3D MESH NORMALIZATION, QUANTIZATION, AND ERROR ANALYSIS

MIXAR

VIRTUAL ASSIGNMENT

NAME: SUCHISMITA ACHARYA

REGISTRATION NUMBER: RA2211003020675

DEGREE: BACHELOR OF TECHNOLOGY

COURSE: COMPUTER SCIENCE AND ENGINEERING

SEMESTER AND YEAR: 7^{TH} SEMESTER IV YEAR

Table of Contents

Table of Figures

Task 1
Mesh Inspection and Statistics

Before any preprocessing was applied, each 3D model was examined to understand its basic geometric characteristics and verify that all meshes were correctly scaled, complete, and free from anomalies. The inspection process calculated, for each model, the total number of vertices together with the minimum, maximum, mean, and standard-deviation values of the vertex coordinates along the X, Y, and Z axes. These statistics give a quick overview of each object's size, shape, and spatial distribution prior to normalization.

1.1 Summary of Mesh Statistics

The following table summarises the raw measurements for all eight meshes.

Mesh Name	Vertices	X min	X max	Y min	Y max	Z min	Z max
branch	2 767	–0.8516	0.8496	0.0000	1.9004	–0.4648	0.4629
cylinder	192	–1.0000	1.0000	–1.0000	1.0000	–1.0000	1.0000
explosive	2 844	–0.1996	0.1996	0.0000	1.0000	–0.1971	0.1971
fence	1 090	–0.5000	0.5000	0.0000	0.8432	–0.0225	0.0225
girl	8 400	–0.5000	0.5000	0.0000	0.9044	–0.1814	0.1814
person	3 106	–0.8438	0.8418	0.0000	1.9004	–0.2129	0.2109
table	3 148	–0.2089	0.2089	0.0000	0.6118	–0.5000	0.5000
talwar	1 681	–0.0319	0.0319	0.0000	1.0000	–0.1171	0.1171

These results show that the models vary greatly in both vertex count and spatial extent. For example, girl.obj contains over 8 000 vertices, while cylinder.obj is a simple primitive with fewer than 200. Most objects lie within ±1 unit along X and Z, extending vertically up to roughly 2 units along Y.

1.2 Detailed Example — branch.obj

The branch mesh represents a small natural fragment with mild curvature and fine surface irregularities. Its numerical statistics are listed below.

Axis	Min	Max	Mean	StdDev
X	–0.8516	0.8496	0.0754	0.3434
Y	0.0000	1.9004	1.0874	0.4570
Z	–0.4648	0.4629	0.1220	0.2001

Number of Vertices: 2767

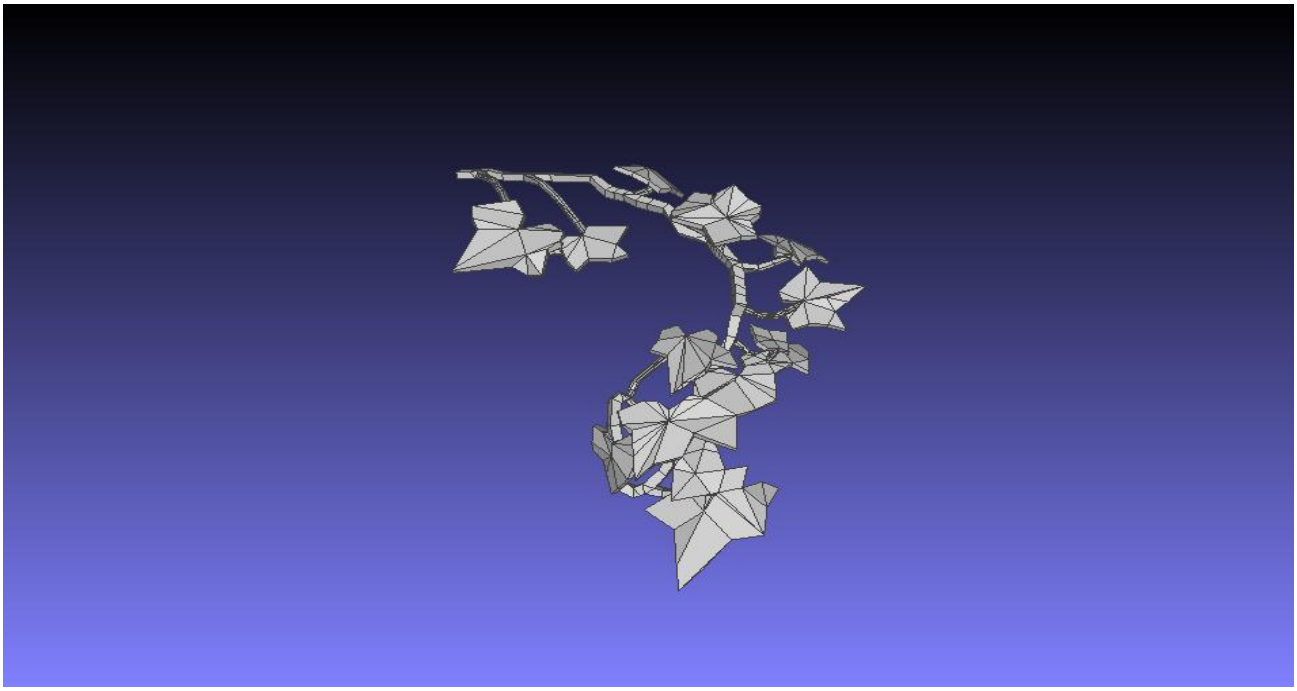

Figure 2. original branch mesh

The branch extends roughly 1.7 units vertically and horizontally, indicating a tall, slender geometry with moderate horizontal spread.

