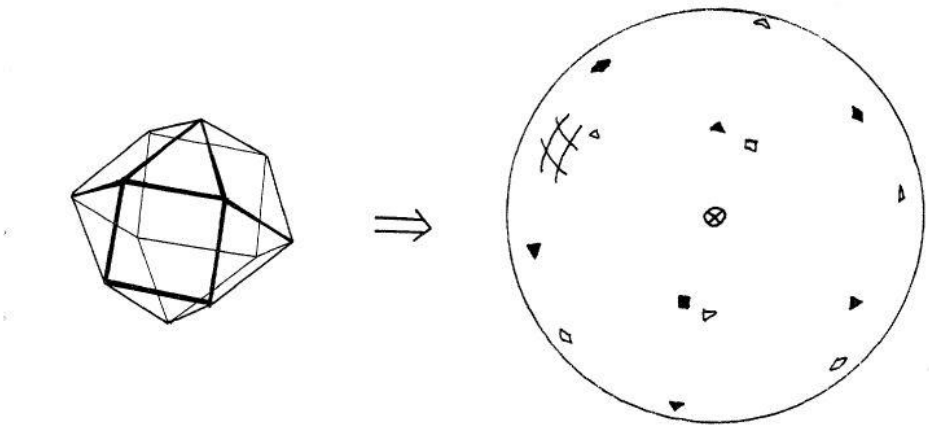
4 GAUSSOVA SFERA

Gaussova sfera je izris komponent normalnih vektorjev točk, ki pripadajo površju. Vsak normalni vektor predstavlja točko na Gaussovi sferi. Obstaja tudi drug princip izdelovanja Gaussovih sfer, ki se imenuje razširjena Gaussova sfera. Princip je zelo podoben sami Gaussovi sferi, le da se vsaki točki na Gaussovi sferi, ki jo normala poda, pripiše tudi površina ploskve s to normalo (Horn, 1984). Ta metoda je lahko uporabna pri teoretičnem obravnavanju posameznih objektov, kot tudi za strojno učenje s pomočjo umetne inteligence. V nalogi smo se osredotočili na metodo, pri kateri se površine ploskev ne pripišejo točkam na Gaussovi sferi. Tako smo samo s pomočjo nagibov in postavitvijo točk na Gaussovi sferi poskusili pridobiti posneta telesa iz oblaka točk.

Z Gaussovo sfero lahko obravnavamo diskretne ali pa kontinuirane primere (Horn, 1984).

4.1 Diskretni primeri

Pri diskretnih primerih prepoznavamo telesa, ki so sestavljena iz manjših ravnih ploskev. Za vsako ravnino na telesu je potrebno izračunati normalni vektor ravnine. Na sferi je rezultat viden kot množica diskretnih točk. Vsaka od teh točk opisuje orientacijo posamezne ravnine, ki je sestavni del telesa, ki ga prepoznavamo v oblaku točk (slika 3). Razširjena Gaussova sfera enolično (do translacije) opredeljuje konveksni polieder (Horn, 1984).

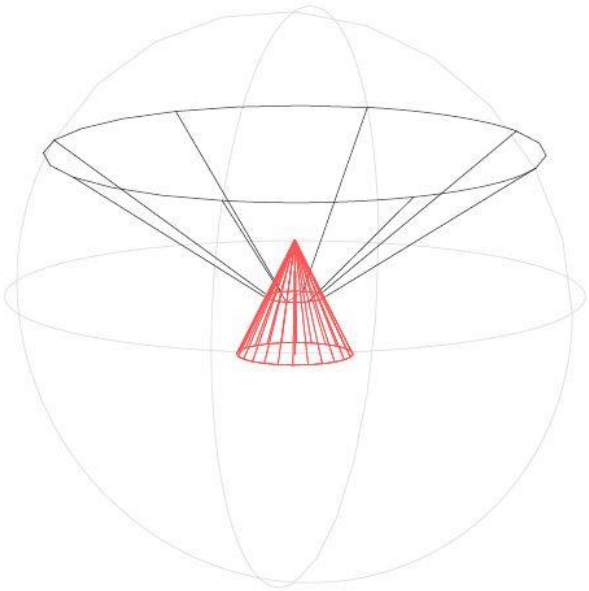


Slika 3: Primer diskretnega prikaza na Gaussovi sferi (Horn, 1984).

Na sliki 3 je prikazan konveksen polieder, ki ga obravnava Horn (1984). Vidna je diskretna točka za vsako ravnino poliedra. Za boljšo predstavo so točke izrisane z liki enake geometrije kot so stranice na poliedru. Že iz takšne Gaussove sfere lahko razberemo orientacijo poliedra v prostoru.

4.2 Kontinuirani primeri

Kadar obravnavamo gladke zvezne ploskve, je postopek drugačen. Ker nimamo ravnin, moramo izračunati normalo v vsaki točki, da ne izgubljamo podatkov o površini. Določanje normal je povezano z integriranjem po gladki ploskvi (Horn, 1984). Gaussova sfera nato ne kaže diskretnih točk za vsako ravnino, vendar pokaže krivulje na površini sfere. Vsaki kontinuirani zaobljeni ploskvi pripada krivulja na Gaussovi sferi, če imamo neskončno mnogo izračunanih normal na ploskvi.

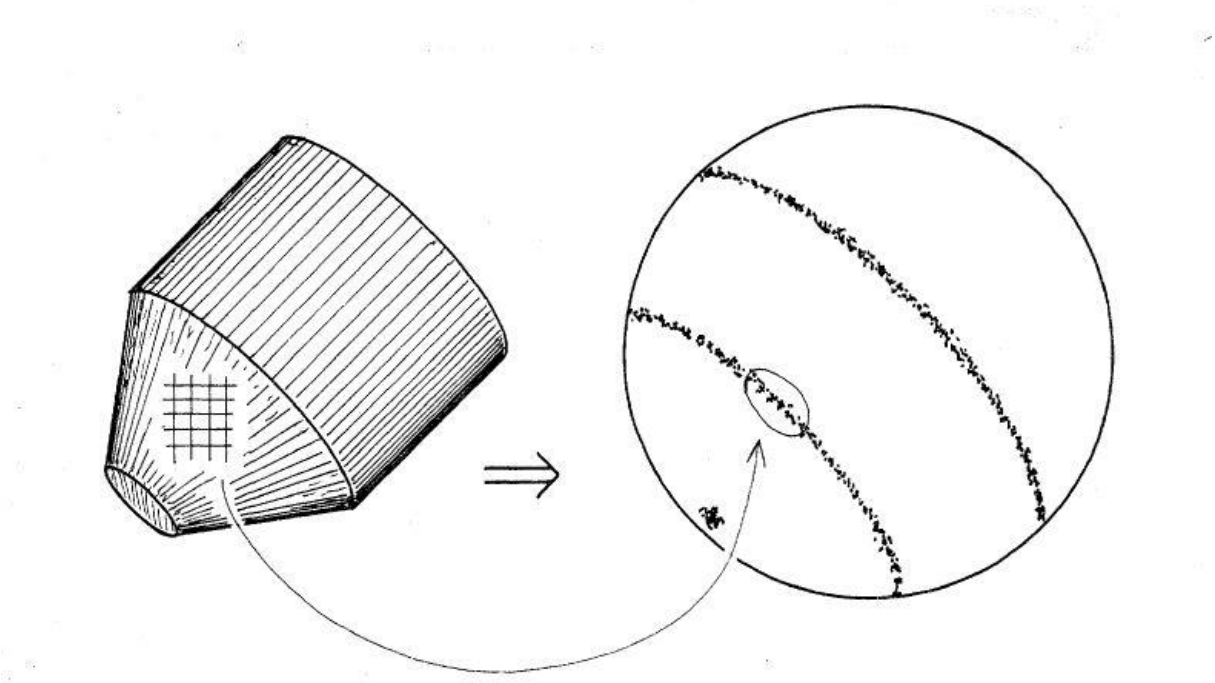


Slika 4: Primer kontinuirane ploskve na Gaussovi sferi.

