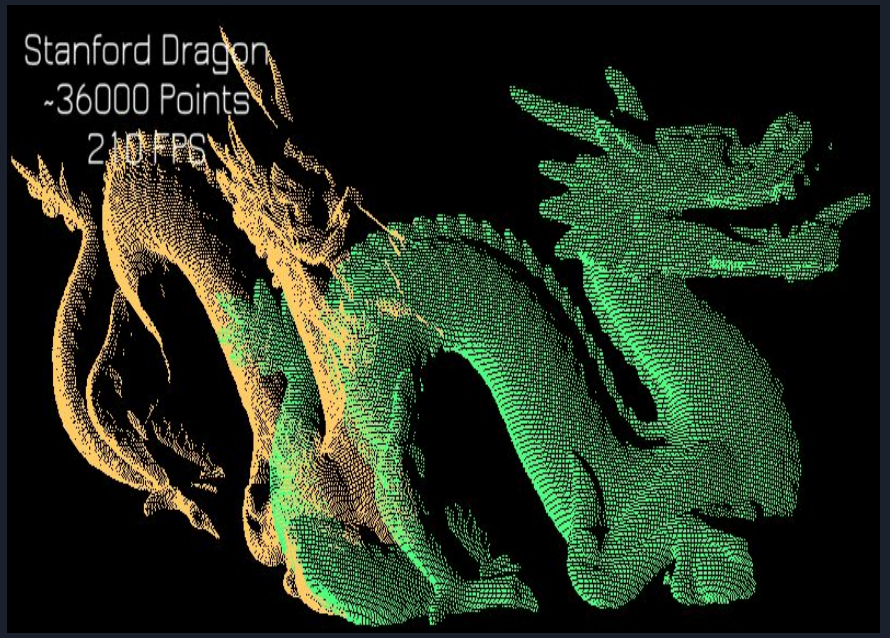
3D Point Cloud Registration using Gaussian Mixture Models

Srinath Rajagopalan, Somanshu Agarwal, Dhruv Karthik

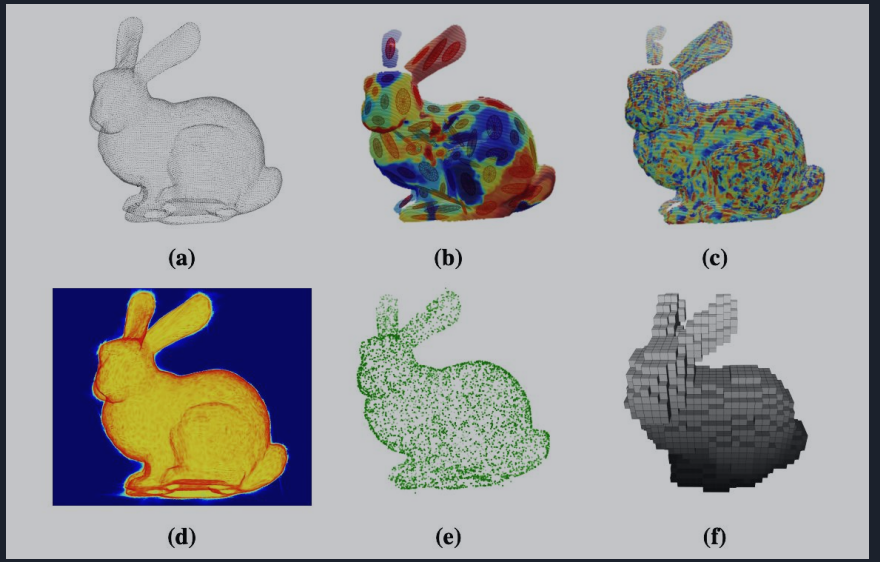
Point Cloud Registration

- Point Cloud Registration (PCR) is the task of aligning two or more point clouds by estimating the relative transformation between them.
- PCR is extensively used in the computer vision and robotics for 3D object mapping, SLAM, dense reconstruction, 3D pose estimation.



Why Probabilistic Representation?

