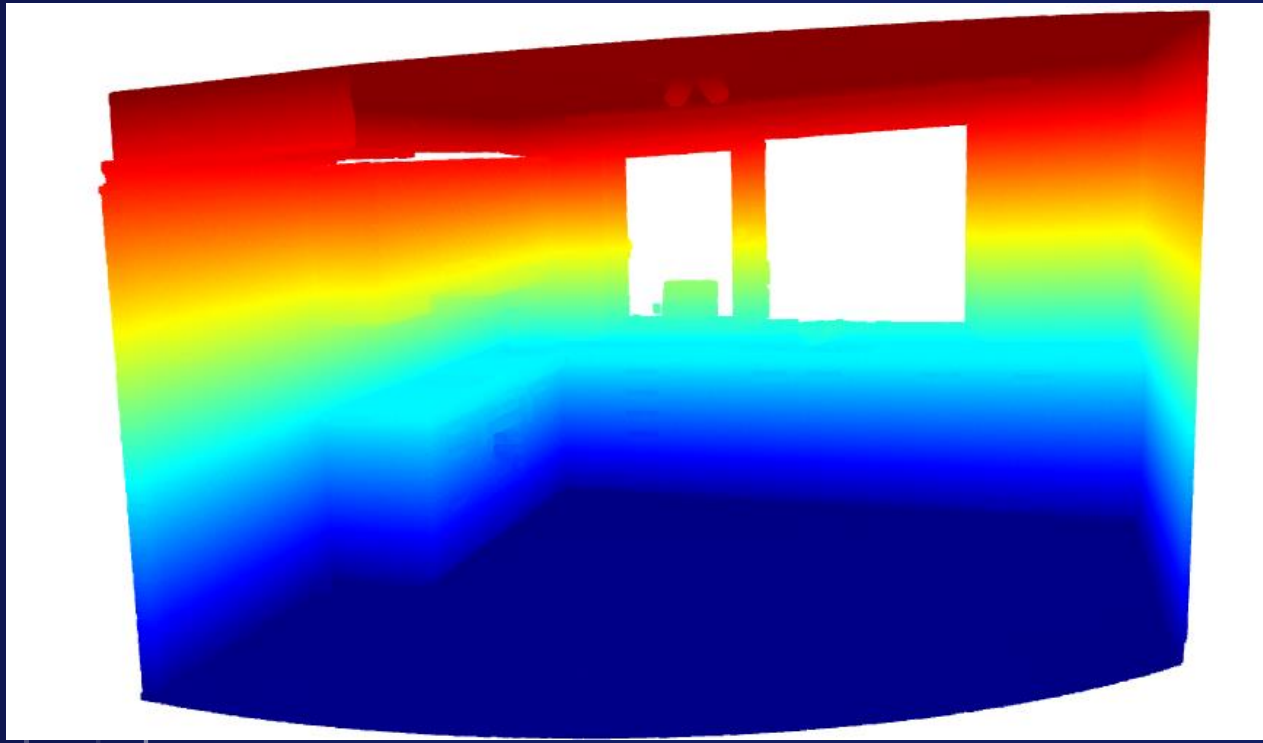
Mračno bodov

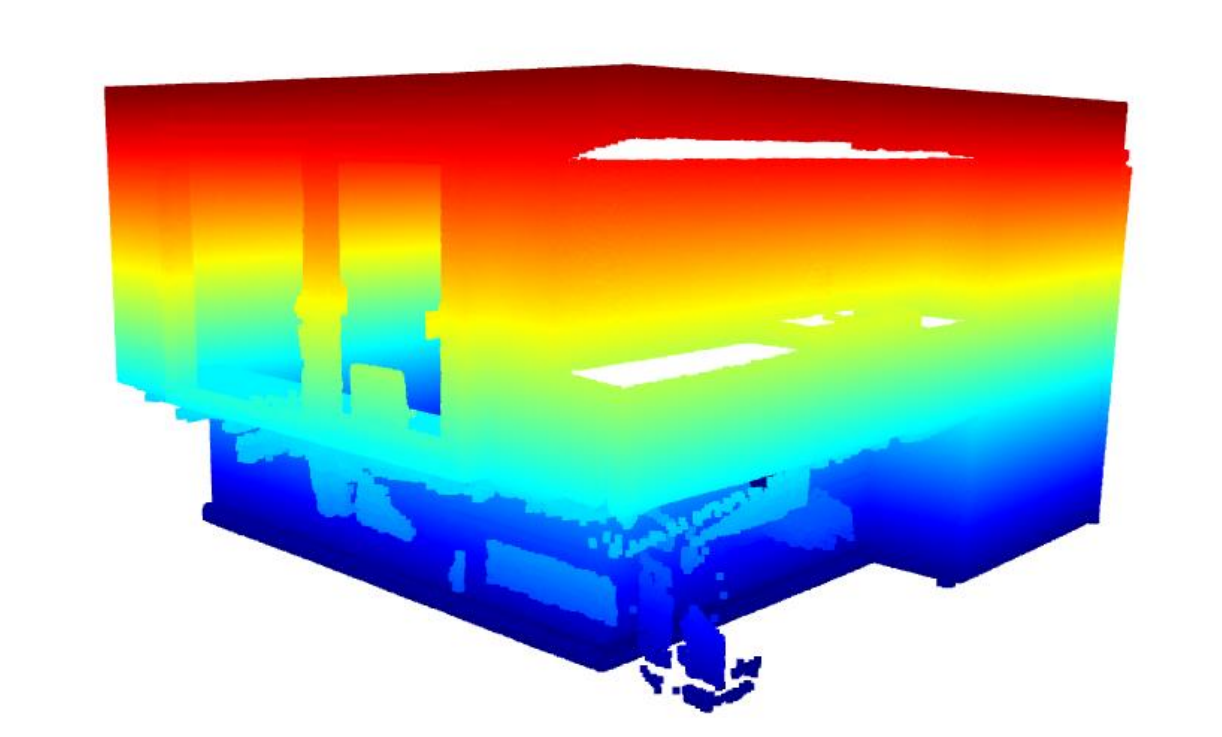
Jozef Košecký
Peter Dobiáš

Obsah

- Vytvorenie a načítanie mračna bodov
- RANSAC
- DBSCAN
- Gaussian mixture

Načítanie mračna bodov

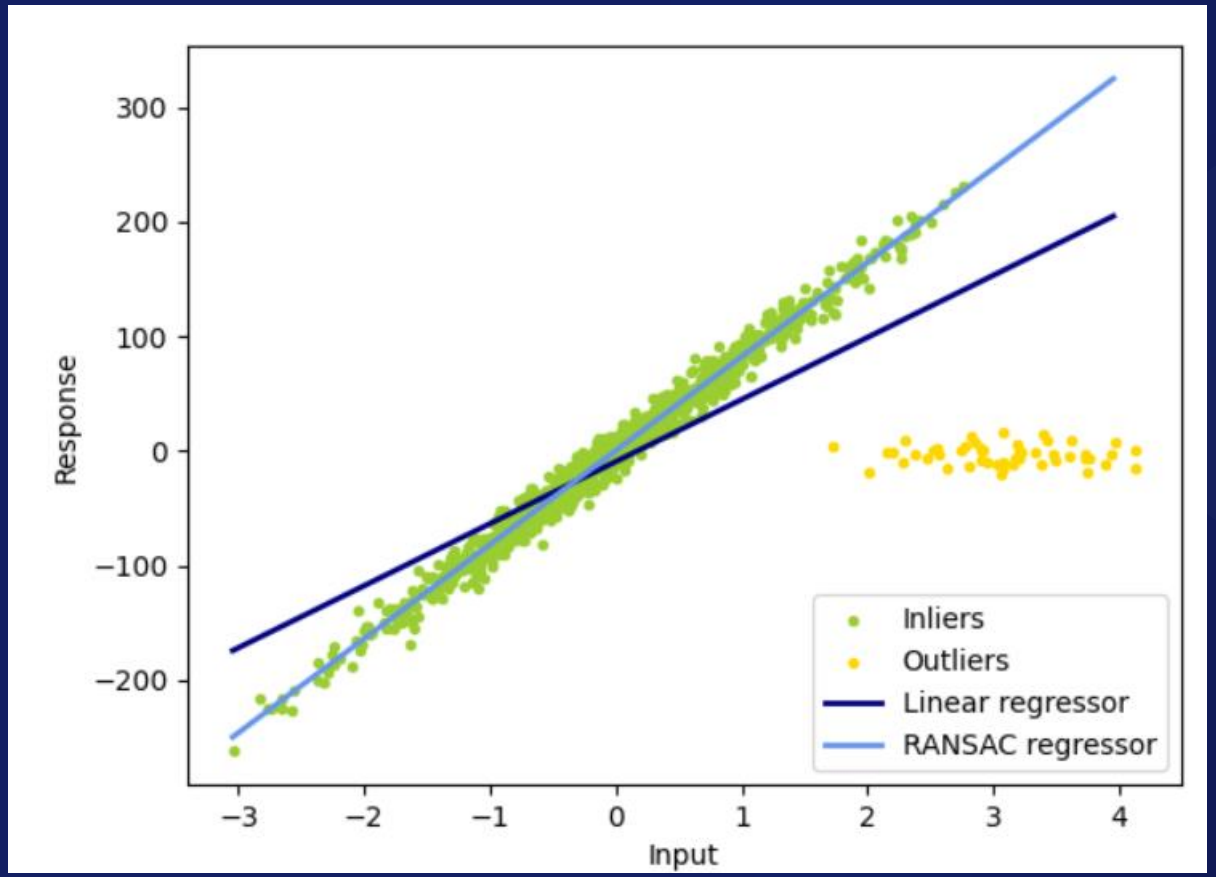




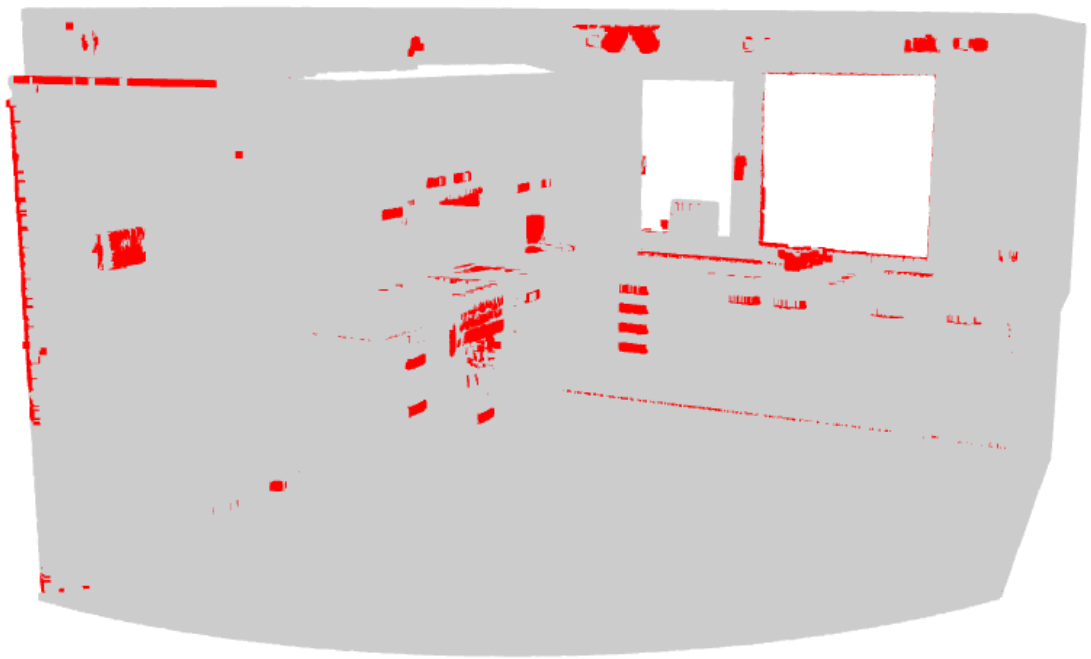
Vytvorenie mračna bodov

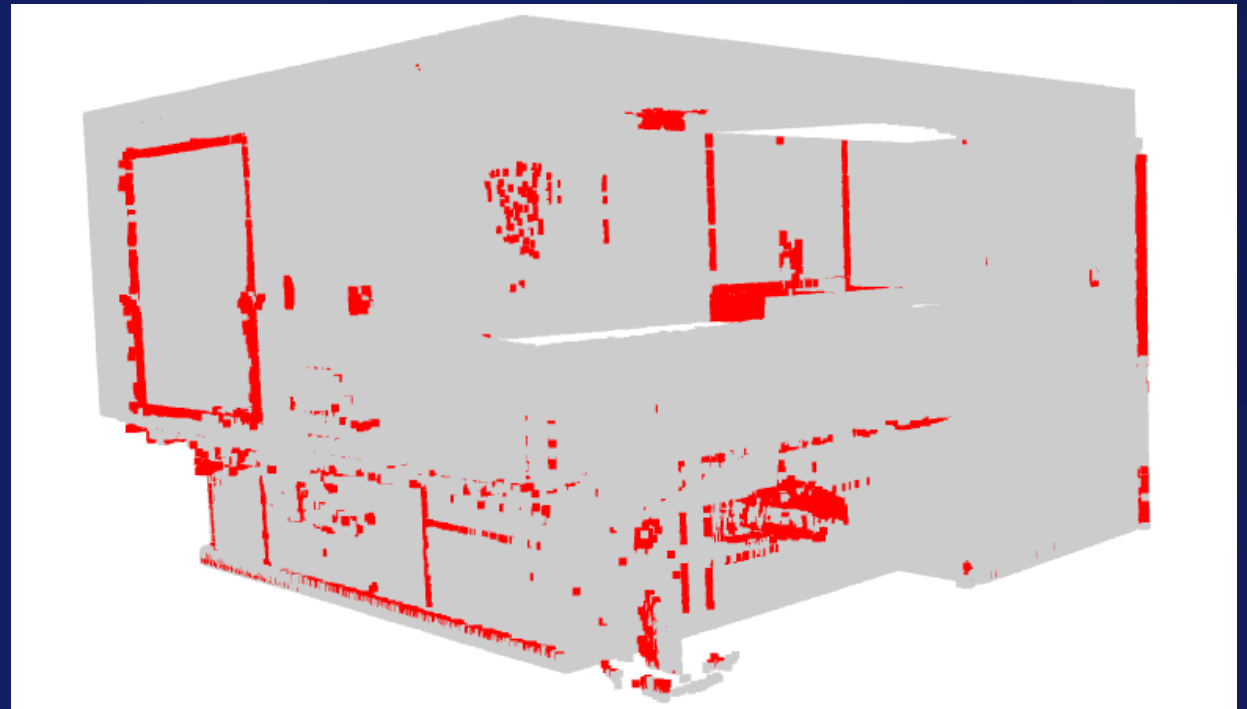
RANSAC

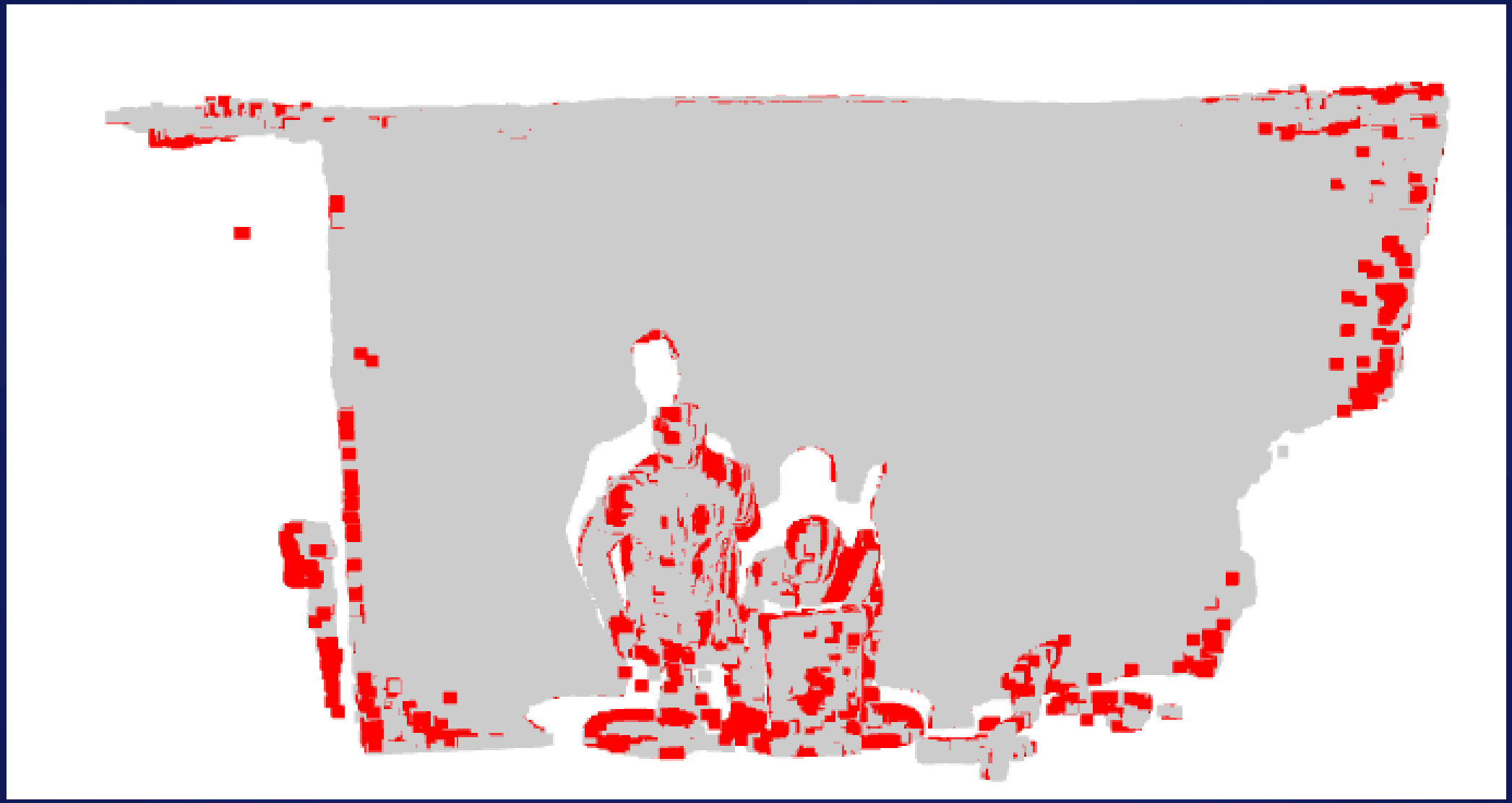
- RANdom Sample Consensus
- Využitý na odstránenie šumu
- Rozlišuje Outliers, Inliers



```python
# read point cloud from file
pcd = o3d.io.read_point_cloud(file_path)
pcd.paint_uniform_color([0.6, 0.6, 0.6])
# visualize
o3d.visualization.draw_geometries([pcd])
cv2.waitKey(0)


################ RANSAC ###################
segment_models={}
segments = {}

max_plane_idx=30

rest=pcd
for i in range(max_plane_idx):
    colors = plt.get_cmap("tab20")(i)
    segment_models[i], inliers = rest.segment_plane(
    distance_threshold=0.01,ransac_n=3,num_iterations=1000)
    segments[i]=rest.select_by_index(inliers)
    segments[i].paint_uniform_color(list([0.8, 0.8, 0.8]))
    rest = rest.select_by_index(inliers, invert=True)
    rest.paint_uniform_color([1, 0, 0])
    print("pass",i,"/",max_plane_idx,"done.")

o3d.visualization.draw_geometries([segments[i] for i in range(max_plane_idx)]+[rest])
cv2.waitKey(0)
```

