# Pixel2Mesh2

Created by Peter Heile, Inspired by the Original P2M paper

## DATASET

SHAPENET

55 Classes x 50k Objects

1 Class

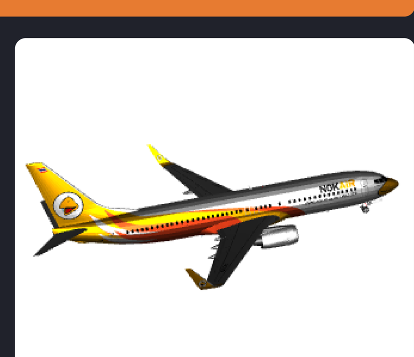
~4k Objects

~20K Images
(5 each Object)

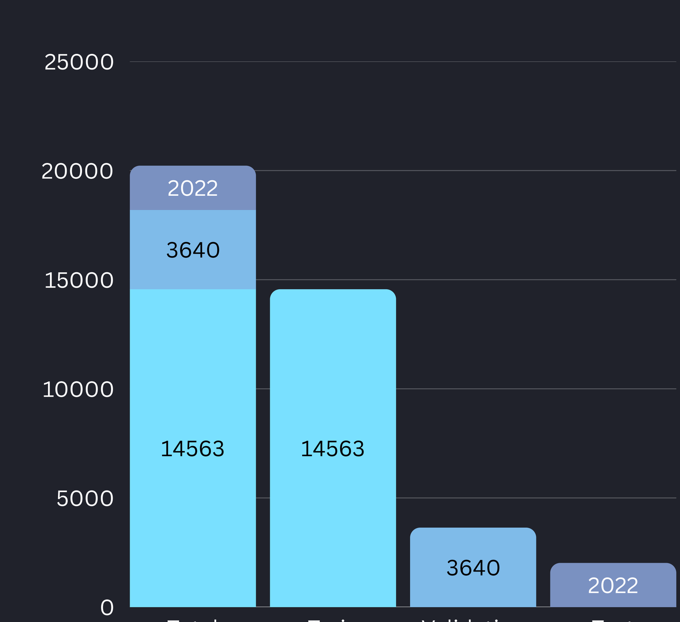
Train/Val: ~18000 Instances
Test: ~2000 Instances

### INPUTS:

There are two key inputs into the entirety of this model. The first being the (224 x 224 x 3) image passed into the Resnet Backbone, this is variablea nd depends on viewpoint. The second is the Initial Ellipsoid mesh passed into the GCN Mesh Deformation Block. This ellipsoid has the shape of (162 x 320 x 3); however will become larger with each Mesh Deformation Block. For each instance, the vertice and surface normal information was collected.

| ResNet Input | | GCN input |
|---|---|---|
|  |  |  |
| (224, 224, 3) | (224, 224, 3) | (162, 320, 3) |

This chart shows the train, testing, and validation split used for the data on the model. 5fold cross validation was used so the train/validation split changed upon each iteration.




## INTRODUCTION

### Goals

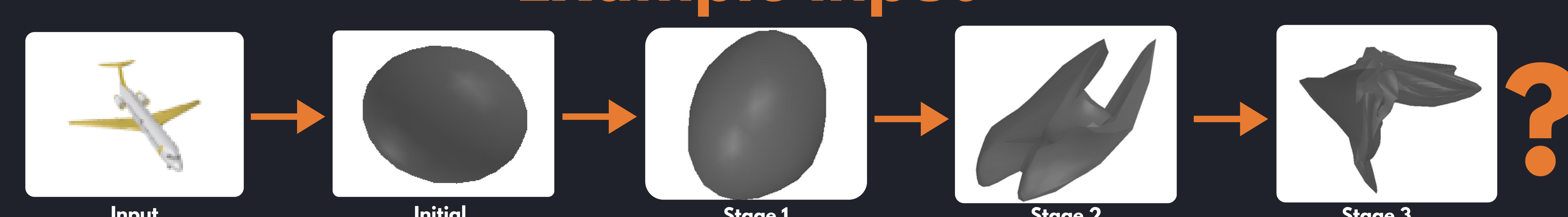
The goal of this research project is to take in an RGB image - in this case only a plane - and produce an accurate representative 3d mesh from that image. This mesh is represented by outputted vertices, faces, and generated surface normals.

