Neural Volumetric Mesh Generator

Yan Zheng
The University of Texas at Austin
yanzheng@cs.utexas.edu

Lemeng Wu
The University of Texas at Austin
lmwu@cs.utexas.edu

Xingchao Liu
The University of Texas at Austin
xcliu@cs.utexas.edu

Zhen Chen
The University of Texas at Austin
zchen96@cs.utexas.edu

Qiang Liu
The University of Texas at Austin
lqiang@cs.utexas.edu

Qixing Huang
The University of Texas at Austin
huangqx@cs.utexas.edu

Abstract

Deep generative models have shown success in generating 3D shapes with different representations. In this work, we propose Neural Volumetric Mesh Generator (NVMG), which can generate novel and high-quality volumetric meshes. Unlike the previous 3D generative model for point cloud, voxel, and implicit surface, volumetric mesh is a ready-to-use representation in industry with details on both the surface and interior. Generating this kind of highly-structured data thus brings a great challenge. To tackle this problem, we first propose to use a diffusion-based generative model to generate voxelized shapes with realistic shape and topology information. With the voxelized shape, we can simply obtain a tetrahedral mesh as a template. Further, we use a voxel-conditional neural network to predict the surface conditioned on the voxels, and progressively project the tetrahedral mesh to the predicted surface under regularization. As shown in the experiments, without any post-processing, our pipeline can generate high-quality artifact-free volumetric and surface meshes.

1 Introduction

Automatic 3D contents creation is a key problem with the development of AR/VR. Although generative models have revealed their power on audio and image synthesis [8, 15, 10, 8, 4, 11, 24, 12, 25], their performance is still unsatisfying in 3D generation. In this work, we consider the problem of directly generating ready-to-use volumetric meshes. Volumetric mesh is one of the most important representations of 3D shapes, which is widely adopted in computer graphics and engineering [23, 9, 13]. Without carefully handling geometric constraints [3, 19, 17, 18, 22], the generated meshes will have various defects, including flipping, self-intersection, large distortion, etc.

Thus, we present a novel pipeline, termed Neural Volumetric Mesh Generator (NVMG), for learning generative models of volumetric meshes. We first generate a rough voxelized shape by a diffusion model adapted to voxels. Splitting voxels into tetrahedra, we can easily get a standardized tetrahedral mesh that then serves as a template, which allows sufficient variety in topological changes [6] compared with pre-defined templates like ellipsoid. Next, NVMG learns a neural network to estimate the continuous surface conditioned on the generated voxelized shape. We progressively deform the outmost vertices on the tetrahedral mesh by projecting them to the predicted surface under a series of regularization terms. The projection objective is carefully designed to promote smoothness of

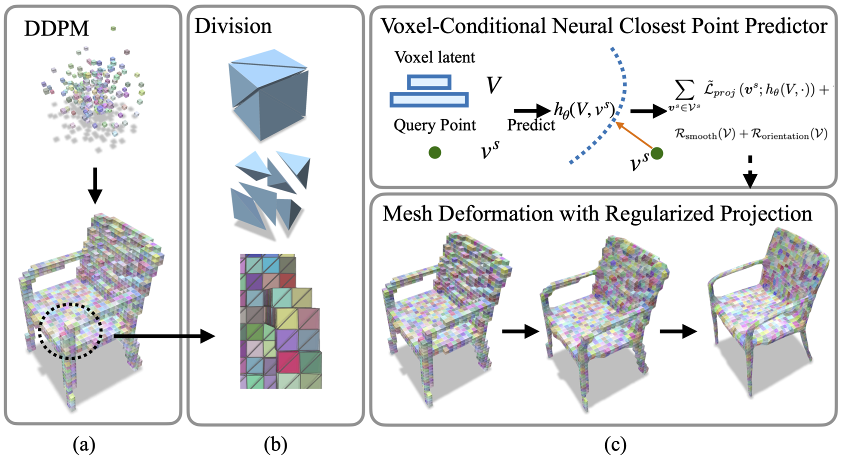
DDPM
Division
Voxel-Conditional Neural Closest Point Predictor
Voxel latent
V
Query Point
v^s
Predict
$h_\theta(V, v^s)$
$\sum_{v^s \in \mathcal{V}^s} \tilde{\mathcal{L}}_{proj}(\boldsymbol{v}^s; h_\theta(V, \cdot)) +$
$\mathcal{R}_{\text{smooth}}(\mathcal{V}) + \mathcal{R}_{\text{orientation}}(\mathcal{V})$
v^s
Mesh Deformation with Regularized Projection
(a)
(b)
(c)