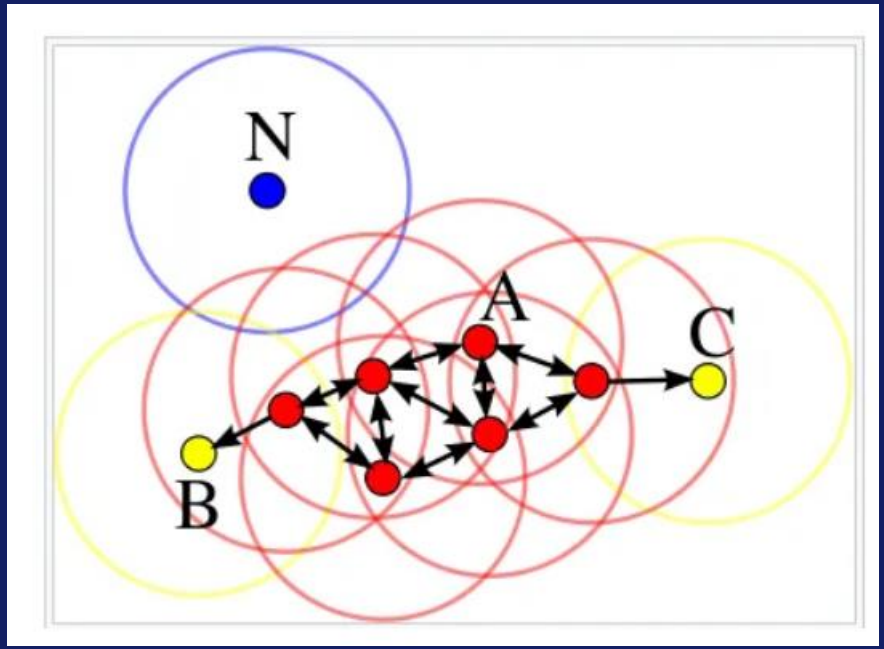




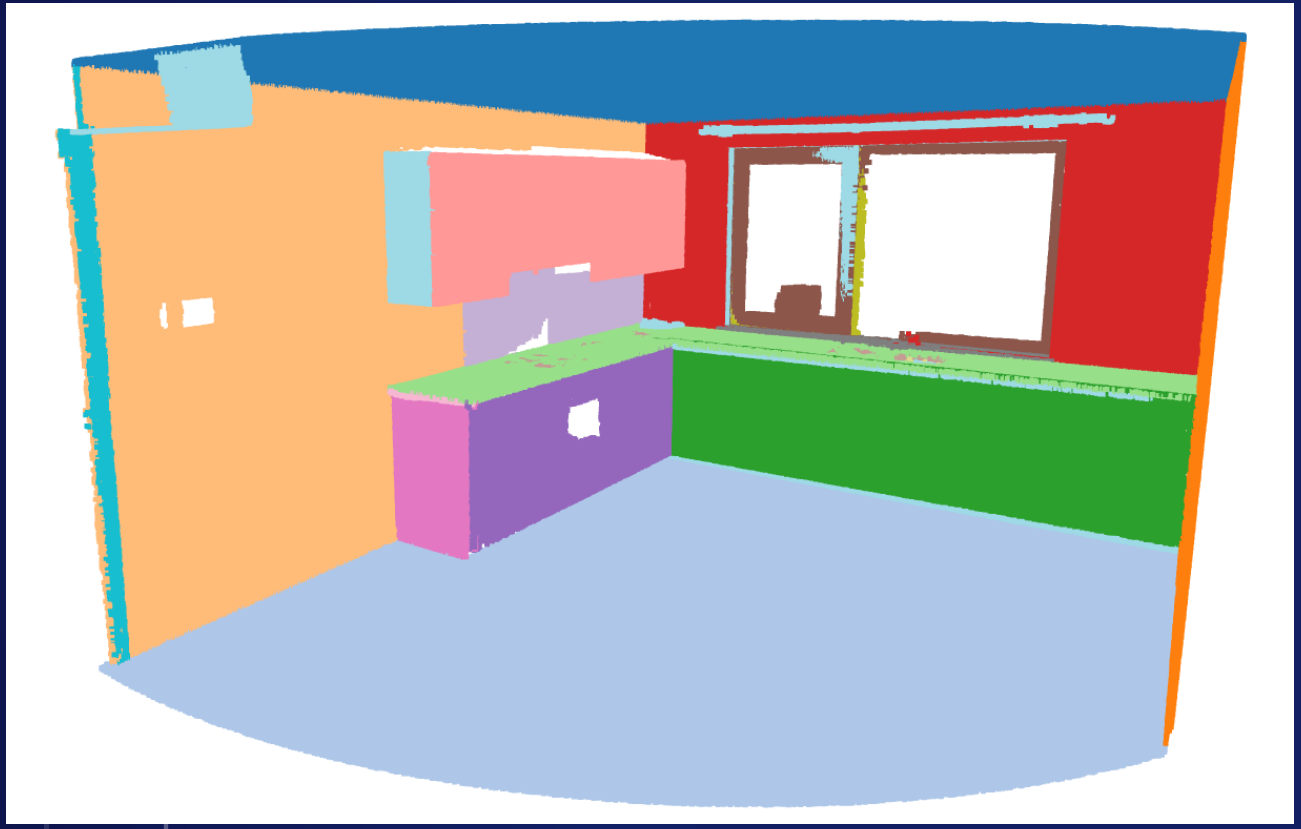


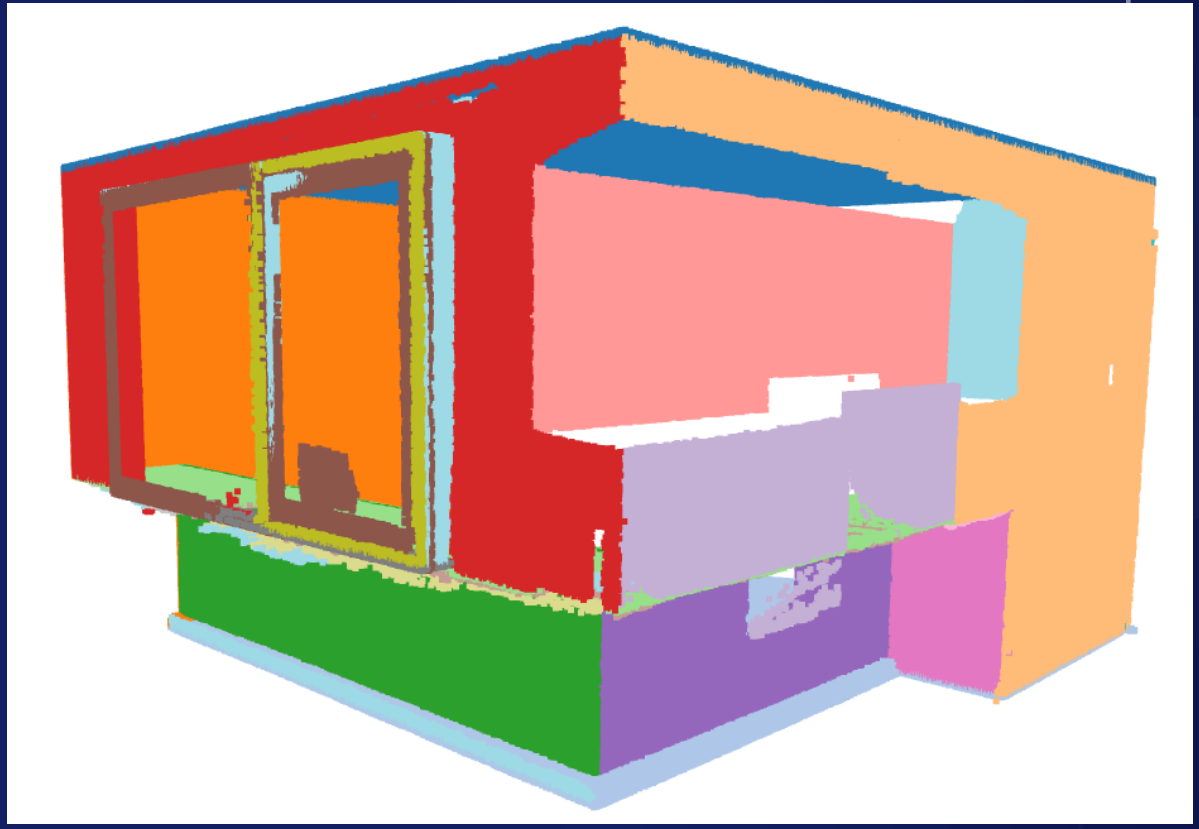
DBSCAN

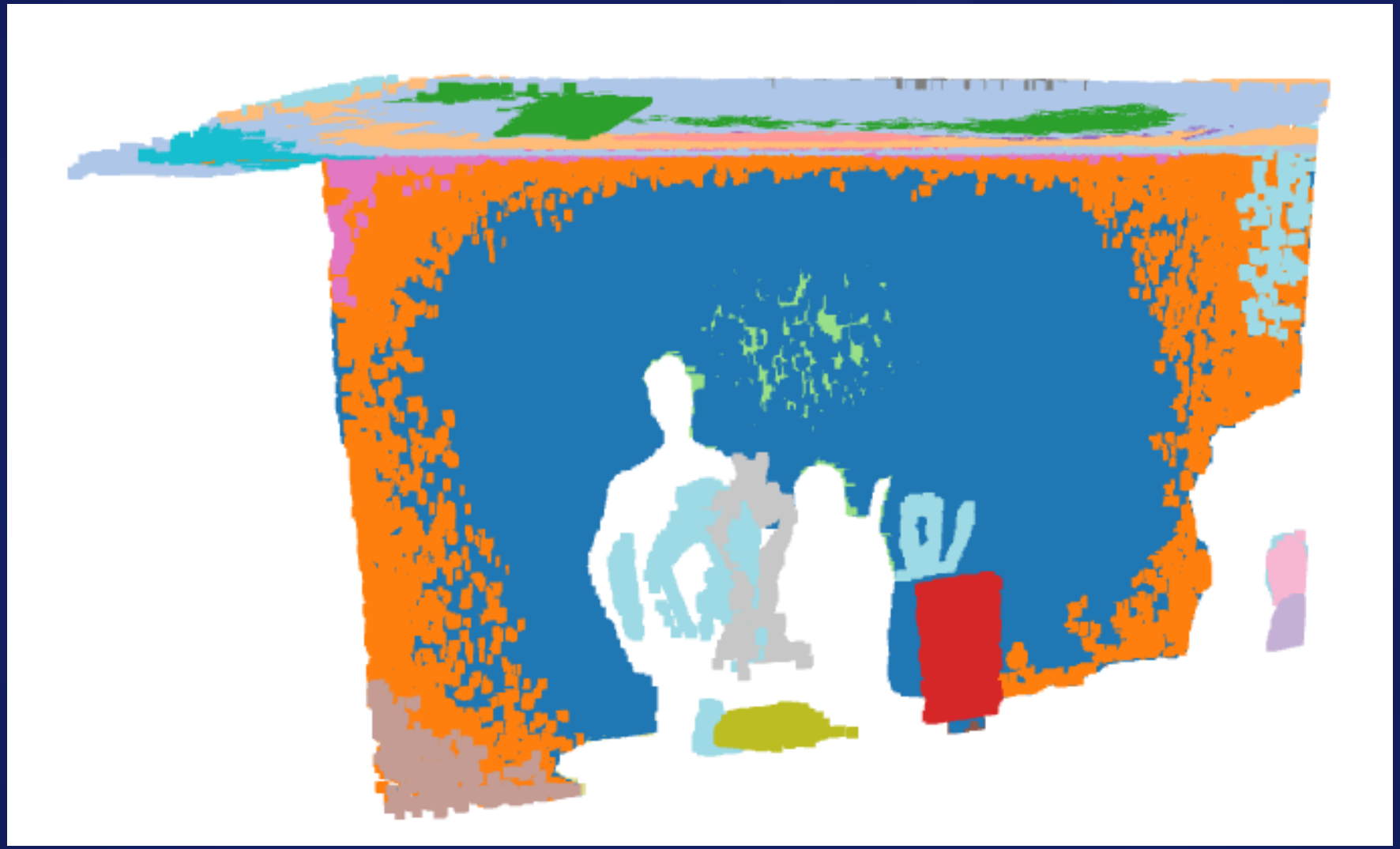
- Density –based spatial clustering of applications with noise
- Identifikuje oblasti s vysokou hustotou bodov v priestore
- Core point
- Border point
- Outlier



```python
for i in range(max_plane_idx):
    colors = plt.get_cmap("tab20")(i)
    segment_models[i], inliers = rest.segment_plane(
        distance_threshold=0.01, ransac_n=3, num_iterations=1000)
    segments[i] = rest.select_by_index(inliers)

    ############### DBSCAN
    labels = np.array(segments[i].cluster_dbscan(eps=0.05, min_points=10))
    candidates = [len(np.where(labels == j)[0]) for j in np.unique(labels)]
    best_candidate = int(np.unique(labels)[np.where(candidates == np.max(candidates))[0]])

    rest = rest.select_by_index(inliers, invert=True)+segments[i].select_by_index(list(np.where(labels!=best_candidate)[0]))
    segments[i]=segments[i].select_by_index(list(np.where(labels==best_candidate)[0]))
    segments[i].paint_uniform_color(list(colors[:3]))

    print("pass", i, "/", max_plane_idx, "done.")
```

