



Differentiable Procedural Models for Single-view 3D Mesh Reconstruction

Albert Garifullin¹ , Nikolay Maiorov¹ , and Vladimir Frolov^{1,2} 

¹Moscow State University, Moscow, Russia

²Keldysh Institute of Applied Mathematics, Moscow, Russia

Abstract

Most existing solutions for single-view 3D object reconstruction are based on deep learning with implicit or voxel representations of the scene and are unable to produce detailed and high-quality meshes and textures that can be directly used in practice. Differentiable rendering, on the other hand, is able to produce high-quality meshes but requires several images of an object. We propose a novel approach to single-view 3D reconstruction that uses procedural generator input parameters as a scene representation. Instead of estimating the vertex positions of the mesh directly, we estimate the input parameters of a procedural generator by minimizing the silhouette loss function between reference and rendered images. We use differentiable rendering and create partly differentiable procedural generators to use gradient-based optimization of the loss function. It allows us to create a highly detailed model from a single image taken in an uncontrolled environment. Moreover, the reconstructed model can be further modified in a convenient way by changing the estimated input parameters.

CCS Concepts

• **Computing methodologies** → *Rendering; Shape modeling;*

1. Introduction

The video game and movie industries require an extensive range of objects to populate virtual environments, often need realistic-looking models based on real-life objects. While modern technologies for scanning and 3D reconstruction can produce high-quality models, they often require complex equipment and significant amounts of time. Additionally, it may be impossible to scan an object or obtain a sufficient number of images. For these cases, single-view 3D reconstruction methods are being developed. Deep learning methods such as [PKS*19] and [YLZ22] have made remarkable progress, but its quality is still often insufficient for practical use. Usually deep learning methods do not consider an important aspect of humans' perception: structure and object components. Procedural generation is an alternative method for obtaining 3D models. Procedural models consist of rules that establish a correlation between input parameters and 3D objects of a specific class. In this paper, we propose a novel approach that utilizes procedural generation for single-view and multi-view mesh and texture reconstruction. Instead of estimating the vertex positions of the mesh directly, we estimate the input parameters of a procedural generator through the silhouette loss function between reference and rendered images. By using differentiable rendering and creating partly differentiable procedural generators for gradient-based. Our approach also allows for convenient model modifications by changing the estimated input parameters, such as creating new lev-

els of detail, altering the geometry and textures of individual mesh parts, and more.

2. Related work

2.1. Single-view 3D reconstruction

Single-view 3D reconstruction presents a significant challenge as it requires prior knowledge of the real world. Therefore, learning-based methods have become dominant in this field. These methods use different scene representations, such as meshes [WZL*18] [NHG*20] [YTG21], voxel grids [CXG*16] [PBF20], point clouds [FSG17] [CHLZ21] or implicit functions [CLG*19]. Among these methods, mesh reconstruction is the most relevant to our work. The majority of single-view 3D mesh reconstruction methods employ an encoder-decoder architecture where the encoder extracts features from the image and the decoder deforms an initial mesh to the target shape. It is noteworthy that these methods are trained and evaluated on the same object categories. Further work [TRR*19] showed that such approaches to single-view 3D reconstruction primarily perform recognition rather than reconstruction. There are also a few research works focused on the generalized single-view 3D reconstruction [ZZZ*18] [YLZ22], yet the quality of models reconstructed from a single image is often insufficient for practical use.

2.2. Differentiable rendering

Physically-based differential rendering [LHK*20] [ZYZ21] is an active area of research related to accurate 3D models and material reconstruction. Such algorithms provide a gradient of a silhouette loss function with respect to the scene parameters that can be then minimized with the gradient descent. An appropriate scene representation is also important for this approach. Mesh-based representation has been the most widely studied, and specific regularization [WFK21] and modification for the Adam optimizer were proposed [WS21] for this task. Recently, Nicolet et al. combined recent advancements in this field in their work [NJJ21] that significantly improved the quality of the resulting mesh. Other scene representations have also been studied. Vicini et al. [VSJ22] proposed an algorithm for differentiable signed distance function rendering and used it for multi-view reconstruction. Differential rendering can also be combined with deep learning [YLZ22] to provide face quality supervision and regularization.

