Homework No.1

- **Objective**

Implement the Iterative Closest Point (ICP) algorithm to align two 3D point clouds. Your implementation should compute the optimal rigid transformation (rotation + translation) that aligns a source point cloud to a target point cloud, and output both the aligned point cloud and the transformation matrix.

- Deadline: three weeks later than the assignment date

1. Problem Statement

You are given:

- **Source point cloud** `P`: N × 3 matrix (N points, each with x, y, z coordinates)
- **Target point cloud** `Q`: M × 3 matrix

Your task is to find a rigid transformation **{R, t}** (rotation matrix $R \in \mathbb{R}^{3x3}$, translation vector $t \in \mathbb{R}^3$) that minimizes the distance between corresponding points in `P` and `Q`:

$$\min_{R,t} \sum_{i=1}^{N} ||R \cdot p_i + t - q_i||^2$$

Where `p_i` is a point in `P`, and `q_i` is its corresponding point in `Q`.

Your program should output:

1. **Aligned point cloud**: The transformed source point cloud after ICP convergence.
2. **Transformation matrix**: The final `[R | t]` matrix (4×4 homogeneous transformation).
3. **Convergence plot**: Plot the registration error vs. iteration number.
4. **Visualization**: Show initial alignment, intermediate steps (optional), and final alignment.

Example output format:

ICP converged after 12 iterations.
Final RMSE: 0.0023

Transformation matrix:
[[0.866, -0.500, 0.000, 0.500],
 [0.500, 0.866, 0.000, -0.300],
 [0.000, 0.000, 1.000, 0.200],
 [0.000, 0.000, 0.000, 1.000]]

Your implementation will be evaluated on:

1. **Correctness** (40%): Does ICP converge to the correct transformation?
2. **Efficiency** (20%): Use of kd-tree for nearest neighbor search.
3. **Robustness** (20%): Handling of outliers/noise (optional: implement outlier rejection).
4. **Visualization** (10%): Clear visualization of alignment process.
5. **Report** (10%): Brief explanation of your implementation choices and results.

BONUS points

1. **Point-to-plane ICP**: Implement the point-to-plane variant for faster convergence on smooth surfaces.
2. **Weighted correspondences**: Weight pairs based on distance or normal similarity.
3. **Multi-resolution ICP**: Implement coarse-to-fine registration.
4. **Benchmarking**: Test on standard datasets (Stanford Bunny, Armadillo).

Deliverables

Submit a single ZIP file containing:

1. `icp_registration.py` : Your complete implementation.
2. `test_data.npz` : Sample source and target point clouds.
3. `results/` folder with:
   - `aligned_cloud.ply` : Final aligned point cloud.
   - `transformation.txt` : Final transformation matrix.
   - `convergence_plot.png` : Error vs. iteration plot.
   - `visualization.png` : Before/after alignment visualization.
4. `report.pdf` (≤ 2 pages): Brief description of your implementation and results.