SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Mračno bodov

POČÍTAČOVÉ VIDENIE A SPRACOVANIE OBRAZU

4.ZADANIE

2023 **Adam Bucko, Daniel Fundárek**

Obsah:

Zadanie: 3

Vytvorenie mračna bodov: 4

RANSAC: 6

Výsledky RANSAC: 6

Segmentácia priestoru: 7

K-means 7

DBSCAN 10

Výsledky DBSCAN: 11

Záver: 12

Bibliography 12

Zadanie:

Cieľom zadania, je oboznámiť sa s pracou s mračnom bodov (point cloud) a segmentácia objektov v priestore. Študent si vyskúša vytvorenie vlastného mračna bodov a aplikáciu metód na získanie segmentovaného priestoru. Použitie externých knižníc ako open3d, sklearn, opencv, a iných je dovolené a odporúčané. Zadanie pozostáva z viacerých úloh:

1. Vytvorenie mračna bodov pomocou Kinect v2 pre testovanie. Nájdite online na webe mračno bodov popisujúce väčší priestor (väčší objem dát aspoň 4x4 metre) pre testovanie algoritmov a načítajte mračno dostupného datasetu (2B)
2. Pomocou knižnice (open3d - python) načítate vytvorené mračno bodov a zobrazíte. (2B)
3. Mračná bodov očistite od okrajových bodov. Pre tuto úlohu je vhodne použiť algoritmus RANSAC. (5B)
4. Segmentujete priestor do klastrov pomocou vhodne zvolených algoritmov (K-means, DBSCAN, BIRCH, Gausian mixture, mean shift ...). Treba si zvoliť aspoň 2 algoritmy a porovnať ich výsledky. (5+5B)
5. Detailne vysvetlite fungovanie zvolených algoritmov. (4B) (Keďže neimplementujete konkrétny algoritmus ale používate funkcie tretích strán je potrebné rozumieť aj ako sú funkcie implementované)
6. Vytvorte dokumentáciu zadania (popis implementovaných algoritmov, Grafické porovnanie výstupov, vysvetlite rozdiel v kvalite výstupov pre rozdielne typy algortimov) (2B).

Vytvorenie mračna bodov:

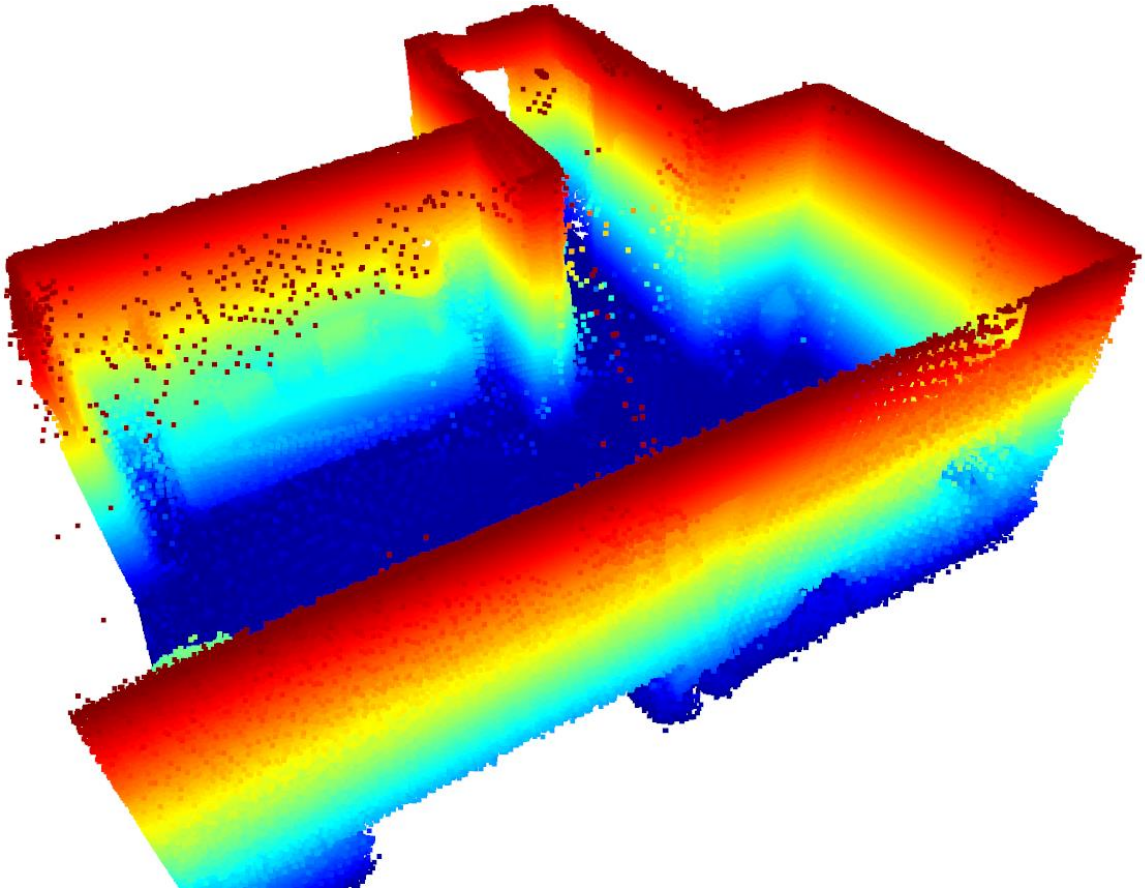
Vytvorili sme svoj vlastný pointcloud za pomoci Kinectu.

