# Discovering 3-D Structure of LEO Obects

**Jacob Lucas, Trent Kyono, Julia Yang**
*The Boeing Company*
**Justin Fletcher**
*Odyssey Systems Consulting*

**Abstract**

A 3-D object model provides invaluable information for space domain awareness. However, in practice, there are countless reasons why a 3-D object model may not exist or is unattainable. In this work, we aim to investigate the discoverability of 3-D object structure from passive observations of objects. Specifically, we aim to investigate the feasibility of using Neural Networks via Neural Radiance Fields for uncovering 3-D object models. Experimentally, we demonstrate feasibility on a simulated set of satellite images, where we can compare object structure to ground-truth.

## 1. Introduction

Space Situational Awareness (SSA) depends on the accumulation of information. There have been many forays into the application of deep learning methods to glean additional information from existing SSA data products [1] [2] [3] [4] [5]. Many of these methods could be improved with a known 3-D model of the object of interest. In this paper we explore the viability of recent machine learning approaches that seek to recover 3-D structure from sets of conventional images via gradient descent methods. More specifically we leverage the recently released Pytorch3D library [6] and attempt 3-D structure recovery by fitting with a deformable mesh, and using a Neural Radiance Field (NeRF) [7].

We begin with a set of rendered images intended to be representative of space based images of a satellite. In this exploratory work these renders are idealized, without camera degraders or other artifacts. We then leverage the Pytorch3D package to learn representative camera parameters for each render. Armed with a set of renders and the now characterized relative motion, we proceed to fitting the render sets with both a deformable mesh, and a Neural Radiance Field. We can then compute quantitative metrics measuring the similarity between images rendered with the learned model and images rendered with the truth model. This procedure is outlined in Figure 1.

We provide a brief discussion of related works in Section 2. Section 3 contains a description of our dataset, training architecture, methods, and our experimental results. We conclude with brief remarks in Section 4.

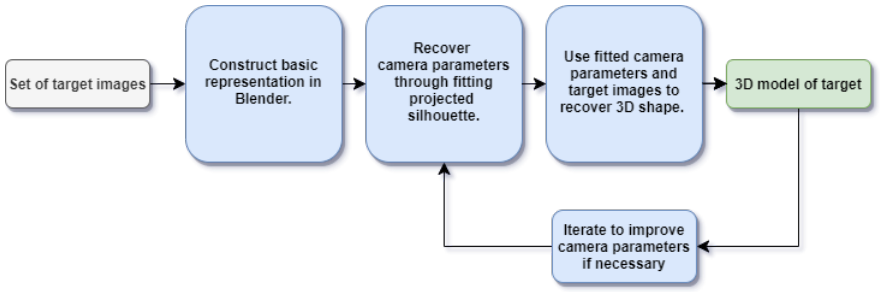


Figure 1: Flow diagram describing the process steps. The examples herein utilize renders as a controlled surrogate for real data.

## 2. Related Works

### 2.1. Mesh Based Methods

Representing 3D shape with meshes is well documented and explored. With [8] requiring only a single image and outputting a triangle mesh, and [9] using a slightly different approach for the same goal. In order to construct 3D shapes without annotations, [10] promotes differentiable rasterization, while [11] proposes a differentiable ray consistency term. Both methods require multiple views. Our approach is similar to [10], where we use a differentiable renderer to enable a gradient based 2D to 3D fitting.

### 2.2. NeRF Based Methods

Neural radiance fields were introduced by [7], and much has been done with them in the short time since introduction. Efforts to reduce the number of images required [12] and to apply them to dynamic scenes [13] are two examples. More recently, while in the process of exploring this work, [14] has applied NeRF and GRAF [15] to images of spacecraft with impressive results. The key differences between this supporting work and that presented here are our inclusion of a camera parameter recovery step, and comparison to mesh recovery.