2.3. Alternative scene representations

Mesh-based or surface-based representations enable efficient rendering and produce easy-to-use meshes as a result of their work. However, image-based optimization of surface geometry can be challenging due to the non-convexity of such an optimization. Volumetric representations [LSS*19] [VJK21] can reliably reach a desirable minimum, but are usually unable to capture finer details. Alternatively, point-based shape representations have also been shown to produce high-quality scene reconstructions [YSW*19] [RFS22]. Another key tool for scalable scene representation is the use of coordinate-based neural networks, also known as neural fields [MST*21] [FKYT*22] [MESK22], which push beyond the resolution limits of discretized grids and generalize to higher-dimensional signals, such as directional emission [MST*21]. Neural fields [MESK22] have demonstrated the ability to handle complex scenes and produce compelling results within seconds. However, to utilize the obtained results, one typically needs to convert them from an alternate representation into a 3D mesh, as most rendering engines work primarily with meshes. Such transformations can pose a significant challenge, resulting in suboptimal models with a large triangle count and lower visual quality.

2.4. Procedural generation

Procedural content generation is a well-known approach to creating diverse virtual worlds. It is used on different levels, ranging from individual objects to large-scale open worlds and game scenarios [HMVDV113] [FE17]. Many procedural object generators work as functions that transform a vector of numerical parameters into a 3D mesh of a specific class, e.g., trees, buildings. There are a few works focusing on inverse procedural generation, which involves estimating the input parameters for the generator given an input model or image [SPK*14] [GJB*20] [GSF22]. However, as non-trivial procedural generators are non-invertible functions, existing works have limited generalization ability and focus on very specific procedural models.

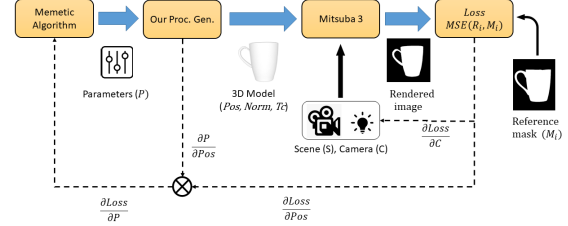


Figure 1: The mesh reconstruction process consists of a procedural generator creating a mesh from input parameters, which is then rendered using a differentiable renderer. The silhouette loss function is then calculated between the rendered output and reference image, with the gradient of this function being back-propagated to the optimizer to modify the input parameters.

3. Method overview

The proposed method implements object and texture reconstruction from one or multiple images using differentiable rendering and procedural generators. The method consists of three steps: mesh reconstruction, texture reconstruction, and post-processing. Both mesh and texture reconstruction are based on the minimization of loss functions (silhouette and texture loss, respectively). We define all scene parameters as $\pi = (P, T, S, C)$, where P is the input parameters of the procedural generator, T is an object's texture (or textures), S is global scene parameters (e.g., light sources, ambient light), and C is cameras' parameters.

Silhouette loss is defined as follows:

$$Loss_s(P, T, S, C) = \frac{1}{N} * \sum_{i=1}^N MSE(Render_s(Gen(P), T, S, C), M_i)$$

Texture loss:

$$Loss_t(P, T, S, C) = \frac{1}{N} * \sum_{i=1}^N MSE(Render_t(Gen(P), T, S, C), R_i)$$

R_i and M_i are reference images and binary masks for these images respectively, N is a number of reference images given, $Gen(P) = (pos, norm, tc)$ is a mesh created by procedural generator, $Render_s$ is an image of a scene rendered in silhouette mode and $Render_t$ is an image of a scene rendered in default mode, with textures and light.

In the first step, we find $P^*, C^* = \arg \min_{P, C} \{Loss_s(P, T, S, C)\}$

or $P^* = \arg \min_P \{Loss_s(P, T, S, C)\}$ if cameras' positions and settings are given.

In the texture reconstruction steps we find $T^* = \arg \min_T \{Loss_t(P^*, T, S, C^*)\}$ with the fixed cameras and geometry. For both steps, we use an iterative gradient-based optimization strategy. More details about the optimization process are provided below, but the key point is that it requires gradients of the loss function with respect to scene parameters. $\frac{dLoss_s}{dT}$ and $\frac{dLoss_s}{dC}$ can be easily obtained from the differentiable renderer. Our implementation

uses Mitsuba 3 [JSR*22] for this purpose. Obtaining $\frac{dLoss_s}{dP}$ is a bit more complicated. Considering the chain rule and the fact that silhouette does not depend on model's normal vectors and texture coordinates, we get $\frac{dLoss_s}{dP} = \frac{dLoss_s}{dPos} * \frac{dPos}{dP}$. $\frac{dLoss_s}{dPos}$ also comes from differentiable renderer and jacobian $\frac{dPos}{dP}$ is obtained from procedural generator. The whole process of optimizing the silhouette loss function is illustrated in Figure 1.

