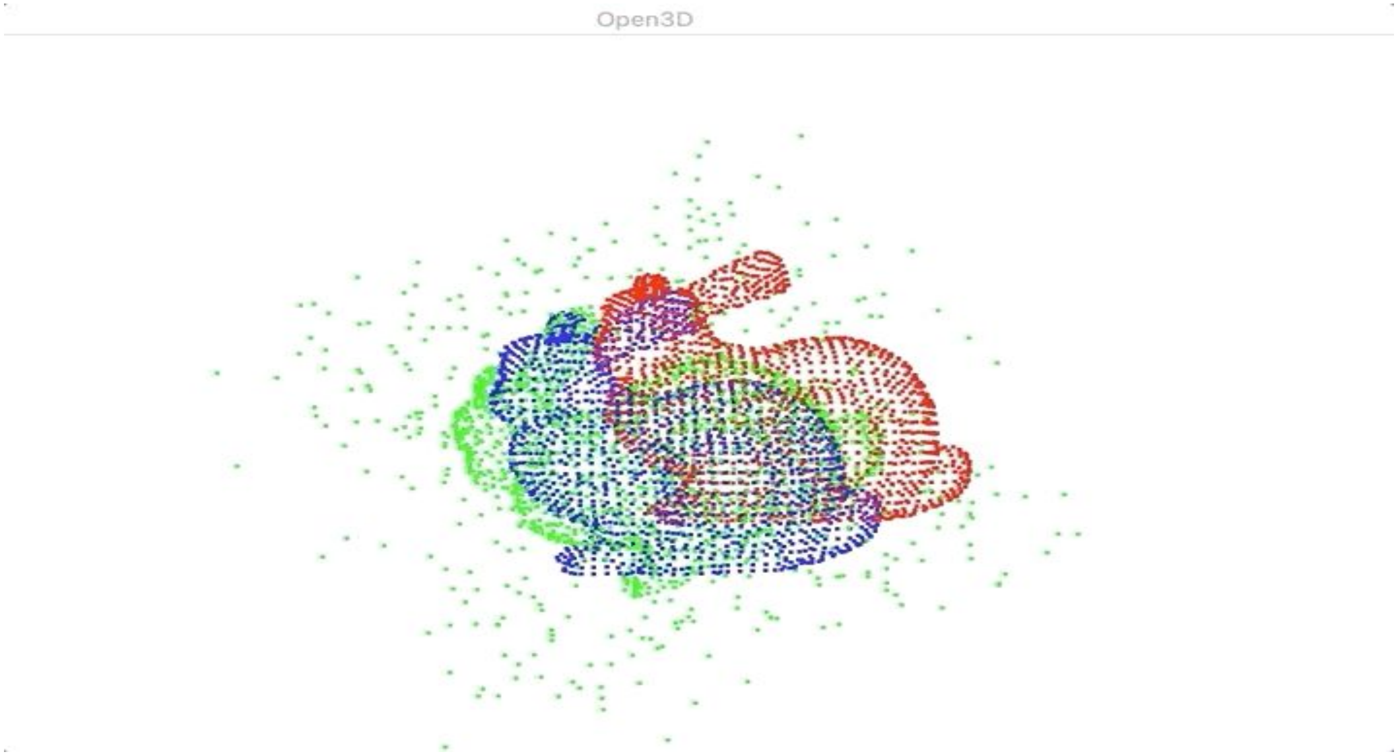
3D Point Cloud Registration with Gaussian Mixture Models- Milestone 2

