

1. Ik toon aan dat ik succesvol gebruik kan maken van Cloud points

Zie bewijslasten 2 (is een document die ik gemaakt heb uit periode 1 voor Emit IT)

2. Ik toon aan dat ik OOP2 haal

Ik heb deze nog niet gehaald en de herkansing is nog niet geweest

3. Ik toon aan dat ik Python leer en beheers

Hieronder het ball pivot programma wat ik gemaakt had in periode 1 voor Emit IT in Python

```
import numpy as np
import open3d as o3d
import laspy
import time
import os

def log_time(message, start):
    elapsed = time.time() - start
    print(f"{message} (duurde {elapsed:.1f} sec)")
    return time.time()

# === Mapselectie ===
default_dir = "./output_meshes"
output_dir = input(f"Voer de uitvoermap in (standaard: {default_dir}): ").strip()
if not output_dir:
    output_dir = default_dir

if not os.path.exists(output_dir):
    os.makedirs(output_dir)
    print(f"Map aangemaakt: {output_dir}")
else:
    print(f"Map gevonden: {output_dir}")
```

```

# === Aantal runs kiezen ===

default_runs = 5

num_runs_input = input(f"Hoeveel runs wil je uitvoeren? (standaard: {default_runs}): ").strip()
num_runs = int(num_runs_input) if num_runs_input.isdigit() else default_runs
print(f"Aantal runs ingesteld op: {num_runs}")


# === Start ===

total_start = time.time()

print("Bestand laden...")

step_start = time.time()


las = laspy.read("segmented_filtered_sub_pointcloud.laz")
points = np.vstack((las.x, las.y, las.z)).transpose()
print(f"Aantal punten: {len(points)}")

step_start = log_time("Bestand ingeladen", step_start)


# === Eventueel verminderen ===

if points.shape[0] > 1_000_000:
    idx = np.random.choice(points.shape[0], size=750_000, replace=False)
    points = points[idx]
    print(f"Punten verminderd tot: {len(points)}")
    step_start = log_time("Punten verminderd", step_start)


# === Gemiddelde afstand inschatten ===

pcd_sample = o3d.geometry.PointCloud()
pcd_sample.points = o3d.utility.Vector3dVector(points[:min(20000, len(points))])
avg_dist = np.mean(pcd_sample.compute_nearest_neighbor_distance())
print(f"Geschatte gemiddelde puntafstand: {avg_dist:.5f}")


# === Runs uitvoeren ===

```

```

for i in range(1, num_runs + 1):

    print(f"\n=== RUN {i}/{num_runs} ===")

    run_start = time.time()

    # Detailniveau berekenen

    if i <= 5:

        detail_factor = 1.0 - (i - 1) * 0.15 # Originele schaal

    else:

        # Na run 5: blijf detail verder opvoeren (meer precisie)

        detail_factor = 0.4 * (0.85 ** (i - 5)) # Exponentieel dalend

    voxel_size = avg_dist * (3.0 * detail_factor)

    print(f"Downsampling met voxel_size={voxel_size:.5f} (detail_factor={detail_factor:.3f})...")

    # Downsamplen

    pcd = o3d.geometry.PointCloud()

    pcd.points = o3d.utility.Vector3dVector(points)

    pcd = pcd.voxel_down_sample(voxel_size=voxel_size)

    # Outliers verwijderen

    pcd, _ = pcd.remove_statistical_outlier(nb_neighbors=20, std_ratio=2.0)

    step_start = log_time("Downsampled en opgeschoond", step_start)

    # Normals berekenen

    print("Normals berekenen...")

    pcd.estimate_normals(

        search_param=o3d.geometry.KDTreeSearchParamHybrid(

            radius=avg_dist * 6.0,

            max_nn=100

        )

    )

```

```

pcd.orient_normals_consistent_tangent_plane(1000)
step_start = log_time("Normals berekend", step_start)

# === Alleen Ball Pivoting ===
print("Start Ball Pivoting (met automatische radii)...")
radii = o3d.utility.DoubleVector(np.linspace(avg_dist * 2.0, avg_dist * 12.0, 8))
mesh = o3d.geometry.TriangleMesh.create_from_point_cloud_ball_pivoting(pcd, radii)

# Mesh optimaliseren
mesh.remove_duplicated_vertices()
mesh.remove_degenerate_triangles()
mesh.remove_duplicated_triangles()
mesh.remove_unreferenced_vertices()
mesh = mesh.filter_smooth_laplacian(number_of_iterations=2)
mesh.compute_vertex_normals()

num_tris = len(mesh.triangles)
num_verts = len(mesh.vertices)
print(f"Ball Pivoting resultaat: {num_tris} driehoeken, {num_verts} vertices")
if num_tris < 50:
    print("Waarschuwing: nauwelijks driehoeken gevormd – radii mogelijk te klein of puntwolk te dun.")

step_start = log_time("Ball Pivoting + opschonen klaar", step_start)

# === Opslaan ===
output_file = os.path.join(output_dir, f"mesh_run_{i}_ballpivot_only.ply")
o3d.io.write_triangle_mesh(output_file, mesh)
step_start = log_time(f"Mesh opgeslagen als {output_file}", step_start)

print(f"Totale duur run {i}: {time.time() - run_start:.1f} sec")
print(f"\n=== Alle {num_runs} runs voltooid in {time.time() - total_start:.1f} sec ===")

```