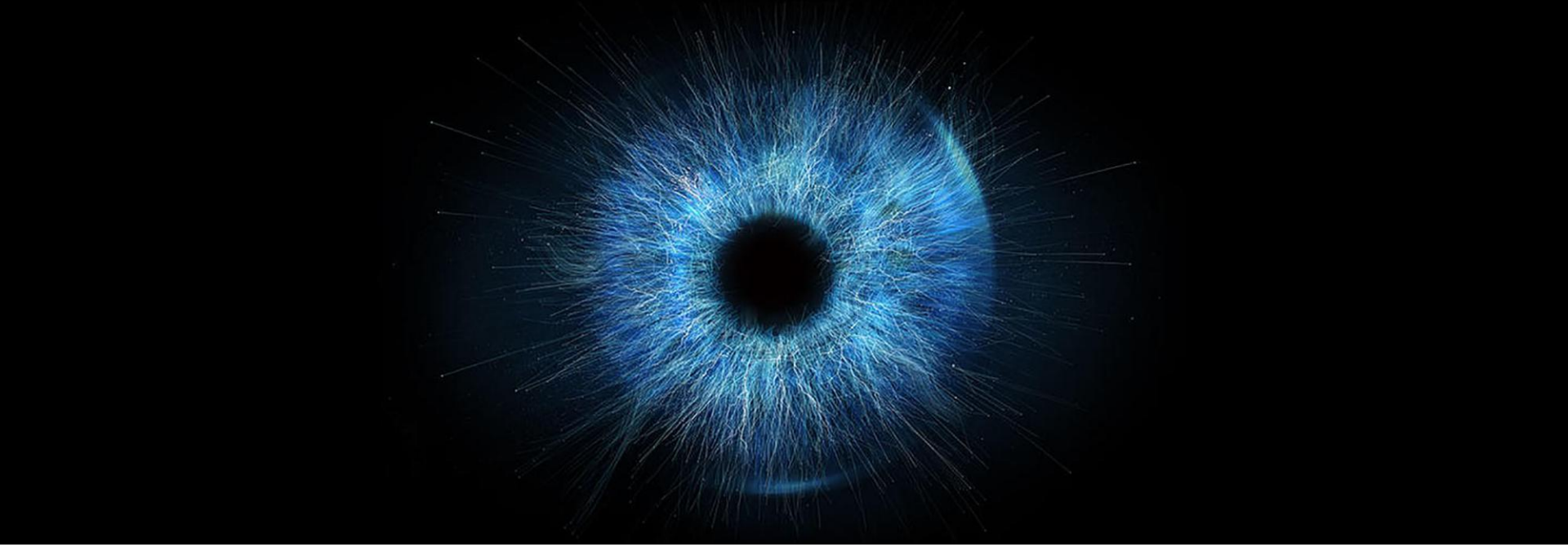


Napredna obrada PC-a
(case study – detekcija i manipulacija objektima)

Tara Knežević, mag.ing.mech.
Doc.dr.sc. Filip Šuligoj

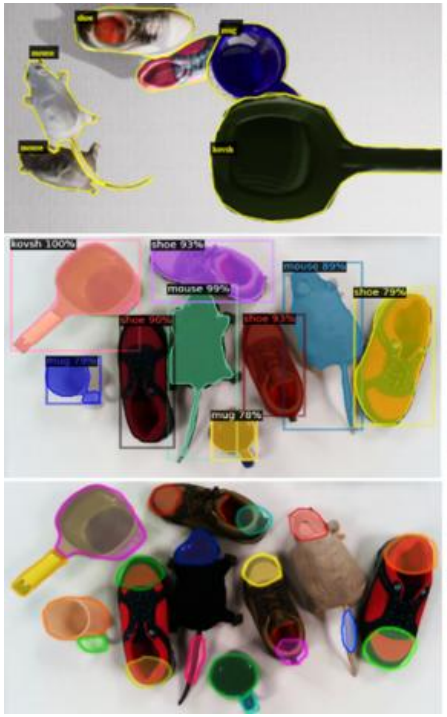
University of Zagreb

FSB Faculty of mechanical engineering and naval arhitecture

Detekcija i manipulacija objektima

- Nije cilj samo **'vidjeti'** objekt, već razumjeti **što gledamo**, **gdje** se nalazi i **što** s njim treba **napraviti**

DETEKCIJA



MANIPULACIJA

Detekcija i manipulacija objektima

DETEKCIJA

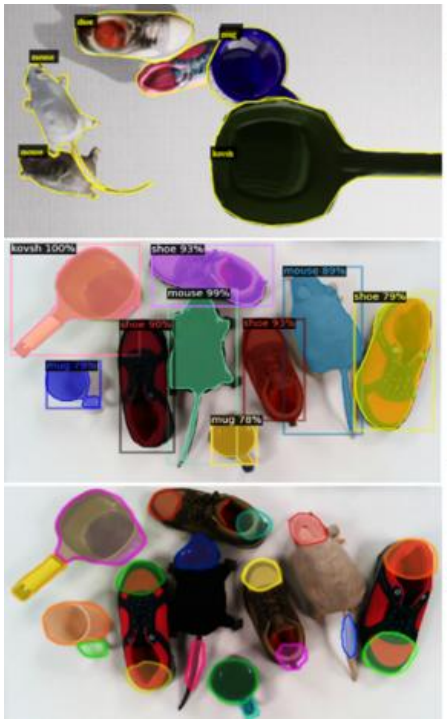
DETEKCIJA je sposobnost robota (ili računala) da na slici prepozna **što** se na njoj nalazi, **gdje** se nalazi, i eventualno **kojeg** je **tipa**

Detekcija i manipulacija objektima

MANIPULACIJA je sposobnost robota da **fizički upravlja objektom** – da ga uhvati, pomakne, rotira, obradi ili sastavi

Detekcija i manipulacija objektima

DETEKCIJA



MANIPULACIJA

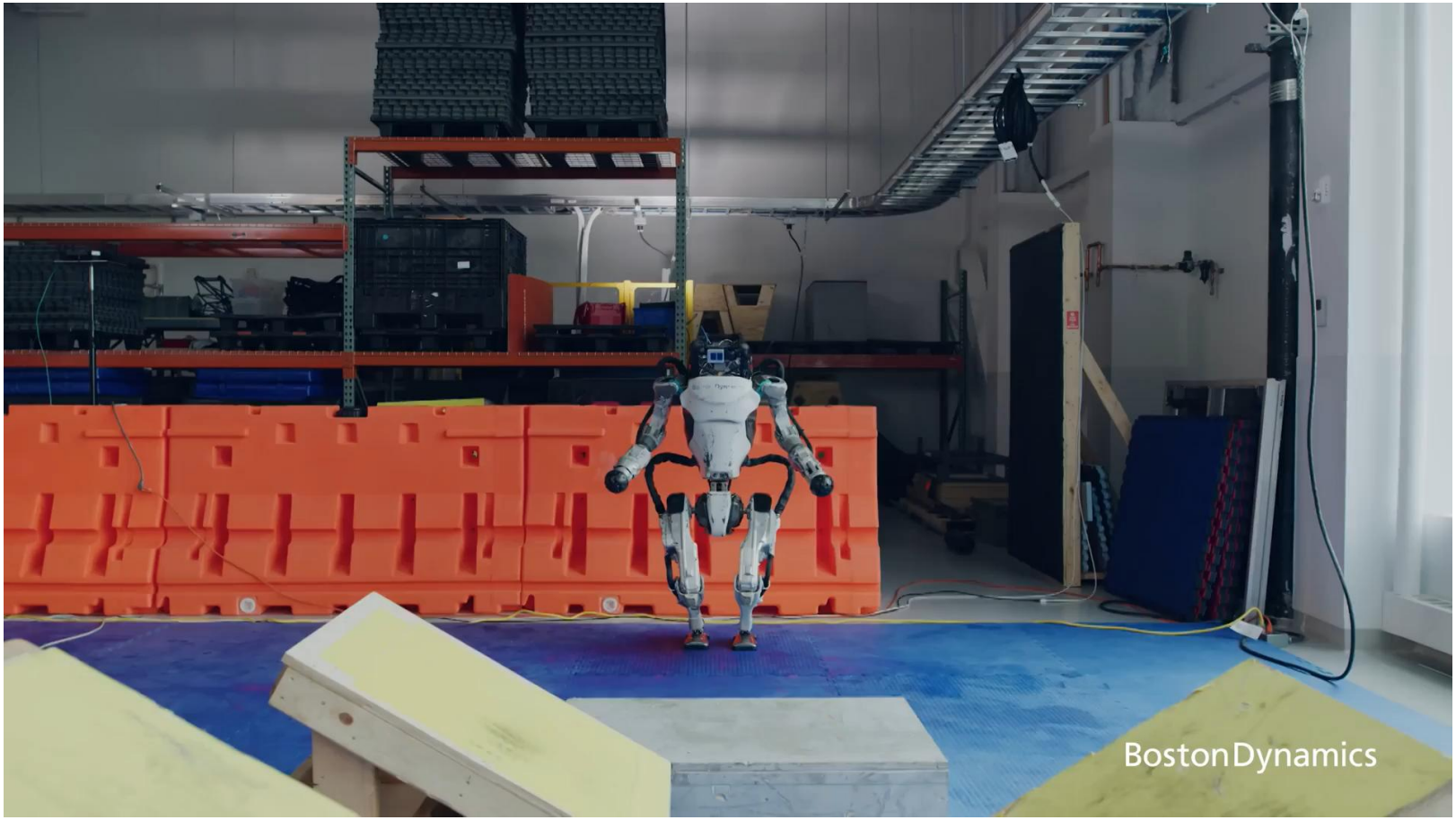
- Vizualna percepcija i fizička manipulacija moraju raditi zajedno da bi robot mogao učinkovito obavljati **zadatke u stvarnom svijetu**

Zašto nam trebaju robusni sustavi percepcije i manipulacije?

Što radimo robusno

Zašto nam trebaju robusni sustavi percepcije i manipulacije?

Što želimo raditi robusno



Zašto nam trebaju robusni sustavi percepcije i manipulacije?

Realnost – istraživanje

Zašto nam trebaju robusni sustavi percepcije i manipulacije?

Realnost – industrija

Zašto nam trebaju robusni sustavi percepcije i manipulacije?

