CS230

2D-Supervised Multi-View Mesh Reconstruction With Attention

Ivan Lizarraga
iva123@stanford.edu
https://github.com/ilizarr/img2mesh

Abstract

The task of predicting an object's 3D mesh structure from 2D images is a challenging exercise for machine learning algorithms. However, this same task seems almost trivial for us humans given our built-in experience and knowledge. Perhaps one reason why this task is a natural one for us is that we have an intuitive understanding of how 3D objects are rendered to 2D images and can readily apply this knowledge in the other direction. The following work explores this 3D to 2D image understanding as a potential key tool for enabling machine learning algorithms to reconstruct object structure. The approach taken here is that of using a *differentiable* renderer to aid in the generation of a loss signal that tells the model how its predicted mesh would appear in rendered images and how that differs from the original input. We showcase a fully 2D-supervised model that can look at multiple views of a given object through an attention layer and produce a 3D mesh. The model is shown to produce believable results for both 2 and 3 views at the same time.

1 Introduction

3D object understanding provides a key advantage when trying to process 2D images. Understanding how a 2D image is rendered from a 3D object can potentially be valuable for many tasks like mesh reconstruction, classification, detection, and segmentation. For this project we focus on the first. Mesh reconstruction is not only intrinsically valuable for asset pipelines and artists but it is also an active area of research. Contributing to this effort is important in order to advance our knowledge on how image understanding helps machine learning algorithms in general.

This work explores the task of generating 3D meshes from multiple views of the same object **without** 3D supervision. This approach means we don't have to rely on labeled 3D mesh data since that can be very hard to come by in the large quantities required by very deep neural networks. This is achieved through the use of a renderer that is able to back-propagate gradients to the mesh generation network.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

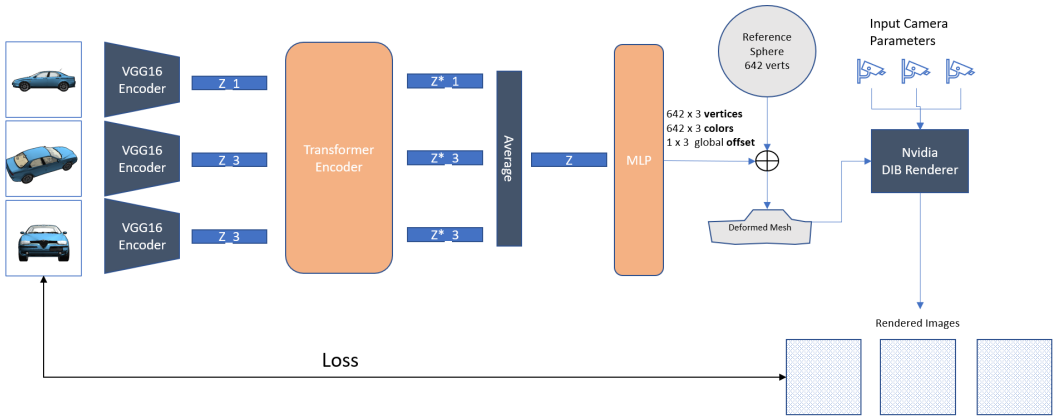

Figure 1: Baseline model architecture and pipeline

The baseline model for this task is shown in figure 1. The input to the model is a set of images for the target object rendered from different view points. We can theoretically support as many simultaneous input viewpoints as can fit in memory. The model also requires as input the original camera positions and orientations, per view, as well as a binary mask to separate the foreground object from the background. The mask can typically be easily derived from the alpha channels of computer rendered images. Finally, the model uses a starting 3D mesh that it will deform to produce the final reconstruction.

The input images are passed through an encoder network in order to produce a low-dimensional latent representation of each image. These latent representations are then fed through an attention network in order to encourage the model to "attend" to the common object features that show up inside each image from different viewpoints. The output from the transformer encoder is then averaged and flattened into a vector that is finally given to the multi-layer perceptron (MLP) network generating the output.