## 3. Experiments

This section will cover how data was generated and processed, the different methods implemented to recover 3D information based on the generated data, and finally how the methods were graded as well as their measured performance.

### 3.1. Datasets

For this study we chose to generate simulated data in order to have a baseline for performance comparison. Data generation began with a 3D mesh of the Hubble Space telescope obtained as an open resource from

NASA [16]. Using this model and the Pytorch3D package, we rendered sets of 60 256x256 pixel images with a variety of mesh poses intended to be representative of satellite motion relative to the observer. Image sets were rendered as both silhouettes and as detailed images. Camera parameters used for rendering, such as rotation and translation matrices, were held aside for comparison. These rendered datasets will be referred to as truth images for clarity.

### 3.2. Training Methods and Performance

The premise for this study required that the data input be limited to images of the target object. The methods used here, as well as many similar Structure-from-Motion (SfM) techniques, require camera parameter/image pairs in order to recover the mesh or structure. Many of these techniques have built in processes for recovering camera parameters from imagery [17]; however, when testing, we found that we had a difficult time fitting camera parameters to the monochromatic and sparse satellite renders. To alleviate this shortcoming, we implemented a bootstrapping method where we begin with a simple 3D structure constructed by hand in Blender [18] and based on the structure observed in the images. We then learned the camera position and orientation for each truth image by applying a gradient descent optimization routine that minimized the image difference between a rendered silhouette of this simple model and the truth silhouette. The difference between the silhouette images was characterized with a Dice loss [19]. Following the fitting procedure the camera parameters were smoothed to enforce the assumption that the satellite is undergoing little to no acceleration. The camera rotation and translation matrices are then calculated with the fitted camera positions. Using this approach to learn camera parameters we were able to recover camera rotation to less than 10 degrees mean absolute error (MAE) over all 3 axes, and translation parameters to less than 5% MAE. The iterative step outlined in Figure 1 was not necessary if the initial model was carefully chosen to represent the primary features well. In practice, we noticed little qualitative difference in 3D fit between the truth camera parameters and camera parameters recovered with this method.

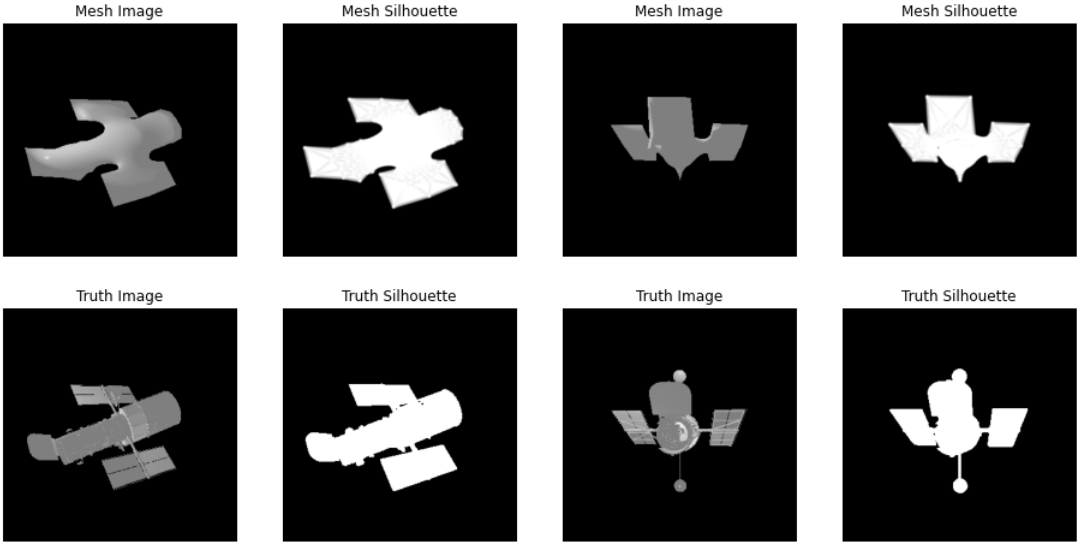


Figure 2: Example renders from the fitted mesh recovered model compared to renders from the truth 3D mesh object. Note that there is no fine detail present.