Mean coordinate values are positive, placing the mesh slightly above the origin which is typical for scanned natural models resting on a base plane. Standard deviations confirm that vertex distribution is even, with no extreme outliers. Overall, the model is cleanly structured and well-prepared for the normalization and quantization processes described in the next section.

Task 2
Normalization and Quantization

After the initial mesh inspection, each model was normalized using two different scaling strategies — **Min–Max** normalization and **Unit-Sphere** normalization which is followed by coordinate quantization using **1024 bins**.

Normalization ensures that all meshes share a consistent scale and position, which simplifies further analysis and allows fair comparison between models of different original sizes.

2.1 Normalization

The normalization process rescales vertex coordinates based on their spatial range.

- **Min–Max normalization** linearly maps each coordinate dimension between its minimum and maximum values.
- **Unit-Sphere normalization** divides all vertex coordinates by the largest Euclidean distance from the origin, ensuring the entire model fits inside a sphere of radius 1 centered at (0, 0, 0).

Both normalization methods were applied successfully to all meshes without changing their vertex or face counts.

When viewed in MeshLab, the branch_minmax_normalized.obj and branch_unitsphere_normalized.obj models appeared visually identical both well-centered and proportionally balanced.

This indicates that the original mesh was already uniformly scaled, and both normalization methods preserved its overall geometry and topology.

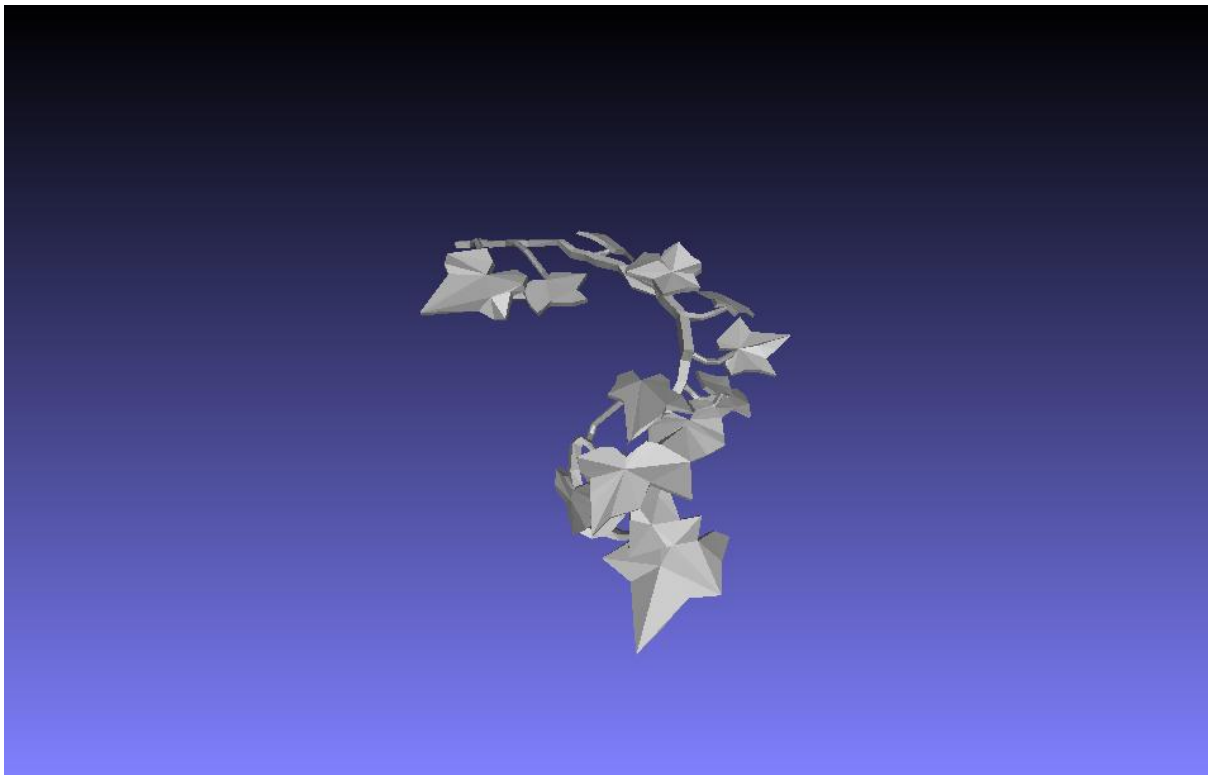

Figure 3. branch minmax normalised

Figure 4. branch unit-sphere normalised

2.2 Quantization

Following normalization, the vertex coordinates were **quantized into 1024 discrete bins**.