Figure 1: NVMG mainly contains three modules. (a) a generative model to generate voxelized shapes from noise distribution; (b) Voxel division for the initial volumetric mesh; (c) Physically correct mesh deformation with voxel-conditional neural closest point predictor and regularized projection.

the mesh, preserve the topology of the tetrahedra and suppress the magnitude of distortion. Like squeezing the clay, the deformation of the outmost vertex on the surface is propagated into the interior region implicitly by our regularization terms. After the deformation, NVMG generates a ready-to-use volumetric mesh without defects like flipping or high distortion.

We empirically evaluate NVMG on unconditional volumetric mesh generation. We show our generated volumetric has a competitive generation quality compared with state-of-the art point cloud generator [27]. When we compare our mesh deformation module with the state-of-the-art voxel-to-mesh method, Neural Dual Contouring, we obtain a robust performance on converting generated voxel to mesh.

2 Neural Volumetric Mesh Generator

2.1 Voxel Generation with Diffusion Models

Voxel Generation with DDPM The first step of our framework is voxel generation. We exploit denoising diffusion probabilistic model (DDPM) [12] for this task because of its simplicity and effectiveness. In our scenario, shapes are normalized to a unit cube with $r \times r \times r$ voxels. A voxel representation of a shape is encoded by a binary tensor $V \in \{0, 1\}^{r \times r \times r}$, where $V(x, y, z) = 0/1$ means that the (x, y, z)-th voxel is unfilled/filled. The training set for voxel generation is created by discretizing the original meshes in the training set to their voxel representations with specific resolution. We adopt a similar 3D convolutional neural network as in [27] to parameterize the noise predictor $\boldsymbol{\epsilon}_\phi$, with several modifications to accommodate it to the voxel generation task.

Tetrahedral Mesh from Voxel Division We then divide each voxel into 6 tetrahedra as in Figure 1(Division) to achieve the face-to-face consistency across the tetrahedral mesh. Viewing each voxel as a cube mesh, we can extract a hexahedral mesh $\mathcal{H}$ from the voxel representation $\mathcal{H} = (\mathcal{V}, \mathcal{F}_{quad}) = (\bigcup_c \{\boldsymbol{v}_c^i\}_{i=1}^{i=8}, \bigcup_c \{\boldsymbol{f}_c^j\}_{j=1}^{j=6})$, where $\mathcal{V}$ is vertex set, $\mathcal{F}_{quad}$ is qudrilateral face set, $\boldsymbol{v}_c^i$ is the i-th vertex of the c-th cube and $\boldsymbol{f}_c^j$ is the j-th quadrilateral face of the c-th cube. By splitting each cube mesh into 6 tetrahedra, we can further extract a tetrahedral mesh $\mathcal{T} = (\mathcal{V}, \mathcal{F}_{tri}) = (\bigcup_c \bigcup_{k=1}^{6} \{\boldsymbol{v}_{c,k}^i\}_{i=1}^{i=4}, \bigcup_c \bigcup_{k=1}^{6} \{\boldsymbol{f}_{c,k}^j\}_{j=1}^{j=4})$, where $\mathcal{V}$ is the vertex set same as hexahedral mesh, $\mathcal{F}_{tri}$ is triangle face set, $\boldsymbol{v}_{c,k}^i$ is the i-th vertex of the k-th tetrahedron in the c-th cube and $\boldsymbol{f}_{c,k}^j$ is the j-th triangular face of the k-th tetrahedron in the c-th cube. For volumetric meshes, as there are interior meshes, we use $\mathcal{V}^s$ to denote the set of the outmost vertices on the surface.

2.2 A Voxel-Conditional Neural Closest Point Predictor

The voxelized shape and its corresponding tetrahedral mesh is a coarse representation the shape of interest. In this section, we use a neural network to predict the fine-grained surface given the voxelized shape. We represent the surface by the closest point function. Given a surface S, the closest point function maps any point $\boldsymbol{x} \in \mathbb{R}^3$ to its closest point on S, $\mathrm{CP}(\boldsymbol{x}) = \underset{\boldsymbol{p} \in S}{\arg\min} \|\boldsymbol{p} - \boldsymbol{x}\|_2$. However, in generation tasks, S is not known beforehand. Therefore, we use a neural network $h_\theta(V, \boldsymbol{x}) : \{0,1\}^{r^3} \times \mathbb{R}^3 \to \mathbb{R}^3$ to learn the closest point function conditioned on the voxelized shape V. Given V, the surface S is implicitly defined by $S = \{\boldsymbol{x} \in \mathbb{R}^3 \mid h_\theta(V, \boldsymbol{x}) - \boldsymbol{x} = \boldsymbol{0}\}$.