- Automatizacija u industriji više nije stvar samo brzine ili ponavljanja, već sposobnosti da se robot snađe u **nepredvidivim, nestrukturiranim okruženjima**
- **Bez dobre percepcije nema dobre manipulacije** - ako robot "ne zna" što vidi i kako da to uzme, sav mehanički hardver je beskoristan

Seeker projekt

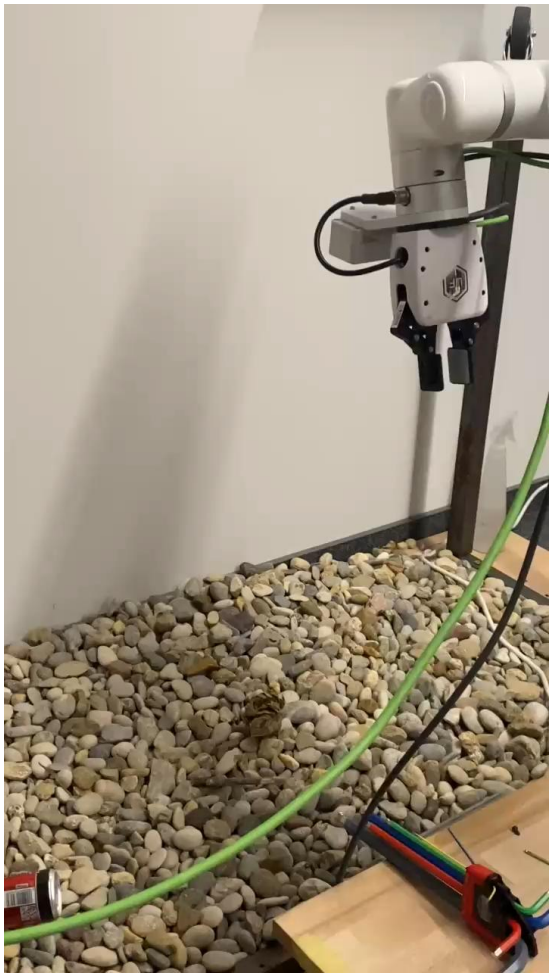
Robot dizajniran za rad u vanjskim uvjetima s primjenom prikupljanja različitih vrsta otpada - uključujući plastiku, organski otpad i druge zagađivače

Projekt se fokusira na razvoj:

- robusnih pogonskih sustava
- sustava ovjesa
- sposobnosti autonomne vožnje
- **integracije senzora za detekciju otpada**
- **upravljačkih algoritama robotskih ruku**

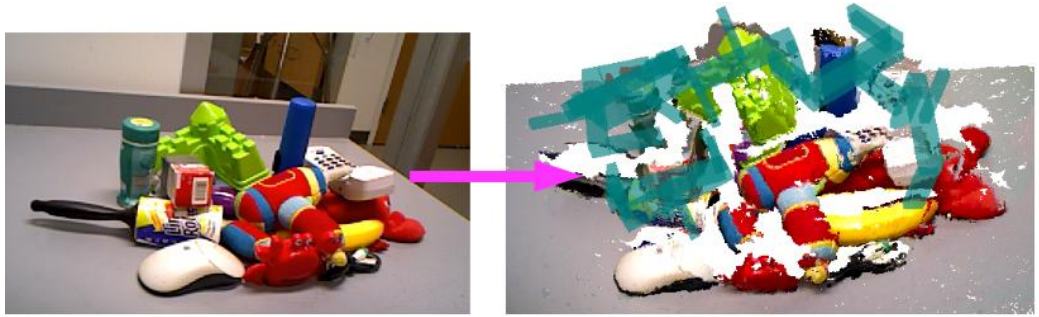
Seeker projekt

Seeker projekt

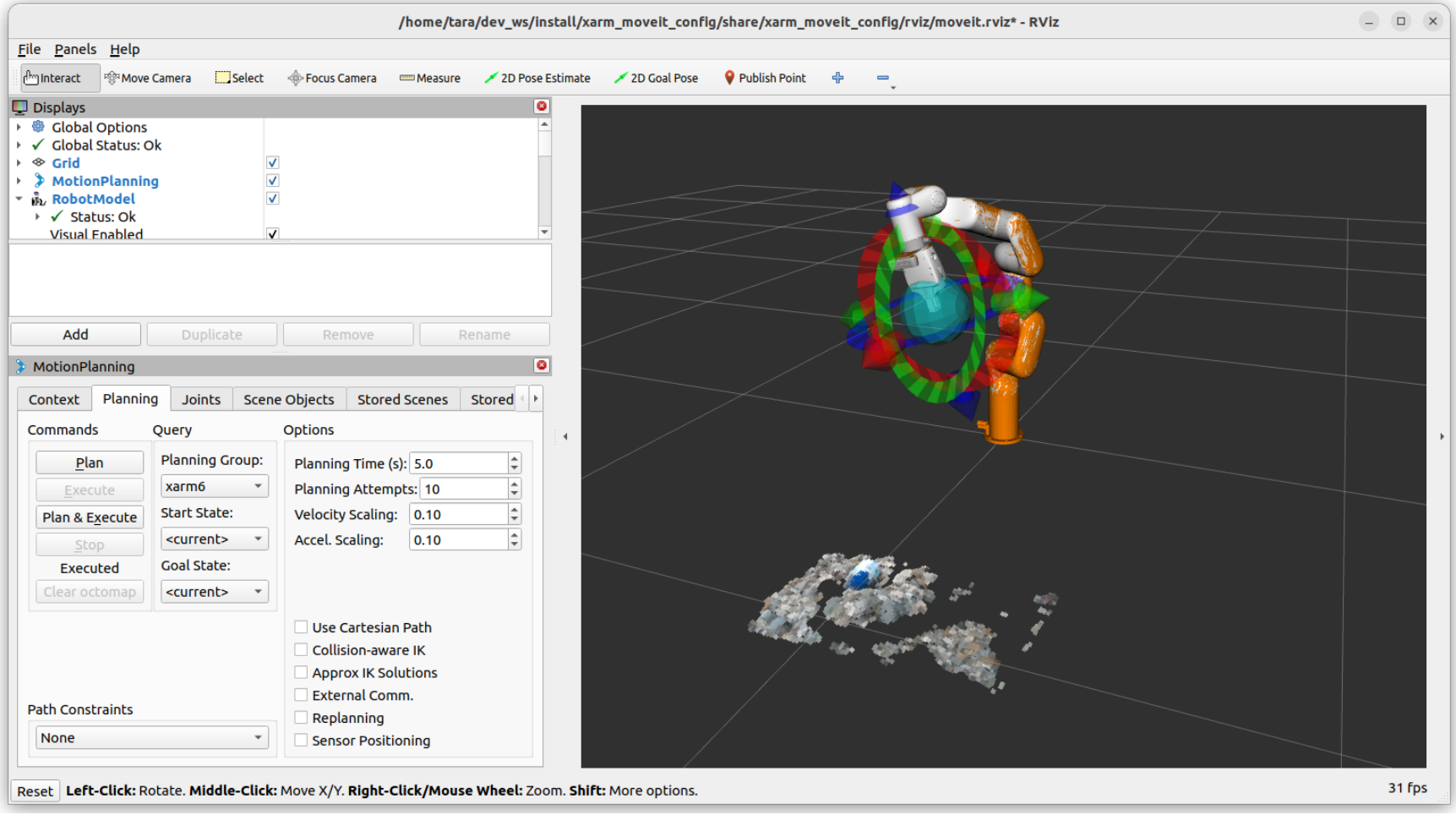


U projektu korišteni su:

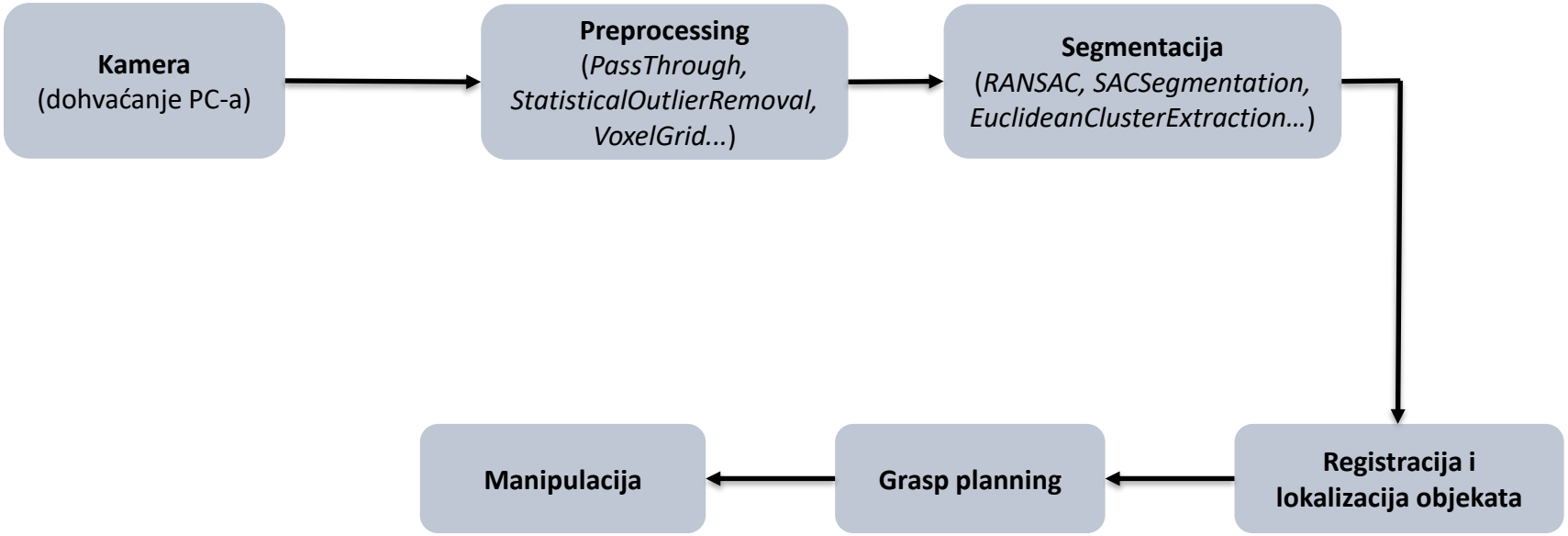
- ROS2 (Humble, C++ kod)
- PCL
- Grasp Pose Detection (GPD) - https://github.com/atenpas/gpd