4. Differentiable procedural generators

The pipeline described in the previous section treats the procedural generator as a black-box function that can produce a mesh $(pos, norm, tc)$ and jacobian $\frac{dPos}{dP}$ from an input vector P . In practice, it is impossible to make a generator of non-trivial objects that works as a smooth, differentiable function. This is primarily due to objects having discrete properties, such as the number of floors in a building. For consistency, we still include these parameters in the vector P , but for each parameter P_d of this type, we assume that $\frac{dPos}{dP_d} = 0$.

For this work, two different procedural generators are created: for dishes and buildings. Both were developed from scratch using the CppAD automatic differentiation library [Bel12] to obtain the required derivatives. The dishes generator (Figure 2) is an example of a simple and easy-to-differentiate algorithm that only has one binary parameter. In contrast, buildings have numerous discrete features, and the building generator (Figure 3) was created to demonstrate the ability of our method to handle such challenges.

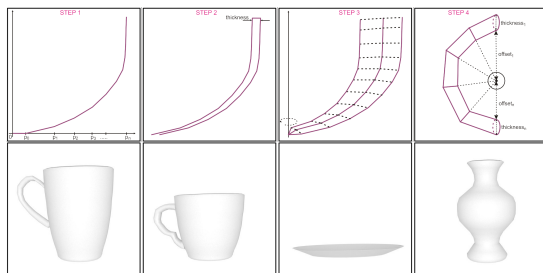


Figure 2: Dishes generator. Top row: generation steps. 1) Create a spline from a vector of vertical offsets. 2) Transform it to create a closed spline with thickness. 3) Rotate it to get the dish body. 4) (Optional) Create a handle from a circle spline and a vector of offsets. The number of points in splines can be changed to produce different levels of detail. Bottom row: some examples of generated dishes. 1) Mug 2) Tea cup 3) Bowl 4) Jar

5. Optimization

Even simple procedural models from previous section prove to be extremely challenging functions to optimize due to two factors: the large number of non-linear internal dependencies in the generator and heterogeneous input parameters, some of which are integers with a limited set of possible values. Previous efforts at 3D reconstruction with procedural generators [GSF22] relied on a specific implementation of the genetic algorithm [Mit98] to find solutions

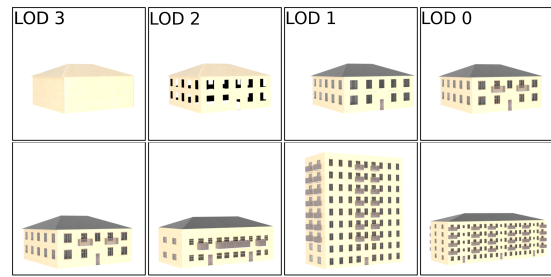


Figure 3: Building generator. Top row: Levels of detail for the same building. 1) Box with a roof. 2) Only outer walls. 3) Building without small details 4) Full detailed building Bottom row: some examples of generated buildings



Figure 4: Results of model reconstruction after each optimization step. From left to right: reference image, genetic algorithm with 128x128 rendering resolution, gradient descent with resolution of 256x256 and 512x512 respectively

without gradient calculation, but only for a restricted set of problems. Being able to calculate the gradient of the loss function expands the list of available methods for optimization. However, we still cannot obtain the derivatives with respect to parameters that represent the procedural model's discrete features.

To address this issue, we use the memetic algorithm [NC12], which combines genetic algorithm with gradient-based optimization. We start with an initial population, a set of initial parameter values, which is taken from presets that have been prepared in advance. Each preset is a set of input parameters representing an adequate model (like those shown on Figure 2). The memetic algorithm performs random mutations and recombination of the initial population, along with gradient-based optimization. Although this process requires several thousand iterations, it can be performed on models with low level of detail and low rendering resolution. To further improve the quality, we use the solution obtained from the memetic algorithm as an initial approximation for the next round of gradient-based optimization with higher levels of detail and higher rendering resolution. For texture reconstruction, only the gradient-based optimization step is performed. The results at each step are shown in Figure 4.

6. Results

In this section, we present the results of single-view and multi-view 3D reconstruction using our method. We used both real-life photos and rendered images as input for our algorithm. Figure 5 shows the results of our single-view reconstruction on photos taken in an uncontrolled environment. Our algorithm was able to accurately reconstruct the mesh for both cups with high precision. Additionally,

the texture was reconstructed, with some assumptions made about the invisible areas. For the building generator, we tested two different reconstruction methods. The first one, "textured box", is to create only the simplest box model and then wrap it with a reconstructed texture, it can be enough for distant objects. The second is to create detailed mesh but skip the texture reconstruction, as the input resolution is not enough to catch fine texture details. Figure 6 shows the results of both methods.