- Deal with outliers and noise
- Continuous geometry - representation is differentiable!
- Avoids artifacts introduced by discretization
- Re-sampling, Heat Maps, Occupancy Grid Maps



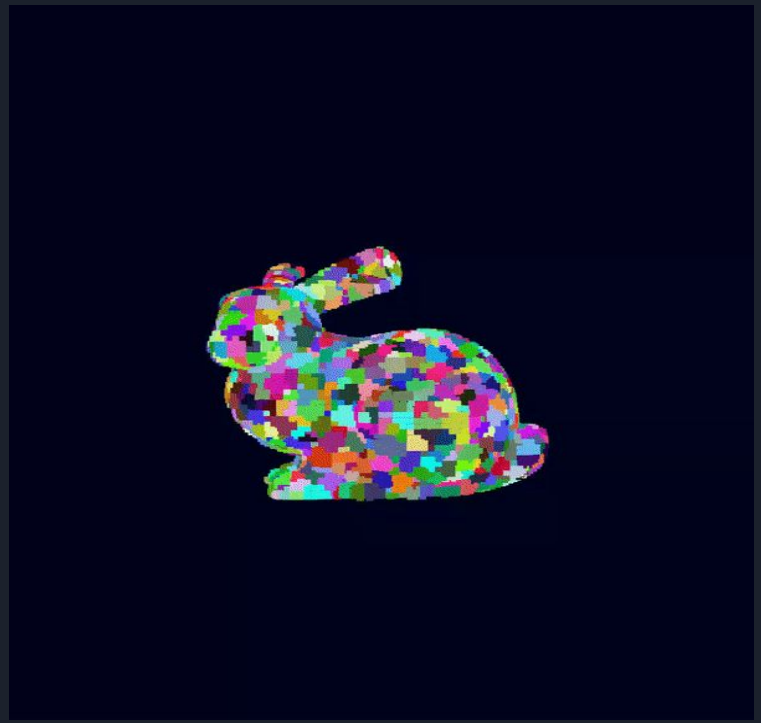
Source - Eckart et al 2016

Gaussian Mixture Models - 2D

Clusterted into one Gaussian

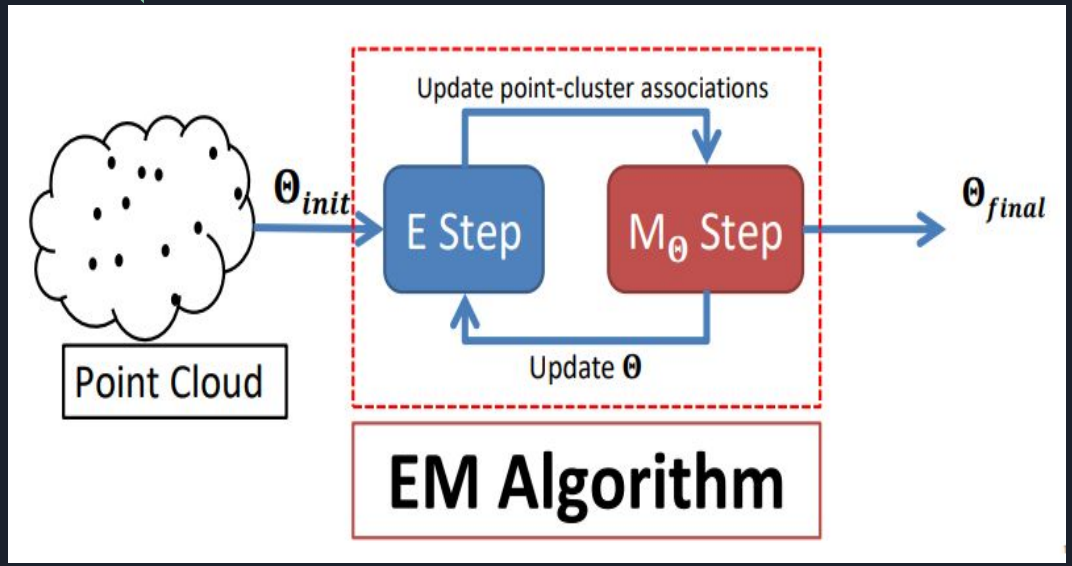
Gaussian Mixture Models - 3D

100 GMM Components



800 GMM Components

Expectation-Maximization Algorithm



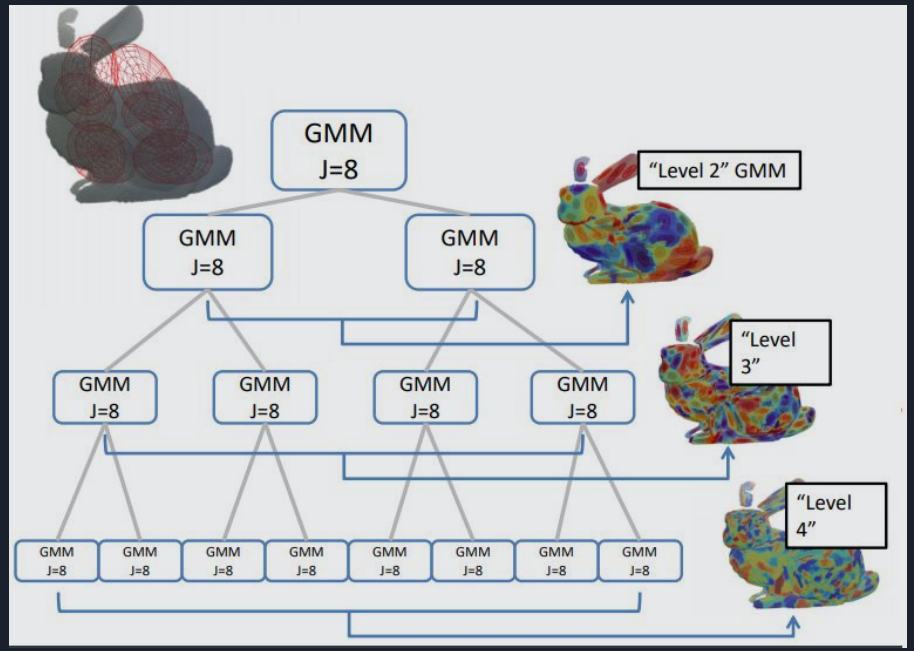
- Point Cloud Data is fed as input to the EM algorithm with some parameter initializations.