Team Members:
Somanshu Agarwal
Srinath Rajagopalan
Dhruv Karthik

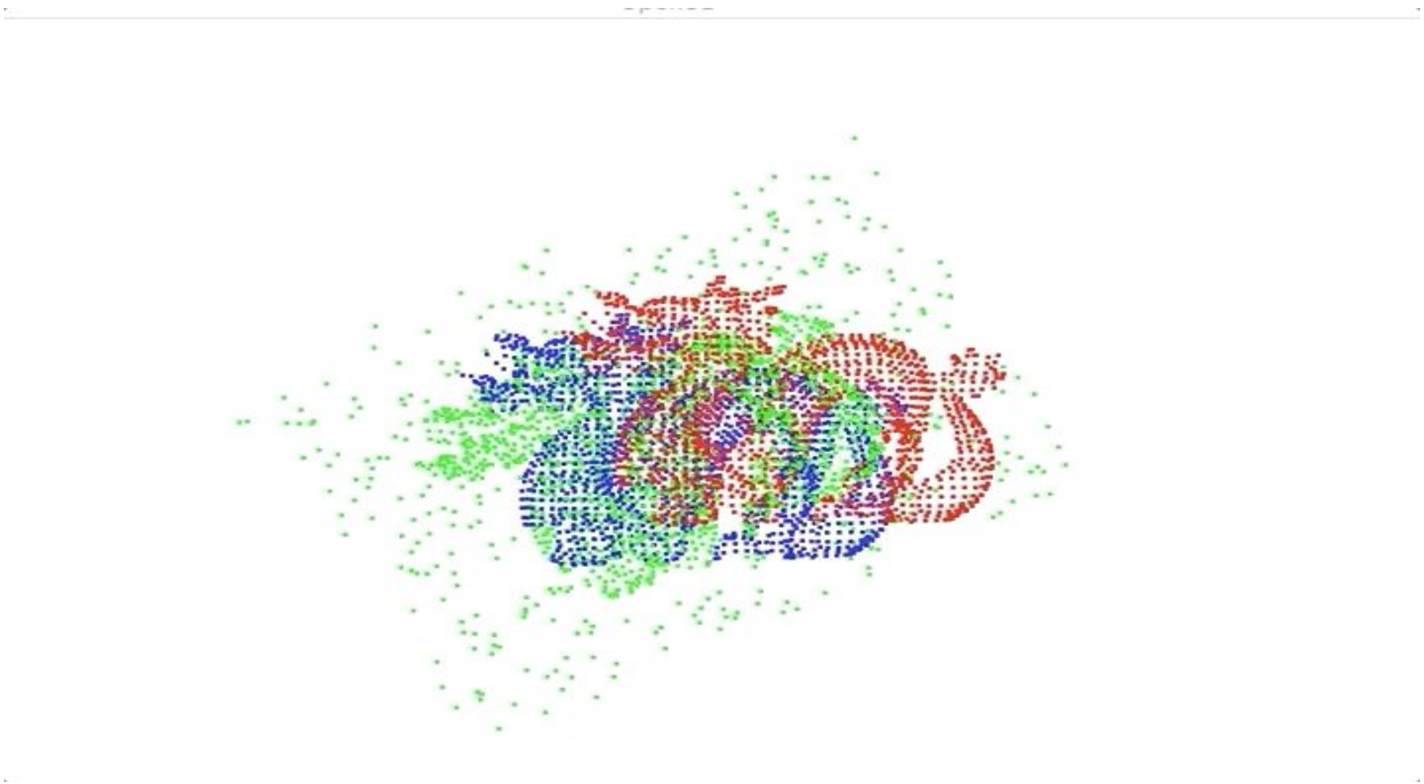
Tasks Accomplished -

- Prototype with Python, Scikit-Learn, Open3D
- Implemented Gaussian Mixture Model on CPU
- Implemented Gaussian Mixture Model on GPU
- Tested different visualizers Intel-Labs Open3D vs OpenGL
- Tested BFGS Solvers on CUDA
- Testing concurrency through CUDA Streams

Visualization using Open3D



Visualization using Open3D



Gaussian Mixture Model - Log Likelihood

$$\mathcal{L}\left(\mathcal{D}|\mu,\Sigma\right) = \sum_{n=1}^{N} \log p(x_n) = \sum_{n=1}^{N} \log\left[\sum_{k=1}^{K} \pi_k p\left(x_n|\mu_k,\Sigma_k\right)\right]$$

Gaussian Mixture Model - E Step

- Given a cluster, what is the probability of a point belonging to it?
- Posterior Probability - calculated for all points and for all components

$$P(Z_i = k | X_i) = \frac{P(X_i | Z_i = k) P(Z_i = k)}{P(X_i)} = \frac{\pi_k N(\mu_k, \sigma_k^2)}{\sum_{k=1}^{K} \pi_k N(\mu_k, \sigma_k)} = \gamma_{Z_i}(k)$$

Parallelize with Kernels and Concurrency with Streams

- Compute unnormalized posterior for each component for all points simultaneously
- Launch each kernel in a separate CUDA Stream (10 streams for 10 components at a time)
- Synchronize and Normalize