Obr. 1 Pointcloud za pomoci Kinectu

A taktiež sme našli pointcloud na internete:

Obr. 2: Pointcloud z internetu

Pointcloud z kinectu vieme bez problémov zobraziť v pythone, avšak internetový poitncloud sa zobrazuje s nesprávnymi farbami. Avšak aj na takomto pointcloude budeme vidieť efekty metódy RANSAC a vyskúšame aj metódy segmentácie.



Obr. 3: Internetový pointcloud v Pythone

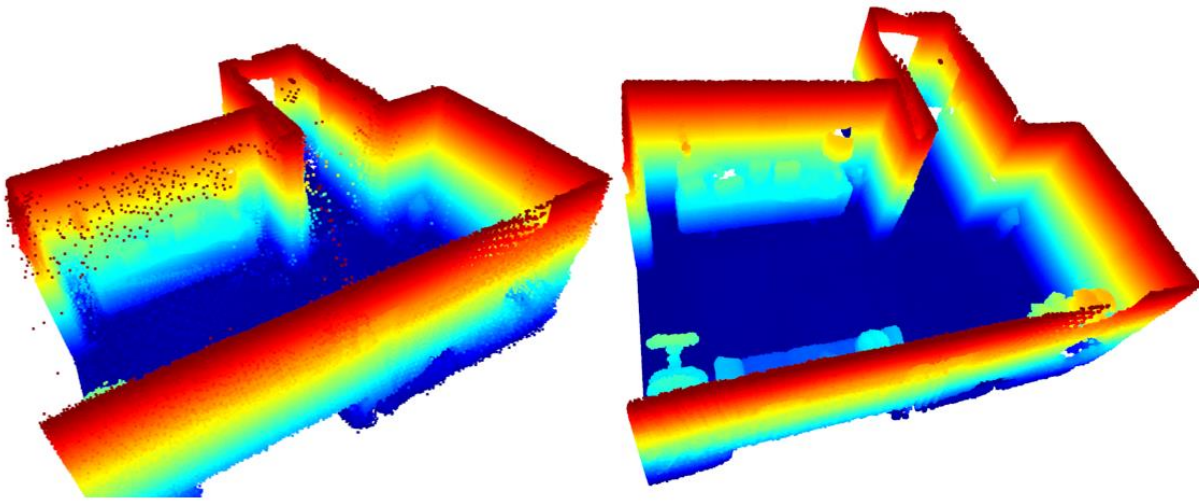
RANSAC:

RANSAC (Random Sample Consensus) je iteratívna metóda používaná na odhad modelov v prítomnosti šumu a odľahčených dát. Je široko používaná v počítačovom videní a spracovaní obrazu na odhad modelov z veľkého množstva dát obsahujúcich veľký počet odchýliek. RANSAC je obzvlášť užitočný pri práci s point cloudami, kde sú body získané z 3D skenovania alebo zobrazenia hĺbky, pretože tieto dáta často obsahujú šum a nepresnosti.