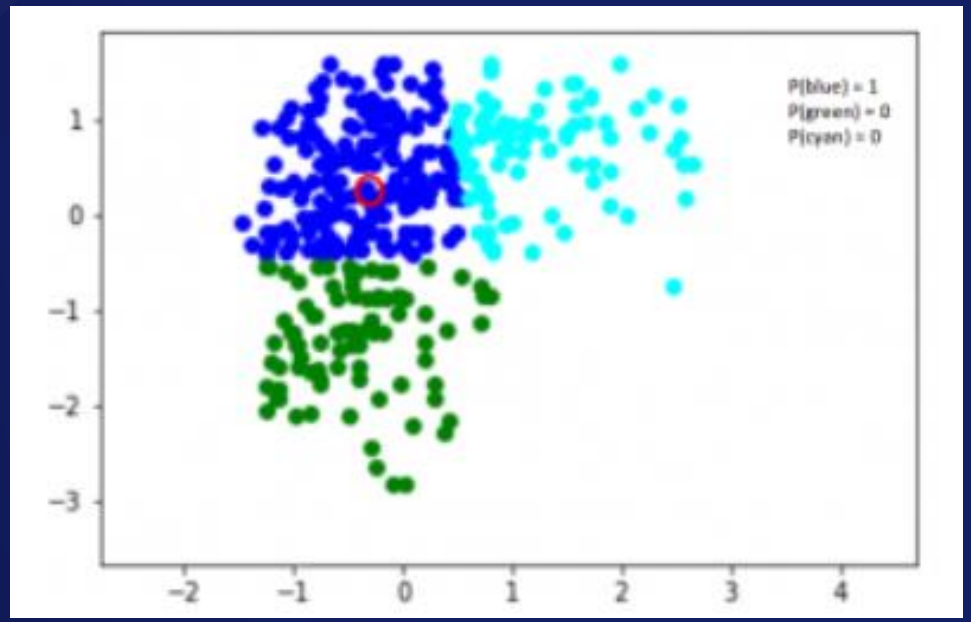


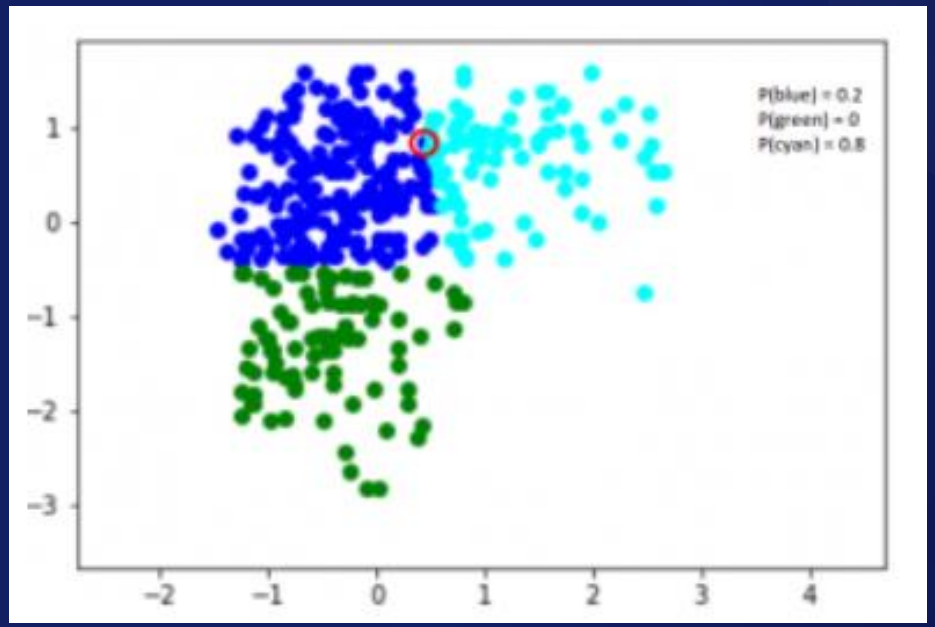




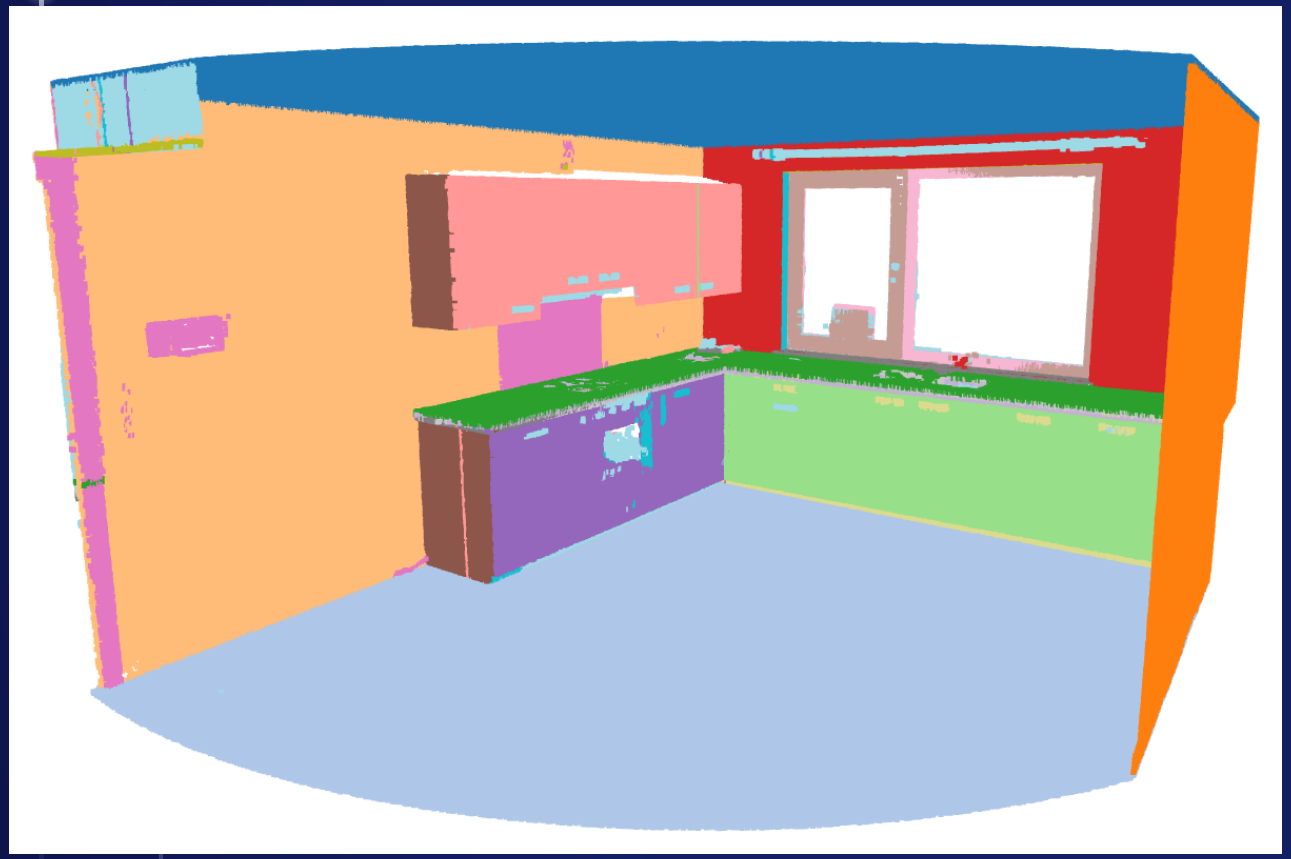
GAUSSIAN MIXTURE

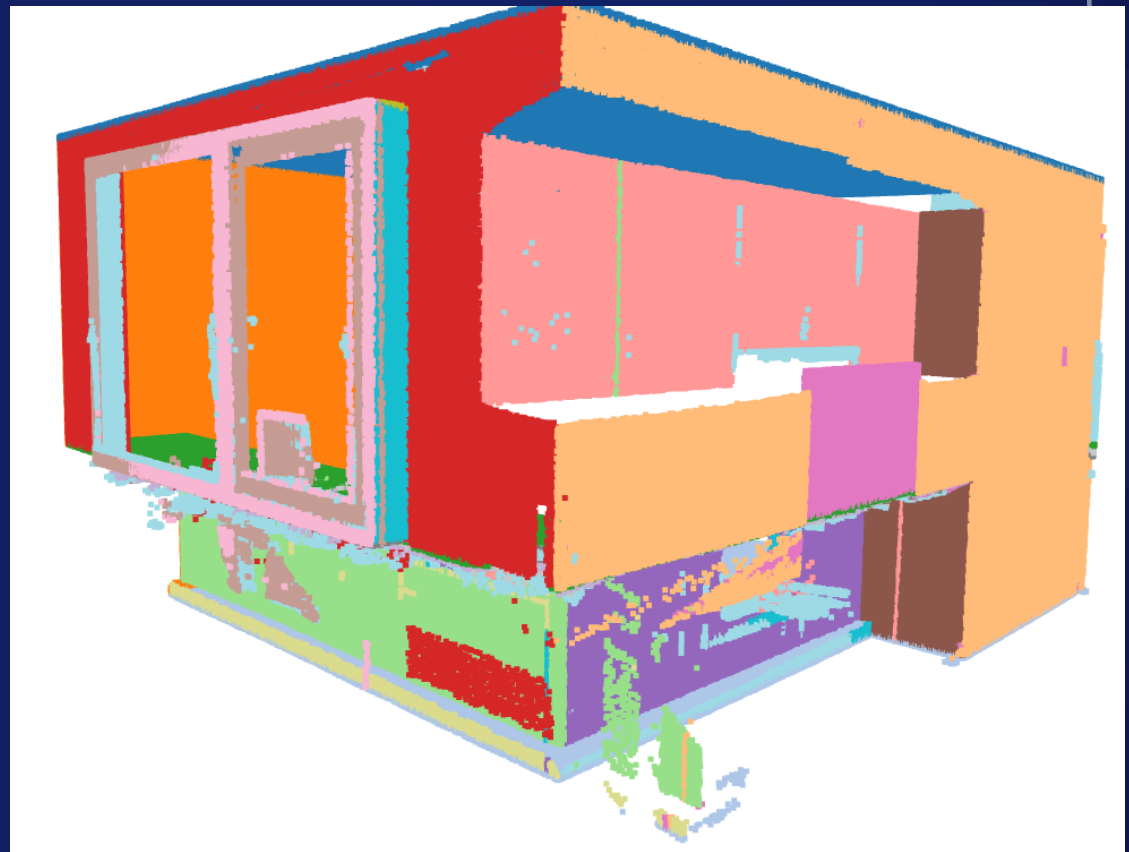
- Gaussovské distribúcie na základe očakávaných zhlukov
- Odhadujú sa priemer, kovariancia a pravdepodobnosť každej distribúcie
- Každý zhluk je reprezentovaný kombináciou distribúcií