Gaussian Mixture Model - M Step

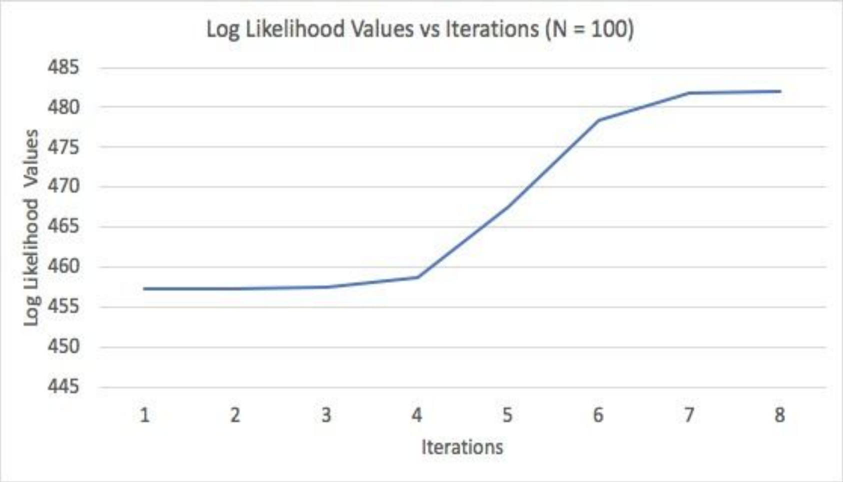
$$\pi_k^{(t+1)} = \frac{\pi_k^{(t)}\Gamma_k}{\sum_{i=1}^{K} \pi_i^{(t)}\Gamma_i} \qquad \log \pi_k^{(t+1)} = \log \pi_k^{(t)} + \log \Gamma_k - a - \log \left[\sum_{i=1}^{K} \exp \left(\log \pi_i^{(t)} + \log \Gamma_i - a \right) \right]$$

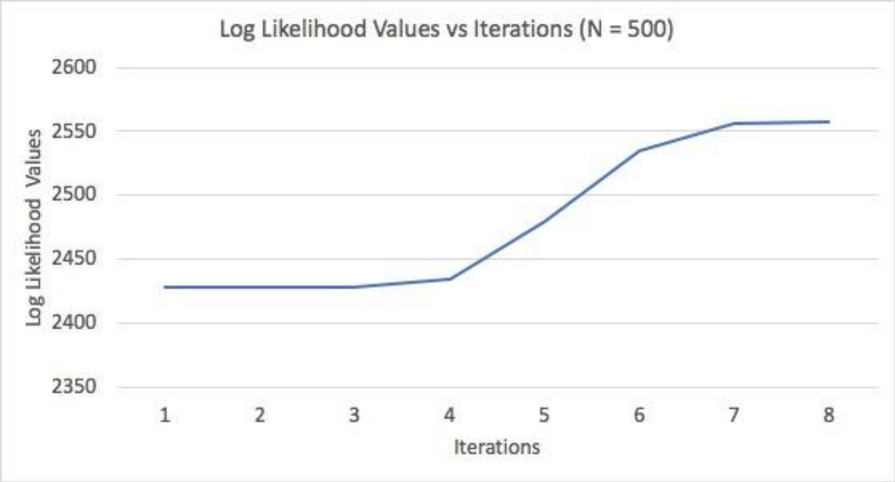
$$\mu_k^{(t+1)} = \frac{\sum_{n=1}^{N} x_n \gamma_{n,k}}{\Gamma_k} \qquad \mu_k^{(t+1)} = \frac{\sum_{n=1}^{N} x_n \exp \log \gamma_{n,k}}{\exp \log \Gamma_k}$$

$$\Sigma_k^{(t+1)} = \frac{\sum_{n=1}^{N} (x_n - \mu_k^{(t+1)})(x_n - \mu_k^{(t+1)})^T \gamma_{n,k}}{\Gamma_k}$$

$$\Sigma_k^{(t+1)} = \frac{\sum_{n=1}^{N} (x_n - \mu_k^{(t+1)})(x_n - \mu_k^{(t+1)})^T \exp \log \gamma_{n,k}}{\exp \log \Gamma_k}$$

Maximum Log Likelihood for EM





The above graphs show that max likelihood is increasing with higher rates initially and then almost saturates to a particular value.