Once camera parameters have been learned for all truth images we can proceed to implementing routines that utilize gradient descent to learn the 3D structure of the source object from the image/camera parameter pairs.

The first routine attempted began with a simple shape, in this case a sphere, which was then deformed and rendered, with the loss comprised of the difference between its rendered silhouette and the truth silhouette. This method couples the silhouette loss (L2) with other losses that enforce smoothness and edge length regularization over the deformed mesh. For this exercise the optimizer was allowed to converge for 15,000 iterations, utilizing Stochastic Gradient Descent (SGD) [20] with a learning rate of 1.2. Despite the additional losses, the implementation used resulted in at best a coarse representation of the truth mesh, with round features appearing faceted in the final product and only the broad shape matching. The assumption is that our loss was ill posed which resulted in chasing local minima, without approaching a global minimum. Figure 2 displays examples of the fitted mesh.

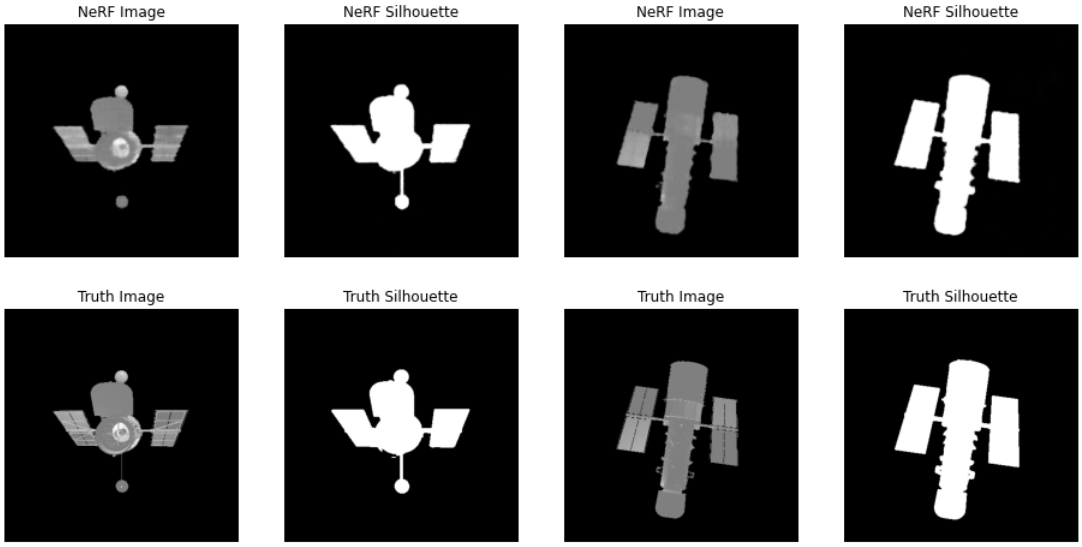


Figure 3: Example renders from the NeRF recovered model compared to renders from the truth 3D mesh object.