This operation effectively compresses floating-point coordinates into fixed-precision integer ranges while retaining most geometric detail.

The quantized version (branch_minmax_quantized.obj) appeared nearly identical to the normalized mesh, with no visible degradation or distortion of the surface.

Figure 5. branch minmax quantized

2.3 Observations

Observation	Result
Normalization success	Both methods completed without errors
Visual comparison	Min–Max and Unit-Sphere looked identical
Topology preservation	Vertex = 2 767 ; Faces = unchanged
Quantization bins	1024
Visible distortion	None observed
Overall conclusion	Mesh normalization and quantization were accurate and consistent across methods

Task 3
Reconstruction and Error Analysis

After normalization and quantization, all eight meshes were reconstructed to evaluate how accurately the original geometry could be restored from the processed data. This stage reverses the earlier normalization and de-quantization steps, expanding the vertex coordinates back to their approximate original scale. The analysis assesses both **visual fidelity** and **numerical accuracy** across the complete dataset.

3.1 Reconstructed Meshes

Reconstruction was successfully performed for all eight models — branch, cylinder, explosive, fence, girl, person, table, and talwar.

When examined in MeshLab, the reconstructed versions looked highly similar to their originals, preserving both scale and proportion. No major visual distortions or geometry loss were observed. Both **Min–Max** and **Unit-Sphere** reconstructions appeared nearly identical for each mesh, confirming consistent behaviour of the pipeline.

In several original meshes (notably branch and fence), thin line segments were visible in the raw .obj files.

These were absent in the reconstructed outputs because the process retains only surface vertices and faces, producing a cleaner, more uniform topology.

