# Depth-Aware Pixel2Mesh

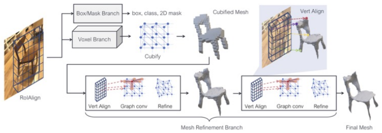
Kabir Jolly, Julian Quevedo, Rohin Manvi
Stanford University

## BACKGROUND

Current advances in the computer vision space have become increasingly accurate in object detection when given 2D inputs.

- Models like **Mask R-CNN**
- **Instance and semantic segmentation**

However, the **world around us lies in 3D**, and there is still much work to be done moving forward in developing a computational understanding of **3D shapes and objects**.



Overview diagram of Mesh R-CNN. Figure from Gkioxari, Malik, and Johnson.

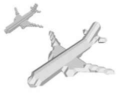
Mesh R-CNN is one major advancement in this space

- Constructs topologically accurate 3D meshes given a 2D RGB image using voxel representations
- Translates these voxel representations into a mesh using a GNN-based approach.

## PROPOSED SOLUTION

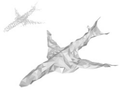
Current systems lack a major component of object recognition that we as humans use to perceive the world around us - *depth*.

Volume as Output



*Choy et al.*

Point Cloud as Output



*Fan et al.*

Pixel2Mesh

*Wang et al.*

We aim to expand upon existing models by enhancing its performance through depth aware inputs and log potential changes in performance as a result.

The depth images will be generated using the MiDaS depth estimation model.

MiDaS on subset of single-view inputs. Figure from Ranftl et al.

## METHODOLOGY AND RESULTS

### Phase I: Differentiable Rendering

In order to provide further supervision on our generated meshes, we considered augmenting the loss with differentiable rendering.

- Utilize a differentiable rasterizer to render a depth map of it the generated mesh
- Error between the rendered depth map and the "ground-truth" depth channel can be measured

This would allow our model to take further advantage of the RGB-D images by creating a mesh that has the same depth characteristics as the input depth map.

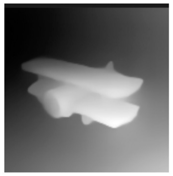
In order to meaningfully compare the rendered depth maps with the ones from the input images we need the following

- The loss must be scale-invariant.
- Rendered depth maps must be from the same camera positions as the input images.

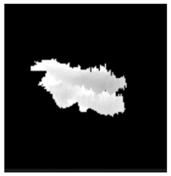
We discovered more difficulties and were unable to fully implement the scale-invariant depth loss.

- MiDaS hallucinates a ground plane beneath the ShapeNet renderings.
- The differentiable renderer simply marks all background points as −1
- MiDaS outputs an inverse depth map

We hope to investigate solving both these problems simultaneously in future work.



Depth Estimation



Mesh Reconstruction

### Phase II: RGB-D Backbone and Mesh Refinement Head

To allow Mesh R-CNN to take RGB-D images as input, we changed the first ResNet layer to learn four-channel filters instead of three-channel filters.

We take advantage of pretraining by copying over the weights of for the first three channels and only train the fourth from scratch.

| Index | Inputs | Operation | Output shape |
|---|---|---|---|
| (1) | Input | Image | $137 \times 137 \times 3$ |
| (2) | (1) | ResNet-50 `conv2_3` | $35 \times 35 \times 256$ |
| (3) | (2) | ResNet-50 `conv3_4` | $18 \times 18 \times 512$ |
| (4) | (3) | ResNet-50 `conv4_6` | $9 \times 9 \times 1024$ |
| (5) | (4) | ResNet-50 `conv5_3` | $5 \times 5 \times 2048$ |
| (6) | (5) | Bilinear interpolation | $24 \times 24 \times 2048$ |
| (7) | (6) | Voxel Branch | $48 \times 48 \times 48$ |
| (8) | (7) | `cubify` | $\|V\| \times 3, \|F\| \times 3$ |
| (9) | (2), (3), (4), (5), (8) | Refinement Stage 1 | $\|V\| \times 3, \|F\| \times 3$ |
| (10) | (2), (3), (4), (5), (9) | Refinement Stage 2 | $\|V\| \times 3, \|F\| \times 3$ |
| (11) | (2), (3), (4), (5), (10) | Refinement Stage 3 | $\|V\| \times 3, \|F\| \times 3$ |

The voxel loss is the binary cross-entropy between the predicted voxel occupation probabilities and the true voxel occupancies.

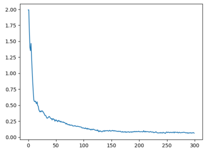
Chamfer distance and the normal distance are used as losses for the mesh. Pointclouds P and Q are sampled from the ground truth and the intermediate mesh predictions from the model.

$$\mathcal{L}_{\text{cham}}(P,Q) = |P|^{-1} \sum_{(p,q)\in\Lambda_{P,Q}} \|p-q\|^2 + |Q|^{-1} \sum_{(q,p)\in\Lambda_{Q,P}} \|q-p\|^2$$

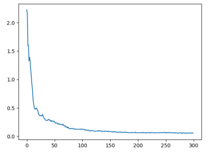
Chamfer Distance

$$\mathcal{L}_{\text{norm}}(P,Q) = -|P|^{-1} \sum_{(p,q)\in\Lambda_{P,Q}} |u_p \cdot u_q| - |Q|^{-1} \sum_{(q,p)\in\Lambda_{Q,P}} |u_q \cdot u_p|.$$

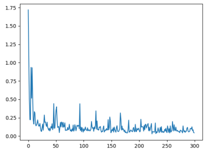
Normal Distance between Point Clouds



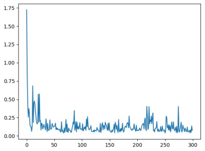
Pixel2Mesh Loss, RGB



Pixel2Mesh Loss, RGB-D



Mesh R-CNN Loss, RGB



Mesh R-CNN Loss, RGB-D

## DATASET AND FEATURES

We trained our model on two datasets: ShapeNet Core (along with renderings from R2N2) and Pix3D.

ShapeNet Core consists of over 50,000 3D meshes, which R2N2 provides rendered images of.

ShapeNet renderings

We use MiDaS to predict each image's depth map which we stack to produce four-channel RGB-D images.

## RESULTS

| Model | Chamfer Dist. (with RGB) | (with RGB-D) |
|---|---|---|
| Pixel2Mesh | 4.915 | **3.707** |
| Mesh R-CNN | **4.729** | 4.791 |

Adding depth resulted in a clear improvement for Pixel2Mesh, but seemed to make little difference for Mesh R-CNN.

## FUTURE WORK

A possible extension is to construct colored meshes

- Accurately represent textures and materials that appear in images
- Since we represent the meshes as graphs, our idea is to incorporate color information as an additional node feature
- Thus, as supervision for our color predictions, we need the meshes in the datasets to have vertex colorings
- To enable Mesh R-CNN to predict the color of each vertex, we plan to increase the dimension of the node features predicted by the mesh refinement stage
- Instead of predicting 3-dimensional features, we will predict 6-dimensional features, where the first three correspond to the vertex coordinate and the second three correspond to RGB values

We also plan to continue the unfinished work on using differentiable rendering and the depth images during training. We hope to overcome the aforementioned roadblocks and hypothesize that due to the additional information fed in during training time, it is likely to outperform the results exhibited by the current depth-aware Pixel2Mesh.