- Algorithm iteratively updates point-cluster associations and the parameters.
- When convergence is reached, the final parameters can be used to model the Point Cloud Data.

Source: NVIDIA - Accelerating GMMs on 3D Point Cloud

GMM and Sparsity

Points in red region unlikely to be clustered with points in yellow region but GMM searches the whole space!

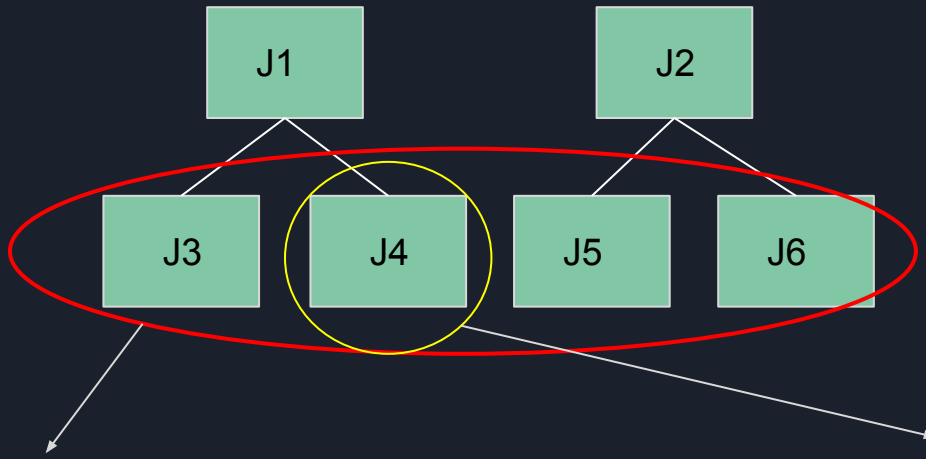
Hierarchical Gaussian Mixture Models



Source: Nvidia - GPU Accelerated 3D Point Clouds

- Conventional GMM evaluates ALL the components for EVERY point
- Hierarchical GMM's limits the search space to logarithmic by leveraging Octree-like division of points in clusters