Na sliki 4 vidimo, kako normale na plašč stožca teoretično opišejo krožnico na Gaussovi sferi. Razmerje med polmerom osnovne ploskve in višine stožca oziroma nagibom plašča, vpliva na položaj prikazane krožnice na Gaussovi sferi. Če upoštevamo to dejstvo, lahko vidimo, da v tem teoretičnem primeru, prikazanem na sliki 4, objektu oz. zakrivljeni ploskvi pripada točno ena krivulja. Da ima vsaka geometrijsko izvorna ploskev svojo zgostitev točk, nam pomaga pri segmentaciji geometrijskih ploskev iz oblaka točk.

4.3 Diskreten prikaz normalnih vektorjev na Gaussovi sferi

Predstavljajmo si, da neko ploskev razdelimo na mnogo majhnih ravnih ploskev in vsaki izračunamo normalo. Recimo, da vsaka normala predstavlja točko na Gaussovi sferi. Zgoščeni položaji točk na Gaussovi sferi opisujejo krivuljo kot je vidno na sliki 5.



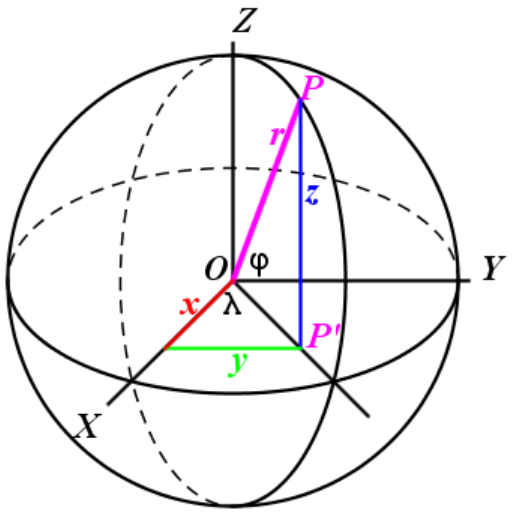
Slika 5: Diskretna aproksimacija objekta (Horn, 1983).

V primeru oblakov točk je to edina možna rešitev prikaza na Gaussovi sferi. V vsaki točki oblaka je potrebno izračunati normalo. Da bi to dosegli, moramo aproksimirati ravnino okoli točke, na kateri računamo normalo. To je mogoče doseči na več načinov, nekateri so opisani v Horn (1984). Logičen in poenostavljen pristop je s pomočjo vektorskega produkta. V tem primeru izračunamo vektorje med točko, za katero smo izračunali normalo, in dvema bližnjima točkama. Vektorski produkt teh dveh vektorjev predstavlja normalo. Končen rezultat je oblak točk s kartezičnimi koordinatami točk ter komponentami normalnih vektorjev. Tako je vzpostavljena povezava med vsako točko in lokacijo normalnih vektorjev na Gaussovi sferi.

4.4 Prepoznavanje ploskev z Gaussovo sfero

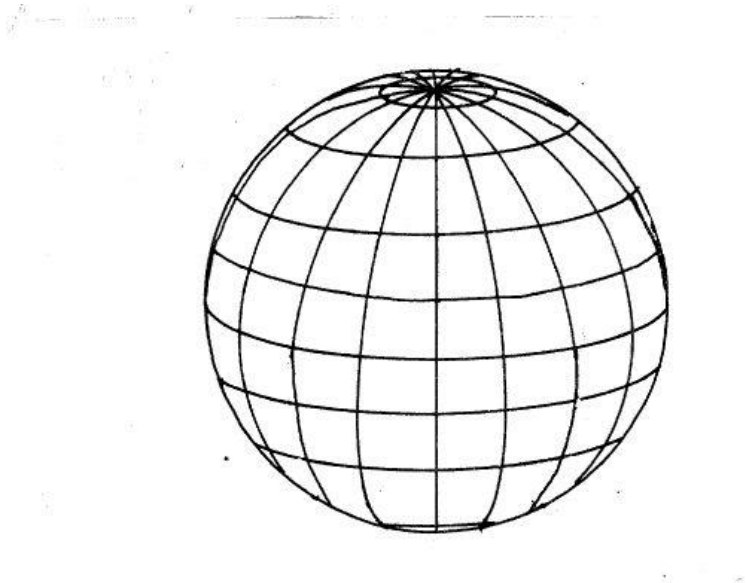
Z Gaussovo sfero pridobimo informacije o nagibih in ukrivljenosti 3D modela. Ostaja vprašanje, kako iz zgoščenih položajev točk na Gaussovi sferi prepoznati, kateri ploskvi v oblaku točk le-te pripadajo.

Obravnava se diskreten prikaz normalnih vektorjev na Gaussovi sferi, izračunanih iz koordinat točk v oblaku. Za lociranje zgostitev normal na Gaussovi sferi je potrebno Gaussovo sfero spremeniti v diskretno obliko s pomočjo sfernih koordinat.



Slika 6: Prikaz sfernih koordinat φ in λ v odvisnosti od kartezičnih koordinat (Earth Coordinates, 2016).

Kot je prikazano na sliki 6 je možen izračun φ in λ iz kartezičnih koordinat. Kot φ lahko zavzema vrednosti od 0° - 90°, kot λ pa vrednosti od 0° - 360°.



Slika 7: Enostavna porazdelitev (Horn, 1984).

Slika 7 prikazuje pretvorbo sfere v enostavno diskretno obliko. Sfero se razdeli v poljubno število zbiralnih celic po obeh sfernih koordinatah φ in λ. Takšna porazdelitev je koristna za izvajanje izračunov s pomočjo računalnika. Z enostavno porazdelitvijo je lahko določiti, katera točka spada v katero zbiralno celico z dvema enačbama sfernih koordinat (Wolfram Mathworld, 2016).

Poleg takšne razporeditve sfere so mogoče tudi drugačne, bolj zahtevne razporeditve (Horn, 1984). V primerih računalniškega obravnavanja zapletenejše razporeditve po Horn-u (1984) niso primerne.

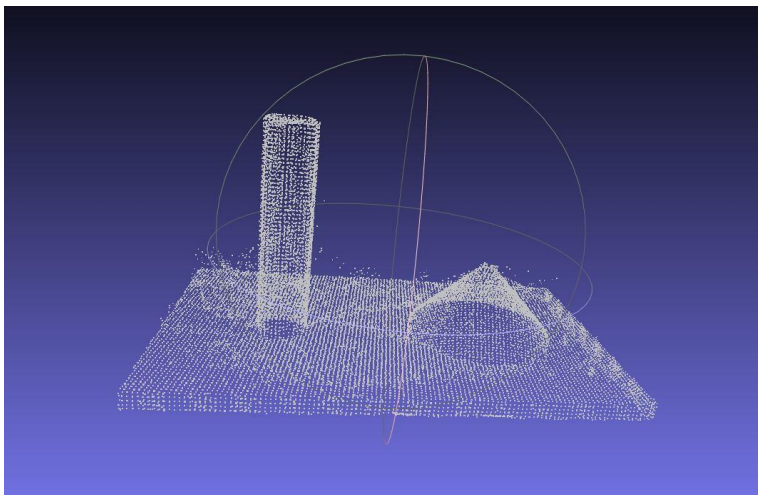
5 PREIZKUS V PRAKSI

V tem poglavju bomo opisali, komentirali in prikazali rezultate praktičnega preizkusa. Navajali bomo zaporedno, od pridobitve oblaka točk pa do končnega rezultata segmentacije.