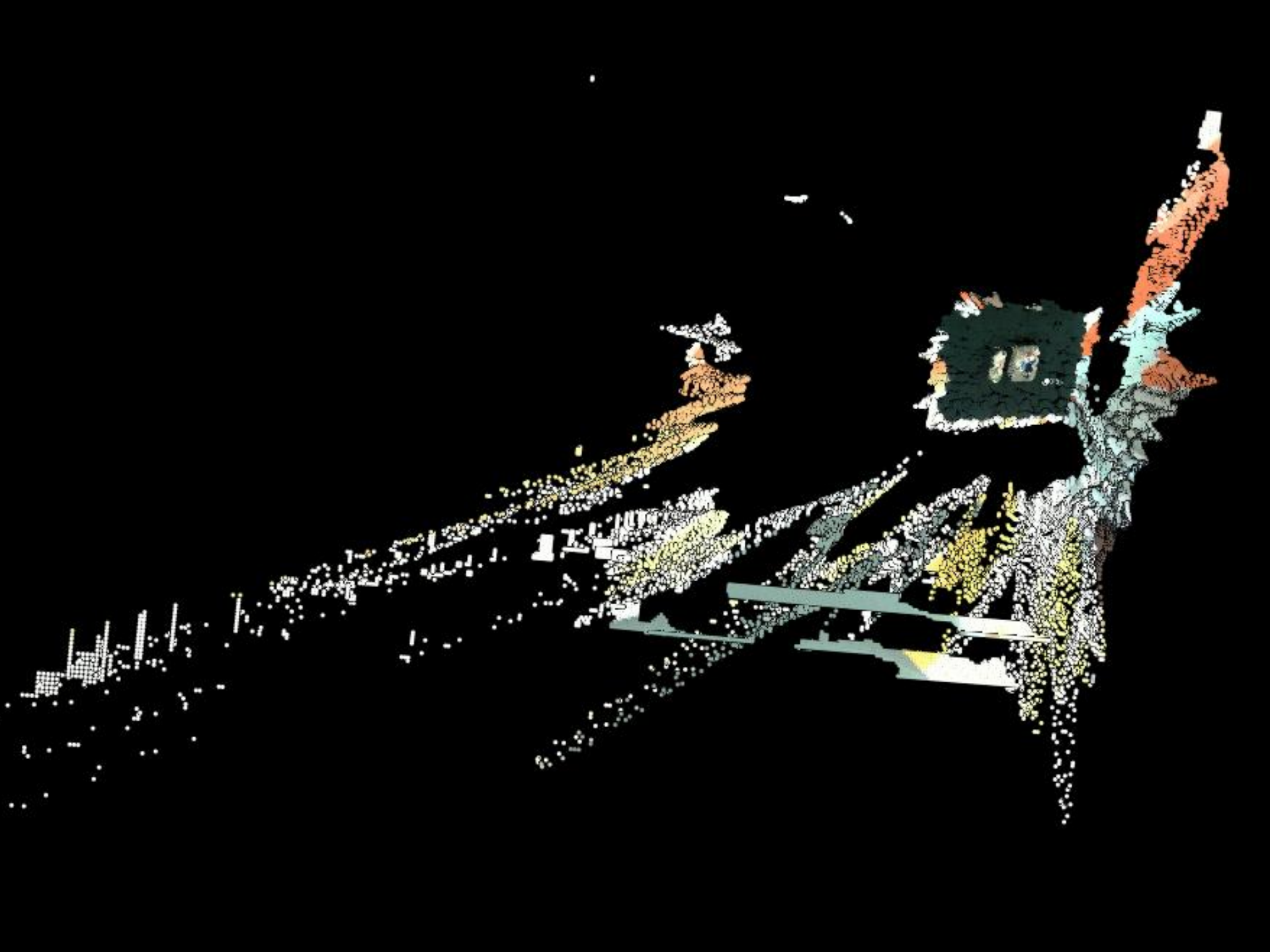


Seeker projekt



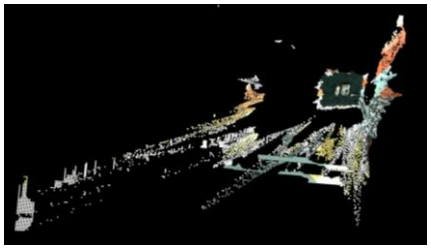
Perceptivno-manipulacijski pipeline





Perceptivno-manipulacijski pipeline

Kamera
(dohvaćanje PC-a)



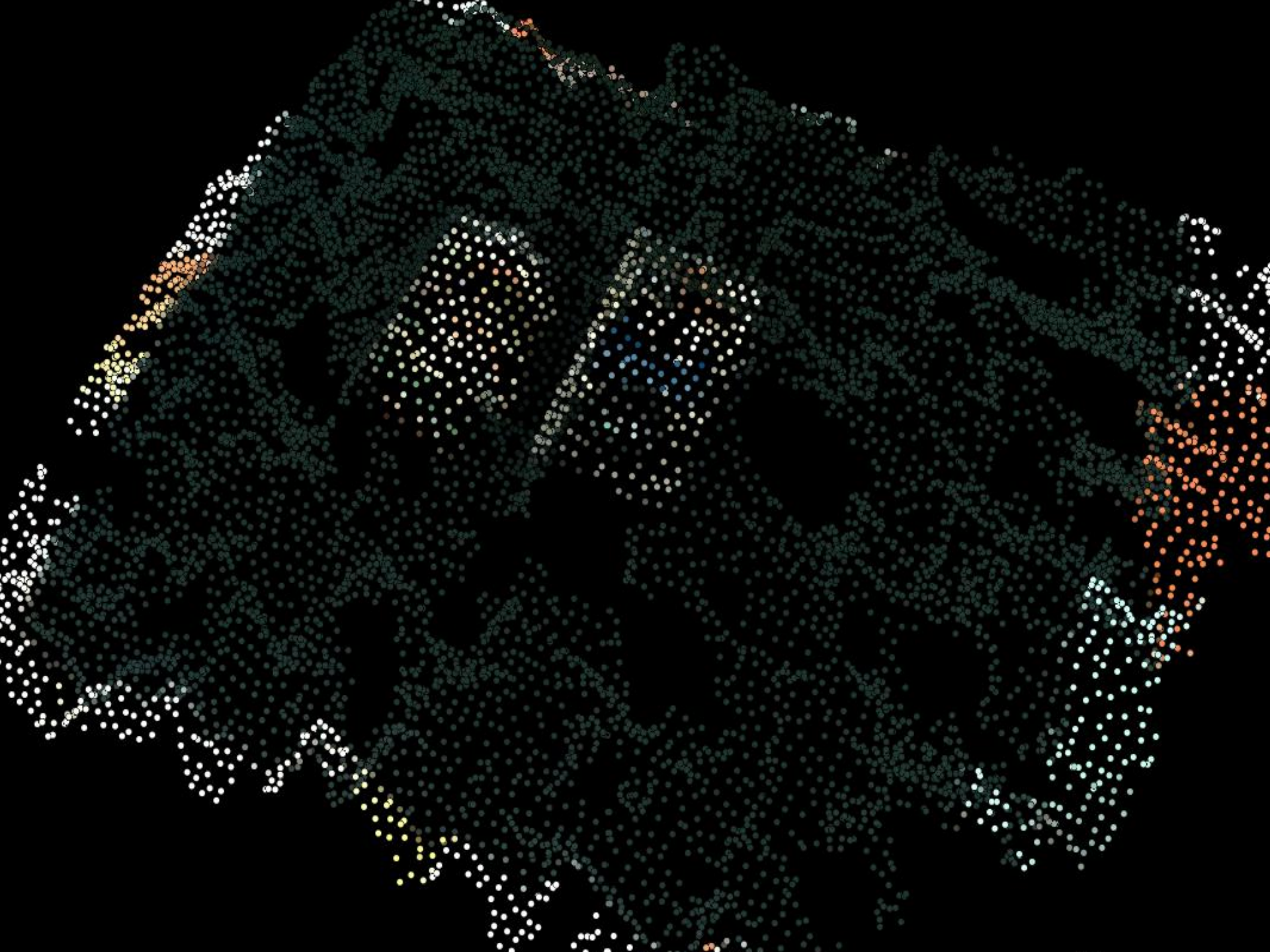
Preprocessing
(*PassThrough, StatisticalOutlierRemoval, VoxelGrid...*)

Segmentacija
(*RANSAC, SACSegmentation, EuclideanClusterExtraction…*)

Registracija i lokalizacija objekata

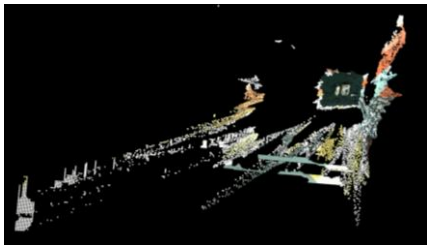
Grasp planning

Manipulacija

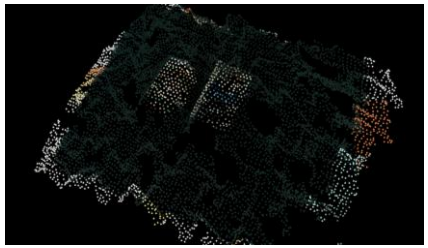


Perceptivno-manipulacijski pipeline

Kamera
(dohvaćanje PC-a)



Preprocessing
(*PassThrough, StatisticalOutlierRemoval, VoxelGrid...*)



Segmentacija
(*RANSAC, SACSegmentation, EuclideanClusterExtraction…*)

Registracija i lokalizacija objekata

Grasp planning

Manipulacija

Segmentacija

- Segmentacija je **proces podjele oblaka točaka** u značajne regije koje predstavljaju dijelove scene ili različite objekte
- **Glavni cilj:** grupirati točke koje imaju slična geometrijska ili semantička svojstva, kako bi se olakšala daljnja obrada
- Kako radi?
  - Definira se neki model ili pravilo
  - Odabiru se točke koje taj uvjet zadovoljavaju i prihvaćaju se kao segment
  - Segmentirane točke izdvajaju se iz PC-a

Segmentacija

- Česte primjene:
  - Odvajanje objekta od pozadine (npr. šalice sa stola)
  - Razdvajanje više objekata u istoj sceni
  - Priprema za registraciju ili hvatanje robota
  - Identifikacija geometrijskih primitiva (npr. ravnina → pod ili zid)

Segmentacija

- Vrste segmentacije u PCL-u:
  - **Geometrijska segmentacija** - koristi se kad se zna što se traži u sceni (ravnina, cilindar, sfera)
  - **Klasterizacija** (clustering) - najčešći pristup za "neka tijela koja nisu spojena"
  - Segmentacija temeljem boje
  - Region Growing - algoritam koji koristi vektore normala i udaljenost za "spajanje" sličnih regija
  - Supervoxel segmentacija - napredna metoda koja koristi voxelizaciju, normalne, boju i kompaktnost. Dobar za složenije scene.