HGMM in CUDA

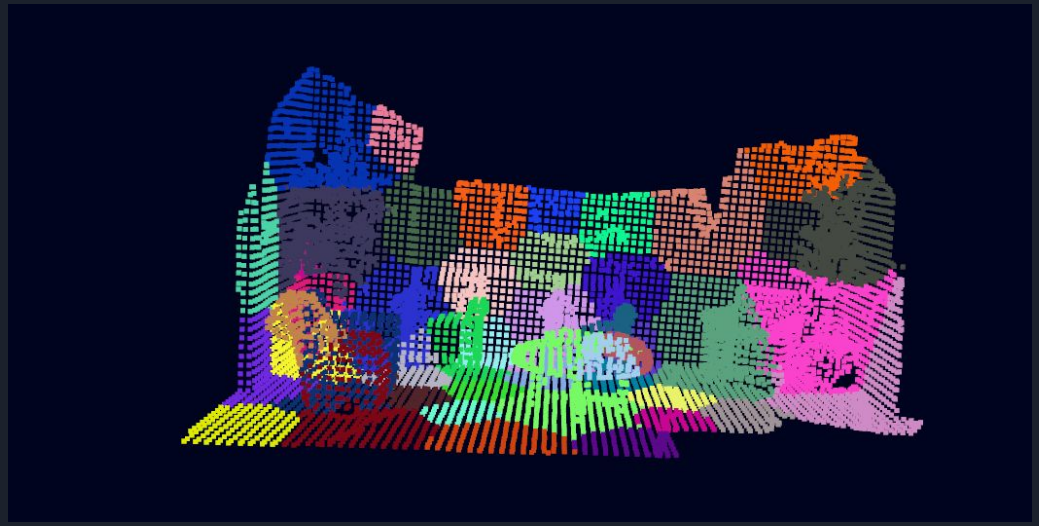


Parallelism per level across points

Accumulate moments per cluster per level with CUDA Atomic Add

Use Cases

Unsupervised Segmentation



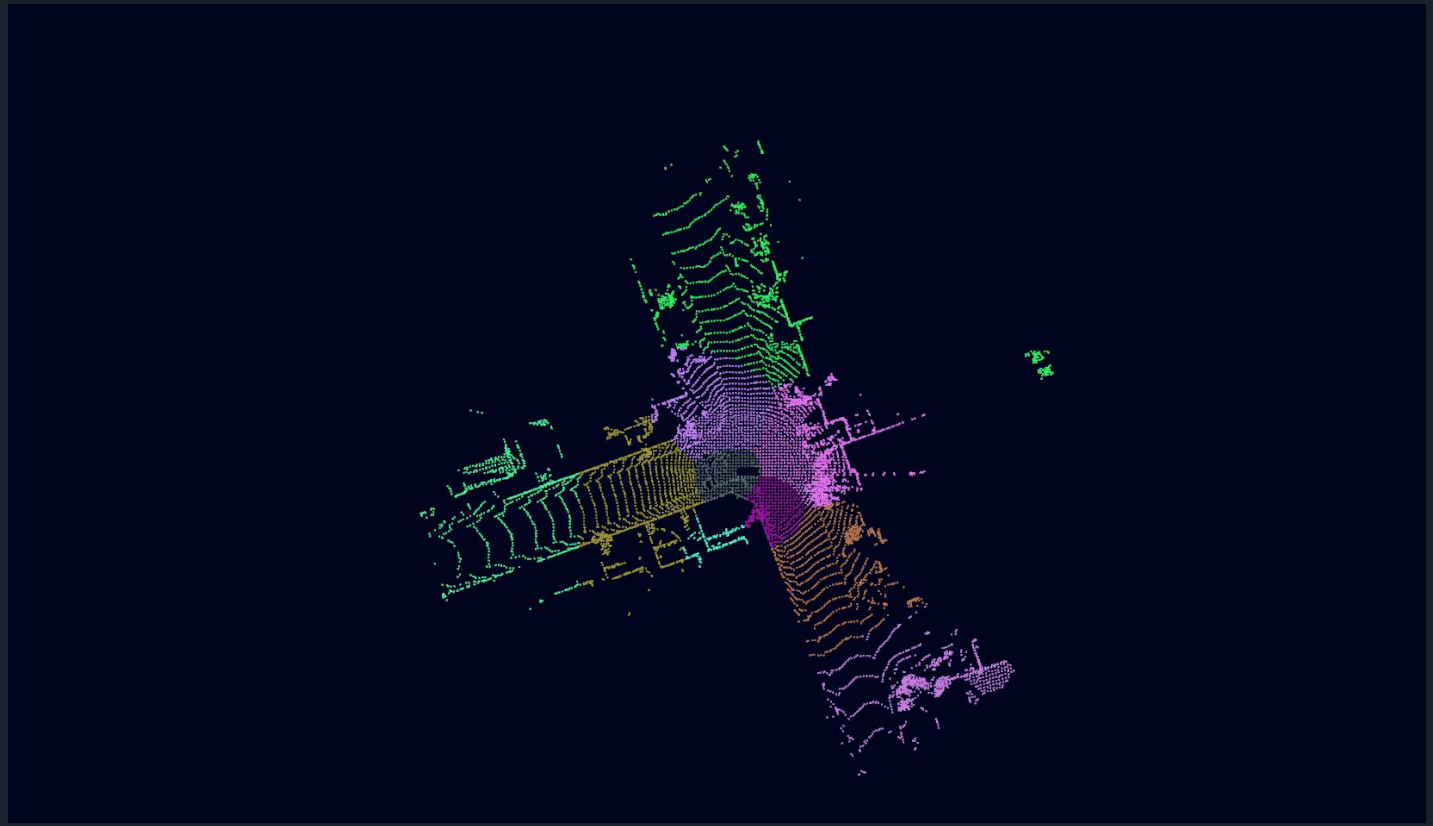
- Good starting point for segmentation when ground truth is unavailable
- Lounge segmented into 10 clusters with all points of a furniture grouped together