The second 3D structure recovery approach attempted here was to fit the image/camera parameter pairs with a Neural Radiance Field. This approach utilized both the rendered image and the rendered silhouette. A NeRF represents the scene as a mapping from a 5D vector valued function defined by the camera parameters (3D location and 2D viewing angle) to an emitted color (r,g,b) and volume density [7]. During training, a random set of rays is projected into the world space, with regular sampling over the length of the ray, creating a 3D sampling of the scene. The 3D scene sampling is then mapped into a harmonic representation, which was shown by [7] to provide an improved optimization landscape. This representation is learned by an MLP (MultiLayer Perceptron) where, in our case, a Huber loss is used for both the rendered silhouette loss and the image loss, and an Adam optimizer [21] was used to manage the gradient descent. We used a simple implementation with a 5 layer MLP used to learn the latent representation. This was then interpreted by a single layer MLP with a single output neuron to determine volume density, followed by a 2 layer MLP with 3 output neurons to interpret color. The routine optimized over 60,000 iterations, with an initial learning rate of 5e-4 that was gradually reduced to 5e-5 as fitting advanced. Using regular sampling over the length of the projected ray is a simplification over the coarse/fine sampling implemented by [7], and likely limits the fine detail accuracy of our approach. Despite the significant memory resources of the GPU used, the memory intensive nature of this technique forced a trade between image size, mini-batch size, ray sampling density, and model complexity. This was exacerbated by pre-loading all rendered images into GPU memory in order to reduce the time required for fitting. We settle on a mini-batch of 4 images, 200 depth samples per ray, and 256 hidden neurons in each of the 5 layers of the MLP.

It was clear that the results of the NeRF implementation were an improvement over the deformable mesh implementation. While taking significantly longer to fit, the rendered silhouettes and images are much closer to the truth data, as show in Figure 3. The images in Figure 3 are examples of poses used during training. We noticed that for our simplified implementation, the quality of the NeRF representation suffered for novel views.

The performance of the approaches were measured by comparing a rendering made with the constructed model to a rendering made with the original 3D mesh. We quantified the difference by computing the IOU (Intersection over Union) or Jaccard index between the truth/recovered silhouette pairs, and the Structural Similarity (SSIM) [22] between the truth/recovered image pairs. This was done over a variety of poses to better characterize overall performance and minimize the effect of spurious views. These resulting values for the mesh approach are in Figure 4, and for the NeRF approach in Figure 5. While the metrics show similar performance it is obvious that the visual quality is quite different between the two approaches. This suggests an alternate similarity metric better correlated to visual quality may be preferable.

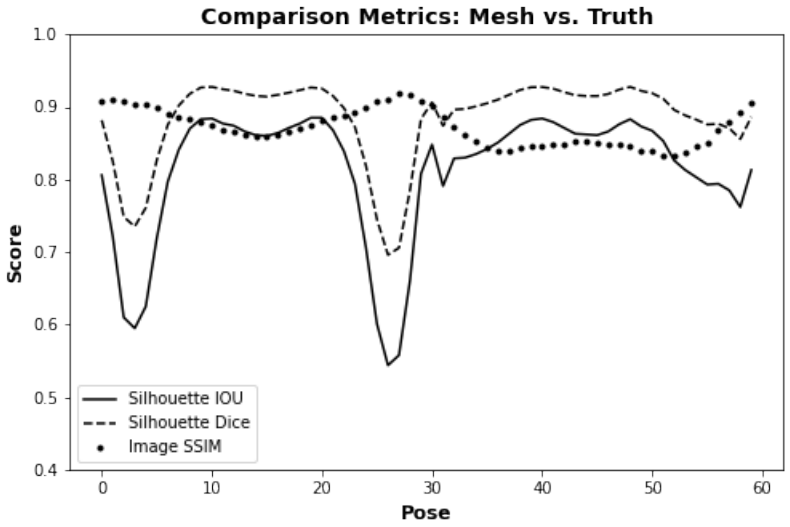


Figure 4: Image comparison metrics for the truth/mesh-recovered image pairs.

### 3.3. Reproducibility

For reproduction, all of the work herein was coded in Python. The optimization and rendering utilized PyTorch3D v0.4.0. Operating system and hardware specifications include Ubuntu Linux 18.04 on an NVidia DGX server with eight Tesla V100 GPUs with 32 GB of memory on each card (all optimization here was performed utilizing a single GPU).

