16-280 Intelligent Robot Systems HW4 (G) - ICP Mapping

INSTRUCTIONS

- **Due: Monday, September 29 at 23:59 EDT.**
- **Format:** You may use a tablet or computer device to digitally fill in PDF, or handwrite your solutions.
- **Policy:** See the course Canvas for homework policies and academic integrity.
- **Submission:**

Collectively submit the filled-in PDF of the assignment and your code in a zip archive titled `GROUP_GROUP#_16280_HW4G.zip`. For example, group 25 would submit the assignment with a zip archive titled `GROUP_25_16280_HW4G.zip`. You must submit a ZIP archive of your ROS package, at minimum including all files you changed. If compressing the entire folder causes issues because of build files, you need not include them. Then, submit a zip containing `point2planepy` and the writeup with Question 3 filled in to the individual assignment submission on Gradescope.

Name	Joseph Jia
Andrew ID	josephji
Hours to complete?	[✓ (0, 2] hours] (2, 3] hours (3, 4] hours (4, 5] hours (5, 6] hours (6, 7] hours (7, 8] hours > 8 hours

We know that tools like **ChatGPT** are readily available and, yes, much of this assignment *is* technically solvable using them. But before you go outsourcing your brain, consider the following:

- You'll miss out on the core ideas and clever insights we carefully designed this assignment to teach.
- You'll be unprepared for exams, interviews, or projects that require this knowledge.
- Least importantly, its an AIV. If we have doubts, you will be asked to explain your implementations in full.

This isn't meant to scare you—just to remind you that learning takes effort, and short-circuiting that process won't help you in the long run. Use AI as a study aid if you must, but not as a substitute for understanding. We're excited for you to tackle the challenges ahead the right way.
If you need any more evidence, read about a recent study by MIT that shows that using LLMs for cognitive tasks literally rots your brain: this article explains how.

Your 16-280 Course Staff

Q1. [21 %] Point to Plane ICP (Individual)

In this problem, you will implement **point-to-plane ICP** in a standalone script. Use the provided starter `point2plane.py` together with synchronized logs `odom_sync.txt` and `scan_sync.txt` located under `data/`. The script will:

- Parse odometry yaw and scans from the provided text logs.
- Convert ranges → (x,y), initialize each scan with odometry, then refine alignment via ICP.
- Accumulate aligned scans and plot the final map.

First, ensure required dependencies are installed:

```
$ pip install matplotlib scipy numpy
```

Once your implementation is complete, you can run it using:

```
$ python3 point2plane.py
```

(a) [7 %] Complete the function `compute_normals(self, points, sensor_origin)` to estimate 2D surface normals using a 3-point central-difference method:

- For $i = 1 \dots N-2$, let tangent $= p_{i+1} - p_{i-1}$, normal $= [-t_y,\, t_x]$, then normalize.
- Flip the normal to face the sensor if $\text{normal}^\top(\text{sensor\_origin} - p_i) < 0$.
- Set boundary normals by copying neighbors: $n_0 = n_1$, $n_{N-1} = n_{N-2}$.

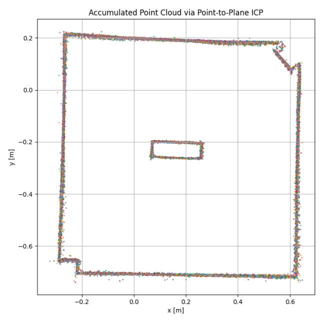
Optionally, you may compare against the provided PCA-based normals by toggling `self.normal_simple = False`.

(b) [7 %] Implement the ICP loop in `icp_point_to_plane(self, ...)`. The starter code already gives you the residual formulation from lecture. You do not need to re-derive it. Follow these steps:

1. Use a KD-tree to find nearest-neighbor correspondences between source and target.
2. For each match, form one row of A and b using the equations given in the starter.
3. Solve $Ax = b$ with SVD to get the incremental update $(\delta\theta, t_x, t_y)$.
4. Apply the update to the source cloud, and repeat until convergence or 15 iterations.

(c) [7 %] Run your implementation with the provided logs (`data/odom_sync.txt`, `data/scan_sync.txt`) and submit:

- A plot of the final accumulated map produced by the script.
- The number of ICP iterations in the last alignment and the final mean absolute point-to-plane residual, which you should print to the terminal from your script and copy the values.



Final Mean Absolute Residual: 0.0004007820243596282
ICP iterations: 15