5.1 Skeniranje testnih podatkov

Skeniranje testnih podatkov smo izvedli v eni izmed predavalnic na UL FGG. Testno polje sestavljajo miza ter modela valja in stožca, ki smo ju postavili na mizo. Uporabili smo terestrični laserski skener RIEGL VZ-400. Skener je bilo potrebno povezati z računalnikom, na katerem smo v programu RiSCAN PRO preko uporabniškega vmesnika upravljali s skenerjem. V bližnji okolici testnega polja smo izbrali 3 stojišča tako, da smo objekte skenirali iz vseh strani. Po skeniranju smo s programom RiSCAN PRO registrirali oblake točk s posameznih stojišč v enoten oblak točk. Oblak točk smo filtrirali z octree filtrom (Schnabel, Klein, 2006) na gostoto 10 milimetrov, da se je zmanjšala količina podatkov za potrebe kasnejšega testiranja algoritmov. Število točk v oblaku je bilo 21.032, kar je relativno malo.

Filtriran oblak točk smo uvozili v program MeshLab (MeshLab, 2016).



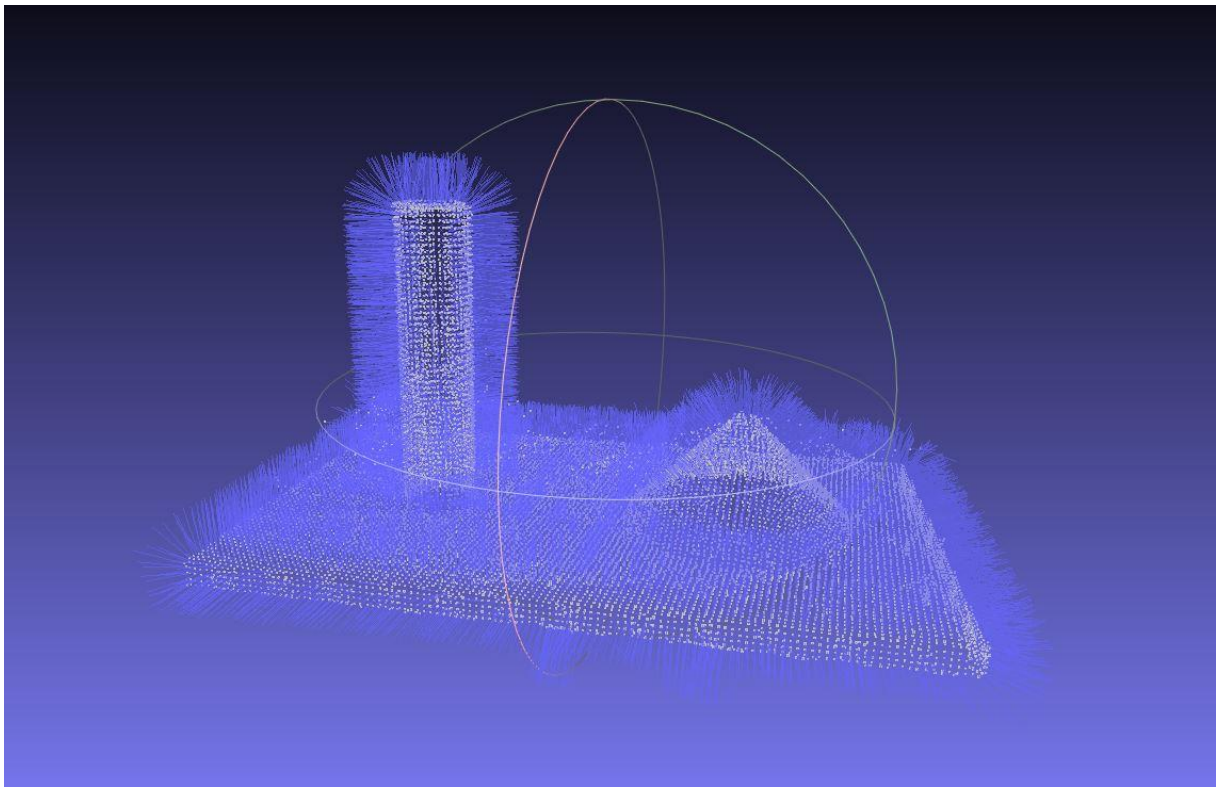
Slika 8: Oblak točk, filtriran s filtrom octree ločljivosti 10 mm in prikazan v MeshLabu.

Na sliki 8 je vidno, da oblak točk vsebuje nekaj šuma. Tega pri skeniranju žal ne moremo preprečiti.

5.2 Izračun normalnih vektorjev

Normalni vektor je vektor, ki je pravokoten na ravnino, zato je potrebno aproksimirati ravnino v vsaki točki oblaka točk. V oblaku točk se ravnina aproksimira glede na sosednje točke. V odvisnosti od razgibanosti oblaka točk je dobro, da se število sosednjih točk, ki bodo prispevale k aproksimaciji ravnine, regulira.

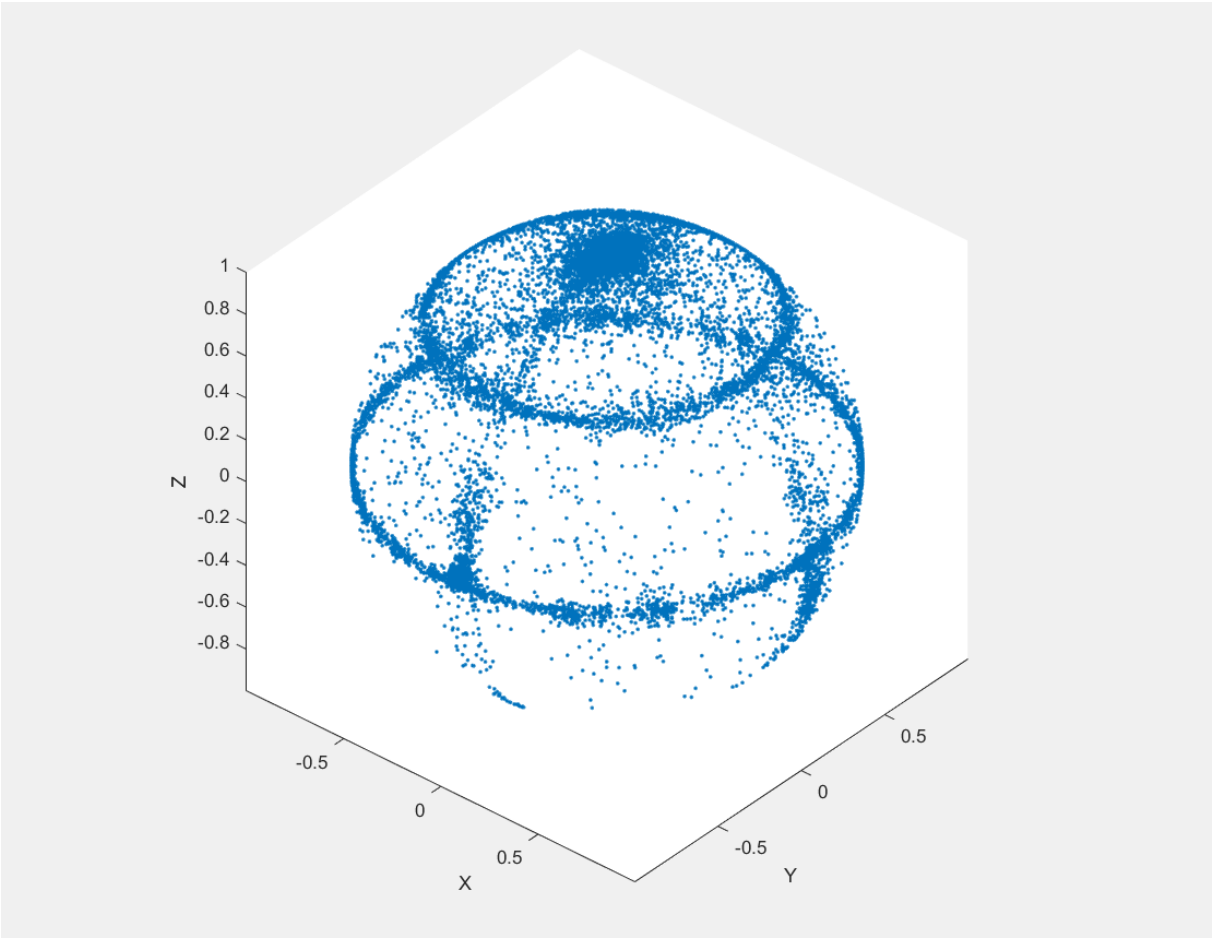
V našem primeru smo normale izračunali s pomočjo programa MeshLab. Edini parameter pri funkciji je že omenjeno število sosednjih točk, ki smo ga nastavili na 20. V MeshLab-u smo izrisali oblak točk z normalnimi vektorji v vsaki točki oblaka (slika 9).