RANSAC (*RANdom SAmple Consensus*) segmentacija

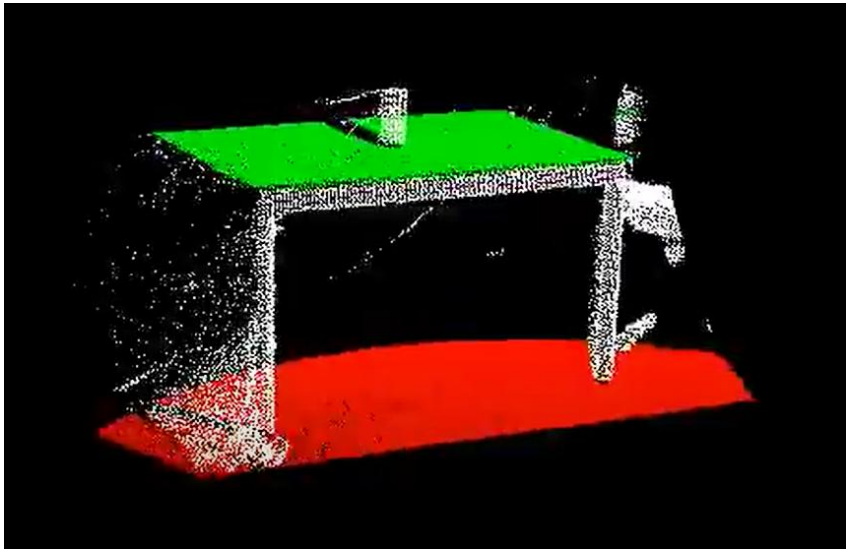
- RANSAC je robustan algoritam za **detekciju geometrijskih modela** (npr. ravnina, cilindar, sfera) u prisutnosti šuma i outliera
- Koristi se kad želiš pronaći **jednostavne geometrijske oblike** u oblaku točaka

RANSAC (*RANdom SAmple Consensus*) segmentacija

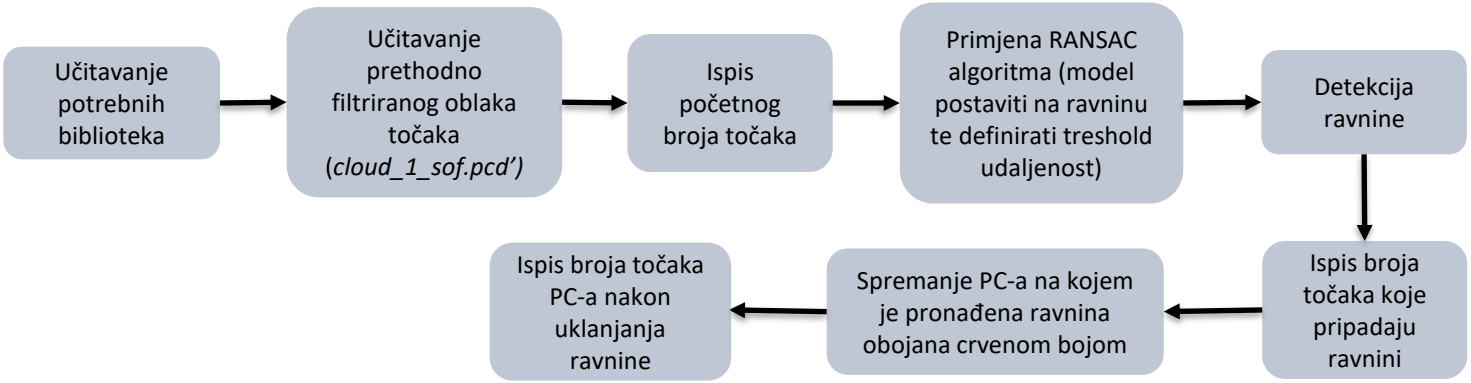
- Kako radi?
  - Nasumično odaberi mali broj točaka potrebnih za model (npr. 3 točke za ravninu)
  - Izračunaj model (npr. jednadžbu ravnine) iz tih točaka
  - Provjeri sve ostale točke – koliko ih "pripada" modelu (tj. udaljenost od modela < prag)
  - Broji inliere (točke koje pristaju modelu)
  - Ponavljaj više puta (npr. 1000 iteracija) → svaki put pokušaj s novim uzorkom
  - Zadrži model s najviše inliera → to je najbolji "fit"

RANSAC (*RANdom SAmple Consensus*) segmentacija

- Česte primjene:
  - Detekcija stola (ravnine) u sceni
  - Uklanjanje pozadine (npr. pod, zid)
  - Priprema za segmentaciju pojedinačnih objekata

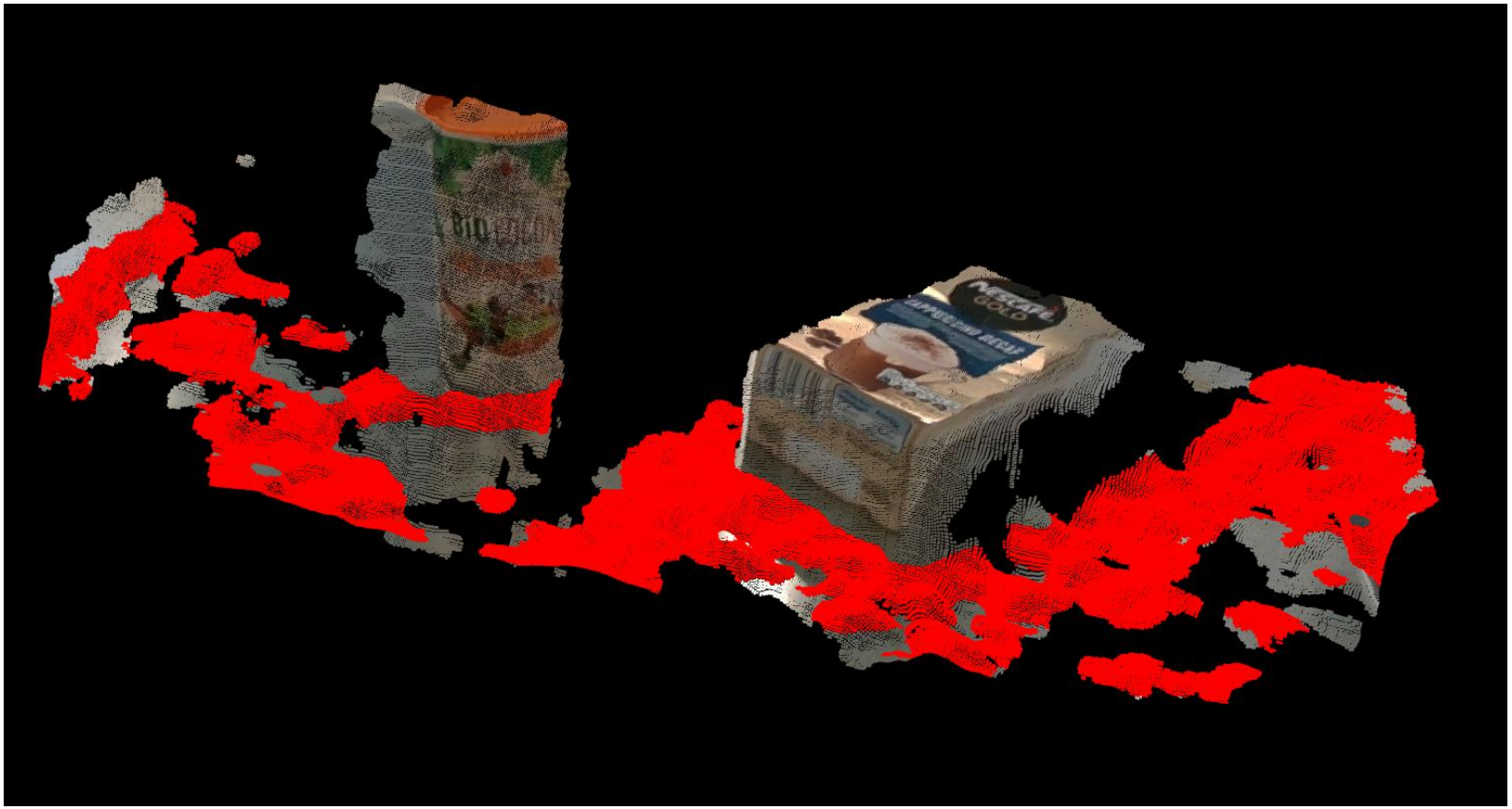


Primjer 07 – RANSAC – pronalazak ravnine



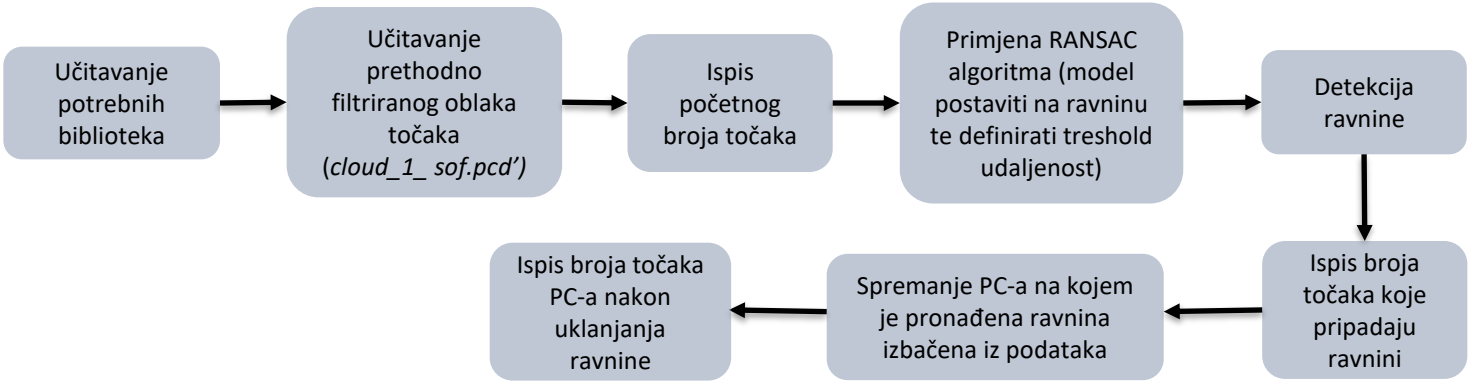
- U primjeru je threshold udaljenosti postavljen na 1 cm
- Model tip je ravnina

Primjer 07 – RANSAC – pronalazak ravnine - vizualizacija



Obrađeni podaci: asv_dataset/scene_01/cloud_1.pcd

Primjer 08 – RANSAC – brisanje ravnine

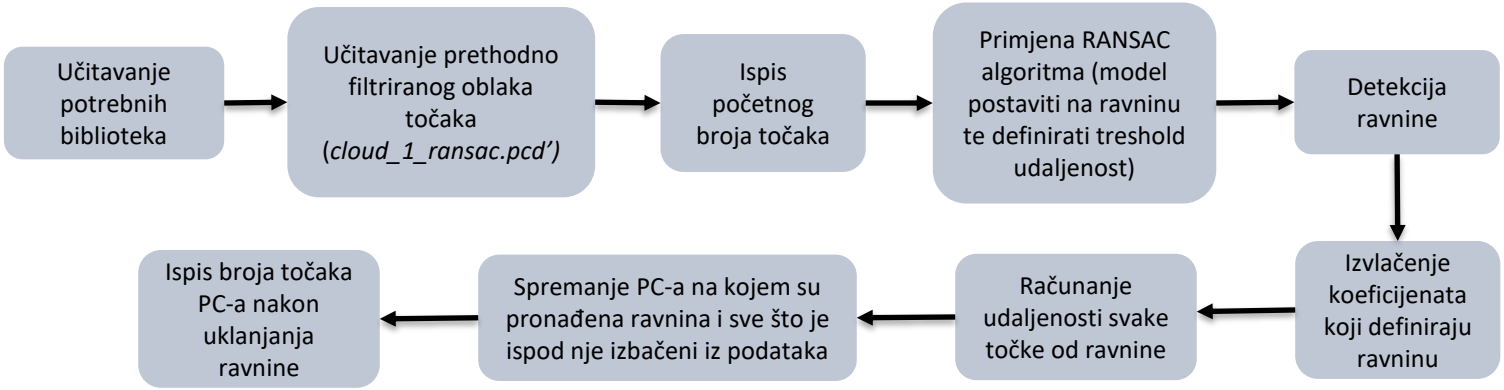


- U primjeru je threshold udaljenosti postavljen na 1 cm
- Model tip je ravnina

Primjer 08 – RANSAC – brisanje ravnine - vizualizacija

Obrađeni podaci: asv_dataset/scene_01/cloud_1.pcd

Primjer 09 – RANSAC – estimacija parametara ravnine i brisanje podataka ispod nje