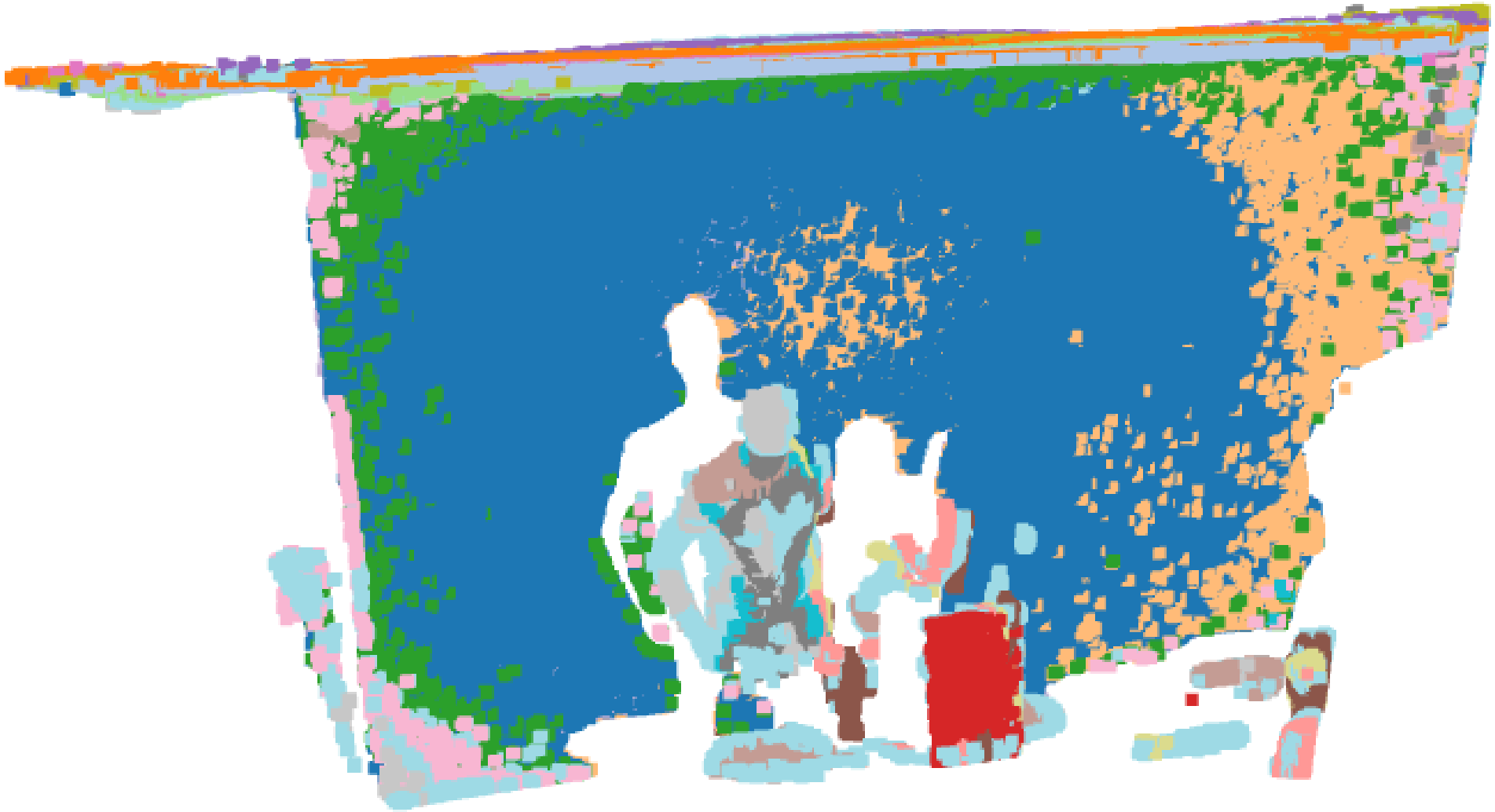




```
max_plane_idx = 30
# Create an instance of the GaussianMixture class
gm = GaussianMixture(n_components=1)

rest = pcd
for i in range(max_plane_idx):
    colors = plt.get_cmap("tab20")(i)
    segment_models[i], inliers = rest.segment_plane(
        distance_threshold=0.01, ransac_n=3, num_iterations=1000)
    segments[i] = rest.select_by_index(inliers)

    ############### DBSCAN
    points = np.asarray(segments[i].points)
    gm.fit(points)
    labels = gm.predict(points)


    candidates = [len(np.where(labels == j)[0]) for j in np.unique(labels)]
    best_candidate = int(np.unique(labels)[np.where(candidates == np.max(candidates))[0]])

    rest = rest.select_by_index(inliers, invert=True)+segments[i].select_by_index(list(np.where(labels!=best_candidate)[0]))
    segments[i]=segments[i].select_by_index(list(np.where(labels==best_candidate)[0]))
    segments[i].paint_uniform_color(list(colors[:3]))
```







Ďakujeme za pozornosť