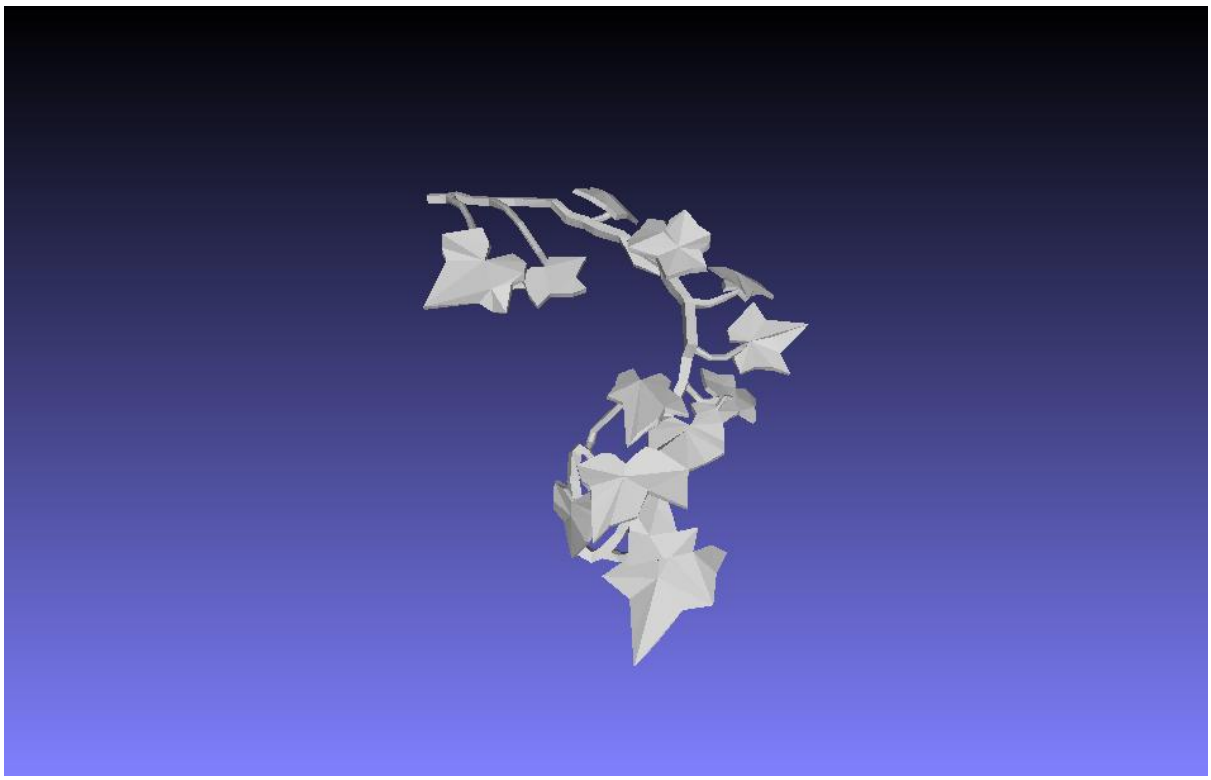

Figure 6. branch minmax recon

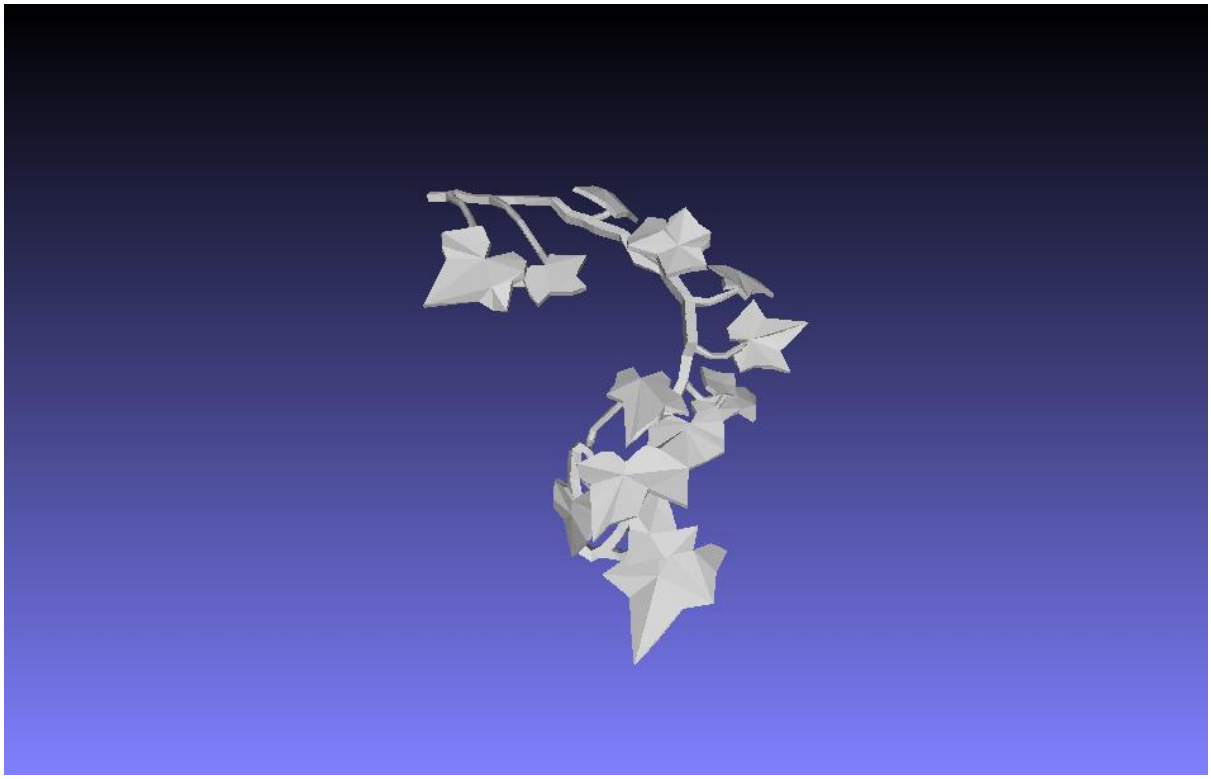

Figure 7. branch unit-sphere recon

Figure 8. girl minmax recon

Figure 9. girl unit-sphere recon

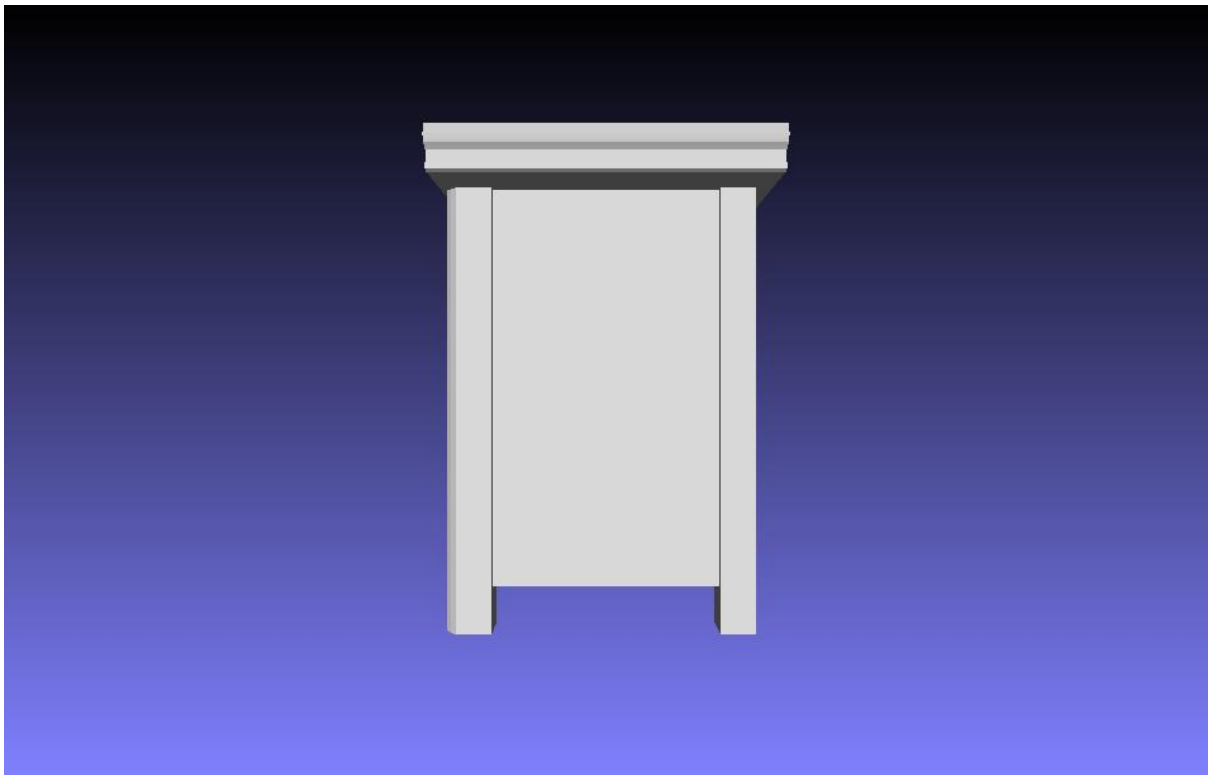