### Applications

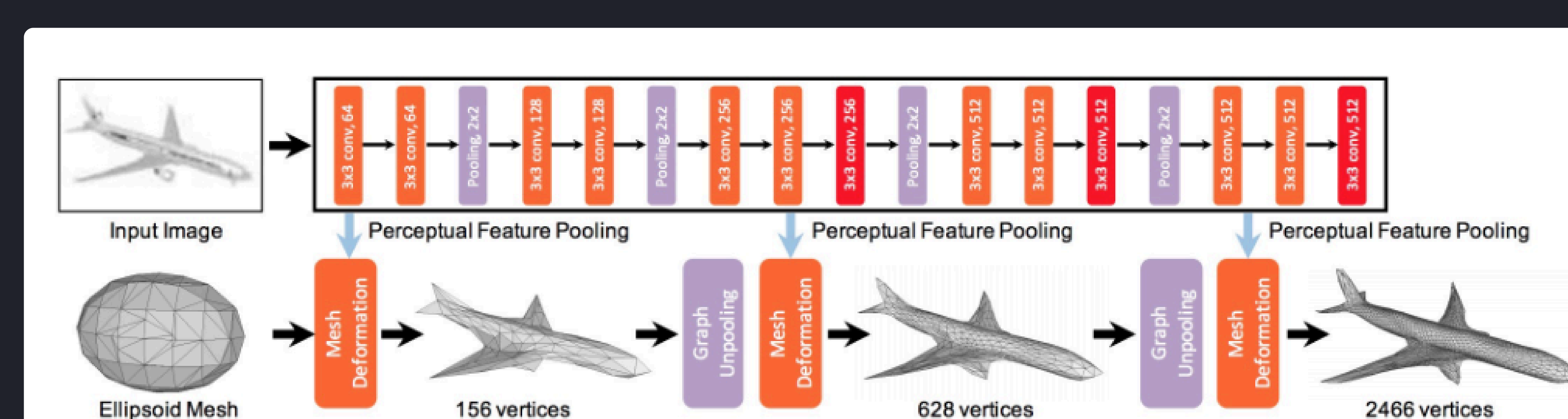
1. Game Development
2. Model Reconstruction
3. Medical Imaging
4. Ecommerce Platforms
5. Robotics & Simulation
6. Dataset Conversions
7. Archeology

**Inspiration:** This paper is inspired by the original P2M paper introduced by Nanyang Wang Et al. I attempt to develop a python-friendly model compatible with Cuda that employs the key architectural components proposed by the paper.

### Example Input




## ARCHITECTURE




*Core Components:

1. **ResNet-18 CNN** – Extracts multi-scale image features from input.
2. **Graph Convolution Layers** – Update vertex features based on mesh connectivity.
3. **GResNet Blocks** – Residual graph-based blocks for stable deep mesh learning.
4. **Feature Pooling** – Projects 3D vertices to 2D image to sample image features.
5. **Mesh Unpooling** – Subdivides the mesh to increase resolution at each stage.
6. **Loss Functions** – Chamfer, Laplacian, Edge, and Normal losses guide learning.

| GCN DEFORM STAGE | MESH SIZE | RESNET LAYER | FEATURE DIMENSION | PROJECT DIMENSION |
|---|---|---|---|---|
| Stage 1 | 162 verts | Layer 2 | 256 | 128 |
| Stage 2 | 642 verts | Layer 3 | 512 | 256 |
| Stage 3 | 2,562 verts | Layer 3 | 512 | 256 |