To prepare training samples, we calculate the closest point $\boldsymbol{p}^s$ on the ground truth object surface to each vertex $\boldsymbol{v}^s \in \mathcal{V}^s$ on triangular mesh surface.

2.3 Physically Robust Mesh Deformation with Regularized Projection

Now we introduce how to get the resulting mesh from the voxelized shape and the trained voxel-conditional neural surface predictor h_θ. A straightforward idea is to directly project the vertices to its closest point inferred by h_θ. Nonetheless, this naive projection does not work well in practice. Therefore, we resort to a regularized projection to optimize the vertices collectively. The regularization terms are designed to encourage the set of vertices to satisfy the necessary conditions for successfully constructing a volumetric mesh. Our optimization problem is formulated as follows,

$$\mathcal{L}(\mathcal{V}) = \sum_{\boldsymbol{v}^s \in \mathcal{V}^s} \tilde{\mathcal{L}}_{proj}\left(\boldsymbol{v}^s; h_\theta(V, \cdot)\right) + \mathcal{R}_{\text{smooth}}(\mathcal{V}) + \mathcal{R}_{\text{orientation}}(\mathcal{V}). \tag{1}$$

$\tilde{\mathcal{L}}_{proj}$ is the projection term and $\mathcal{R}_{\text{smooth}}$ and $\mathcal{R}_{\text{orientation}}$ are the regulatization terms. Note that our regularization term $\mathcal{R}(\mathcal{V})$ only changes the distribution of the vertices in the space, but not modify the target surface S. We define $\mathcal{R}(\mathcal{V})$ to be differentiable so that $\mathcal{L}(\mathcal{V})$ can be easily optimized with any gradient-based optimizer.

1) Smooth term for uniform structure A satisfying volumetric mesh should not only have a surface mesh $\mathcal{T}^s$ that fits the target surface closely, but should also have a uniform inner volumetric tetrahedral mesh. For this purpose, we add $\mathcal{R}_{\text{smooth}}$:

$$\mathcal{R}_{\text{smooth}}(\mathcal{V}) = \lambda_a \mathcal{R}_{\text{edge}}(\mathcal{V}, \mathcal{F}_{tri}) + \lambda_b \mathcal{R}_{\text{laplacian}}(\mathcal{V}, \mathcal{F}_{tri}) + \lambda_c \mathcal{R}_{\text{normal}}(\mathcal{V}^s, \mathcal{F}^s_{tri}). \tag{2}$$

We keep the mesh connectivity $\mathcal{F}_{tri}$ and $\mathcal{F}^s_{tri}$ invariant during optimization. $\mathcal{R}_{\text{edge}}$ penalizes the length of edges to prevent extremely long edges; $\mathcal{R}_{\text{laplacian}}$ is a graph-based term smoothing the mesh and $\mathcal{R}_{\text{normal}}$ enforces normal consistency among adjacent faces on the surface.

2) Orientation term to prevent defects Defects on mesh connectivity like flipping breaks local injectivity[1] and causes invalid tetrahedral mesh. It will make the mesh unqualified for certain downstream applications without applying non-trivial mesh repair. The orientation term, which is widely used in many simulation tasks [17, 18, 22], ensures a minimal positive volume v_0 of tetrahedron, which helps to avoid unexpected artifacts on the generated mesh:

$$\mathcal{R}_{\text{orientation}}(\mathcal{V}) = \sum_{c,k} l(\boldsymbol{M}^{c,k}), \tag{3}$$

$$l(\boldsymbol{M}^{c,k}) = \begin{cases} -(\det(\boldsymbol{M}^{c,k}) - v_0)^2 \log(\frac{\det(\boldsymbol{M}^{c,k})}{v_0}) & , \det(\boldsymbol{M}^{c,k}) \le v_0 \\ 0 & , \det(\boldsymbol{M}^{c,k}) > v_0 \end{cases}, \tag{4}$$

where $\boldsymbol{M}^{c,k} \in \mathbb{R}^{4\times4}$ is the matrix stacked by the homogeneous coordinates of four vertices of the k-th tetrahedron in the c-th cube.