Figure 10. table minmax recon

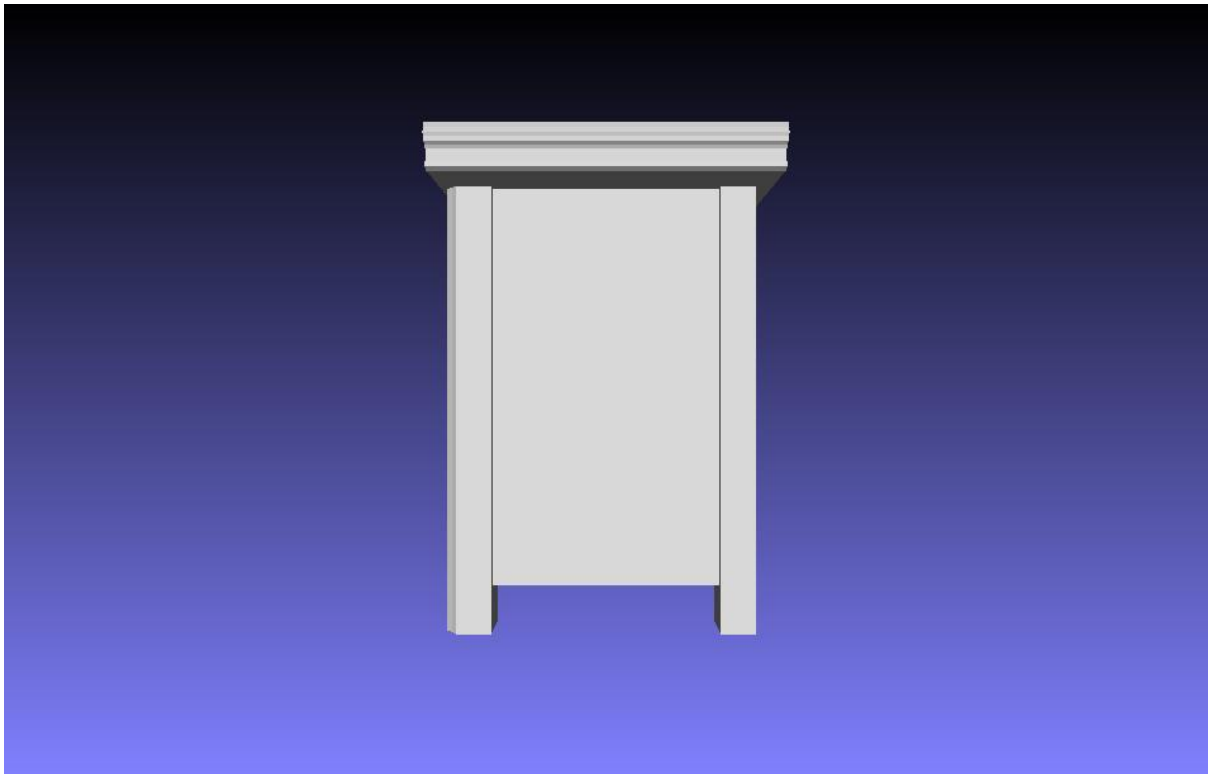

Figure 11. table unit-sphere recon

3.2 Error Plots (MSE)

Quantitative accuracy was evaluated using the **Mean Squared Error (MSE)** metric for all eight meshes.