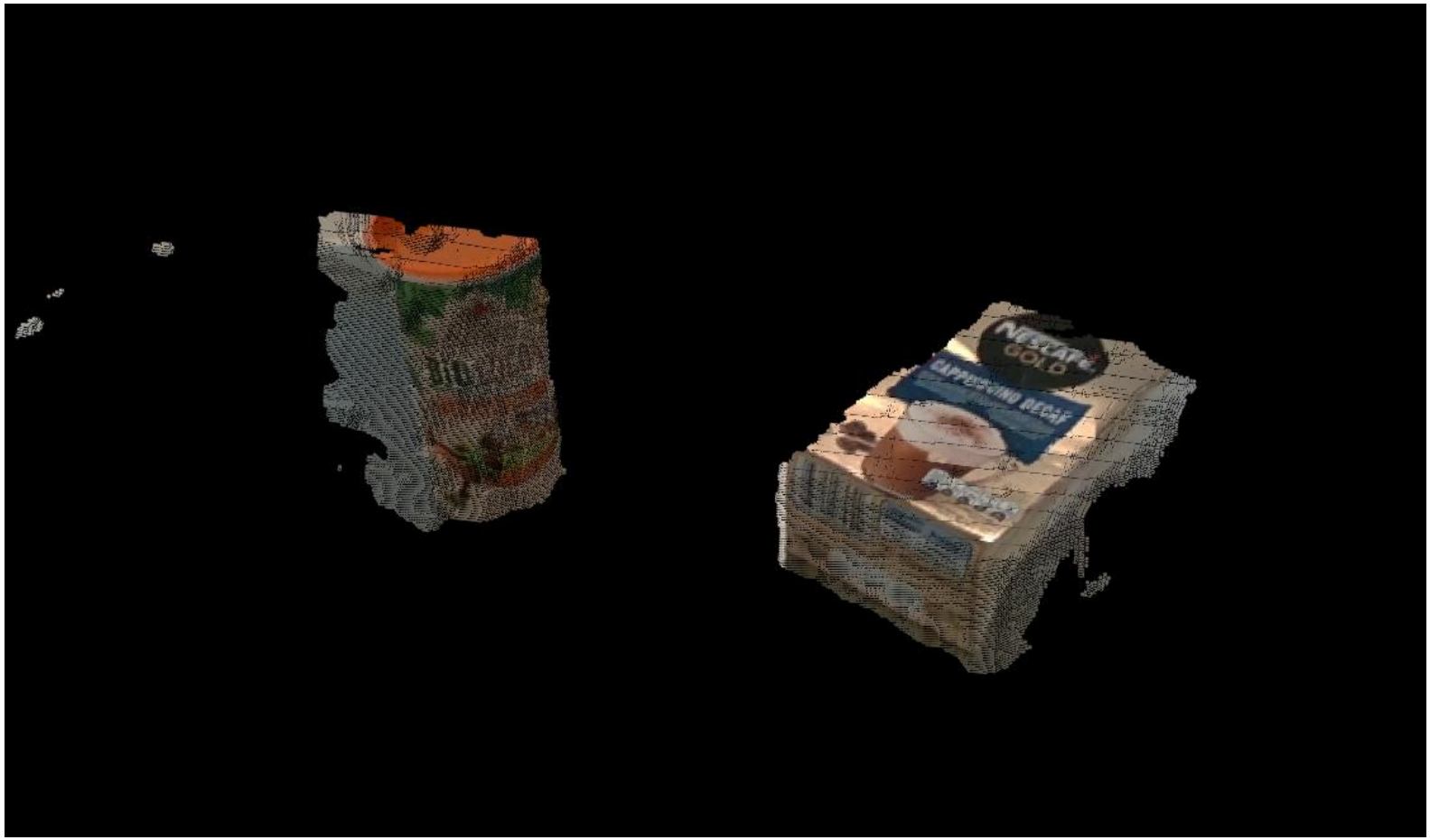
- Koeficijenti koje dobijemo su: a, b, c, d – za njih vrijedi sljedeći izraz (pri izračunu udaljenosti od ravnine):

$$udaljenost = ax + by + cz + d$$

- Sve točke za koje vrijedi sljedeći izraz izbacuju se s PC-a

$$udaljenost \geq 0$$

Primjer 09 – RANSAC – estimacija parametara ravnine i brisanje podataka ispod nje - vizualizacija



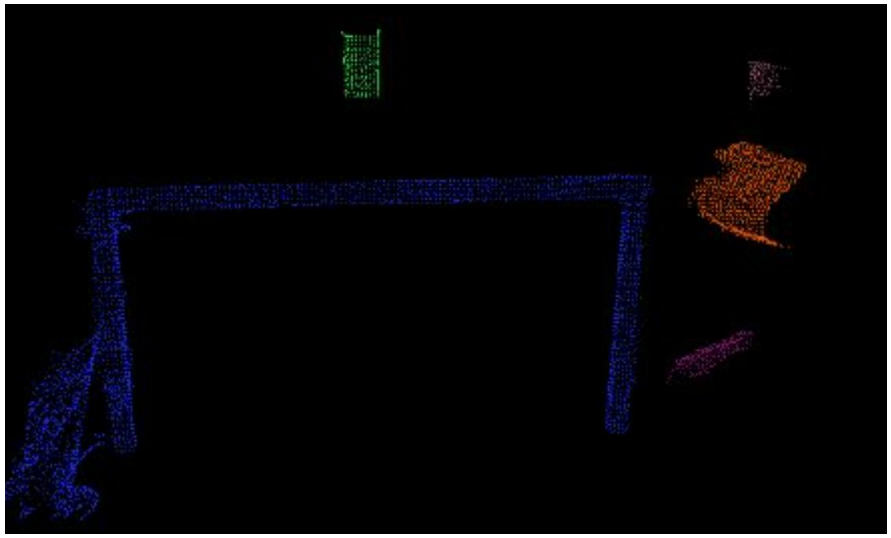
Obrađeni podaci: asv_dataset/scene_01/cloud_1.pcd

Klasterizacija - EuclideanClusterExtraction

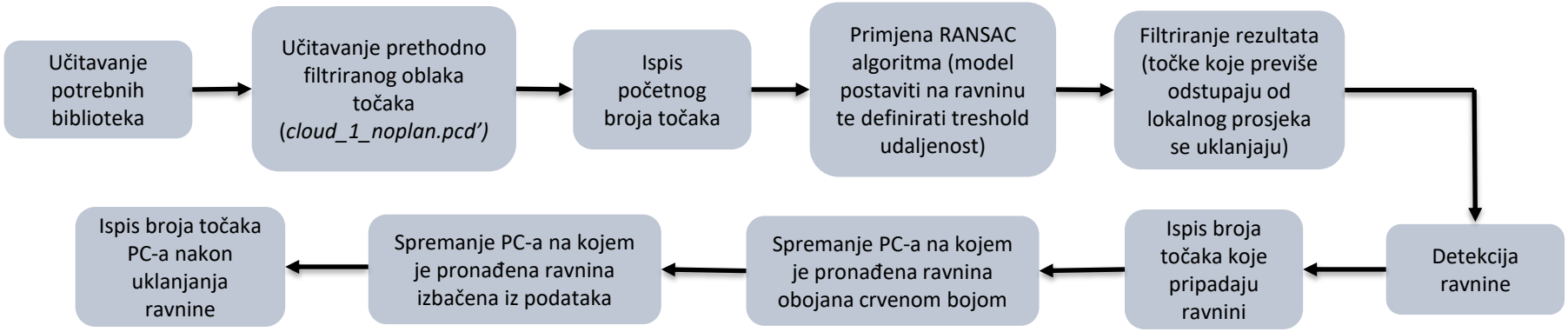
- Klasterizacija je proces **grupiranja točaka** u zasebne objekte na temelju njihove blizine u prostoru
- Svaki klaster se tretira kao potencijalni pojedinačni objekt
- Kako radi?
  - Pretraživanje susjeda (npr. KdTree)
  - Iterativno grupiranje točaka koje su međusobno udaljene manje od zadanog tresholda
  - Svaka povezana grupa točaka postaje novi klaster
  - Klasteri se mogu vizualizirati u različitim bojama ili spremiti kao posebni PC-ovi

Klasterizacija - EuclideanClusterExtraction

- Česte primjene:
  - Detekcija i odvajanje objekata
  - Uklanjanje preostalih točaka nakon uklanjanja pozadine
  - Priprema za klasifikaciju ili grasp planning pojedinačnih objekata

