Parallelizing Chamfer Distance Computation for Point Cloud Similarity

Problem Statement:

All-pairs (naive) Chamfer distance is an $O(N^2)$ algorithm that computes the similarity between two point clouds in $\mathbb{R}^3$. Dividing the points in 3D space and distributing the distance computation using it's associativity can bring incredible speed up to large point clouds (> 1M vertices).

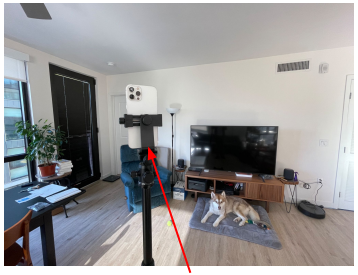
$$\mathcal{D}_{CD}(S_A, S_B) = \frac{1}{|S_A|} \sum_{a \in S_A} \min_{y \in S_B} ||a - y||_2^2 + \frac{1}{|S_B|} \sum_{b \in S_B} \min_{y \in S_A} ||b - y||_2^2$$

Challenges:

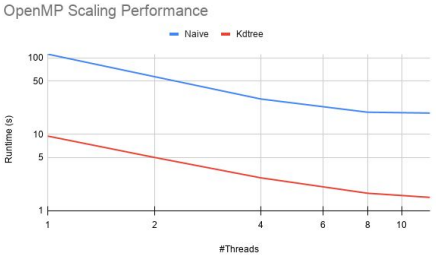
LiDAR data is inherently sparse. Consumer LiDAR sensors are noisy (see Fig. 2.d) and point clouds have non-uniform density therefore there is variance between any two scans.

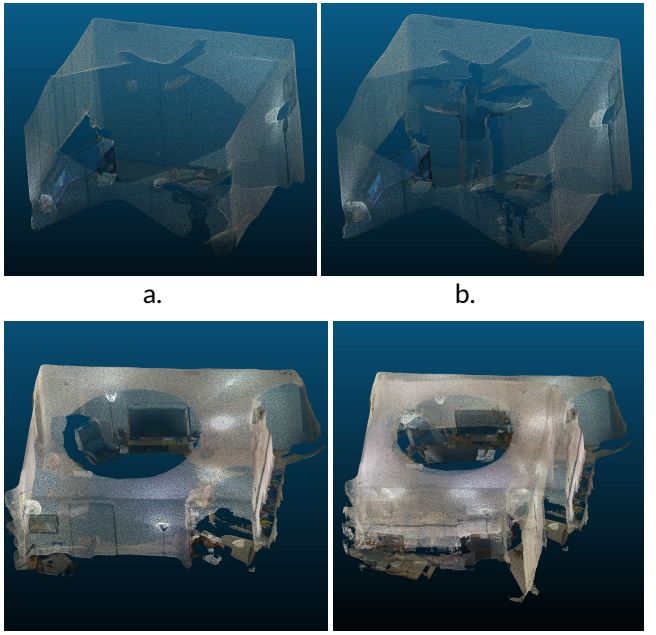
Parallelization Approach:

Notice we only need to look at "local" points. Use an octree for segmenting the 3D space (each thread grabs an octant) and run nearest neighbors to limit the search space. Use work-sharing in outer loops, reduction (for calculating the argmin in the inner loops) thereby distributing the distance computation across many threads (shared-memory) efficiently.



Tripod mounted iPhone with LiDAR sensor. iPhone is rotated slowly by hand for at least 3 full rotations to collect a scan of the environment. More rotations = more points, denser clouds





a. b. c. d.

Figure 2