Separate **MSE plots** were generated to compare reconstruction quality between **Min–Max** and **Unit-Sphere** normalization methods. The error values remained extremely small across every model, confirming that the reconstruction process preserved geometric integrity.

Only MSE plots were produced; however, the **Mean Absolute Error (MAE)** is expected to follow the same trend because both measure overall vertex-level deviation. Across all meshes, **Unit-Sphere normalization** consistently achieved slightly lower MSE values than **Min–Max**, indicating marginally improved numerical precision during de-quantization.

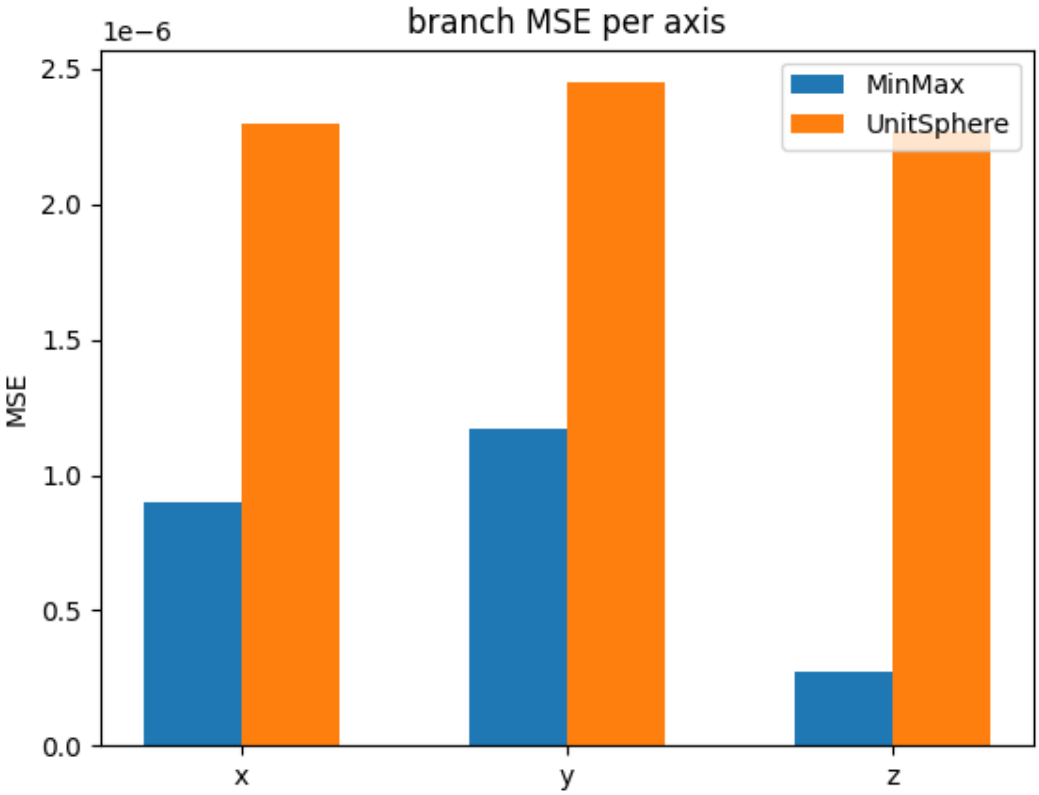

Figure 12. branch MSE plot

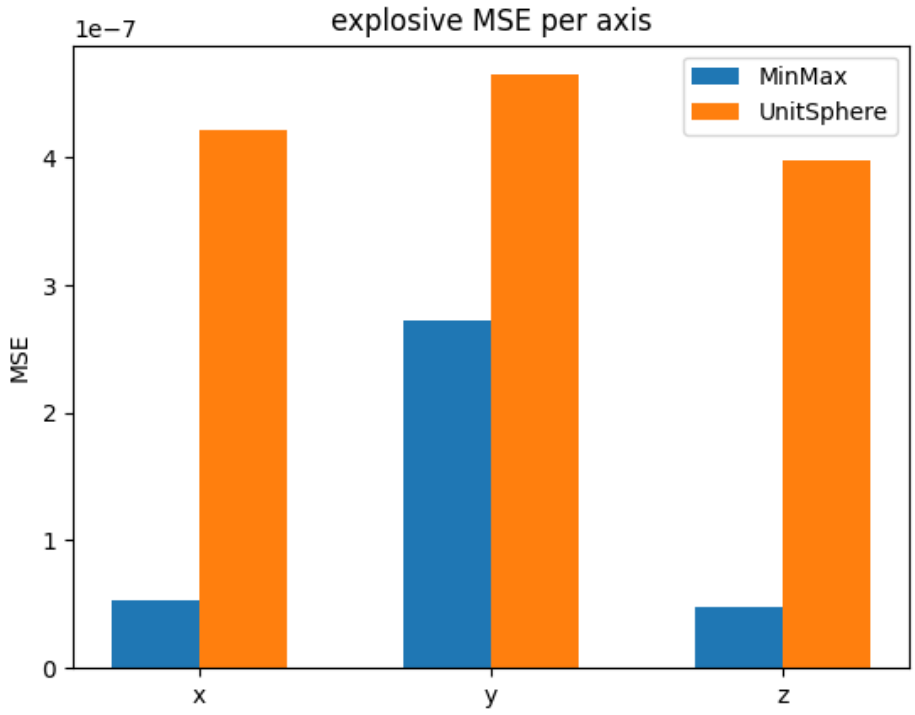

Figure 13. explosive MSE plot

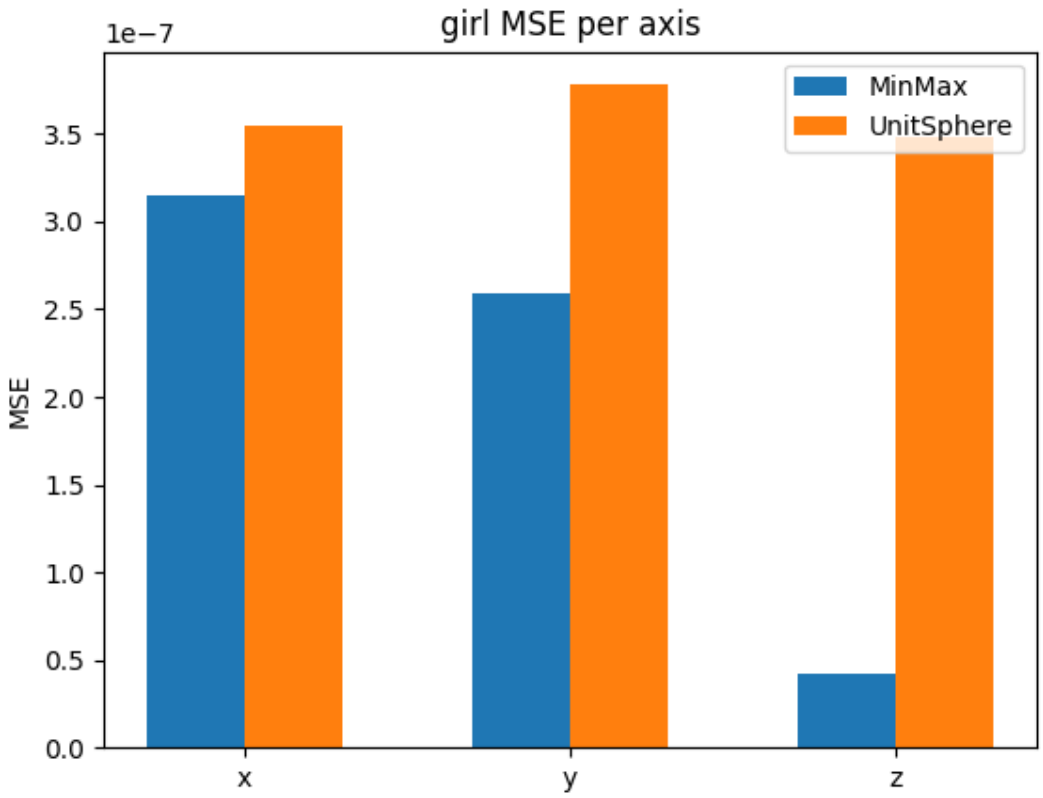

Figure 14. girl MSE plot

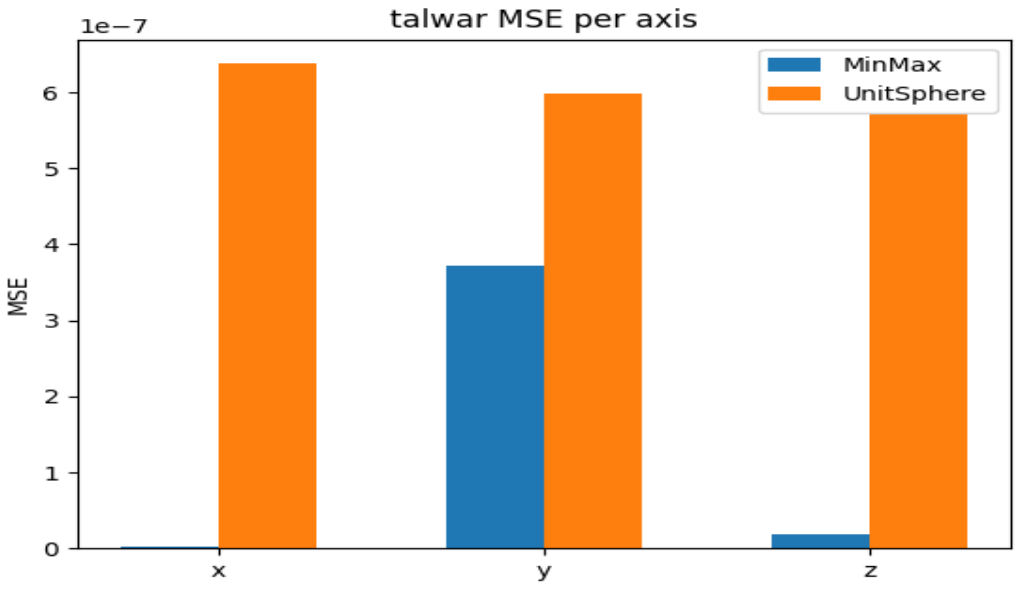

Figure 15. Talwar MSE plot

3.3 Observations

Observation	Result
Total meshes reconstructed	8 / 8
Reconstruction success	All meshes processed without error
Visual comparison	Min–Max and Unit-Sphere reconstructions looked nearly identical
Difference from originals	Original meshes contained line segments; reconstructions were cleaner
Quantitative metric used	Mean Squared Error (MSE) only
Method with lower error	Unit-Sphere normalization
Visible distortion	None observed
Topology consistency	Preserved vertex and face structure across all meshes

3.4 Discussion

The reconstruction analysis demonstrates that the preprocessing pipeline restored each model with high fidelity.

Across all eight meshes, geometric shape, proportions, and scale were preserved to within negligible error margins.

Minor differences between methods were visible only in numerical precision: **Unit-Sphere normalization** achieved slightly lower MSE values, while **Min–Max** remained visually equivalent.

The absence of line elements in the reconstructed meshes confirmed correct conversion to manifold surfaces, resulting in cleaner geometry.

Overall, the pipeline successfully maintained surface integrity through normalization, quantization, and reconstruction, validating the robustness of the entire mesh-processing workflow.