Slika 9: Prikaz normal na oblaku točk.

MeshLab omogoča izvoz koordinat točk ter ostalih atributov v različnih formatih. Pri izvozu smo lahko izbrali možnost, da poleg vsake točke izvozi tudi komponente normalnega vektorja za vsako točko v oblaku in jih zapiše v datoteko želenega formata. Izbrali smo enostavno *.xyz datoteko in izvozili oblak točk z normalnimi vektorji.

Izvožene normalne vektorje smo nato izrisali. Po izrisu je vidno, da konci vektorjev tvorijo sfero, če jih prestavimo v izhodišče koordinatnega sistema, saj MeshLab izračuna enotske normalne vektorje.



Slika 10: Prikaz Gaussove sfere testnih podatkov.

Po izrisu sfere na sliki 10 so vidni pasovi zgostitev. Če kartezične koordinate normalnih vektorjev pretvorimo v sferne koordinate φ in λ, lahko vidimo, da je največja zgostitev, ko se kot φ približuje 90°. Sklepamo, da točke s koordinato φ = 90° pripadajo največji ploskvi v oblaku točk, v našem primeru, ravnini. Če bi bil oblak točk popoln in bi vse točke ležale točno na ravnini, bi bila ravnina na Gaussovi sferi vidna kot točka (poglavje 4.1). Tej zgostitvi pripada tudi zgornja osnovna ploskev valja, ki je vzporedna z ravnino, ter seveda nekaj šuma.

Poleg ravnine sta razvidni tudi dve kontinuirani zgostitvi (poglavje 4.2) na vzporednikih sfere. Če pogledamo sliko 10, vidimo, da edini normalni vektorji s kotom φ = 0° pripadajo valju, saj so edini pravokotni na z os koordinatnega sistema oblaka točk. Zadnja večja zgostitev je vidna v srednjih vrednostih kota φ. Po vsej verjetnosti pripada stožcu, saj nam ploskev plašča stožca edina ne poda pravokotnih ali vzporednih normal na os z v oblaku točk.

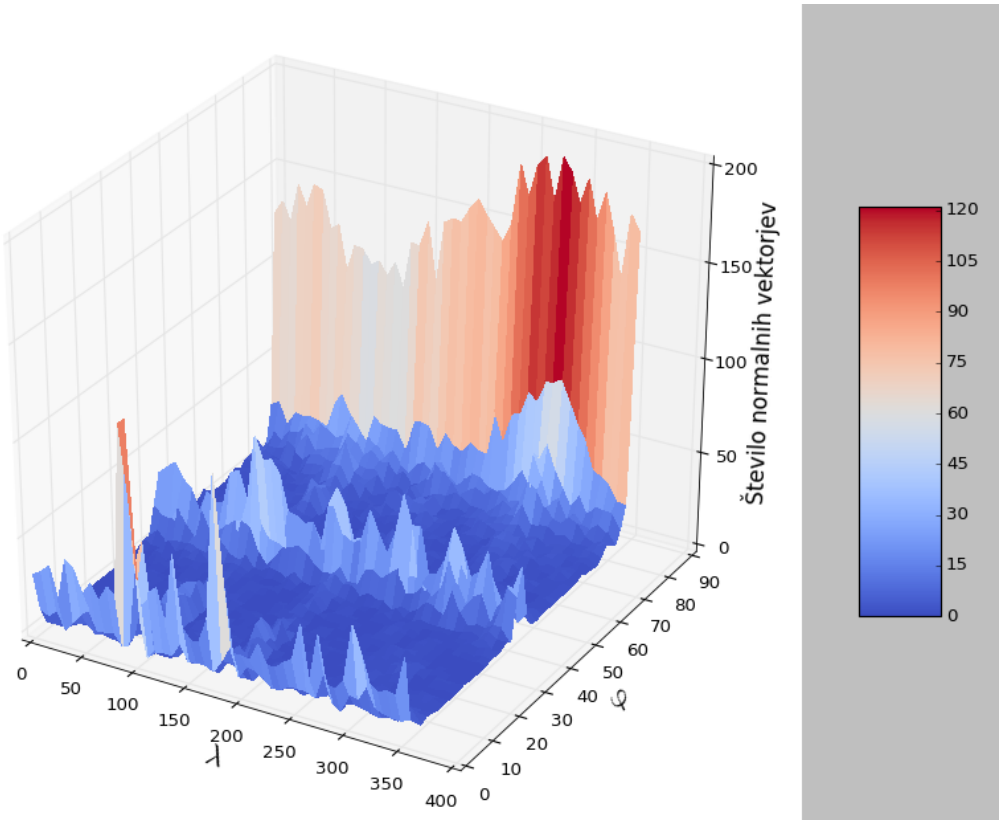
5.3 Razdelitev sfere

Iz prikaza na sliki 10 na Gaussovi sferi je razvidno, da je sfero potrebno razdeliti na poljubno število zbiralnih celic, če želimo zgostitvam določati položaj. Razdeliti je potrebno razpon kotov φ in λ v 2D mrežo, ki predstavlja zbiralni prostor sfere (poglavje 4.4). Za vsak normalni vektor preverimo, v katero zbiralno celico spada glede na koordinati φ in λ.

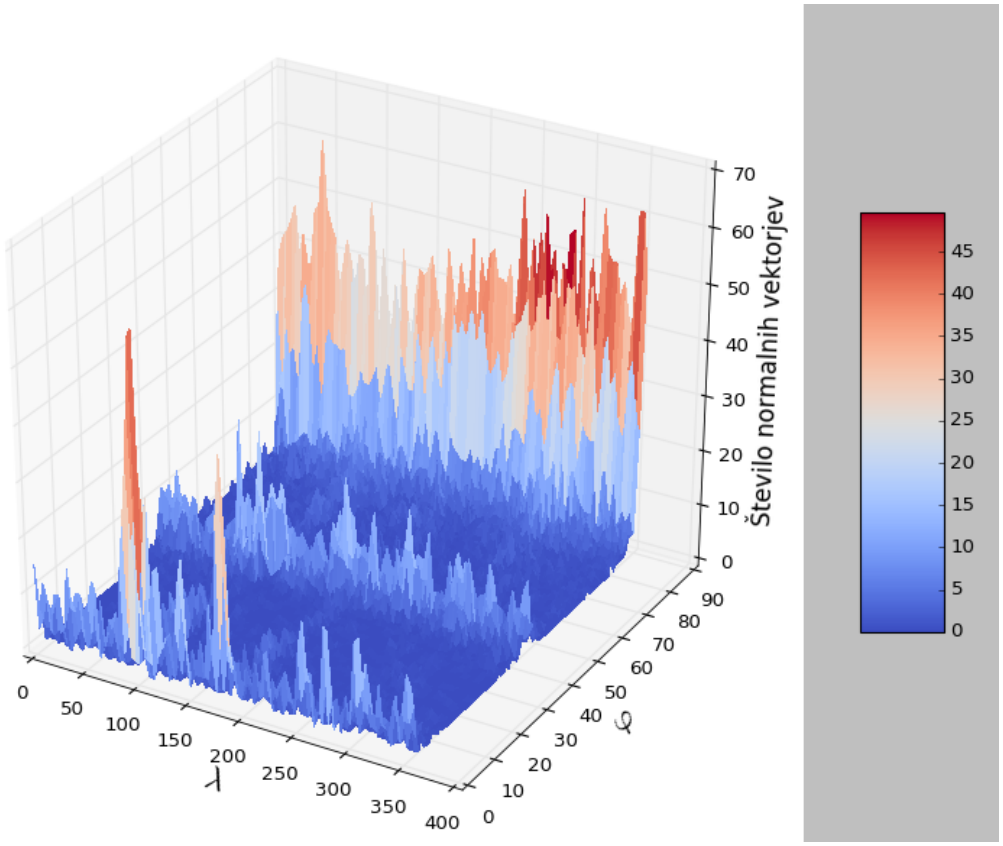
5.3.1 Izdelava 2D slike hemisfere

V obravnavanem primeru normalni vektorji večinoma ležijo v zgornji polovici Gaussove sfere (slika 10). Iz zgornje polovice sfere smo zato izdelali 2D sliko hemisfere. Dimenziji slike hemisfere ustrezata razponu kotov φ (od 0° do 90°) in λ (od 0° do 360°). Na ta način smo dejansko kreirali zbiralne celice, v katere lahko razvrščamo normalne vektorje, za katere smo izračunali kota φ in λ.