The model's output consists of the vertex offsets to apply to the starting mesh, the color to use for each vertex, and a global mesh position that enables the model to zoom in and out. The rest of the model from this point on essentially functions as one big loss function network. It is comprised of the DIB-R renderer [1] which takes the model's mesh and the original camera extrinsics to generate rendered images from the same viewpoints. We then compare the original images with the rendered versions to produce a loss signal that is propagated back to the transformer and MLP components.

2 Dataset and Features

Training primarily used example data made available by the DIB-R[1] examples to come up with an initial model architecture. This dataset consisted of 100 renderings of a red clock from different viewpoints with recorded camera parameters and segmentation masks.

The ShapeNet [2] repository of 3D CAD models was also used to generate custom renderings for additional training.

3 Methods

This report includes results for two models. The baseline model uses a VGG16 encoder network as a front-end. The second model uses a small convolutional neural network that is warmed up with pretrained weights from VGG16 which results in faster predictions at no loss of fidelity in the renderings.

As used in [3] and [1], we employ a multi-part loss function. We use an image loss and a mask loss. We also add terms to regularize the mesh structure: a face angle loss and an edge length loss.

4 Experiments/Results/Discussion

Refer to figures 2 and 3 for model results.

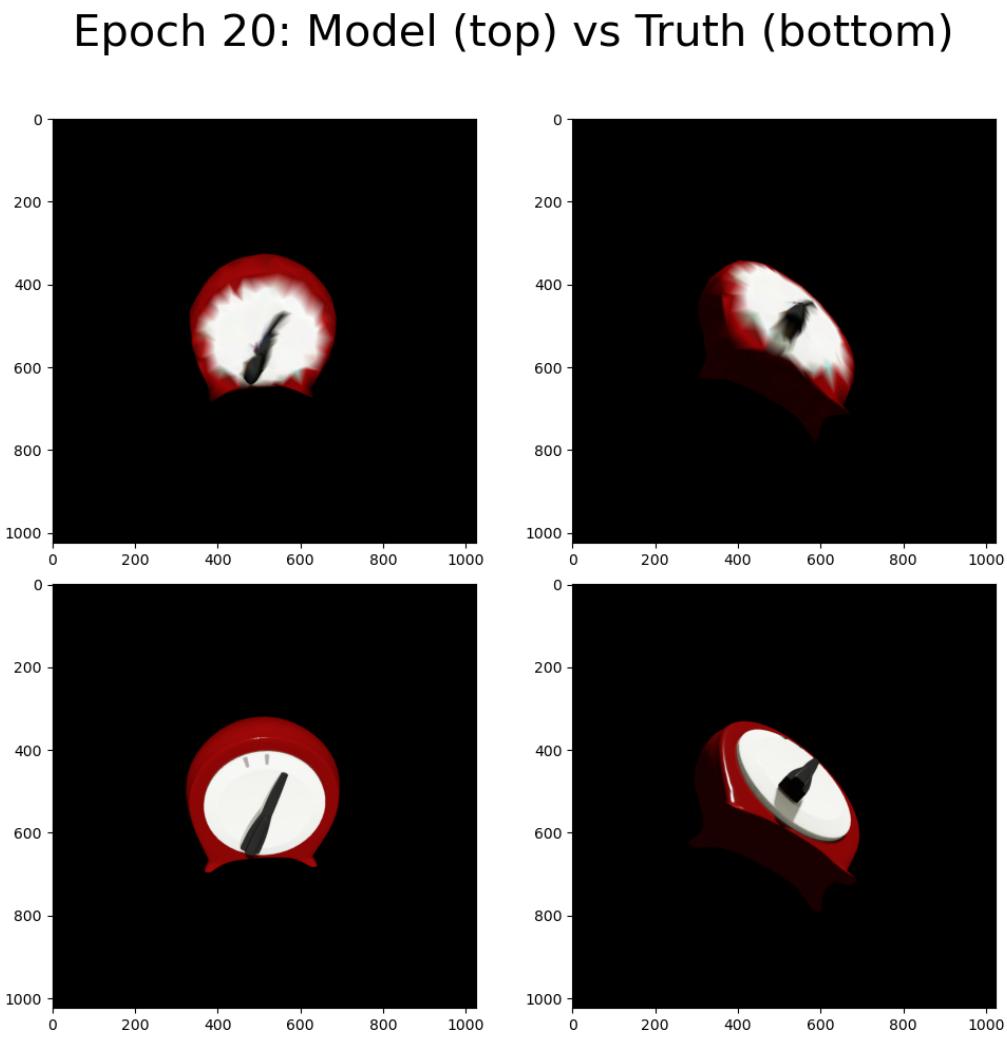

Figure 2: Baseline model results

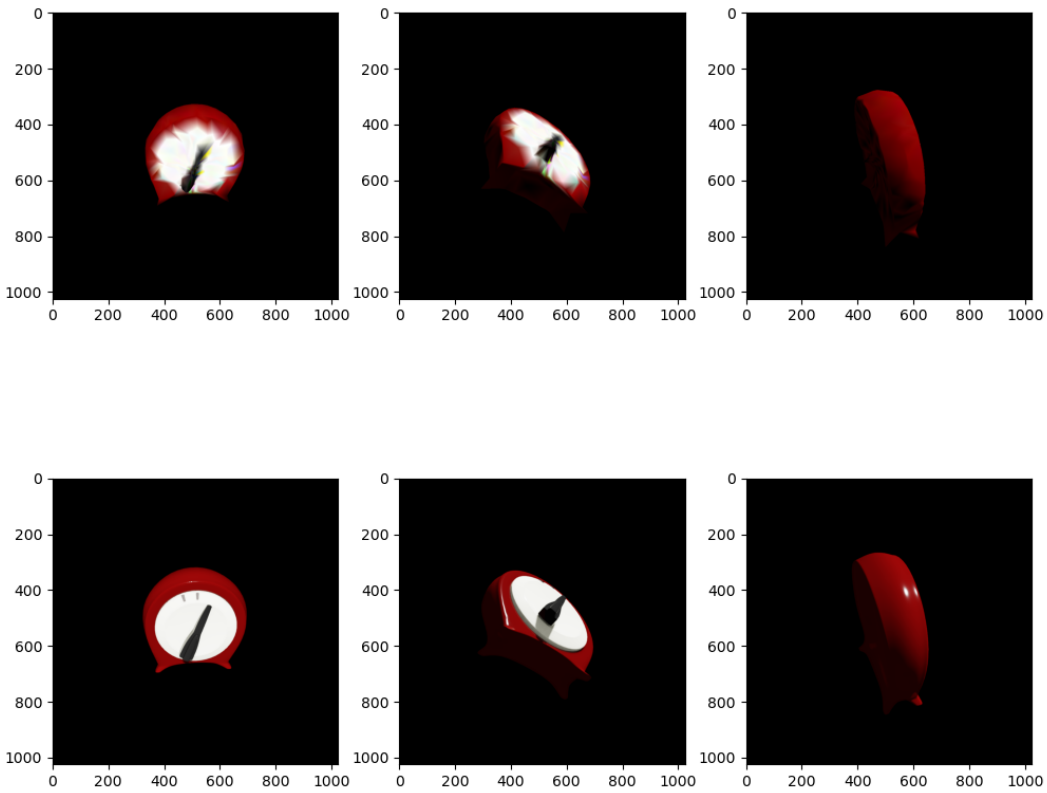

Figure 3: CNN encoder model results with more input viewpoints

5 Conclusion/Future Work

We show that an attention model using VGG encoders and a differentiable renderer can accurately produce 3D meshes from 2D images and camera extrinsics.

References

[1] Wenzheng Chen, Jun Gao, Huan Ling, Edward Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. Learning to predict 3d objects with an interpolation-based differentiable renderer. In *Advances In Neural Information Processing Systems*, 2019.

[2] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository, 2015.

[3] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *ECCV*, 2018.