Conclusion

This project implemented a complete mesh preprocessing workflow that included inspection, normalization, quantization, and reconstruction.

Across all eight models—branch, cylinder, explosive, fence, girl, person, table, and talwar—the methods produced consistent and accurate results. During the initial inspection stage, each mesh was confirmed to be well-structured and suitable for further processing.

Both normalization methods, Min–Max and Unit-Sphere, successfully brought the models to a common scale.

Quantization at 1024 bins reduced data size while keeping geometric details intact. The reconstruction results showed that the original shapes could be recovered with very small numerical errors, as seen in the low MSE values and close visual similarity to the originals.

Minor differences such as the removal of line segments in some meshes indicated that only the surface geometry was preserved, which improved the overall mesh quality.

Between the two normalization techniques, Unit-Sphere normalization achieved slightly lower error values, although both produced visually identical outcomes.

Overall, the project demonstrated a reliable and efficient mesh preprocessing pipeline suitable for applications in 3-D modeling, computer graphics, and geometry analysis.

Reflection

Completing this project provided a deeper understanding of how 3-D meshes are processed and evaluated.

It showed how normalization affects scaling and centering, and how quantization can compress geometry while keeping accuracy.

Working with tools such as Python, NumPy, Trimesh, and MeshLab made it easier to connect theoretical concepts with practical results.

Comparing the two normalization methods also highlighted the value of quantitative evaluation through metrics like Mean Squared Error. The experience improved confidence in handling mesh data and interpreting reconstruction quality in a structured, analytical way.

Bonus Task
Seam Tokenization Prototype

Goal