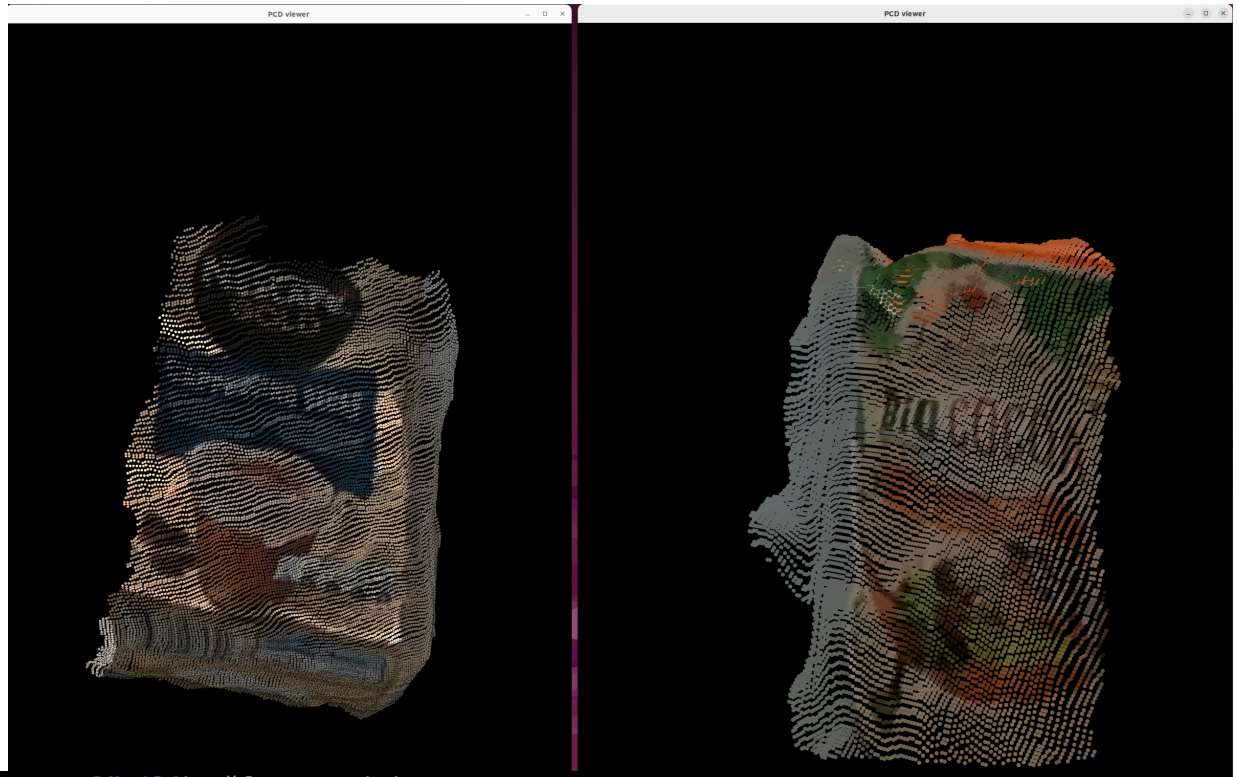


Primjer 10 – EuclideanClusterExtraction



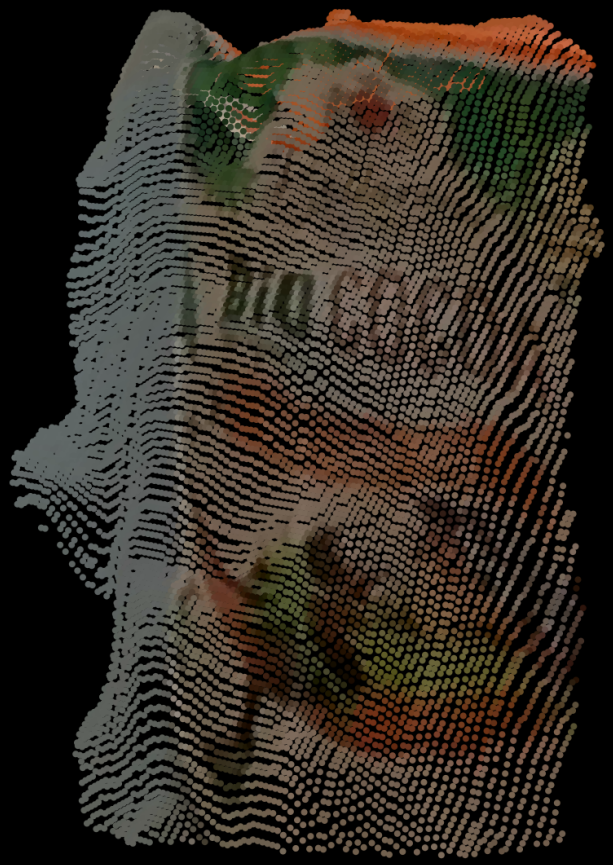
- Maksimalna udaljenost između dvije točke da bi pripadale istom klasteru – 2 cm
- Minimalni broj točaka u klasteru – 1000
- Maksimalni broj točaka u klasteru – 25000
- Sve manje ili veće od ovih granica – izbacuje iz PC-a automatski

Primjer 10 – EuclideanClusterExtraction - vizualizacija



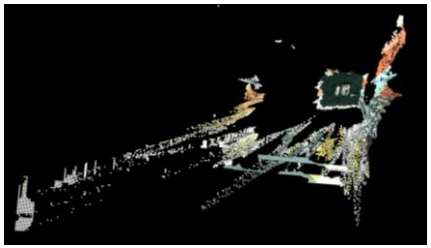

```
asvtara@asvtara-virtual-machine:~/pcl_ws_prof/build$ ./klaster_primjer
Učitano točaka: 30888
Broj pronađenih klastera: 2
Spremljen klaster #0 s 18751 točaka u: /home/asvtara/pcl_ws_prof/data/asv_dataset/scen
e_01_obradeno/cloud_1_klaster_1.pcd
Spremljen klaster #1 s 12080 točaka u: /home/asvtara/pcl_ws_prof/data/asv_dataset/scen
e_01_obradeno/cloud_1_klaster_2.pcd
```

Obrađeni podaci: asv_dataset/scene_01/cloud_1.pcd

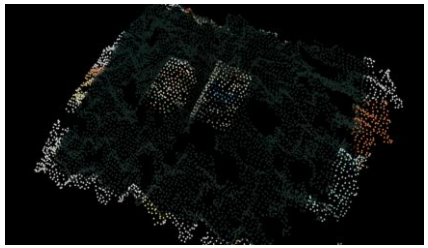


Perceptivno-manipulacijski pipeline

Kamera
(dohvaćanje PC-a)



Preprocessing
(*PassThrough, StatisticalOutlierRemoval, VoxelGrid...*)



Segmentacija
(*RANSAC, SACSegmentation, EuclideanClusterExtraction…*)

Registracija i lokalizacija objekata

Grasp planning

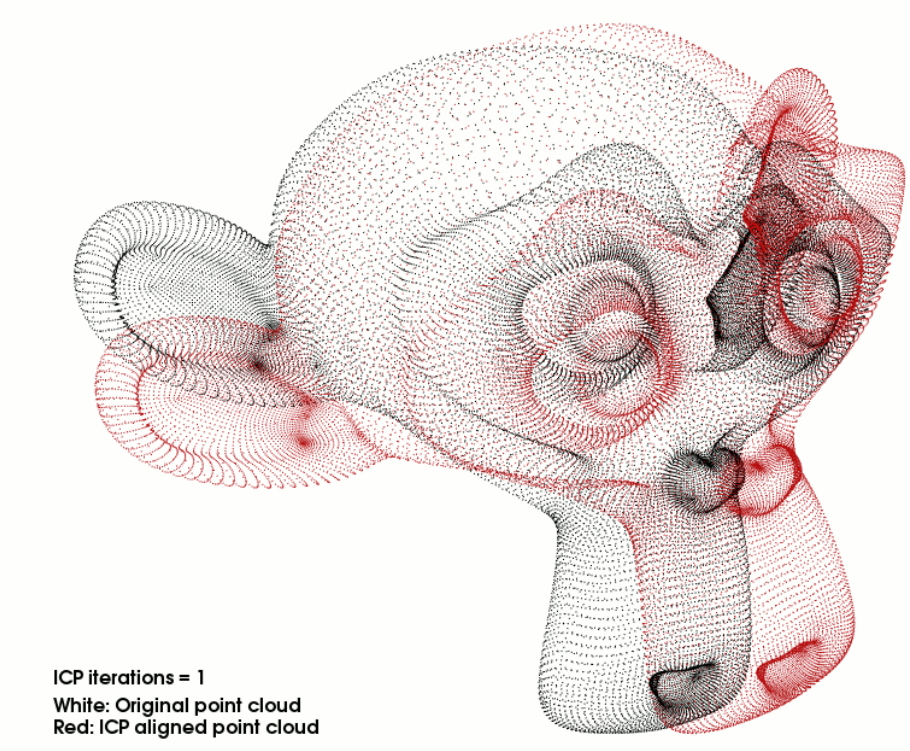
Manipulacija

Registracija objekata

- Registracija je proces **poravnanja** dvaju ili više oblaka točaka u **isti koordinatni sustav**
- Cilj je dovesti ih u prostor gdje se njihovi oblici **poklapaju**, te na taj način **obogaćujemo informacije** o sceni koju promatramo
- **Zašto se koristi?**
  - Spajanje više snimki iz različitih pozicija (npr. s kamere na robotu)
  - Usporedba scene s poznatim modelom (npr. detekcija objekta)

Registracija objekata

- Česte primjene:
  - Stitchanje više pogleda u jedan 3D model
  - Detekcija poznatog objekta u sceni
  - 3D mapiranje prostora
  - Rekonstrukcija predmeta



Transformacija PC-a

- Transformacija oblaka točaka vrši se pomoću **homogene matrice transformacija**, koja kombinira rotaciju i translaciju $\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}$, gdje vrijedi – **R** = matrica, **t** = vektor

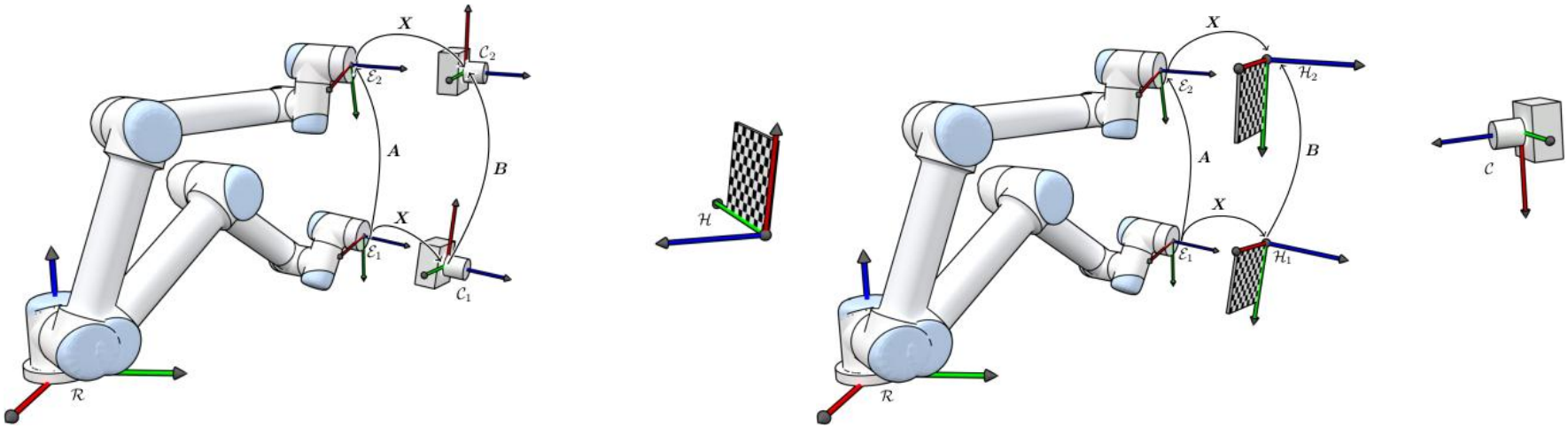
- Ako poznajemo matricu transformacija između dva različita oblaka točaka, možemo ih svesti u **jedan koordinatni sustav** te obogatiti količinu informacija