Figure 5: Single-view reconstruction results with dishes procedural generator. Top row: reference images. Bottom row: reconstruction results.



Figure 6: Single-view reconstruction of building. From left to right: reference image, window mask, result of "textured box" reconstruction, "detailed reconstruction" result with generic textures.

6.1. Comparison

We compared the proposed method with different other solutions for 3D reconstruction to demonstrate that our approach is able to produce visually better results even from a single input image. Figure 7 demonstrates that procedurally generated mesh has more consistent structure and better restores the concave shape of input object. Procedural model applies strict rules on model structure and guarantees that the output mesh will have adequate structure if it is impossible to match the input object precisely. Figure 8 shows how an increase in the number of viewpoints affects quality of reconstruction.

7. Conclusion and future work

In this work, we have presented a novel single-view 3D reconstruction approach that estimates the input parameters of a procedural generator to reconstruct the model. We have implemented an efficient strategy for finding the optimal parameter set based on a single input image. Our methods demonstrates better results compared to existing approaches and produces meshes with fewer artifacts. However, our approach has some limitations, primarily that it only works well on a class of objects that the underlying procedural generator can create. The procedural generators used in this



Figure 7: Results using our method compared to differentiable SDF reconstruction [VSJ22], InstantNGP [MESK22] and Pixel2Mesh [WZL*18]. We measured the average silhouette intersection over union (IoU) between reference and reconstructed models for 64 uniformly distributed viewpoints. Our approach is far better than Pixel2Mesh single-view reconstruction and has comparable results to multi-view reconstruction methods.

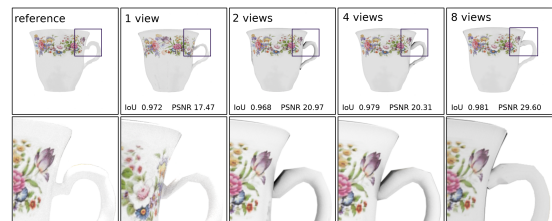


Figure 8: Results of 3D reconstruction with our approach and different viewpoints. One viewpoint is enough for mesh reconstruction, and further increases do not make much improvement. However, more viewpoints allow to perform accurate texture reconstruction.

work were created from scratch and are limited in their abilities. As future research, we plan to generalize our approach to a wider class of objects. It requires either integrating existing generators into our reconstruction pipeline or using generative neural networks.

8. Acknowledgements

The work was sponsored by the non-profit Foundation for the Development of Science and Education "Intellect".