Waymo Open Dataset

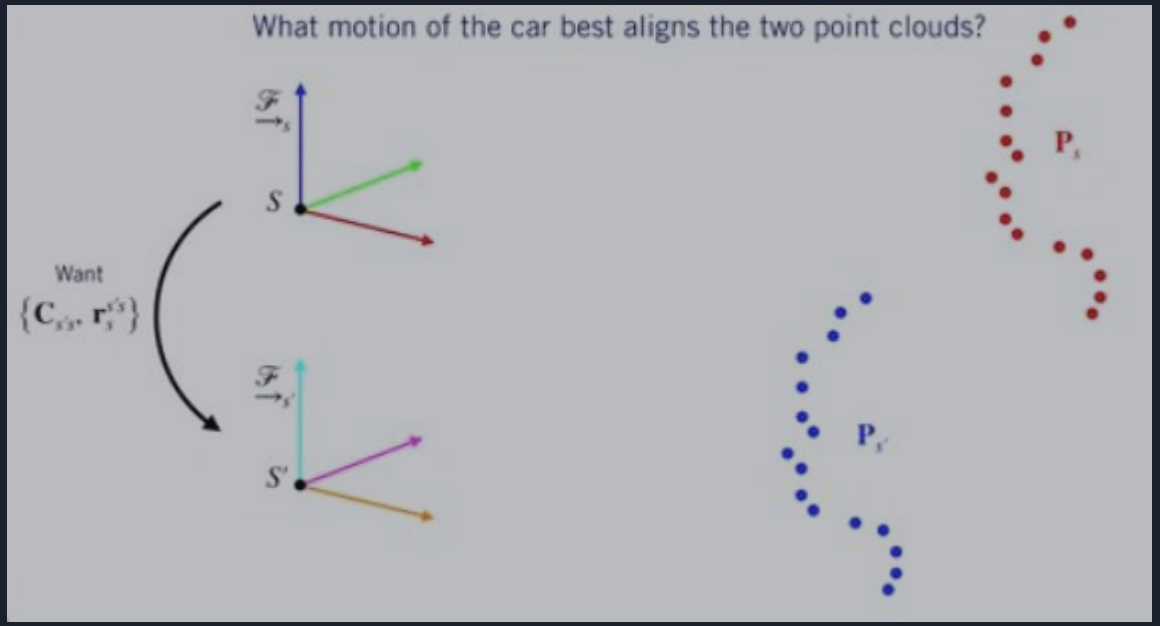
1) 1,000 segments of 20s each, collected at 10Hz (200,000 frames) in diverse geographies and conditions
2) Sensor data -
   - 1 mid-range lidar 4 short-range lidars 5 cameras (front and sides)
   - Synchronized lidar and camera data
   - Lidar to camera projections
   - Sensor calibrations and vehicle poses