Transformacija PC-a snimljenog na robotu

- Često je slučaj da se **kamera instalira na robotsku ruku** ili u njenu okolinu prilikom implementacije nekog zadatka
- Postoje metode kojima možemo **kalibrirati** tu kameru kako bi mogli dobiti informacije o promjenama oblaka točaka
- U tom slučaju, **transformacija oblaka točaka** bitna je kako bi mogli sve što je snimljeno smjestiti u **isti koordinatni sustav** te prema tome vršiti **manipulaciju**

Moguće konfiguracije odnosa robot-kamera

1. **Hand in eye** - odnos pozicije i rotacije kamere u odnosu na end effector robota je konstantan
2. **Hand to eye** - Odnos pozicije i rotacije kamere u odnosu na bazu robota je konstantan jer su kamera i alat spojeni – kamera "gleda iz ruke"

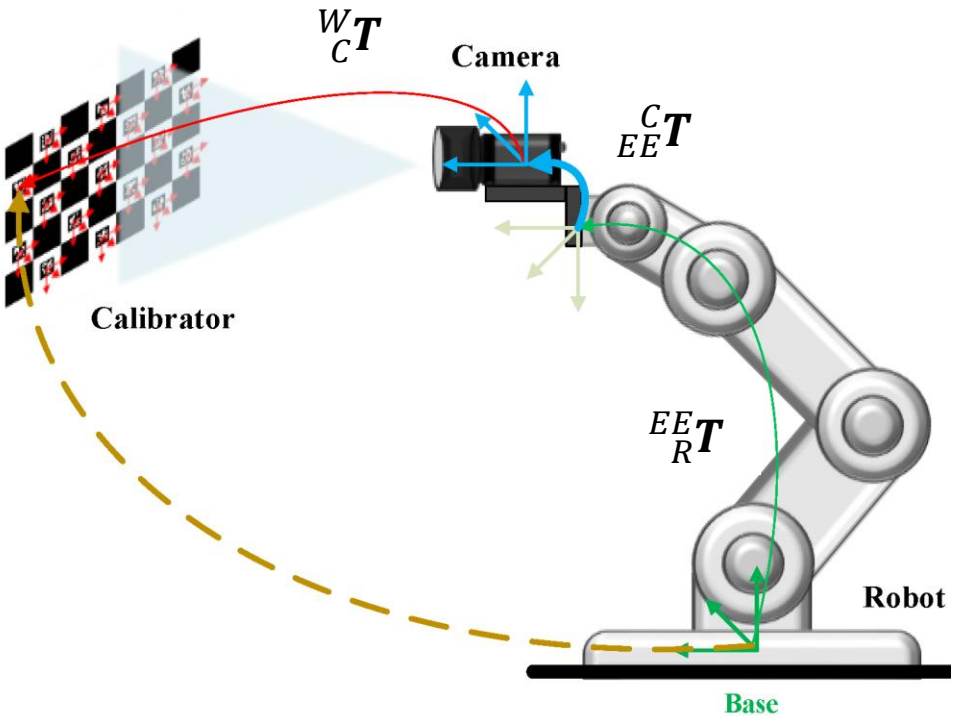


Kalibracija kamere

- Prije rada s kamerom potrebno je provesti **intrinzičnu** i **ekstrinzičnu kalibraciju**
- **Intrinzična** kalibracija služi da se na slici **isprave nesavršenosti** senzora i objektiva
- **Ekstrinzični parametri** daju nam vezu kamere u odnosu na bazu robota (hand to eye) ili vezu kamere i EEF robota (hand in eye)
- Ekstrinzični parametri mogu se zapisati kao **homogena matrica transformacija**, koja opisuje prostorni odnos baze robota i kamere

Ekstrinzična kalibracija kamere

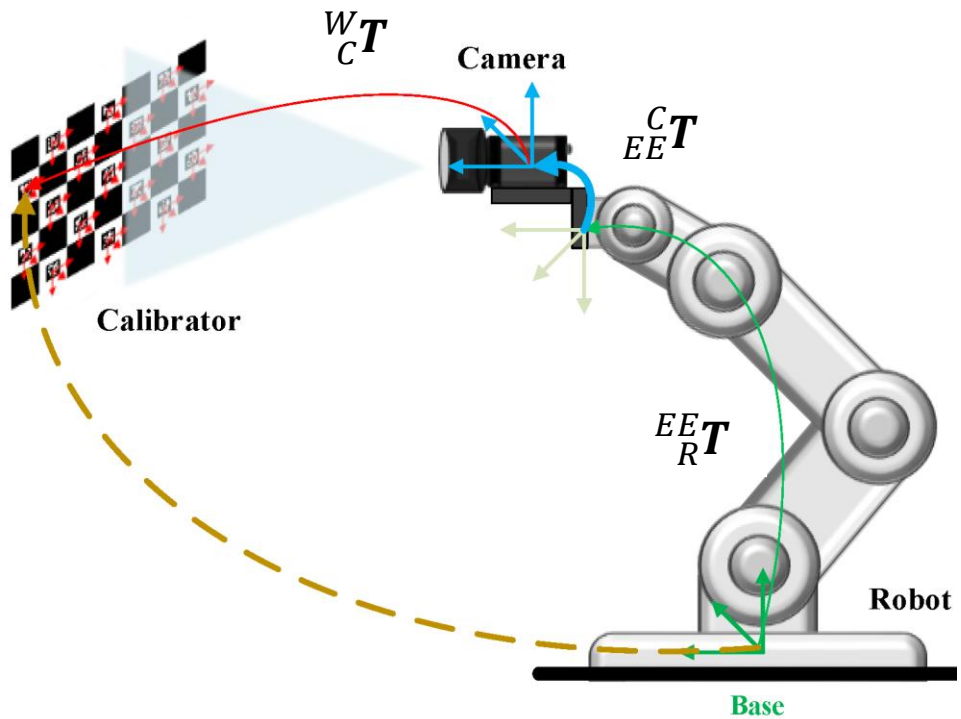
- ${}^{W}_{C}\boldsymbol{T}$ - matrica transformacija od kamere do koordinatnog sustava kalibracijske ploče
- ${}^{EE}_{R}\boldsymbol{T}$ - matrica transformacija od baze robota do TCP-a (end-effectora - EE)
- **Zanima nas: ${}^{C}_{EE}\boldsymbol{T}$ - matrica transformacija od TCP-a do koordinatnog sustava kamere**



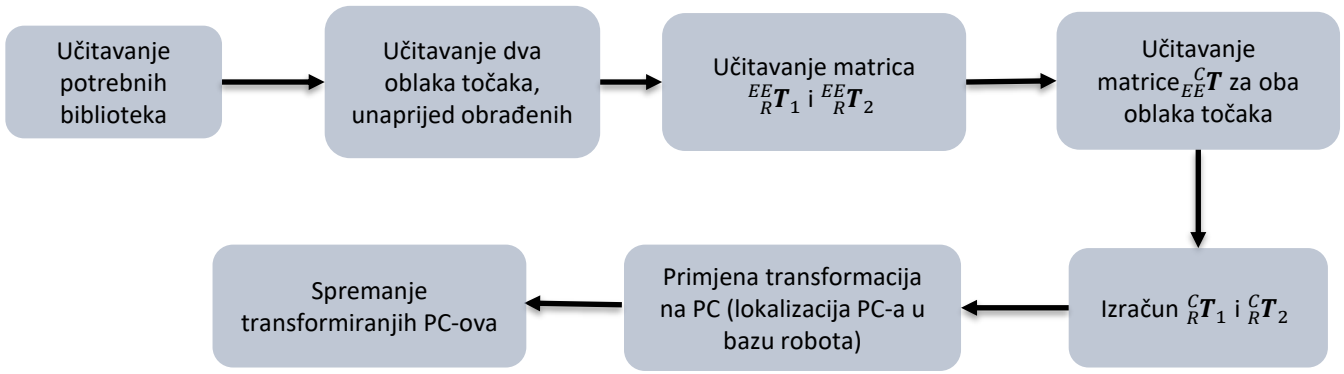
Lokalizacija koordinata predmeta u koordinatni sustav baze robota

- Iz prikaza je jasno da nas zanima odnos: ${}^{EE}_{R}\boldsymbol{T} * {}^{C}_{EE}\boldsymbol{T} * {}^{W}_{C}\boldsymbol{T}$

- Očitamo poziciju predmeta prema koordinatnom sustavu kamere, a zatim transformiramo poziciju predmeta u koordinatni sustav baze robota



Primjer 11 – transformPointCloud



- Prije transformiranja PC-a, potrebno je napraviti cijeli preprocessing postupak, kako bi radili s odgovarajućim podacima
- Matrica ${}^{C}_{EE}\boldsymbol{T}$ je konstantna i uvijek koristimo istu matricu
- Matrice ${}^{EE}_{R}\boldsymbol{T}$ moramo očitati s robota svaki puta kada snimimo PC

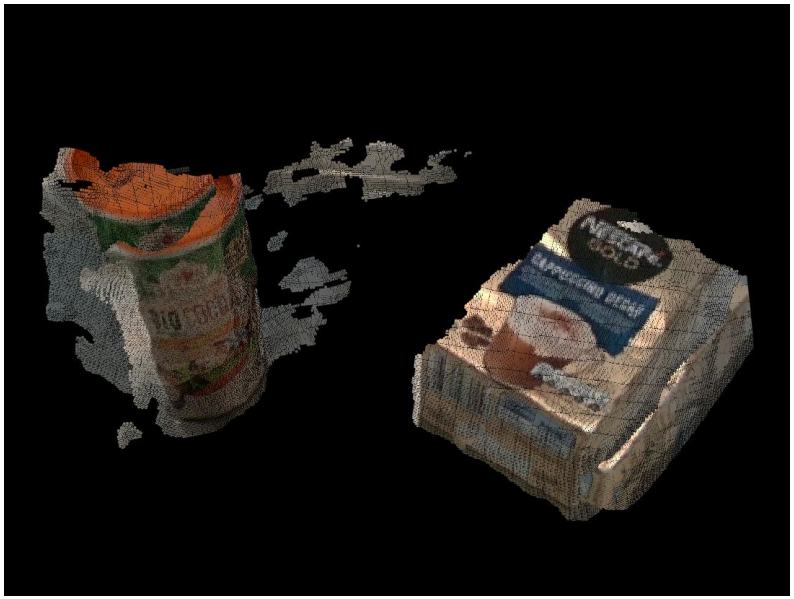
Primjer 11 – transformPointCloud – vizualizacija (prije transformacije)

Obrađeni podaci: asv_dataset/scene_01/cloud_1.pcd i asv_dataset/scene_01/cloud_4.pcd

Primjer 11 – transformPointCloud – vizualizacija (nakon transformacije)

Obrađeni podaci: asv_dataset/scene_01/cloud_1.pcd i asv_dataset/scene_01/cloud_4.pcd

Primjer 11 – transformPointCloud – vizualizacija (usporedba: prije i nakon transformacije)