3) Robust Projection for distortion suppression We empirically observe that, if all the vertices approaches the surface, they will collide with each other, causing excessive gradients, and finally gives meshes with large distortion. To avoid this, we replace $\mathcal{L}_{proj}$ with a robust version,

$$\tilde{\mathcal{L}}_{proj}\left(\boldsymbol{v}^s; h_\theta(V, \cdot)\right) = \|\boldsymbol{v}^s - h_\theta(V, \boldsymbol{v}^s) + k\boldsymbol{n}\|_2, \tag{5}$$

where $\boldsymbol{n} \sim \mathcal{N}(0, 1)$ is a random Gaussian noise, k is a coefficient that gradually decays to 0 as optimization proceeds. By adding this noise term, the 'lucky' vertices arrive at the surface earlier, while the 'unlucky' vertices arrive later. Thus the collisions and the excessive gradients are avoided, and the distortion on the resulting mesh is significantly reduced.

After obtaining the voxelized shape and the tetrahedral mesh from Sec. 2.1, we optimize Eq. (1) to yield the final volumetric mesh.

3 Experiment

In this section, we empirically show that NVMG can generate high-fidelity shapes with fine details on the mesh surface by comparing with other unconditional generative model on 3D shape generation task. For the evaluation metric, as discussed in [26, 27], 1-Nearest Neighbor(1-NN) is a more robust metric comparing with other metrics, so we choose it in our experiment. For data processing, we use Chair and Airplane classes and voxelized the raw mesh to $32 \times 32 \times 32$ into voxel with the scale fitting the bounding box, which means making full use of the voxel resolution. After generating the mesh, we uniformly sample 2048 points on the surface to calculate the 1-NN score.

Result. We see from Table 1, NVMG gets a comparable and even better performance compared with the state-of-the-art point cloud generation baseline. Figure 3 shows the visualization of our generated samples comparing with point cloud generative model PVD [27] by sampling the point on the mesh.

Base rep.	Method	Airplane		Chair	
		CD	EMD	CD	EMD
Point Cloud	1-GAN [2]	87.30	93.95	68.58	83.84
	PointFlow [26]	75.68	70.74	62.84	60.57
	DPF-Net [16]	75.18	65.55	62.00	58.53
	SoftFlow [14]	76.05	65.80	59.21	60.05
	ShapeGF [5]	80.00	76.17	68.96	65.48
	Diffusion [20]	74.14	65.12	56.24	54.28
	PVD [27]	**73.82**	**64.81**	56.26	53.32
Mesh	PolyGen [21]	81.51	72.32	69.12	63.90
	NVMG (w/ NDC [7])	76.41	67.20	58.75	55.38
	NVMG	74.41	65.93	**56.12**	**53.30**

Table 1: Result compared with various point cloud generation baselines. CD indicates Chamfer distance and EMD indicates Earth Mover's Distance. Lower score means better generate quality and diversity.

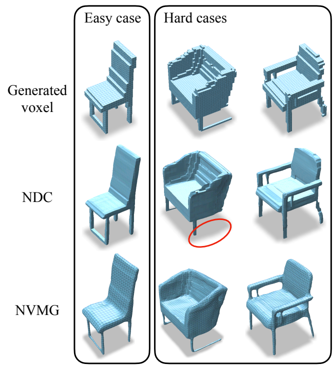

Figure 2: Mesh deformation uses our method and NDC. We notice the heavy artificial and even a missing leg (column 2 in red circle) on the NDC mesh for the hard cases.

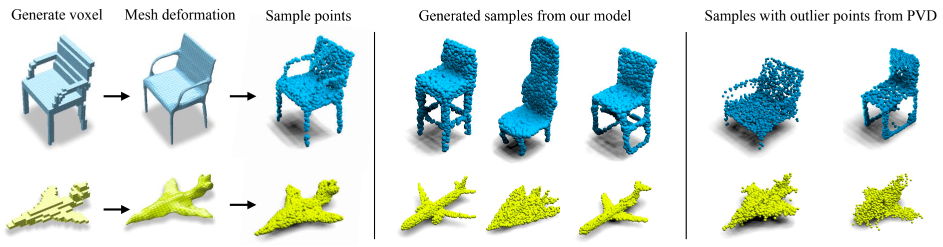
Generate voxel
Mesh deformation
Sample points
Generated samples from our model
Samples with outlier points from PVD

Figure 3: We show the generation procedure to produce point cloud samples, generated point cloud sampled from our model, and the outlier point case in other regular point cloud generation models (PVD [27] as an example here).

