Thingiverse research - calculate curvature

- before we are try to calculate curvature , first we try to understand what is a curvature. So we found **principal curvatures** (k1,k2). it defining the maximal and minimal curvature at a given point of the surface. after we get product of the maximum and minimum principal curvatures we can get Gaussian Curvature.

$$K = k1 \cdot k2$$

This links naturally to the classification of models :

- kg=0 means surface is locally flat in at least one direction (parabolic point)
- K>0 : The surface is locally "bowl-like" (elliptical, like on a sphere or a hill) and K<0 : The surface is locally "saddle-like" (hyperbolic, like a saddle) so they are free form models

[1] R. Vaillant, "Curvature of a triangle mesh: definition and computation," Rodolphe Vaillant, Jun. 6, 2014. [Online]. Available: https://rodolphe-vaillant.fr/entry/33/curvature-of-a-triangle-mesh-definition-and-computation

- so we analyzed again another 50 data set from thingiverse dataset : 1716278 - 1778123

- calculating Gaussian Curvature
- we found a way to get curvature values using pyvista library

```
import pyvista as pv
import numpy as np

# 1. Load the mesh
mesh = pv.read(r"C:\Users\user\Documents\thingi data set\1722406.stl")

# 2. Compute curvature
# The `curvature` filter returns a NumPy array directly.
gaussian_curvatures = mesh.curvature(curv_type='gaussian')

# 3. Print and analyze the values
print("The first 10 curvature values are:")
print(gaussian_curvatures[:10])

print("\n--- Curvature Statistics ---")
print(f"Minimum curvature: {np.min(gaussian_curvatures):.4f}")
print(f"Maximum curvature: {np.max(gaussian_curvatures):.4f}")
print(f"Average curvature: {np.mean(gaussian_curvatures):.4f}")
print(f"Number of values: {len(gaussian_curvatures)}")
```

values we got :

Minimum curvature: -3650.0334
Maximum curvature: 6822.9975
Average curvature: 1.0439
Number of values: 10171

- then we needed to double check that values are accurate

  - for that we used Trimesh lib

```
import trimesh
import numpy as np

# 1. Load the mesh
mesh = trimesh.load(r"C:\Users\user\Documents\thingi data set\1722406.stl")

# 2. Calculate the discrete Gaussian curvature using vertex defects
gaussian_curvatures_trimesh = mesh.vertex_defects

# 3. Print and compare the statistics
print("--- Trimesh Curvature Statistics ---")
print(f"Minimum curvature: {np.min(gaussian_curvatures_trimesh):.4f}")
print(f"Maximum curvature: {np.max(gaussian_curvatures_trimesh):.4f}")
print(f"Average curvature: {np.mean(gaussian_curvatures_trimesh):.4f}")
print(f"Number of values: {len(gaussian_curvatures_trimesh)}")
```

values we got :

Minimum curvature: -1.6116
Maximum curvature: 5.0708
Average curvature: -0.0012
Number of values: 10171

  - and again used PyVista lib with VTK Filter got same results like pyvista code

so after considering these methods we undestood that pyvista and trimesh libs using different mathematical approach

  - **PyVista** measures how a surface bends, similar to a standard mathematical formula
- **Trimesh** measures how pointy or flat a surface is by looking at the angles of the triangles that make it up

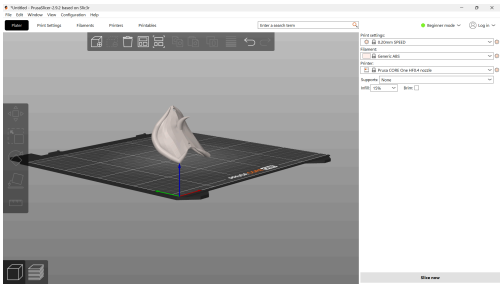
Sample data:

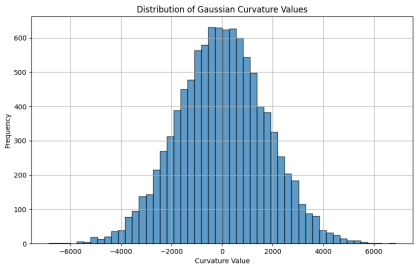
Model	PyVista Min	PyVista Max	PyVista Avg	Trimesh Min	Trimesh Max	Trimesh Avg
1722406.stl	-3650.0334	6822.9975	1.0439	1.8817	9287020.5383	5189.7758
1716278.stl	-1321.4646	438.6815	0.6570	0.0602	629961865.1267	11405.2499

