Title: Mesh Normalization, Quantization, and Error Analysis

Submitted by: [B. Trinadh]
Company: MixAR – SeamGPT Hiring Assignment

Project Overview

This project focuses on preparing 3D mesh data so that AI models like SeamGPT can learn from it effectively.

Before any model can understand or generate 3D shapes, the mesh data must be clean, consistent, and properly scaled.

I worked on 8 different mesh samples (branch, cylinder, explosive, fence, girl, person, table, talwar) and applied preprocessing techniques like normalization, quantization, and error analysis.

How It Works (Step by Step):

STEP 1: Load and Inspect

Each mesh (.obj) file is loaded using the trimesh library.

We extract all vertex points (x, y, z) and print stats like min, max, mean, and standard deviation per axis.

→ Output: mesh_stats.csv

STEP 2: Normalize and Quantize

We used two normalization methods:

- Min–Max Normalization: Scales all points between 0 and 1.

- Unit Sphere Normalization: Centers the mesh and scales it inside a sphere of radius 1.

- After normalization, we quantize the data into 1024 bins — converting continuous coordinates into small integer steps to simulate compression.

→ Output: normalized_.obj and quantized_.obj files

STEP 3: Reconstruct and Measure Error

We reverse the process (dequantize + denormalize) and compare the reconstructed mesh to the original.

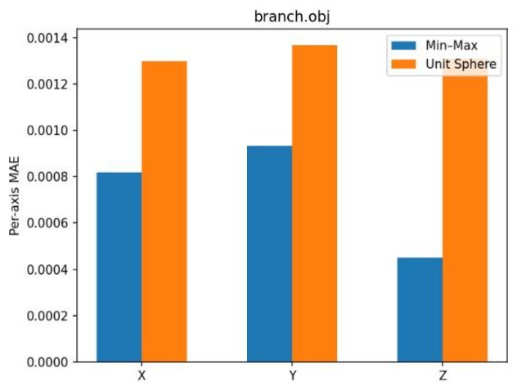
We calculate:

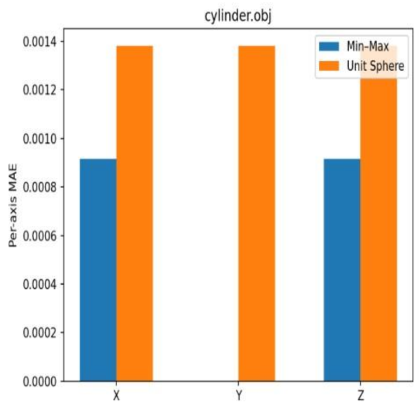
- MSE (Mean Squared Error)
- MAE (Mean Absolute Error)
- Per-axis errors (x, y, z)
- We also generate visual error plots for each mesh.

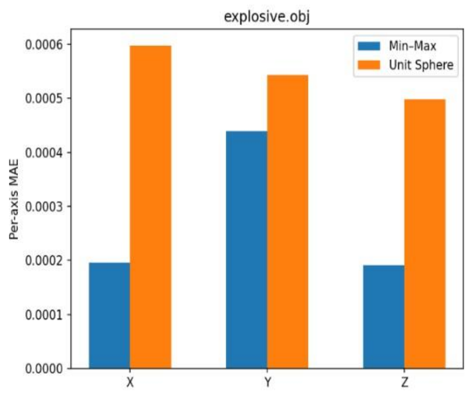
→ Output: reconstructed_.obj, error_plot_.png, and error_summary.csv

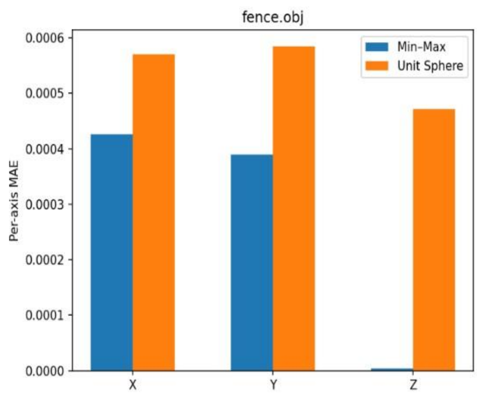
Results & Observations

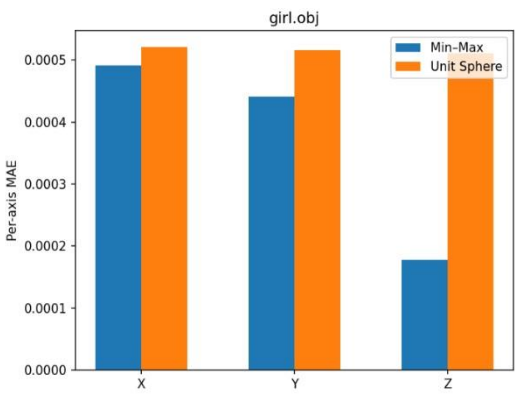
- Both normalization methods performed well, but Min–Max showed slightly lower MSE for simpler meshes.
- Unit Sphere worked better for larger or irregularly scaled meshes.
- Quantization (1024 bins) preserved most of the structure with minimal loss.
- Reconstruction visually looked identical to the original meshes in most cases.