- Vidljivo je znatno poboljšanje, ali i dalje nije u potpunosti dobro preklopljeno (oku su vidljive značajne pogreške)

Obrađeni podaci: asv_dataset/scene_01/cloud_1.pcd i asv_dataset/scene_01/cloud_4.pcd

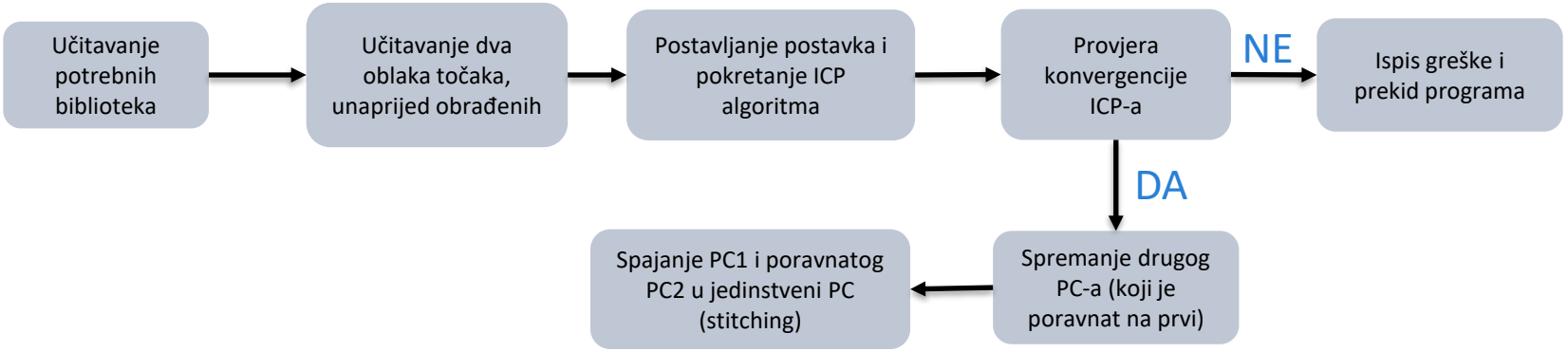
ICP registracija (Iterative Closest Point)

- Koristi se kad već imamo **dva oblaka** koji su "**blizu**" jedan drugome → **fine tuning algoritam**
- Nije robusno na outliere (koristi se s **filtriranim** i/ili **segmentiranim podacima**)
- Kako radi? Iterativno se:
  - Traži najbliže točke između izvora i cilja
  - Estimira transformaciju
  - Primijeni transformaciju
  - Ponavlja sve dok ne konvergira

Stitchanje više oblaka (multiview merging)

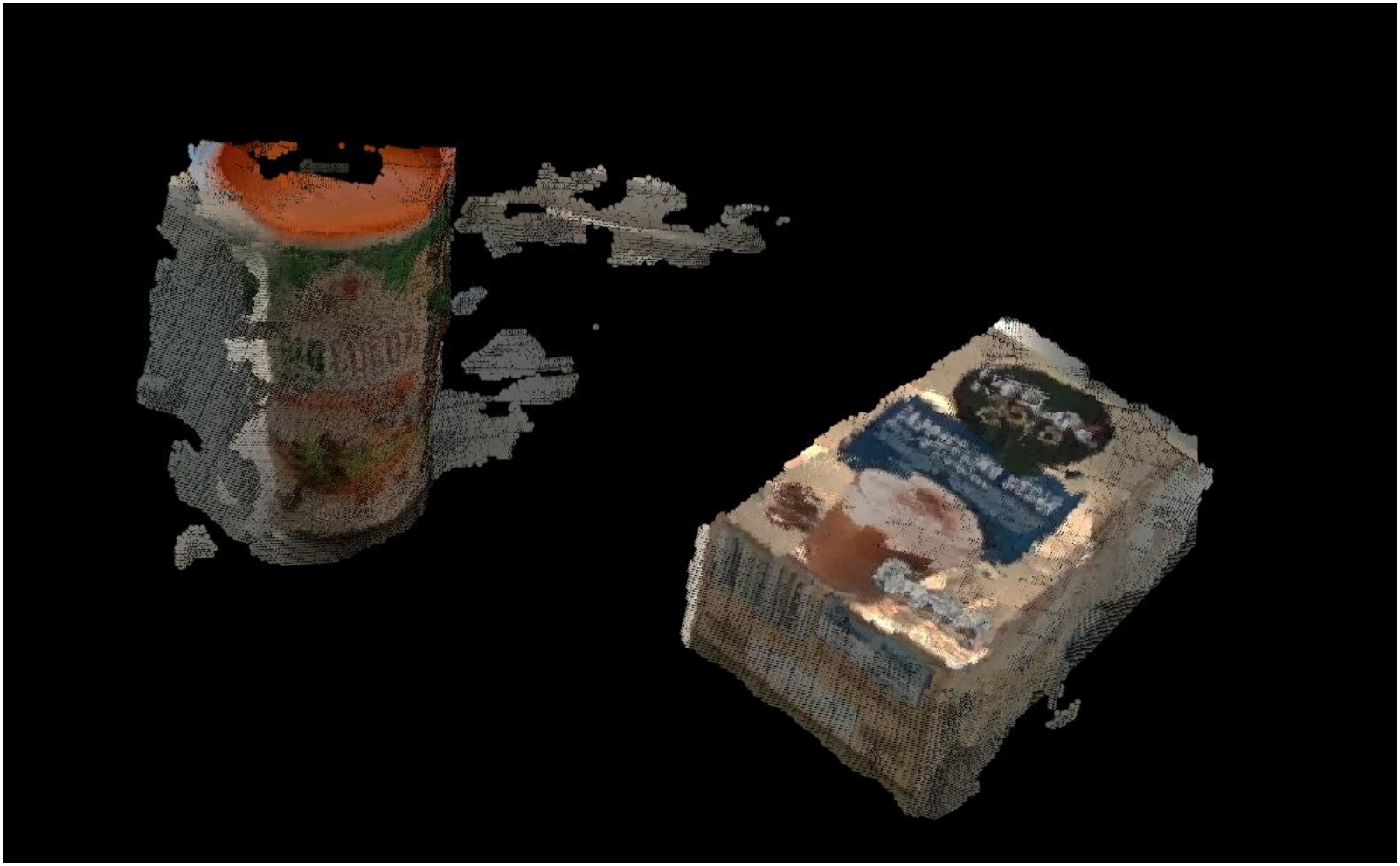
- Cilj je spojiti više PC datoteka snimljenih iz različitih pozicija što rezultira bogatijim 3D modelom
- Kako radi?
  - Svaki point cloud transformiraš u zajednički koordinatni sustav
  - Registriraš ih međusobno (transformPointCloud, ICP, feature matching...)
  - Nakon toga sve PC datoteke spojiš u jednu

Primjer 12 – ICP & stitching



- Prije transformiranja PC-a, potrebno je napraviti cijeli preprocessing postupak, kako bi radili s odgovarajućim podacima
- Ako ICP algoritam konvergira, ali i dalje nismo zadovoljni s dobivenim → malo ručno promijeniti transform drugog PC-a (u primjeru je translacija po x-osi za 1 cm, po z-osi za -1 cm te rotacija oko x-osi za 5 stupnjeva)

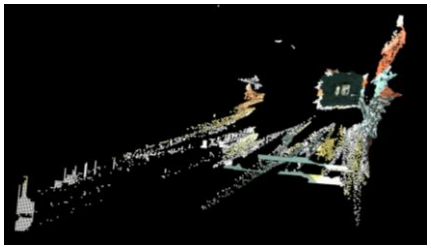
Primjer 12 – ICP & stitching



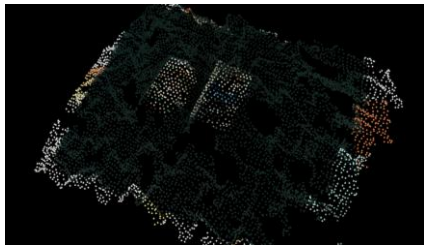
Obrađeni podaci: asv_dataset/scene_01/cloud_1.pcd i asv_dataset/scene_01/cloud_4.pcd

Perceptivno-manipulacijski pipeline

Kamera
(dohvaćanje PC-a)

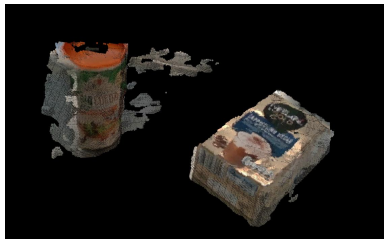


Preprocessing
(*PassThrough, StatisticalOutlierRemoval, VoxelGrid...*)



Segmentacija
(*RANSAC, SACSegmentation, EuclideanClusterExtraction…*)

Registracija i lokalizacija objekata



Grasp planning

Manipulacija

Grasp planning – pose estimation

- Nakon uspješne registracije i lokalizacije predmeta u bazu robota, kako bi omogućili manipulaciju predmetom potrebno je odrediti odnos glavnih osi objekta u odnosu na bazu robota te središte objekta za prihvat kod jednostavnih objekata

Student assignment (seminar)

Student assignment (seminar) - klasterizacija

Zadatak 1: iz zadanog PC-a provesti cijeli pipeline kako bi iz početnog PC-a saznali informaciju o **broju predmeta na slici** te svaki spremili kao **zaseban point cloud**

- Potrebno je napraviti preprocessing
- Na završnim point cloudovima ne smije biti outliera
- Podaci nalaze se unutar mape 'asv_dataset'
- **Napomena:** svaki student radi sa svojim PC-om (zadano u tablici)
- **Predajete** konačne pointcloud datoteke **(.pcd)** te zapis (redoslijed) operacija koje ste koristili i glavne parametre pri njihovim implementacijama **(.pdf izvješće)**

Student	**Point cloud**
Akrap	scene_01/cloud_3
Ćuk	scene_02/cloud_4
Gugo	scene_01/cloud_5
Hartl	scene_02/cloud_3
Noršić	scene_01/cloud_4
Pongračić	scene_02/cloud_5

Student assignment (seminar) – registracija i lokalizacija

Zadatak 2: za proizvoljno odabrana 2 PC-a (iz mape asv_dataset/scene_03) provesti cijeli pipeline rekonstrukcije objekta s dva različita pogleda

- Potrebno je napraviti preprocessing
- Na završnim point cloudovima ne smije biti outliera
- Potrebno je provesti registraciju i lokalizaciju PC-ova te na kraju rezultat stitchati (iz dva PC-a dobiti jedan)
- Podaci nalaze se unutar mape ‘asv_dataset/scene_03’
- **Napomena: proizvoljno odaberite dva PC-a koja ćete poravnati i stitchati**
- **Predajete** konačnu pointcloud datoteku **(.pcd)** te zapis (redoslijed) operacija koje ste koristili i glavne parametre pri njihovim implementacijama **(.pdf izvješće)**