RANSAC algoritmus:

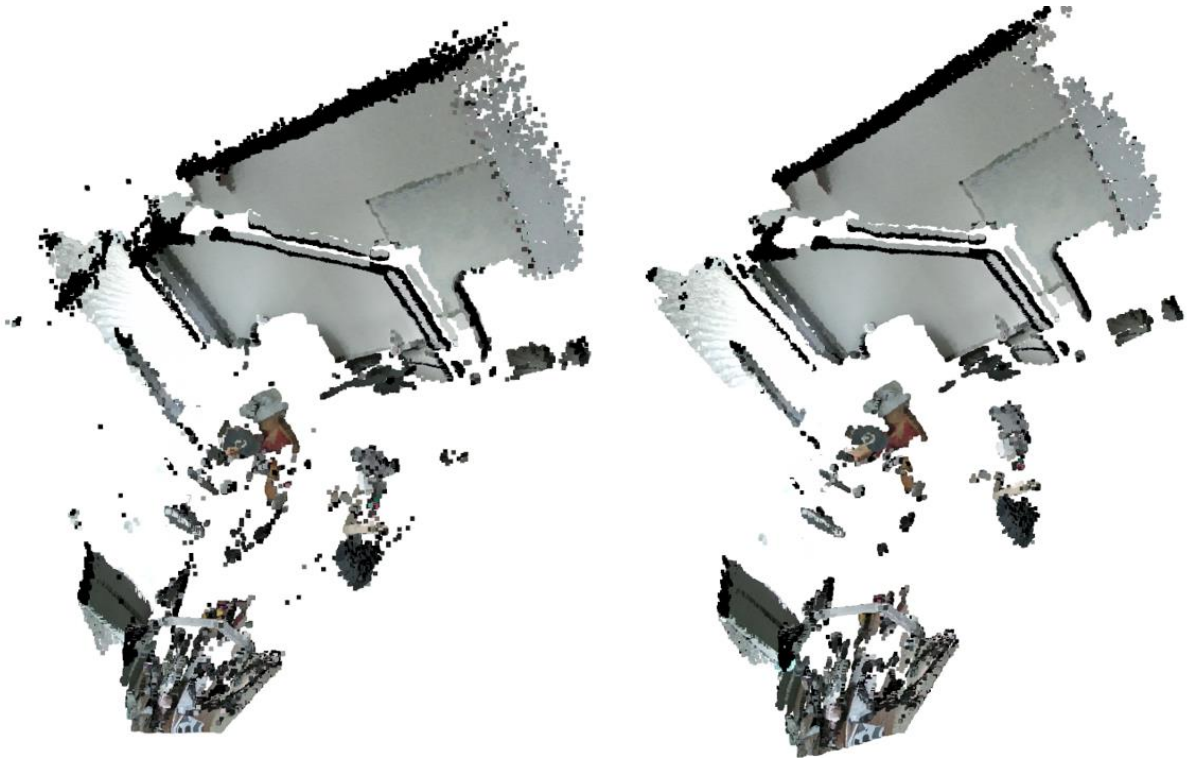
1. Náhodne vyberie minimálnu množinu bodov potrebných na odhad modelu (napr. priamka, rovina, kruh atď.) z vstupných dát.
2. Vypočíta model na základe tejto minimálnej množiny bodov.
3. Určí množinu inliereov, teda bodov, ktoré sú v súlade s odhadnutým modelom do určitej preddefinovanej tolerancie.
4. Opakuje kroky 1-3 preddefinovaný počet iterácií alebo pokiaľ nie je dosiahnutá požadovaná presnosť.
5. Výber modelu s najväčším počtom inliereov a jeho ďalšie vylepšenie pomocou týchto inliereov.