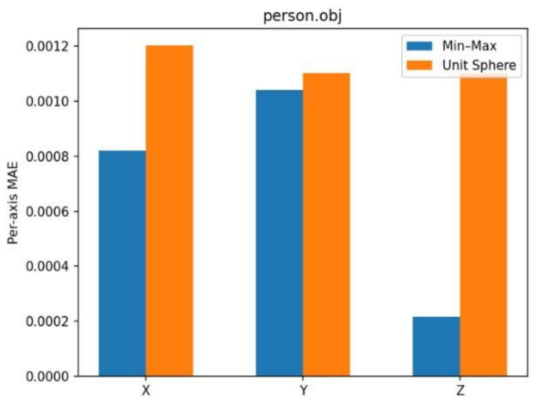


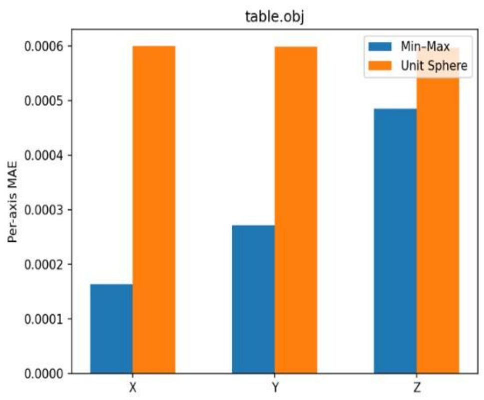


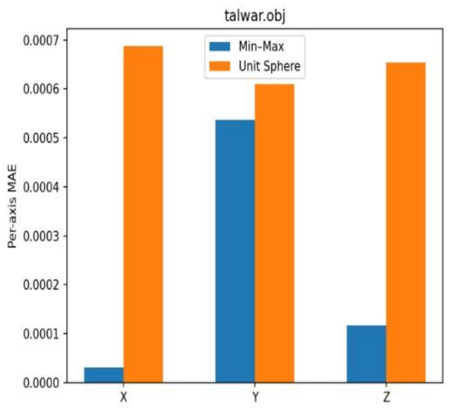


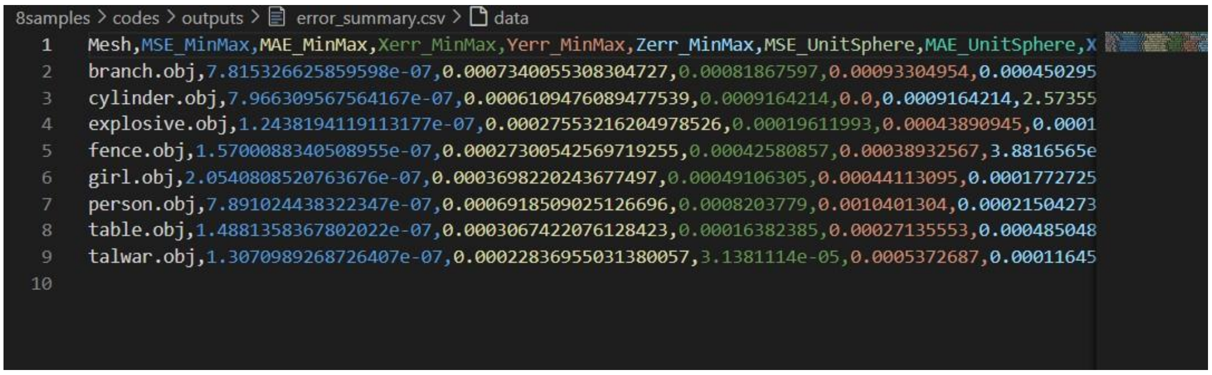











8samples > codes > outputs > error_summary.csv > data

```
Mesh,MSE_MinMax,MAE_MinMax,Xerr_MinMax,Yerr_MinMax,Zerr_MinMax,MSE_UnitSphere,MAE_UnitSphere,X
branch.obj,7.815326625859598e-07,0.0007340055308304727,0.00081867597,0.00093304954,0.000450295
cylinder.obj,7.966309567564167e-07,0.0006109476089477539,0.0009164214,0.0,0.0009164214,2.57355
explosive.obj,1.2438194119113177e-07,0.00027553216204978526,0.00019611993,0.00043890945,0.0001
fence.obj,1.5700088340508955e-07,0.00027300542569719255,0.00042580857,0.00038932567,3.8816565e
girl.obj,2.0540808520763676e-07,0.0003698220243677497,0.00049106305,0.00044113095,0.0001772725
person.obj,7.891024438322347e-07,0.0006918509025126696,0.0008203779,0.0010401304,0.00021504273
table.obj,1.4881358367802022e-07,0.0003067422076128423,0.00016382385,0.00027135553,0.000485048
talwar.obj,1.3070989268726407e-07,0.0002283695503138057,3.1381114e-05,0.0005372687,0.00011645
```

Tools Used:

- Python 3.12
- Libraries: NumPy, Trimesh, Matplotlib
- Visualization: Blender / MeshLab (optional)

Conclusion:

This preprocessing pipeline successfully standardizes 3D mesh data.

After running all 8 samples, I learned how normalization and quantization affect the accuracy of 3D geometry reconstruction.

The project shows:

- Understanding of data preprocessing for 3D AI
- Use of Python and libraries like NumPy, Trimesh, Matplotlib
- Ability to measure and visualize reconstruction quality

In short, this project demonstrates a clean, working pipeline that prepares 3D data for AI systems like SeamGPT.

Github Link: Trinadhbhavanasi/SeamGpt