Robustness of the mesh deformer. We compare our mesh deformation part with the state-of-the-art method Neural Dual Contouring (NDC) [7] by taking our generated voxel samples as input to show our designed mesh deformer is more robust and suit for our proposed pipeline. From Table 1 last two lines, we observed a higher performance of our method compared with NDC. Further in Figure 2, we visualize some cases and see our result is more robust with less artifact in the hard cases. The above result indicates our design of the deformation-based method and robustness regularization loss works better in our proposed pipeline comparing with the off-the-shell marching cube based voxel to mesh method.

References

[1] Sayed Mazdak Abulnaga. *Volumetric mesh parameterization to a canonical template*. PhD thesis, Massachusetts Institute of Technology, 2018.

[2] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pp. 40–49. PMLR, 2018.

[3] Noam Aigerman and Yaron Lipman. Injective and bounded distortion mappings in 3d. *ACM Transactions on Graphics (TOG)*, 32(4):1–14, 2013.

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018.

[5] Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *European Conference on Computer Vision*, pp. 364–381. Springer, 2020.

[6] Li Chen and Yongwu Rong. Linear time recognition algorithms for topological invariants in 3d. In *2008 19th International Conference on Pattern Recognition*, pp. 1–4. IEEE, 2008.

[7] Zhiqin Chen, Andrea Tagliasacchi, Thomas Funkhouser, and Hao Zhang. Neural dual contouring. *ACM Transactions on Graphics (Special Issue of SIGGRAPH)*, 41(4), 2022.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[9] Si Hang. Tetgen, a delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Softw*, 41(2):11, 2015.

[10] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

[11] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pp. 2722–2730. PMLR, 2019.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[13] Yixin Hu, Qingnan Zhou, Xifeng Gao, Alec Jacobson, Denis Zorin, and Daniele Panozzo. Tetrahedral meshing in the wild. *ACM Trans. Graph.*, 37(4):60–1, 2018.

[14] Hyeongju Kim, Hyeonseung Lee, Woo Hyun Kang, Joun Yeop Lee, and Nam Soo Kim. Softflow: Probabilistic framework for normalizing flow on manifolds. *Advances in Neural Information Processing Systems*, 33:16388–16397, 2020.

[15] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[16] Roman Klokov, Edmond Boyer, and Jakob Verbeek. Discrete point flow networks for efficient point cloud generation. In *European Conference on Computer Vision*, pp. 694–710. Springer, 2020.

[17] Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M. Kaufman. Incremental potential contact: Intersection- and inversion-free, large-deformation dynamics. *ACM Trans. Graph.*, 39(4), jul 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392425. URL `https://doi.org/10.1145/3386569.3392425`.

[18] Minchen Li, Danny M. Kaufman, and Chenfanfu Jiang. Codimensional incremental potential contact. *ACM Trans. Graph.*, 40(4), jul 2021. ISSN 0730-0301. doi: 10.1145/3450626.3459767. URL `https://doi.org/10.1145/3450626.3459767`.

[19] Xin Li, Xiaohu Guo, Hongyu Wang, Ying He, Xianfeng Gu, and Hong Qin. Harmonic volumetric mapping for solid modeling applications. In *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, pp. 109–120, 2007.

[20] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2837–2845, 2021.

[21] Charlie Nash, Yaroslav Ganin, SM Ali Eslami, and Peter Battaglia. Polygen: An autoregressive generative model of 3d meshes. In *International conference on machine learning*, pp. 7220–7229. PMLR, 2020.

[22] Ruiqi Ni, Teseo Schneider, Daniele Panozzo, Zherong Pan, and Xifeng Gao. Robust amp; asymptotically locally optimal uav-trajectory generation based on spline subdivision. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7715–7721, 2021. doi: 10.1109/ICRA48506.2021.9561272.

[23] Matthias Nieser, Ulrich Reitebuch, and Konrad Polthier. Cubecover–parameterization of 3d volumes. In *Computer graphics forum*, volume 30, pp. 1397–1406. Wiley Online Library, 2011.

[24] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[25] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020.

[26] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4541–4550, 2019.

[27] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5826–5835, 2021.