Source - Waymo Open Dataset

GMM on a stream of LIDAR Point Clouds

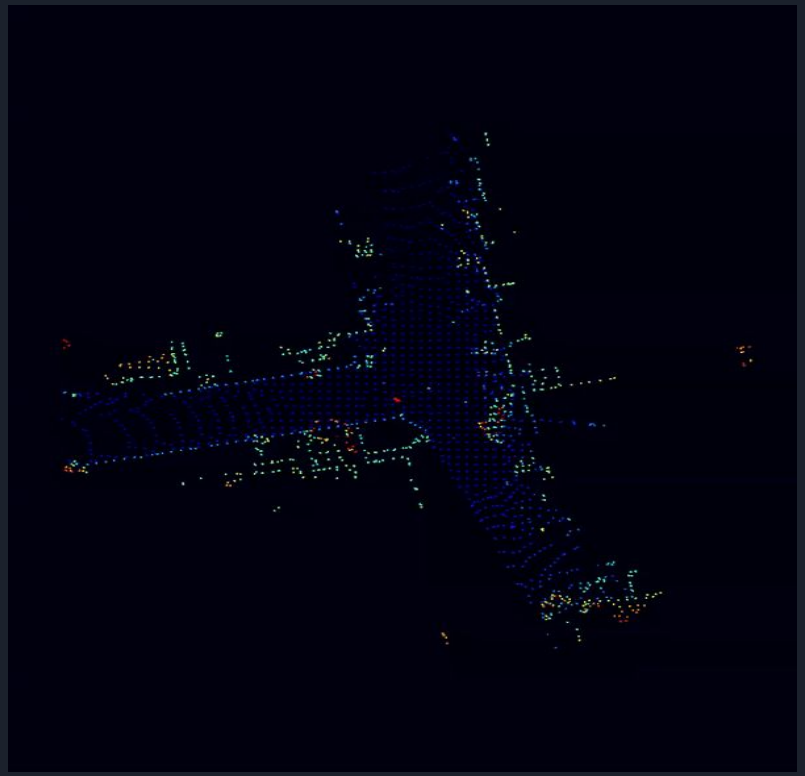


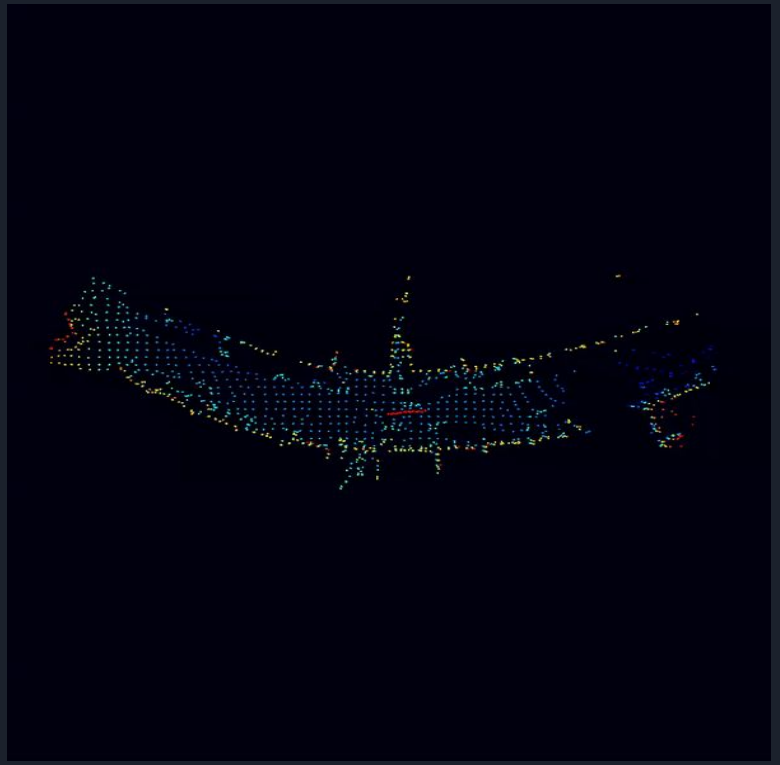
Localization with PCR



Source - State-Estimation and Localization for Self-Driving Cars

Localization with PCR





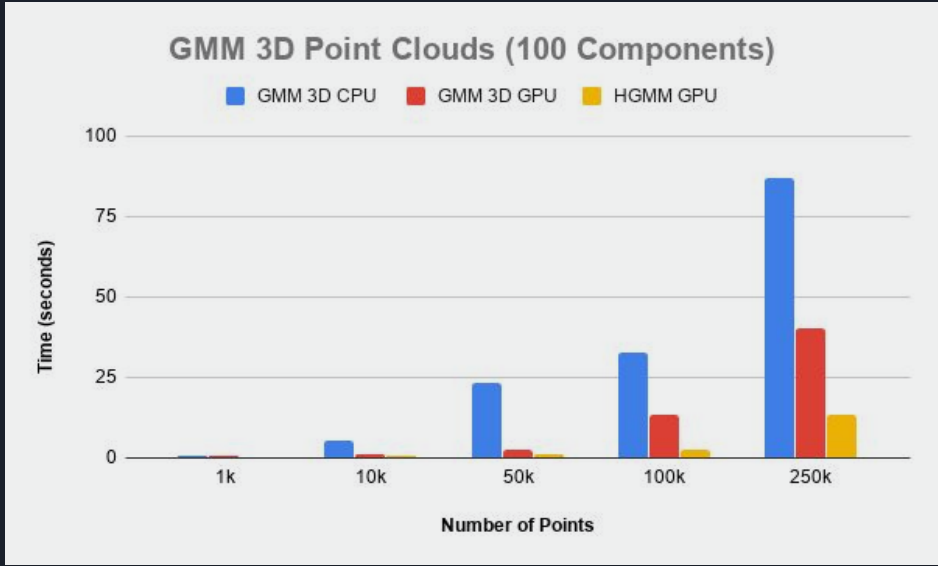
Implementation and Performance Analysis

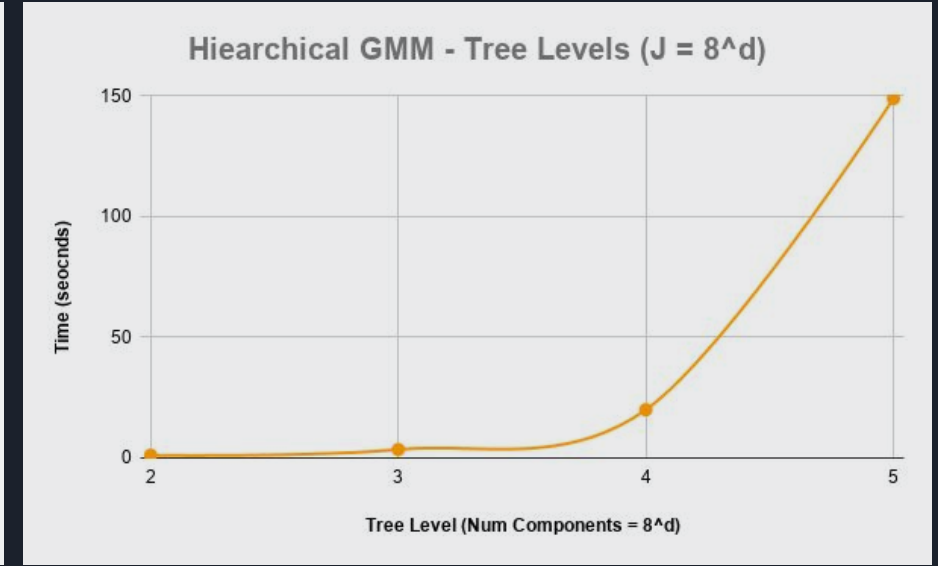
CuPy and Numba