GMM Registration - Minimize L2 Distance

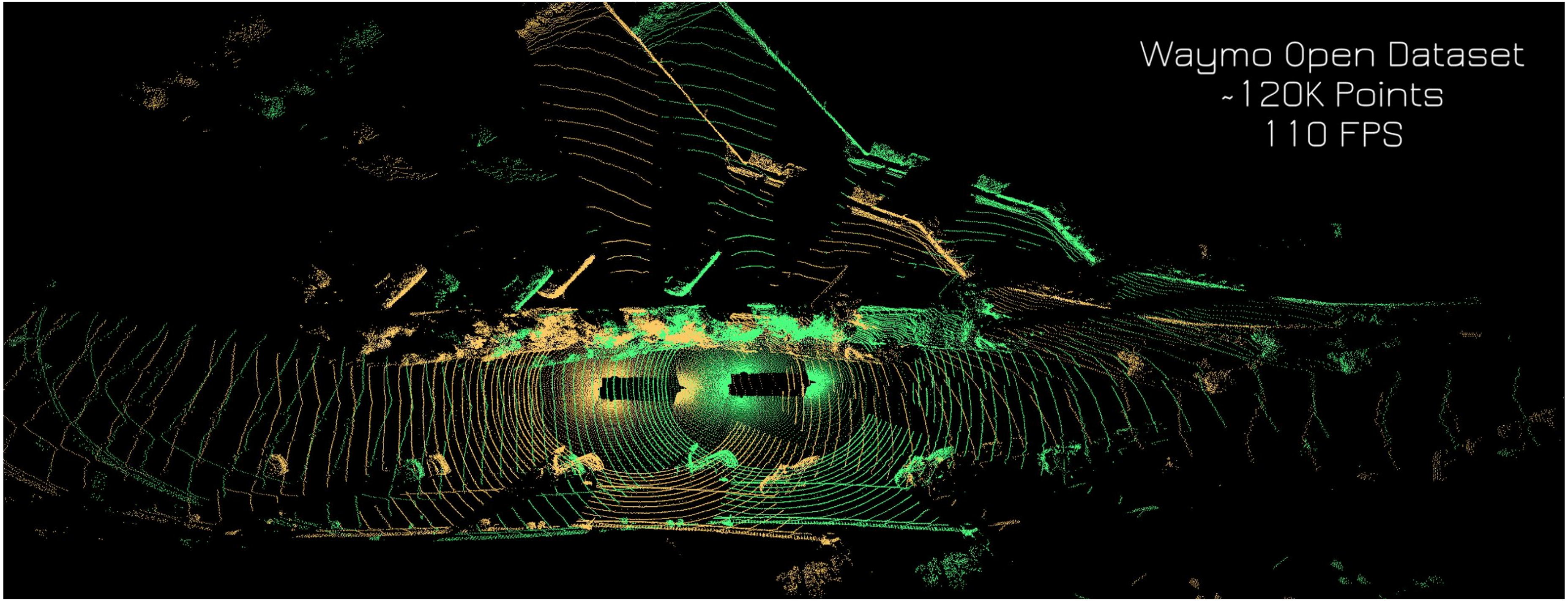
$$d_{L_2}(\mathcal{S},\mathcal{M},\theta)=\int\left(gmm(\mathcal{S})-gmm(T(\mathcal{M},\theta))\right)^2 dx,$$

$$d(f,g,\mathbf{R},\mathbf{t})=\int(f_{\mathbf{R},\mathbf{t}}^2-2f_{\mathbf{R},\mathbf{t}}g+g^2)dx$$

$$\int f_{\mathbf{R},\mathbf{t}}g \;=\; \sum_{i=1}^{m}\sum_{j=1}^{n}\alpha_i\beta_j\phi(0|\mathbf{R}\mu_i+\mathbf{t}-\nu_j,\mathbf{R}\Sigma_i\mathbf{R}^T+\Gamma_j)$$

Testing several BFGS Optimizers in CUDA

Visualization using OpenGL



Future RoadMap

1. Milestone 3 (12/02) :
   a. Identify bottlenecks in GMM implementation
   b. Complete registration with GMM with BFGS solver on CUDA
   c. Implement Hierarchical GMM on GPU
2. Milestone 4 (12/09) :
   a. Incorporate Adaptive Scaling for HGMM
   b. Implement Real-Time Localization using HGMM Registration
   c. Test on a live dataset