V našem primeru smo 2D sliko hemisfere razdelili na enako število zbiralnih celic po razponu kotov φ in λ. Obravnavali smo dva primera in sicer razdelitev na 45 (ločljivost: $\varphi = 2°$, $\lambda = 8°$) ter 90 (ločljivost: $\varphi = 1°$, $\lambda = 4°$) zbiralnih celic po obeh oseh (sliki 11 in 12). Ker iščemo celice z največ normalnimi vektorji kot tudi koordinate točk, katerih normalni vektorji spadajo v ustrezne zbiralne celice na 2D sliki hemisfere, smo izdelali dve prazni zbiralni matriki z dimenzijami, ki ustrezajo številu zbiralnih celic 2D slike hemisfere. Ti dve matriki predstavljata zbiralna prostora 2D slike hemisfere. V eno izmed zbiralnih matrik smo na mesta, kamor spada normalen vektor na 2D sliki hemisfere, prištevali 1, v drugo pa pripeli zaporedno število točke iz izvornega oblaka točk (slika 8). Zaporedna števila (indeksi) točk v drugi zbiralni matriki so pomembna za kasnejše iskanje tistih točk v izvornem oblaku točk, za katere smo s segmentacijo normalnih vektorjev ugotovili, da pripadajo določenemu geometrijskemu telesu.



Slika 11: Prikaz 2D slike hemisfere z ločljivostjo $\varphi = 2°$, $\lambda = 8°$ (45 zbiralnih celic po obeh oseh).



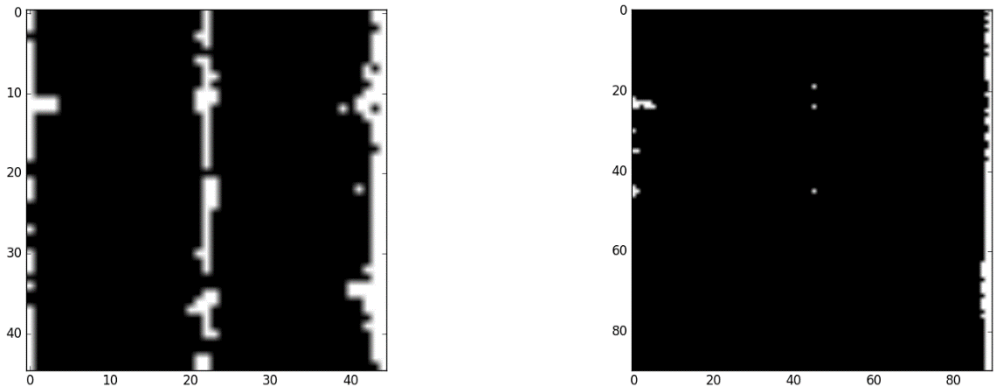
Slika 12: Prikaz 2D slike hemisfere z ločljivostjo $\varphi = 1°$, $\lambda = 4°$ (90 zbiralnih celic po obeh oseh).

Razlika med slikama 11 in 12 je predvsem v številu točk v vsaki zbiralni celici, ki je odvisno od ločljivosti, ki smo jo uporabili za razdelitev sfere (2° za φ na sliki 11 in 1° za φ na sliki 12), vendar vseeno obe sliki primerno prikažeta zgostitve na Gaussovi sferi v odvisnosti od ločljivosti kotov φ in λ.

Predvsem se iz slik 11 in 12 vidi, da posamezna telesa v zbiralnem prostoru opisujejo premice. Zato smo se odločili, da bomo zbiralni prostor binarizirali in premice poiskali s Houghovo transformacijo.

5.4 Binarizacija matrike

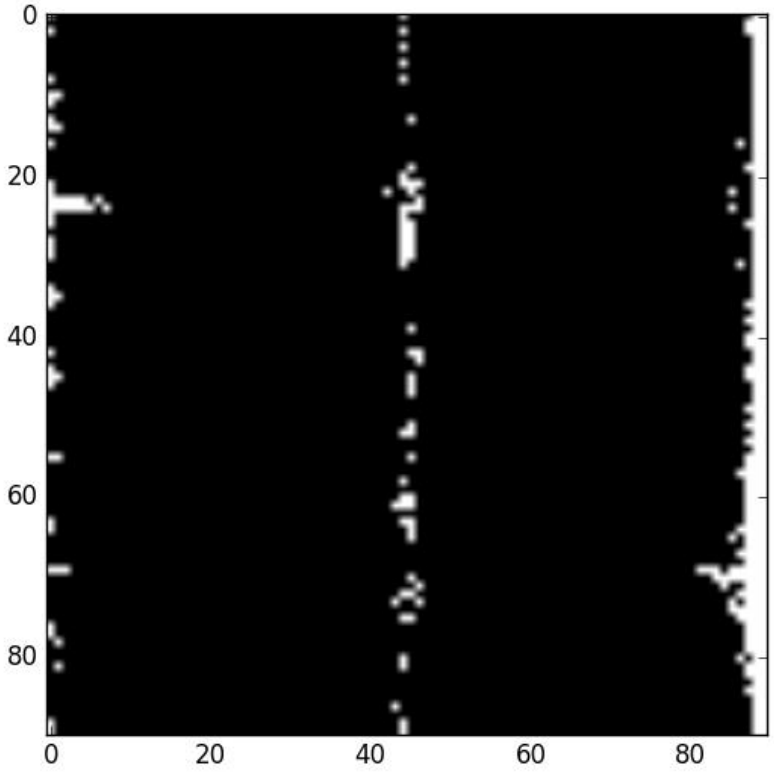
Zbiralno matriko 2D slike hemisfere smo v nadaljevanju pretvorili v binarno rastrsko sliko. V binarni sliki imajo piksli le dve vrednosti, ki sta po navadi 0 in 1 (črno in belo). Določitev vrednosti v binarni sliki je odvisna od kriterijev (mejne vrednosti) pretvorbe v binarno sliko. V našem primeru je bila mejna vrednost minimalno število točk v zbiralni matriki 2D slike hemisfere. Za mejno vrednost smo uporabili 20 točk.



Slika 13: Binarizirani rastrski sliki z dimenzijami 45x45 (levo, ločljivost: φ = 2° , λ = 8°) in 90x90 pikslov (desno, ločljivost: φ = 1° , λ = 4°) pri mejni vrednosti 20.

Na sliki 13 lahko opazimo pomembnost števila zbiralnih celic v 2D zbiralnem prostoru. Če je zbiralnih celic v 2D zbiralnem prostoru manj, je število točk, ki k določeni celici spadajo, večje. Sklepamo, da so položaji maksimumov bolj generalizirani. Mejno vrednost za binarizacijo je zato potrebno izbrati glede na število točk na Gaussovi sferi ter število zbiralnih celic 2D slike hemisfere (sliki 11 in 12). Slika 13 prikazuje dve binarni sliki. Pri izračunu parametrov premice s Houghovo transformacijo se uporabijo samo celice z vrednostjo 1. Hitro lahko ugotovimo, da desna binarna slika (slika 13) ne vsebuje dovolj belih celic za ustrezno določitev parametrov premice, zato bi morali določiti manjši prag (mejno vrednost) za binarizacijo in tako dobiti več belih pikslov.