- CuPy - numpy arrays on the GPU. NumPy API but computations accelerated with the GPU
- Numba - write CUDA kernels in Python (JIT compiled to CUDA-C code)

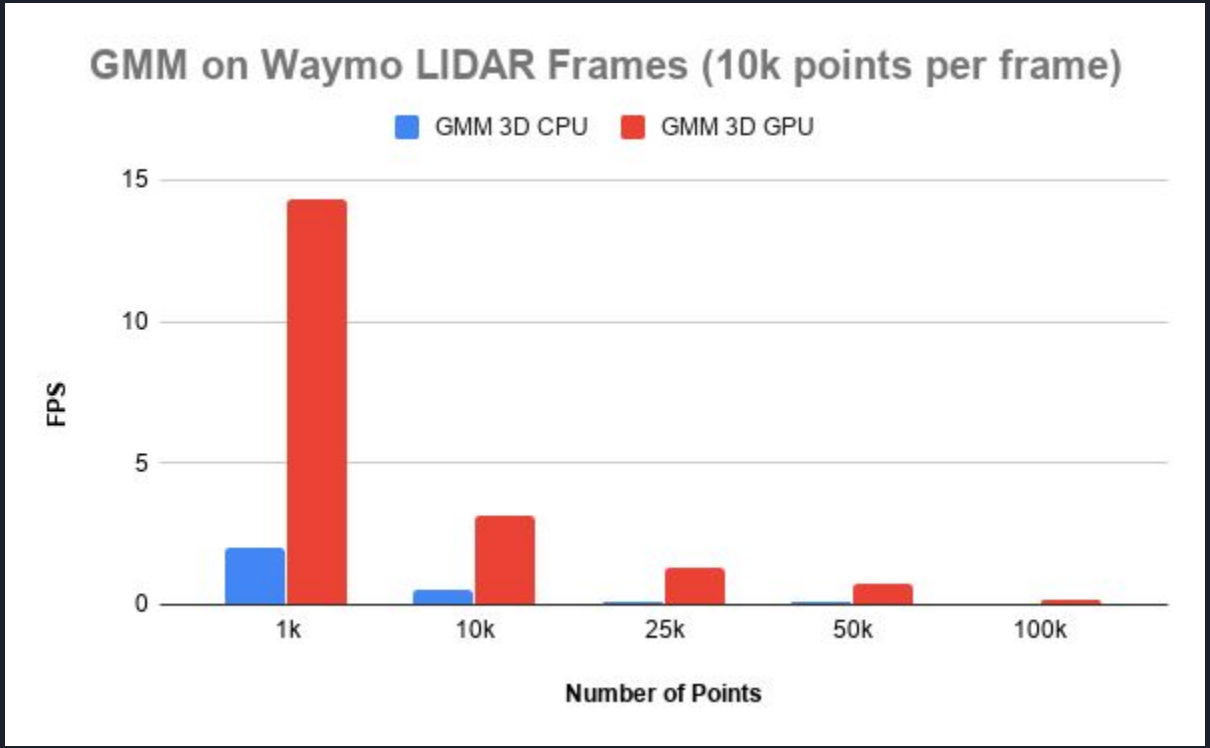
```
from numba import cuda
@cuda.jit
def increment_by_one(an_array):
    # Thread id in a 1D block
    tx = cuda.threadIdx.x
    ty = cuda.blockIdx.x
    bw = cuda.blockDim.x

    idx = tx + ty * bw
    if idx < an_array.size:
        an_array[idx] += 1
```