## RESULTS

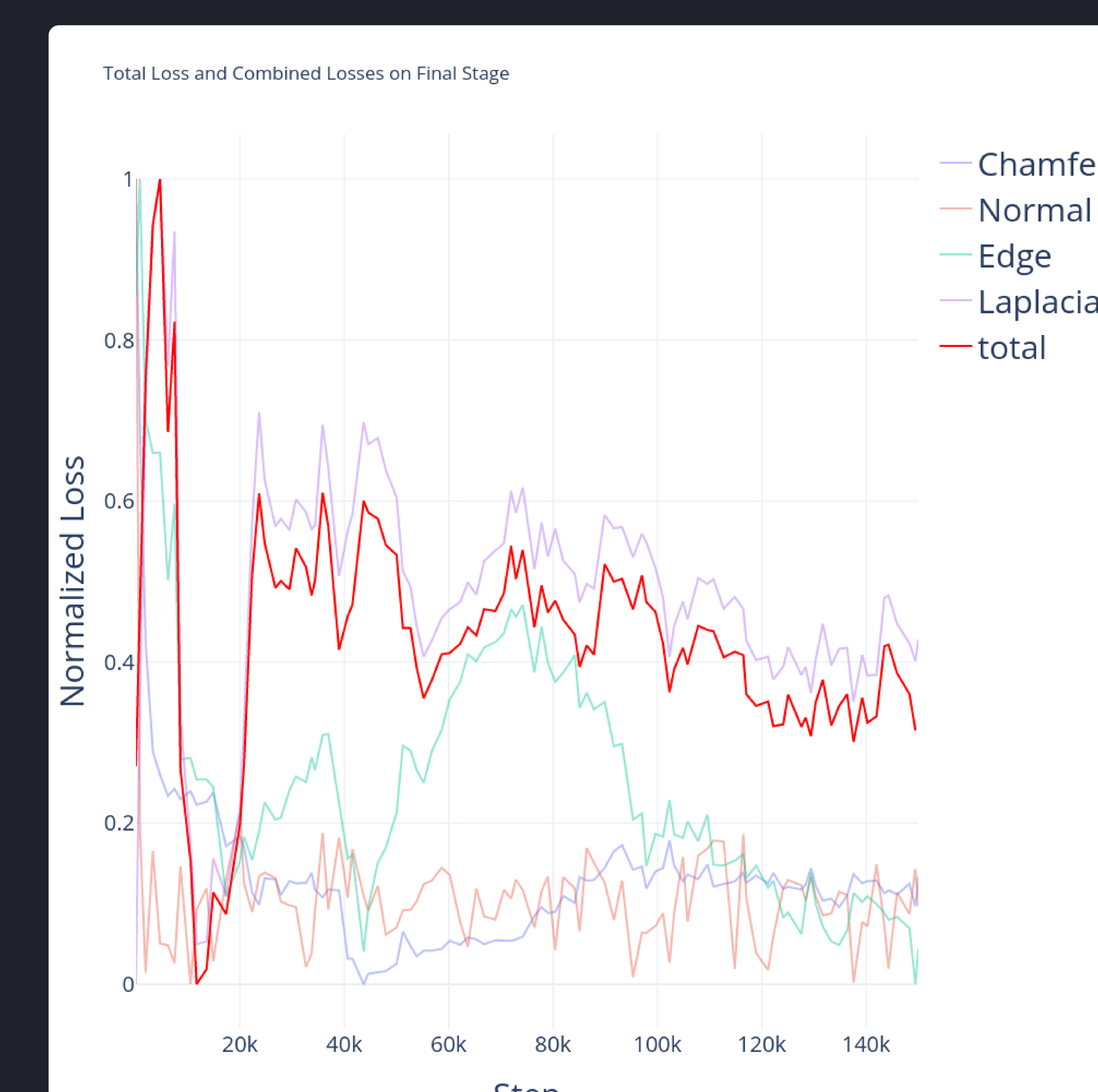
**Chamfer Loss:**
Finds distance between two point clouds.
Weight: 1.0

**Laplacian Loss:**
Promotes similar vertices locally.
Weight: 0.1

**Edge Loss:**
Minimizes each edge length.
Weight: 0.1

**Normal Loss:**
Ensures similar angular differences of vertices.
Weight: 0.05

**Combined Loss:**
Summed losses weighted and summed by stages.




**Accuracy:**
Final accuracy value of the model on testing set.

# 0.41