Výsledky RANSAC:



Obr. 4: Ransac aplikovany na pointcloud z internetu

Ako môžeme vidieť outlier body z fotky nalavo sa uz an fotke napravo neobjavujú.



Obr. 5: Ransac aplikovany na pointcloud z Kinectu

Ransac nám odstránil outlier body aj na fotke z Kinectu.

Segmentácia priestoru:

Na segmentáciu priestory sme si zvolily algoritmus Kmeans a algoritmus DBscan.

K-means

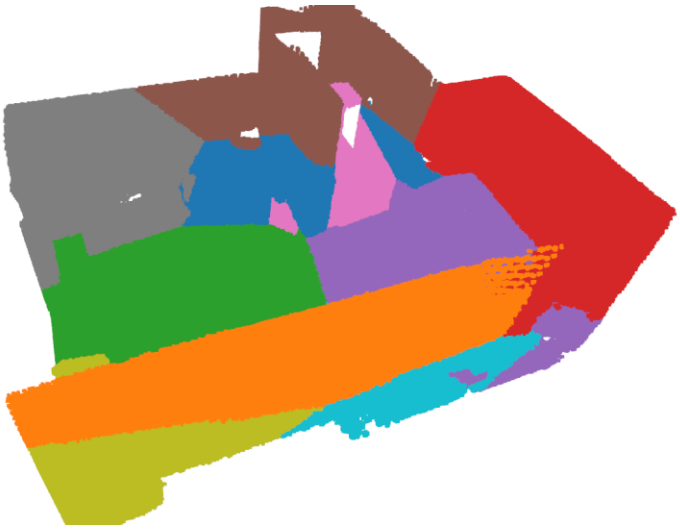
K-means je stredový zhlukovací algoritmus používaný na n-rozmerných dátach, ako sú pointcloudy. Algoritmus K-means funguje na základe iteratívneho priraďovania bodov k najbližším stredom zhlukov a následného výpočtu nových stredov zhlukov na základe priemerných hodnôt priradených bodov. Pri segmentácii pointcloudov je cieľom rozdeliť množinu bodov na zhluky, ktoré majú podobné vlastnosti, ako sú poloha, farba alebo normály.

Nasledujúce kroky opisujú, ako funguje algoritmus K-means pre segmentáciu pointcloudov:

1. Inicializácia: Určenie počtu zhlukov (K) a náhodný výber K počiatočných stredov z vstupného pointcloudu. Počet zhlukov môže byť zvolený na základe predchádzajúcych informácií o scéne alebo pomocou heuristík.
2. Priraďovanie bodov: Pre každý bod v pointcloude sa vypočítajú vzdialenosti ku všetkým stredom zhlukov. Každý bod sa priradí k zhluku, od ktorého stredu má najmenšiu vzdialenosť.
3. Aktualizácia stredov zhlukov: Pre každý zhluk sa vypočíta nový stred ako priemerná hodnota bodov priradených k tomuto zhluku. Nové stredy zhlukov sa stanú stredmi pre nasledujúce iterácie.
4. Konvergencia: Opakujú sa kroky 2 a 3, pokiaľ sa stredy zhlukov neprestanú meniť alebo nebudú menšie ako určená tolerancia. V tomto prípade sa algoritmus považuje za konvergovaný.
5. Výstup: Výsledkom je segmentovaný pointcloud, kde každý bod je priradený k jednému zo zhlukov.

K-means je jednoduchý a rýchly algoritmus, ale má niekoľko obmedzení, ako je citlivosť na výber počiatočných stredov zhlukov a potreba zadať počet zhlukov vopred. Alternatívne metódy, ako sú DBSCAN alebo HDBSCAN, sú schopné automaticky určiť počet zhlukov a sú menej citlivé na výber parametrov.