Performance Analysis



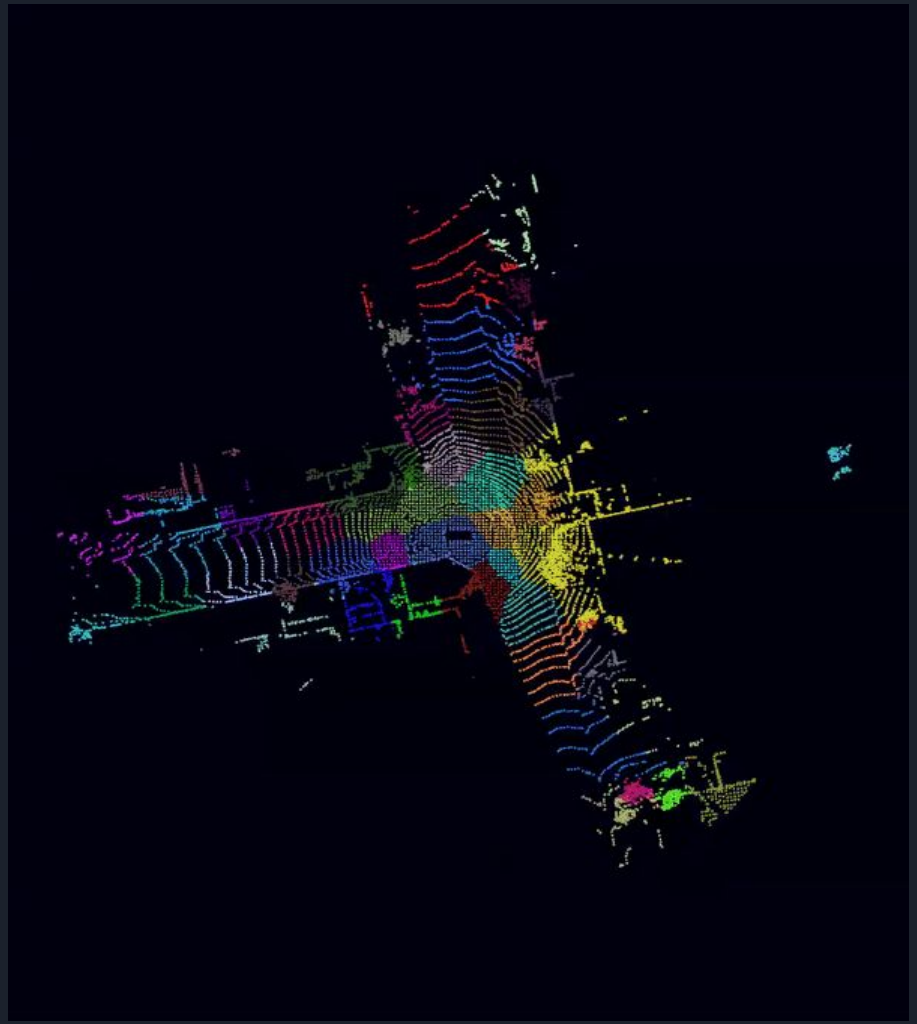


Performance Analysis



Conclusion

- GMMs (CPU and GPU) on different 3D point cloud datasets
- Point Cloud Registration using GMM
- Hierarchical GMMs (CPU and GPU) for scaling to large point clouds
- GMMs on LIDAR point clouds from Waymo Open Dataset
- Localization with LIDAR point clouds using PCR




Waymo Open Dataset
~120K Points
110 FPS