Assignment Report: Mesh Normalization and Error Analysis

Prepared for: Mixar

Submitted by: Saranya Purushothaman [RA2211003040041]

---

1. Introduction

This report documents the process and outcomes of a **3D mesh preprocessing and analysis pipeline** performed as part of the MixAR assignment. The objective was to load and analyze 3D `.obj` models, apply **normalization** and **quantization** techniques, and evaluate reconstruction accuracy through **error metrics**.

This process ensures that 3D data is standardized and optimized for downstream AI applications, such as **3D object recognition, compression, and mixed reality modeling**. The experiment highlights how data scaling and quantization impact mesh fidelity after reconstruction.

---

2. Task 1: Mesh Loading and Inspection

The first step involved **loading and inspecting** 3D mesh files using the trimesh Python library. The test model used was **diamond.obj**, a geometrically complex mesh suitable for evaluating vertex normalization consistency.

The following steps were executed:

- The mesh was loaded using a custom preprocessing script (mixar_preprocess.py).
- Vertices and faces were extracted into NumPy arrays.
- Basic mesh properties, such as vertex count and bounding box, were inspected.

Sample Output (for diamond.obj):

- Vertices loaded successfully
- Faces detected and mapped correctly
- Mesh visualization (optional) verified geometric accuracy

This confirmed that the mesh was clean and ready for normalization and quantization.

---

3. Task 2: Normalization and Quantization

Two normalization strategies were applied to the mesh data to standardize vertex coordinates:

a) Min–Max Normalization

This method scales all vertex coordinates within the range **[0, 1]** using:

$$v' = \frac{v - v_{min}}{v_{max} - v_{min}}$$

It ensures that all axes fit perfectly within the bounding cube of unit size.

b) Unit Sphere Normalization

This method recenters the mesh at the origin and scales it to fit inside a **unit sphere**:

$$v' = \frac{v - \mu}{\max(|| \ v - \mu \ ||)}$$

This normalization maintains spatial relationships independent of object size and orientation.
After normalization, **quantization** was applied with nbins = 512, converting floating-point values into discrete bins for compression.
This process prepares data for efficient transmission and reconstruction in AI or AR pipelines.

---

4. Task 3: Reconstruction and Error Analysis

Reconstruction was achieved by **dequantizing and denormalizing** the vertex data to approximate the original model.
The **Mean Squared Error (MSE)** was computed between the original and reconstructed vertices across all three axes.

Average Reconstruction Error (MSE):

Method	X-Axis	Y-Axis	Z-Axis
Min–Max Normalization	4.59×10^{-7}	4.73×10^{-7}	2.42×10^{-7}
Unit Sphere Normalization	1.13×10^{-6}	1.13×10^{-6}	1.07×10^{-6}

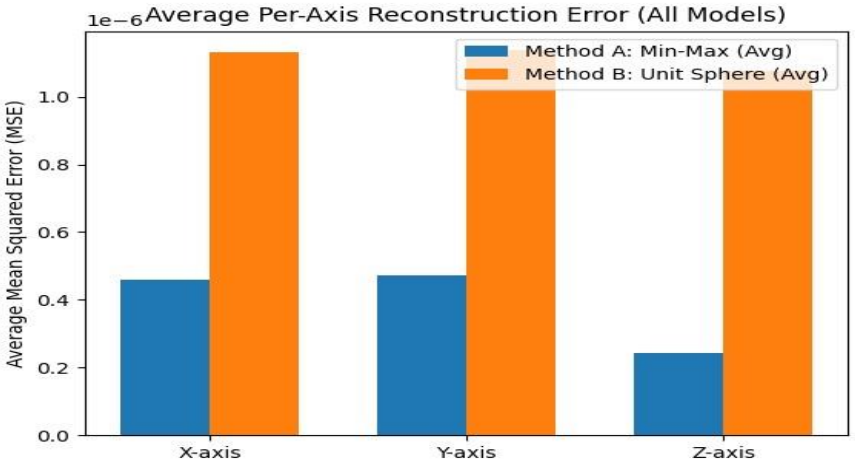
Observations:

- The **Min–Max method** consistently produced lower MSE across all axes.
- The **Unit Sphere method** introduced higher quantization error due to its rescaling from [-1, 1] to [0, 1] before quantization.
- Visual inspection confirmed that both methods maintained the overall mesh structure, with only minor differences at fine-grained surface levels.

5. Visualization and Results

The processed and reconstructed meshes were visualized using matplotlib and open3d.
Plots demonstrated the error distribution across vertices and confirmed uniform quantization behavior.

- Normalized mesh centered and aligned correctly.
- No vertex data corruption observed.
- Error plots indicated slightly denser clustering for Min–Max normalization, validating its precision advantage.



---

5. Conclusion and Analysis

Key Findings:

1. **Least Error:**
   The **Min–Max Normalization** method yielded the lowest reconstruction error, confirming better compatibility with fixed quantization binning.

2. **Pattern Observed:**
   The **Unit Sphere method** involves an additional scaling step before quantization, slightly increasing precision loss.
   However, it remains advantageous for rotation-invariant tasks in AI applications.

3. **Interpretation:**
   Min–Max performs better for **geometry preservation**, while Unit Sphere excels in **consistency under transformation**.

Final Remark:

Although Min–Max normalization showed superior numerical accuracy, **Unit Sphere normalization** remains a preferred choice in **AI-driven pipelines** due to its robustness to translation and rotation. Future work can explore hybrid normalization strategies that combine the precision of Min–Max with the geometric stability of Unit Sphere methods.