## 4. Conclusion

It is clear that given sufficient view diversity of a satellite there are methods available that enable the discovery of a 3D structure representation. The utility of these approaches is limited by the number and diversity

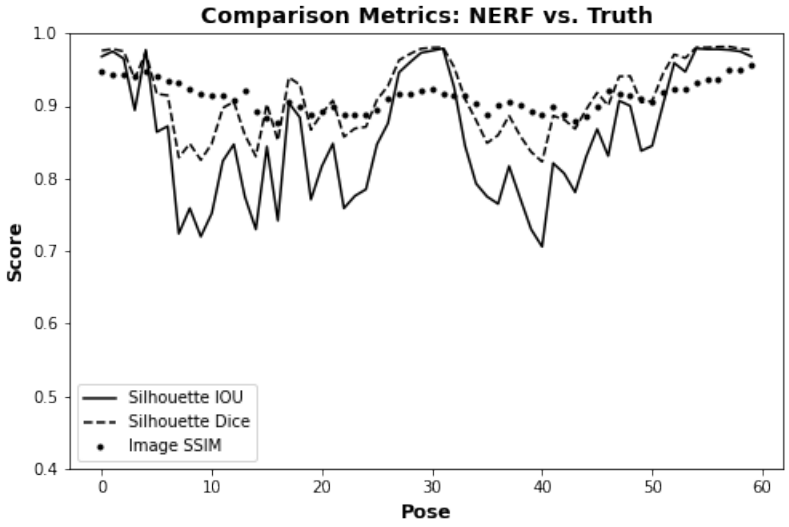


Figure 5: Image comparison metrics for the truth/NeRF-recovered image pairs.

of views required for recovery, method of recovery, quality of the images, and conditioning of the loss. Of the methods tested here, our simplified NeRF implementation shows the most promise. Indeed [14] shows exceptional quality with a more developed implementation. Both approaches warrant further study to better understand the limiting factors. Our initial implementations are likely the most prominent limitation. Additionally, we have shown that a pose fitting method utilizing a similar differentiable rendering approach can effectively provide virtual camera parameters and support the functionality of both methods. Future work will build on this start with more advanced implementations aimed towards a more robust and detailed reconstruction.

## References

[1] B. Jia, K. D. Pham, E. Blasch, Z. Wang, D. Shen, and G. Chen. Space object classification using deep neural networks. In *2018 IEEE Aerospace Conference*, pages 1–8, March 2018.

[2] Michael Werth, Jacob Lucas, Trent Kyono, Ian McQuaid, and Justin Fletcher. Quality-weighted iterative deep convolution (qwid). In *Advanced Maui Optical and Space Surveillance (AMOS) Technologies Conference*. 2020.

[3] Jacob Lucas, Trent Kyono, Ian McQuaid, and Justin Fletcher. Deep learning approach for satellite orientation determination from ground based resolved images. In *Advanced Maui Optical and Space Surveillance (AMOS) Technologies Conference*. 2020.

[4] Jacob Lucas, Michael Werth, Trent Kyono, Ian McQuaid, and Justin Fletcher. Automated interpretability scoring of ground-based observations of leo objects with deep learning. In *IEEE Aerospace*. 2020.

[5] Michael Werth, Trent Kyono, Ian McQuaid, and Justin Fletcher. Guiding multi-frame blind deconvolution with image recoveries from neural networks. In *Advanced Maui Optical and Space Surveillance (AMOS) Technologies Conference*. 2020.

[6] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020.

[7] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis, 2020.

[8] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn, 2020.

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

[10] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer, 2017.

[11] Shubham Tulsiani, Tinghui Zhou, Alexei A. Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency, 2017.

[12] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images, 2021.

[13] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes, 2020.

[14] Anne Mergy, Gurvan Lecuyer, Dawa Derksen, and Dario Izzo. Vision-based neural scene representations for spacecraft, 2021.

[15] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis, 2021.

[16] All resources, Jan 2021.

[17] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[18] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam,

[19] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation, 2016.

[20] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[21] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[22] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.