Výsledky Kmeans:



Obr. 6: Segmentovaný internet-pointcloud za pomoci Kmeans

Obr. 7: Segmentovaný kinect-pointcloud za pomoci Kmeans

Vidíme že algoritmus zlyháva v priestoroch, kde sú objekty blízko seba. Alebo kde sú veľké objekty (ako napríklad steny). Keď je napríklad v pointcloude 9 rôznych objektov a jedna veľká stena, a zadáme teda nech Kmeans zdeteguje 10 rôznych objektov, tak to zväčša dopadá tak, že na stene zdeteguje objektov niekoľko a ostatné menšie objekty potom spája do väčších celkov.

DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) je zhlukovací algoritmus založený na hustote. Na rozdiel od K-means, DBSCAN nevyžaduje zadanie počtu zhlukov vopred a dokáže sám zistiť počet zhluky rôznych tvarov a veľkostí.

DBSCAN funguje na základe dvoch kľúčových parametrov:

Epsilon (ε): Maximálna vzdialenosť medzi dvoma bodmi, ktoré sa považujú za susedné.

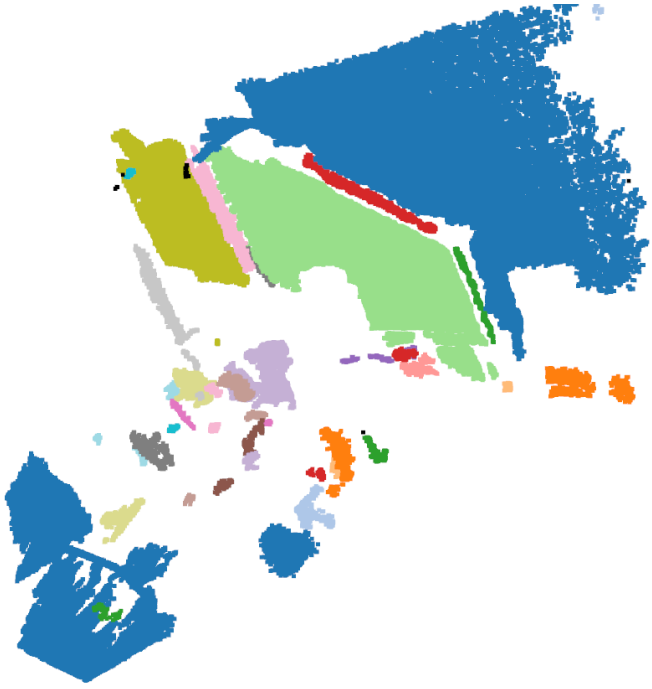
MinPts: Minimálny počet bodov, ktoré sú potrebné na vytvorenie zhluku.

Nasledujúce kroky opisujú, ako funguje algoritmus DBSCAN pre segmentáciu pointcloudov:

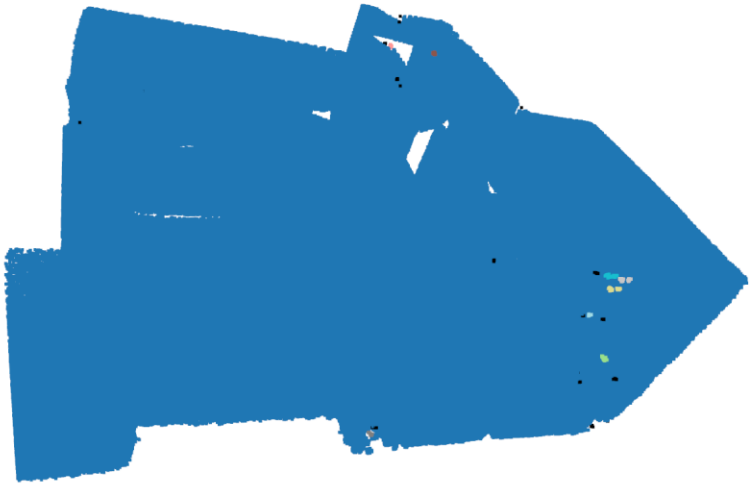
1. Pre každý bod v pointcloude sa zistia susedné body, ktoré ležia v guľovom okolí s polomerom ε.
2. Body sa klasifikujú ako jadro (core), hraničné body (border) alebo šum (noise) podľa nasledujúcich pravidiel:
    a. Jadro: Bod, ktorý má aspoň MinPts susedov vrátane seba v okolí s polomerom ε.
    b. Hraničné body: Bod, ktorý má menej ako MinPts susedov, ale aspoň jeden zo susedov je jadro.
    c. Šum: Bod, ktorý nie je ani jadro, ani hraničný.
3. Pre každé neoznačené jadro :
    a. Vytvorte nový zhluk a priraďte toto jadro k novému zhluku.
    b. Pre každý susedný bod jadra, ktorý ešte nebol priradený k žiadnemu zhluku, platí, že sa priradí k aktuálnemu zhluku a ak je tento bod tiež jadro, zopakuje sa tento krok pre všetkých jeho susedov.
4. Výstup: Výsledkom je segmentovaný pointcloud, kde každý bod je priradený k jednému zhluku alebo označený ako šum.