References

- [Bel12] BELL B. M.: Cppad: a package for c++ algorithmic differentiation. *Computational Infrastructure for Operations Research* 57, 10 (2012), 3
- [CHLZ21] CHEN C., HAN Z., LIU Y.-S., ZWICKER M.: Unsupervised learning of fine structure generation for 3d point clouds by 2d projections matching. In *Proceedings of the IEEE/CVF international conference on computer vision* (2021), pp. 12466–12477. 1
- [CLG*19] CHEN W., LING H., GAO J., SMITH E., LEHTINEN J., JACOBSON A., FIDLER S.: Learning to predict 3d objects with an interpolation-based differentiable renderer. *Advances in neural information processing systems* 32 (2019). 1
- [CXG*16] CHOY C. B., XU D., GWAK J., CHEN K., SAVARESE S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VIII 14* (2016), Springer, pp. 628–644. 1
- [FE17] FREIKNECHT J., EFFELSBURG W.: A survey on the procedural generation of virtual worlds. *Multimodal Technologies and Interaction* 1, 4 (2017), 27. 2
- [FKYT*22] FRIDOVICH-KEIL S., YU A., TANCİK M., CHEN Q., RECHT B., KANAZAWA A.: Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 5501–5510. 2
- [FSG17] FAN H., SU H., GUIBAS L. J.: A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 605–613. 1
- [GJB*20] GUO J., JIANG H., BENES B., DEUSSEN O., ZHANG X., LISCHINSKI D., HUANG H.: Inverse procedural modeling of branching structures by inferring l-systems. *ACM Transactions on Graphics (TOG)* 39, 5 (2020), 1–13. 2
- [GSF22] GARIFULLIN A., SHCHERBAKOV A., FROLOV V.: Fitting parameters for procedural plant generation. 2, 3
- [HMVDVI13] HENDRIKX M., MEIJER S., VAN DER VELDEN J., IOSUP A.: Procedural content generation for games: A survey. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 9, 1 (2013), 1–22. 2
- [JSR*22] JAKOB W., SPEIERER S., ROUSSEL N., NIMIER-DAVID M., VICINI D., ZELTNER T., NICOLET B., CRESPO M., LEROY V., ZHANG Z.: Mitsuba 3 renderer, 2022. <https://mitsuba-renderer.org>. 3
- [LHK*20] LAINE S., HELLSTEN J., KARRAS T., SEOL Y., LEHTINEN J., AILA T.: Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–14. 2
- [LSS*19] LOMBARDI S., SIMON T., SARAGIH J., SCHWARTZ G., LEHRMANN A., SHEIKH Y.: Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751* (2019). 2
- [MESK22] MÜLLER T., EVANS A., SCHIED C., KELLER A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)* 41, 4 (2022), 1–15. 2, 4
- [Mit98] MITCHELL M.: *An introduction to genetic algorithms*. MIT press, 1998. 3
- [MST*21] MILDENHALL B., SRINIVASAN P. P., TANCİK M., BARRON J. T., RAMAMOORTHY R., NG R.: Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM* 65, 1 (2021), 99–106. 2
- [NC12] NERI F., COTTA C.: Memetic algorithms and memetic computing optimization: A literature review. *Swarm and Evolutionary Computation* 2 (2012), 1–14. 3
- [NHG*20] NIE Y., HAN X., GUO S., ZHENG Y., CHANG J., ZHANG J. J.: Total3dunderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 55–64. 1
- [NJJ21] NICOLET B., JACOBSON A., JAKOB W.: Large steps in inverse rendering of geometry. *ACM Transactions on Graphics (TOG)* 40, 6 (2021), 1–13. 2
- [PBF20] POPOV S., BAUSZAT P., FERRARI V.: Corenet: Coherent 3d scene reconstruction from a single rgb image. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16* (2020), Springer, pp. 366–383. 1
- [PKS*19] PONTES J. K., KONG C., SRIDHARAN S., LUCEY S., ERIKSSON A., FOOKES C.: Image2mesh: A learning framework for single image 3d reconstruction. In *Computer Vision—ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part I 14* (2019), Springer, pp. 365–381. 1
- [RFS22] RÜCKERT D., FRANKE L., STAMMINGER M.: Adop: Approximate differentiable one-pixel point rendering. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–14. 2
- [SPK*14] STAVA O., PIRK S., KRATT J., CHEN B., MÈCH R., DEUSSEN O., BENES B.: Inverse procedural modelling of trees. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 118–131. 2
- [TRR*19] TATARCHENKO M., RICHTER S. R., RANFTL R., LI Z., KOLTUN V., BROX T.: What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2019), pp. 3405–3414. 1
- [VJK21] VICINI D., JAKOB W., KAPLAYAN A.: A non-exponential transmittance model for volumetric scene representations. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–16. 2
- [VSJ22] VICINI D., SPEIERER S., JAKOB W.: Differentiable signed distance function rendering. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–18. 2, 4
- [WFK21] WICKRAMASINGHE U., FUA P., KNOTT G.: Deep active surface models. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 11647–11656. doi: 10.1109/CVPR46437.2021.01148. 2
- [WS21] WANG Y., SOLOMON J.: Fast quasi-harmonic weights for geometric data interpolation. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–15. 2
- [WZL*18] WANG N., ZHANG Y., LI Z., FU Y., LIU W., JIANG Y.-G.: Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European conference on computer vision (ECCV)* (2018), pp. 52–67. 1, 4
- [YLZ22] YANG X., LIN G., ZHOU L.: Zeromesh: Zero-shot single-view 3d mesh reconstruction. *arXiv preprint arXiv:2208.02676* (2022). 1, 2
- [YSW*19] YIFAN W., SERENA F., WU S., ÖZTIRELI C., SORKINE-HORNUNG O.: Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–14. 2
- [YTG21] YE Y., TULSIANI S., GUPTA A.: Shelf-supervised mesh prediction in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 8843–8852. 1
- [ZYZ21] ZHANG C., YU Z., ZHAO S.: Path-space differentiable rendering of participating media. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–15. 2
- [ZZZ*18] ZHANG X., ZHANG Z., ZHANG C., TENENBAUM J., FREEMAN B., WU J.: Learning to reconstruct shapes from unseen classes. *Advances in neural information processing systems* 31 (2018). 1