Za izračun v nadaljevanju smo zato uporabili zbiralno matriko z ločljivostjo 1° po φ in 4° po λ (90 zbiralnih celic po obeh oseh) ter jo pretvorili v binarno rastrsko sliko dimenzije 90×90 pikslov s pragom 10 (slika 14).



Slika 14: Binarna rastrska slika z dimenzijami 90×90 pikslov (ločljivost: φ = 1° , λ = 4°) pri mejni vrednosti 10.

5.5 Houghova transformacija na binarni sliki

Princip delovanja Houghove transformacije na binarni sliki je podoben kot v primeru vektorskih točk, ki ga opisujemo v poglavju 3. Koordinate v tem primeru predstavljajo piksli z vrednostjo 1 (slika 14).

Najprej je potrebno definirati parametričen prostor. Parametričen prostor je dvorazsežen in ima dva elementa, kot α in dolžino d (poglavje 3). Element α lahko zavzame vrednost med -90° in 90°, element d pa ima razpon odvisen od števila zbirnih celic (pikslov) na sliki 14. Če je to število imenovano C, lahko element d zajema vrednosti $[-\sqrt{C^2+C^2}, \sqrt{C^2+C^2}]$ oz. od negativne do pozitivne vrednosti diagonale rastrske slike. Oba razpona smo shranili v spremenljivko in določili, na koliko zbiralnih celic naj se razpona razdelita. Določili smo zbiralni prostor s 180-imi vrednostmi po elementu α z ločljivostjo 1°. Binarna rastrska slika v našem primeru ima dimenzije 90×90 piklsov (slika 14), velikost diagonale

tako znaša približno 127 enot. Zbiralni prostor smo razdelili na 254 delov (2×127) po elementu d z ločljivostjo ene enote.

Vrednosti v zgoraj definiranima parametričnem prostoru in v zbiralnem prostoru so med sabo povezane preko indeksov oz. zaporedne številke zbirne celice. Za vsak položaj pikslov z vrednostjo 1 v binarni sliki in za vsak kot α iz definiranega parametričnega prostora smo izračunali parameter d in v ustrezno mesto v zbiralnem prostoru prišteli 1. Najvišje vrednosti v zbiralnem prostoru preko indeksov v zbiralnem prostoru posredno podajajo parametre najverjetnejših premic.

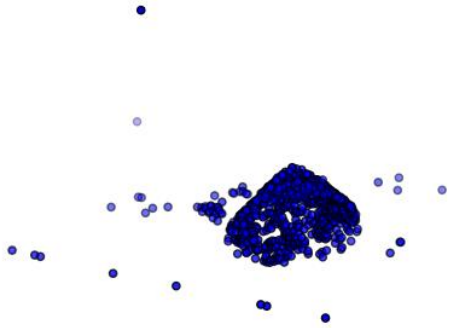
	Dolžina (d)	Kot (α)
Ravnina	89,8509803	0,000153
Stožec	45,6784313	0,000153
Valj	0,50196078	-0,000153

Preglednica 1: Najverjetnejši parametri premic za vse objekte iz testnih podatkov.

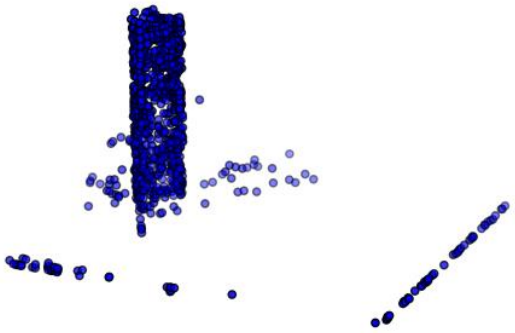
Enote dolžin v preglednici 1 so odvisne od velikosti zbirnih celic. Razlika med zaporednima zbirnima celicama znaša 1 enoto. Koti α so podani v decimalnih stopinjah. Iz parametrov premic v preglednici 1 vidimo, da so premice med sabo vzporedne. Razbrali smo, da sta osnovni ploskvi valja in stožca med sabo in ravnino vzporedni, saj imajo vsi objekti skoraj enak parameter α. Po parametru d je mogoče ločiti 3 skenirane objekte, ki jih obravnavamo. V našem primeru smo maksimume parametrov med seboj ločili s kriterijem, da se dolžine (d) med sabo razlikujejo za vrednost 20, saj drugače transformacija poda najbolj verjetne parametre, kateri bi vsi pripadali ravnini, ker ta vsebuje največ točk v izvornem oblaku točk. S tem kriterijem smo poiskali lokalne maksimume na 2D sliki hemisfere. Kriterij bi lahko še spreminjali glede na stanje na Gaussovi sferi in glede na različnost skeniranih objektov v oblaku.

5.6 Segmentacija iz oblaka točk

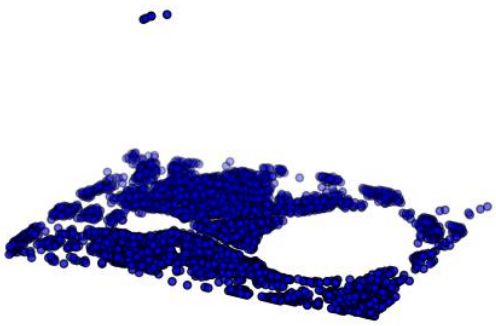
Dimenzije binarne slike uporabljene za Houghovo transformacijo so enake kot dimenzije zbiralne matrike v poglavju 5.3.1. Zato leži premica v zbiralni matriki (slika 12) na enakem mestu kot na binarni rastrski sliki. S parametri premic v rastru smo pridobili piksle v rastru, skozi katere najverjetneje poteka premica. Pridobljeni piksli predstavljajo zbiralne celice in posledično indekse v zbiralnih matrikah iz poglavja 5.3.1, ki so najbližje premici. Ker smo v poglavju 5.3.1 ustvarili zbiralno matriko, ki tvori povezavo z izvornim oblakom točk, smo lahko iz indeksov v izvornem oblaku točk poiskali kartezične koordinate točk, katerih normalni vektorji spadajo v določeno zbiralno celico (indeks).



Slika 15: Segmentiran stožec.



Slika 16: Segmentiran valj.



Slika 17: Segmentirana ravnina.

Na slikah 15, 16 in 17 so prikazani segmentirani objekti iz skeniranega oblaka točk. Segmentacija je bila izvedena uspešno. Razvidno je, da se največ šuma nahaja ob robovih objektov, zato ne dobimo lepo zaključenih objektov. V ravnini (slika 17) sta vidni dve praznini, kamor spadata osnovni ploski valja in stožca, saj so imele normale skeniranih točk na tem delu drugačno usmerjenost. Vidno je tudi nekaj točk zgornje osnovne ploskve valja nad eno izmed lukenj.

6 ZAKLJUČEK

V nalogi smo opisali segmentacijo na podlagi Gaussove sfere. Iz oblaka točk smo pridobili normalne vektorje in jih nato prikazali na Gaussovi sferi. Sfero smo razdelili s sfernimi koordinatami in izdelali 2D sliko hemisfere, ki prikazuje zgostitve točk na sferi. Rastrsko sliko smo binarizirali. Binarna slika je bila vhodni podatek za izvedbo Houghove transformacije, s katero smo pridobili parametre premic na rastru. Ker je binarna rastrska slika enakih dimenzij kot diskretna 2D slika hemisfere, smo lahko s parametri premic določili zaporedna števila (indekse) zbiralnih celic diskretne 2D slike hemisfere, ki ležijo na premici. Preko indeksov smo dostopali do koordinat točk iz izvornega oblaka točk, katerih normalni vektorji spadajo v z indeksi določene celice v zbiralni matriki 2D slike hemisfere.

