Module Plan

Tavish Mankash

Overview

Short, practical breakdown of the codebase. Each module has a small, clear contract so we can iterate fast.

Modules

1. data

Purpose: get terrain into memory.

- `load_synthetic_terrain(shape)`: quick, deterministic heightmap (for dev/testing).
- *(later)* `load_laz(path)`: read LAZ/point cloud and return gridded heightmap.
- Inputs: file path or shape; Outputs: $H \in \mathbb{R}^{m \times n}$ height grid.

2. preprocess

Purpose: convert raw points to a usable grid; light smoothing if needed.

- *(placeholder)* `voxelize(points, voxel_size)`.
- Inputs: point cloud; Outputs: grid/voxels.

3. graph

Purpose: neighbors and distances for grid search.

- `iter_neighbors(idx, shape, connectivity)`: valid 4/8-neighbors.
- `step_distance(a,b)`: 1 or $\sqrt{2}$ for diagonals.
- Inputs: grid index; Outputs: neighbor indices and base step cost.

4. cost

Purpose: movement cost + heuristic.

- `slope_cost(H,a,b; k,p)`: penalize slope as in the design doc (tan-based penalty).
- `heuristic_euclidean(a,b)`: admissible planar distance.
- Inputs: height grid, two cells; Outputs: scalar costs.

5. planner

Purpose: path search.

- `a_star(start, goal, neighbors, h, move_cost)`: generic A*.
- Inputs: start/goal, callbacks; Outputs: path (list of cells), cost.

6. cli

Purpose: quick demo and smoke tests.

- `python -m terrain_navigate`: runs a small demo on synthetic terrain.
- Flags: `-size`, `-k`, `-p`.

Notes

Keep functions small and testable. Prefer simple numpy-only ops first; add heavier deps only when needed.