Výhody DBSCANu oproti iným zhlukovacím metódam, ako je K-means, je schopnosť zistiť zhluky rôznych tvarov a veľkostí, automaticky určiť počet zhlukov a rozlíšiť šum od zhlukov. Nevýhodou je, že DBSCAN môže byť citlivý na voľbu parametrov ε a MinPts, čo môže mať významný vplyv na výsledky segmentácie.

Výsledky DBSCAN:



Obr. 8: Segmentovaný kinect-pointcloud za pomoci DBSCAN



Obr. 9: Segmentovaný internet-pointcloud za pomoci DBSCAN

Ako môžeme vidieť pointcloud z kinectu bol pekne segmentovaný, Takýto výsledok bol dosiahnutý aj vďaka tomu že jednotlivé zhluky bodov majú medzi sebou jasné medzery a tým pádom DBSCAN vie určiť, kde daný zhluk končí a kde začína nový. Na poitncloude z internetu žiaľ DBSCAN spravil veľmi zlú segmentáciu. Toto je zapríčinené práve tým že pointcloud je jeden veľký prepojený objekt a tým pádom DBSCAN deteguje len jeden veľký zhluk bodov.

Záver:

Vygenerovali sme si pointcloud za pomoci Kinectu v2 a taktiež sme našli jeden pointcloud na interenete. Pomocou knižnice open3d sme si vedeli tieto pointcloudy vyzualizovať, avšak pointcloud z internetu sa nevyzualizoval so správnymi farbami.

Oba pointcloudy sme očistili od outlier bodov za pomoci metódy RANSAC. Tento prístup fungoval veľmi pekne pri oboch poitncloudoch.

Následne sme segmentovali vyfiltrované pointcloudy. Na túto úlohu sme si zvolili metódu DBSCAN a metódu K-means. Princíp oboch funkcií sme detailne popísali.

Metóda K-means ma mnoho obmedzení a edge-casov kedy zlyháva. Toto je dôvod prečo segmentácia oboch pointcloudov nedopadla tak ako by sme chceli. Avšak na demonštráciu toho ako táto metóda funguje je tento výsledok dostačujúci.

Metóda DBSCAN veľmi pekne segmentovala pointclou z Kinectu, keďže medzi jednotlivými objektami boli predelenia v ktorých neležali body. Pri pointcloudu z internetu zlyhala práve z toho dôvodu že pointcloud interpretovala ako jeden veľký jednoliaty objekt.

Bibliografia

[1] **Pavlovičová, Jarmila.** *Prednášky z PVSO.* Bratislava : s.n., 2023.

[2] **scikit.** KMeans. *scikit-learn.* [Online] April 2023. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html.

[3] **scikit-learn.** DBSCAN. *scikit-learn.org.* [Online] April 2023. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html.

[4] **Stachniss, Cyrill.** RANSAC - 5 Minutes with Cyrill. *Youtube.* [Online] 2023 April. https://www.youtube.com/watch?v=9D5rrtCC_E0.

[5] **Wikipedia.** DBSCAN. *Wikipedia.* [Online] April 2023. https://en.wikipedia.org/wiki/DBSCAN.

[6] —. k-means clustering. *Wikipedia.* [Online] April 2023. https://en.wikipedia.org/wiki/K-means_clustering.