1717685.stl	-24587.0989	174072.1667	4.9859	238905.3333	3894574.2365	754103.3559
1722417.stl	-12.3997	10.1269	0.0393	23.4998	4709532.7443	14264.8586
1722422.stl	-2292.4593	23017.2252	4.5531	0.4812	4235483.6964	2126.9556
1722424.stl	-46258.8188	6636.0004	-13.1371	0.2032	11877795.3175	4541.3557
1750620.stl	-3963.8649	6319.8287	0.5918	0.0353	9213399.2000	13670.4512
1764652.stl	-901.0032	623.0561	0.1056	0.1059	8937999.4827	3222.5003
1772543.stl	-0.0448	0.2138	0.0389	0.0617	4.4708	1.2170
1778123.stl	-9367.3261	13117.3249	1.2927	0.0149	9580395.7007	18048.8179

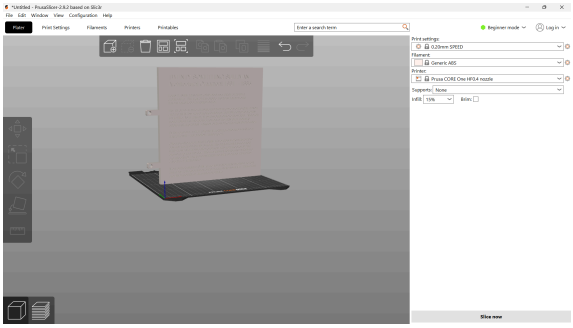
got a code to generate histogram , bcs using that we can identify type of the model.

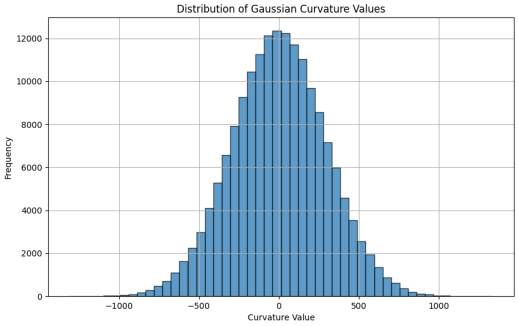
1. 1722406.stl



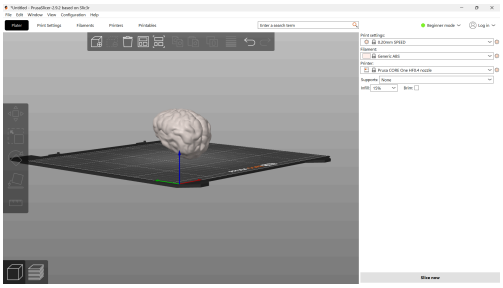


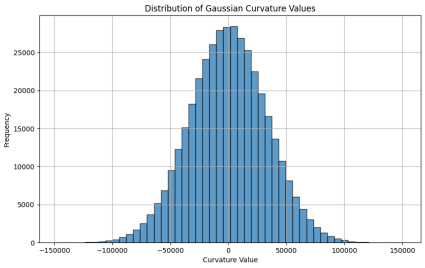
2. 1716278.stl



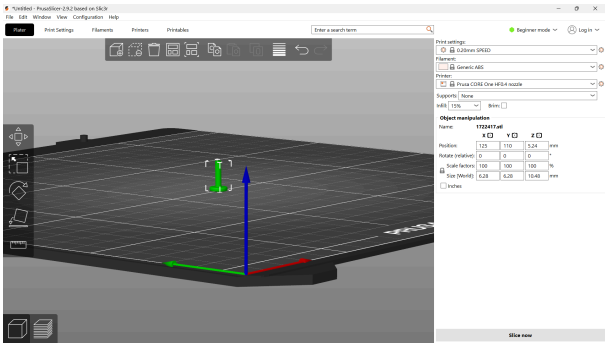


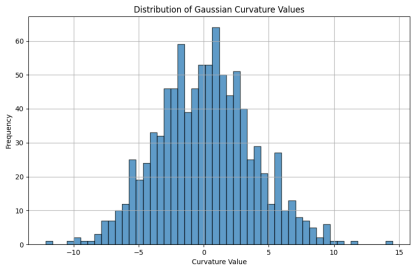
3. 1717685.stl





4. 1722417.stl





5. 37003.stl