Končni rezultat je segmentiran oblak točk. Vse objekte, ki smo jih v nalogi obravnavali, smo tudi uspešno izločili iz oblaka točk. Ugotovili smo, da popolna segmentacija ni tako samoumevna in imamo pri vseh izmed iskanih objektov kar veliko šuma. Ker smo kot testne objekte uporabili pravilne geometrijske oblike, ki jih lahko opišemo z matematičnimi enačbami, bi lahko šum izločili na podlagi oddaljenosti točk v oblaku od teoretično izračunane oblike, kar pa v naši nalogi nismo naredili.

Dodajmo še nekaj podrobnosti, ki bi jih lahko upoštevali v prihodnosti. Ustrezno ločljivost razdelitve Gaussove sfere v diskretno 2D sliko hemisfere bi lahko določili glede na število skeniranih objektov v oblaku in splošno število točk v izvornem oblaku točk. Če je število točk v oblaku relativno majhno, naj bo majhno tudi število zbiralnih celic v zbirnem prostoru. V obratnem primeru se lahko zgodi, da v celico, ki naj bi podajala maksimum, spada premajhen del normalnih vektorjev in je posledično potrebno regulirati mejno vrednost pri pretvarjanju v binarno obliko. Enako velja pri zbirnem prostoru uporabljenem v Houghovi transformaciji.

Po končani nalogi lahko ugotovimo, da je segmentacija z Gaussovo sfero precej uporabna. Poudariti moramo, da je težko segmentirati velik oblak točk z veliko objekti, saj imajo mnogi enake odtise na Gaussovi sferi. V takšnem primeru bi morali oblak razdeliti na več manjših delov in na vsakem izvesti segmentacijo z ustreznimi kriteriji. Zanimivo bi bilo poskusiti segmentirati strehe hiš iz oblaka točk zajetega z letalom, saj je večina hiš v Sloveniji dvokapnic. Vsaka kap hiše bi podala približno zgostitev na Gaussovi sferi, na podlagi katere bi lahko strehe segmentirali.

Napisan algoritem ni popoln in bi ga za zgoraj omenjen poskus morali dopolniti. Ravnina na 2D sliki hemisfere ne bi bila prikazana kot linija, če ne bi bila vzporedna z xy ravnino koordinatnega sistema oblaka točk. Za določanje ravnin, ki niso vzporedne z xy osjo koordinatnega sistema oblaka točk na podlagi Gaussove sfere, tako Houghova transformacije ne bi prišla v poštev in bi morali v algoritem dodati iskanje lokalnih zgoščišč, ki na 2D sliki hemisfere ne predstavljajo premic.

VIRI

Earth Coordinates. 2016. Nosco.ch. http://www.nosco.ch/mathematics/en/earth-coordinates.php (Pridobljeno 30.8.2016.)

Grigillo, D. 2009. Samodejno odkrivanje stavb na visokoločljivih slikovnih virih za potrebe vzdrževanja topografskih podatkov. Doktorska disertacija. Ljubljana, Univerza v Ljubljani, Fakulteta za gradbeništvo in geodezijo (samozaložba D. Grigillo): 156 f.

Horn, B. K. 1984. Extended Gaussian Images. Cambridge: Massachusetts Instutute of Technology. Proceedings of IEE 72, 12.

Meshlab. 2016. Computer Science department of University of Pisa. http://meshlab.sourceforge.net/ (Pridobljeno 31. 8. 2016.)

Schnabel, R., Klein, R. 2006. Octree-Based Point-Cloud Compression. Institut für Informatik II, Universität Bonn. Bonn, The Eurographics Assosiation.

Shan, J., Toth, C. K. 2009. Topographic Laser Ranging and Scanning: Principles and Processing. Boca Raton, Florida, CRC Press.

Wolfram Mathworld. 2016. Spherical Coordinates. http://mathworld.wolfram.com/SphericalCoordinates.html (Pridobljeno 31. 8. 2016.)

Vosselman, G., Mass, H.-G. 2010. Airborne and Terrestrial Laser Scanning. Dunbeath, Scotland: Whittles Publishing.

PRILOGE

UPORABLJEN PROGRAM

```
# -*- coding: utf-8 -*-

import os
import numpy as np
import pylab as pl
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter


#spremeni working dir (datoteke shrani v enako mapo kot se nahaja ta datoteka)
full_path = os.path.realpath(__file__)
this_dir = str(os.path.dirname(full_path))
os.chdir(this_dir)

# kart_nor = open("Osnovno_oct10mm_nekaj.txt","r")
# k_nor = kart_nor.readlines()

def sfr_max (st_raz, k_nor, izris = 0):
    '''
    st_raz(int) = stevilo razredov(vecje - bolj natanca segmentacija)
    st_objektov = int, kolkim objektom naj izvozi indekse iz oblaka tock
    k_nor = list stringov, prebran oblak tock npr. ["x y z xn yn zn", " "," "]

    Vrne matriko s stevilom normal v celici ter matriko z indeksi normal iz
    izvornega oblaka tock v vsaki celici
    '''
    xn = []
    yn = []
    zn = []
    fis = []
    lams = []
    k  = 0


    #postavitev spremenljivk za shranjevanje
    raz = []
    coun = []
    for i in range(st_raz):
        raz.append([])
        coun.append([])
        for j in range(st_raz):
            raz[i].append([])
            coun[i].append(0)

    #prebere kartezicne koordinate normal iz oblaka tock
    # in jih shrani po oseh x, y, z

    for i in range(len(k_nor)):
        k +=1
        vrstica = k_nor[i]
        vrstica = vrstica.split()
        xi = float(vrstica[3])
        yi = float(vrstica[4])
        zi = float(vrstica[5])


        xn.append (xi)
        yn.append (yi)
        zn.append (zi)
    print (k)

    #za vsako točko iz normal izračuna sferne koordinate normale na gausovi
    #sferi in jih porazdeli v st_raz(int)^2 celic. Za vsak razred shrani indeks
    #tock iz te celice v raz spremenljivko. coun deluje kot stevec tock v vsaki
```