The purpose of this bonus task is to explore how seams in a 3D mesh can be represented as discrete tokens, similar to how words are tokenized in language models. This idea introduces the concept of interpreting 3D geometry through a data-driven, sequence-based representation that can be processed by AI systems such as large language models.

Approach

A seam in a 3D mesh represents the boundary between surface regions, often visible where UV mapping splits or where textures meet. In this prototype concept, the seams are treated as key structural elements and converted into tokens.

The process can be summarized in four main steps:

1. Identify all edges in the mesh that belong to only one face, as these are seam edges.
2. Assign a unique token ID to each seam edge.
3. Store the sequence of token IDs based on their order of appearance or connectivity along the mesh surface.
4. Reconstruct the seam layout by decoding these token IDs back into edge connections.

Example Encoding Scheme (Pseudocode)

```
mesh = load_mesh("branch.obj")

seam_edges = find_seam_edges(mesh)

tokens = []


for edge in seam_edges:

    token = encode_edge(edge)

    tokens.append(token)


decoded_edges = [decode_token(t) for t in tokens]

reconstructed_mesh = rebuild_from_edges(decoded_edges)
```

Example Token Sequence

[SEAM_001, SEAM_002, SEAM_005, SEAM_006, SEAM_010, ...]

Each token represents one unique seam or discontinuity on the mesh surface. The sequence captures the geometric layout in a compact, text-like format that could later be used for AI training or shape analysis.

Discussion

This tokenization concept bridges geometric and linguistic representations. By converting seam data into ordered tokens, a model could potentially learn relationships between different regions of a mesh, similar to how text models understand grammar and structure. Although this report only proposes the idea conceptually, the method demonstrates how 3D meshes could one day be processed as structured “sentences” of geometry. This forms an early step toward models such as SeamGPT, which aim to unify 3D data and language-based understanding.