celici

```
    fik = 90/st_raz
    lamk = 360/st_raz

    for i in range(len(xn)):
        xi = xn[i]
        yi = yn[i]
        zi = zn[i]

        if xi == 0:

            continue

        di = np.sqrt(xi**2 + yi**2)
        R = np.sqrt(xi**2 + yi**2 + zi**2)
        fi_r = np.arcsin(zi/R)
        lam_r = np.arctan2(yi,xi)

        fi = fi_r * 180 / np.pi
        lam = lam_r * 180 / np.pi
        lam = lam + 180


        lams.append(lam)
        fis.append(fi)

        lamp = 0

        for j in range(st_raz):
            if lamp < lam <= lamp+lamk:
                fip = 0
                for k in range(st_raz):
                    if fip < fi <= fip+fik:
                        coun[j][k] +=1
                        raz[j][k].append(i)
                    fip += fik
            lamp += lamk


    # Izris
    if izris != 0:

        fig = plt.figure()
        ax = fig.add_subplot(111, projection='3d')
        x1 = np.arange(0,360, lamk)
        y1 = np.arange(0,90, fik)

        y1, x1 = np.meshgrid(y1, x1)

        SUR = ax.plot_surface(x1, y1, coun,rstride=1, cstride=1, cmap=cm.coolwarm,
                              linewidth=0, antialiased=False)

        ax.set_xlabel(r'$\lambda$', fontsize=20)
        ax.set_ylabel(r'$\varphi$', fontsize=20)
        ax.set_zlabel('Število normalnih vektorjev', fontsize=15)
        fig.colorbar(SUR, shrink=0.5, aspect=5)
        plt.show()



    return (coun, raz)
```

```
# -*- coding: utf-8 -*-

import os
import numpy as np
import pylab as pl
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
from maximum_sfera import sfr_max


#spremeni working dir (datoteke shrani v enako mapo kot se nahaja ta datoteka)
full_path = os.path.realpath(__file__)
this_dir = str(os.path.dirname(full_path))
os.chdir(this_dir)


def binar(rast, edge):
    '''
    rast = matrika s stevilom normal v celici. Oblika je list v list [[],[],[],[]]
    edge = celo stevilo. kakšna je meja za spremembo binarne vrednosti

    Vzame vhodno matriko in glede na mejo določi binarne vrednosti
    Vrne matriko enakih dimenzij z binarnimi vrednostmi glede na mejo
    '''

    val = rast.copy()

    # Glasovanje
    for i in range(len(val)):
        for j in range(len(val[0])):
            k = val[i][j]
            if k <= edge:
                val[i][j] = 0
            else:
                val[i][j] = 1


    # Izris
    val1 = np.asarray(val)

    fig= plt.figure()
    ax1 = plt.axes()
    y1 = np.linspace(0,360, len(val))
    x1 = np.linspace(0,90, len(val[0]))

    x1, y1 = np.meshgrid(x1, y1)

    plt.imshow(val1, cmap ='Greys_r' )

    plt.show()

    return val

def hough_line(img):
  '''
  img = binarna matrika oblike [[],[],[],[]]

  Izracuna indekse parametrov linije in vrne v kombinaciji s stevilom glasov
  za posamezen par parametrov. Vrne tudi vektorja dolzin in kotov.
  '''

  # vektorja dolžin in kotov
  thetas = np.deg2rad(np.linspace(-90.0, 90.0, 2*90))
  dim = len(img)
  diag_len = int(np.ceil(np.sqrt(dim*dim + dim*dim))  ) # maksimalna dolžina
```

```
    rhos = np.linspace(-diag_len, diag_len, diag_len * 2.0)
    # Vrednosti za nadaljno uporabo
    cos_t = np.cos(thetas)
    sin_t = np.sin(thetas)
    num_thetas = len(thetas)

    # Matrika parametricnega prostora
    accumulator = np.zeros((num_thetas, 2*diag_len))
    accumulator = accumulator.tolist()

    # Indeksi mejnih tock (celic), skozi katere zelimo iskati linijo

    y_idxs, x_idxs = np.nonzero(img)


    # Glas v matriki parametricnega prostora
    for i in range(len(x_idxs)):
      x = x_idxs[i]
      y = y_idxs[i]

      for t_idx in range(num_thetas):
        # Izracun rho. Maksimalna dolzina dodana za pozitiven indeks
        rho = int(round(x * cos_t[t_idx] + y * sin_t[t_idx]) + diag_len)
        accumulator[t_idx][rho] += 1


    #sortiranje
    vred = []
    ind = []

    for j in range(len(accumulator)):
        for k in range(len(accumulator[0])):
            vred.append(accumulator[j][k])
            ind.append([j,k])


    max_ind = sorted (zip(vred, ind), reverse = True)

    return max_ind, rhos, thetas


def local_max (max_ind, rhos, thetas, ods):
    '''
    max_ind, rhos, thetas so izhodni podatki funkcije
    odst = float ali int

    iskanje lokalnih maximumov glede na odstopanje po dolzini
    '''

    locals = []
    locals.append(max_ind[0])

    for i in range(len(max_ind)):
        if locals[-1][1][1] - max_ind[i][1][1] >= ods:
            locals.append(max_ind[i])

    return locals


def get_cells (locals, rast, loc_num, thetas, rhos):
    '''
    locals = izhodni podatek funkcije local_max
    rast = izhodni podatek funkcije sfr_max
    loc_num = int, številka željenega lokalnega maksimuma po vrsti
    thetas, rhos = vektorja dolžin in kotov, izhodni podatek hough_line
```

```
Vrne indekse celic matrike oblike rast skozi katere poteka linija v dveh
enako velikih vektorjih.
'''
dim = len(rast)
diag = int(np.ceil(np.sqrt(dim*dim + dim*dim))  )

#iskanje pravih parametrov
rho_ind = locals[loc_num][1][1]
theta_ind = locals[loc_num][1][0]

rho = rhos[rho_ind]
theta = np.deg2rad(thetas[theta_ind])


#generiranje vektorja za izračun
ys = (np.linspace(0, len(rast)-1, len(rast))).tolist()
xs = []
# izracun y-ov
for yi in ys:
    x = ( (rho-yi*np.sin(theta))/(np.cos(theta)) )
    xs.append(int(round(x)))

return xs, ys
```

```
# -*- coding: utf-8 -*-

import os
import numpy as np
import pylab as pl
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D


#spremeni working dir (datoteke shrani v enako mapo kot se nahaja ta datoteka)
full_path = os.path.realpath(__file__)
this_dir = str(os.path.dirname(full_path))
os.chdir(this_dir)


def v_list(obj_ot):
    '''
    Spremeni readlines metodo branja v 3 liste s koordinatami x, y in z
    '''
    obj_x = []
    obj_y = []
    obj_z = []

    for i in range(len(obj_ot)):
        temp1 = obj_ot[i].split()[0:3]
        obj_x.append(float(temp1[0]))
        obj_y.append(float(temp1[1]))
        obj_z.append(float(temp1[2]))

    return [obj_x, obj_y, obj_z]


def search_plot1 ( OT, bins, x_ras, y_ras):
    '''
    maxs, list z tupli, ki vsebujejo število točk v enem izmed max razredov
    in indexe teh točk kot drugi element v tuplu (st_tock, [indeks])

    ind je št.objekta, ki ga iščem

    v oblaku točk poišče točke z indeksi iz max razredov

    vrne koordinate točk iz oblaka tock
    '''

    indeksi_tock0 = []
    lenx = len(x_ras)

    # Vzame vse indekse tock in jih pospravi v vektor(list)
    for i in range(lenx):
        indeksi_tock0.append( bins [int(y_ras[i])-1] [int(x_ras[i])-1] )

    indeksi_tock = sum(indeksi_tock0, [])

    # Sestavi oblak tock iz orginalnega preko danih indeksov
    obj_ot = []
    for i in indeksi_tock:
        obj_ot.append(OT[i])

    # Spremeni iz string v float matriko
    matrika = v_list(obj_ot)

    #izris tock iz posameznega razreda
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')
    plt.axis('equal')
```

```
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.scatter(matrika[0], matrika[1], matrika[2])

ax.get_xaxis().set_ticks([])
ax.get_yaxis().set_ticks([])
# ax.get_zaxis().set_ticks([])

ax.set_axis_off()

plt.show()

return matrika
```

```
# -*- coding: utf-8 -*-

import os
import numpy as np
import pylab as pl
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from maximum_sfera import sfr_max
from bin_hough import binar, hough_line, local_max, get_cells
from search_plot import search_plot1

#spremeni working dir (datoteke shrani v enako mapo kot se nahaja ta datoteka)
full_path = os.path.realpath(__file__)
this_dir = str(os.path.dirname(full_path))
os.chdir(this_dir)

#Odpri in preberi vhodno datoteko [x,y,z,xn,yn,zn]
kart_nor = open("Osnovno_oct10mm_nekaj.txt","r")
k_nor = kart_nor.readlines()

#Izračunaj matriko z zgoščišči na sferi in shrani rezultate v ločeni
#spremenljivki
podat = sfr_max(45,k_nor,1)
rast = podat[0]
bins_s_tock = podat[1]

#Izdelaj binarno matriko
k = binar(rast, 10)

#Izraunaj parametre linij v lokalnem maksimumu
max_ind, rhos, thetas= hough_line(k)
locals = local_max(max_ind, rhos, thetas, 15)

#Pridobi celice, ki jih linija seka
x_rav, y_rav = get_cells(locals, rast, 0, thetas, rhos)
x_stoz, y_stoz = get_cells(locals, rast, 1, thetas, rhos)
x_valj, y_valj = get_cells(locals, rast, 2, thetas, rhos)

#S pomočjo teh celic pridobi indekse iz orginalnega oblaka in izriši
plot1 = search_plot1(k_nor, bins_s_tock, x_stoz, y_stoz)
plot2 = search_plot1(k_nor, bins_s_tock, x_valj, y_valj)
plot3 = search_plot1(k_nor, bins_s_tock, x_rav, y_rav)
```