Survey of Mesh and Tetrahedron

Hongchi Xia
University of Illinois Urbana-Champaign
hongchix@illinois.edu

Wei-Cheng Huang
University of Illinois Urbana-Champaign
wch7@illinois.edu

Abstract

This survey explores recent advances in tetrahedral meshing and its applications within the field of 3D vision and reconstruction. Tetrahedral meshes play a crucial role in the representation of 3D objects, allowing for efficient simulations and geometric processing. The first part of the survey covers key methodologies such as surface-to-tetrahedral mesh transformation and isosurface extraction using techniques like DMTet and Flexicubes. The survey also explores the evolution of mesh generation from 2D images and 3D point clouds, focusing on data-driven approaches like DefTet and DefGrid. The survey also discusses the broader impact of these advancements on downstream applications such as physics-based simulations, high-fidelity rendering, and text-to-3D generation. The survey highlights how continuous improvements in tetrahedral meshing and surface reconstruction algorithms are enabling new possibilities for complex 3D shape generation, optimization, and real-time rendering.

1. Introduction

The field of 3D vision has experienced remarkable advancements in recent years, driven by both theoretical developments and practical applications. A central problem in this domain is **3D reconstruction**, which refers to the process of creating a digital, three-dimensional representation of objects or scenes from two-dimensional data, such as images or point clouds. This problem is inherently ill-posed due to the loss of depth information in 2D projections. However, modern algorithms, including Structure-from-Motion (SfM) [20] and Simultaneous Localization and Mapping (SLAM) [25], have significantly improved the accuracy and robustness of 3D reconstructions.

Once a 3D model is reconstructed, an essential task is to convert the model into a form suitable for further analysis, simulation, or rendering. **Tetrahedral meshing** is one such representation that has garnered substantial attention [4, 9, 11]. Tetrahedral meshes decompose a 3D object into smaller, non-overlapping tetrahedra, allowing for efficient numerical methods and finite element simulations. This process is crucial in applications like physics-based simulations [24], where the accuracy and stability of the solution depend heavily on the quality of the mesh.

Tetrahedral meshing can be understood through several key concepts: *parameterization*, *representation*, and *Marching Tetrahedra*.

Tetrahedron Parameterization A tetrahedron is the simplest three-dimensional polyhedron, consisting of four vertices, six edges, and four triangular faces. Given four vertices $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4 \in \mathbb{R}^3$, the corresponding tetrahedron can be described using a set of barycentric coordinates $\mathbf{b} = (b_1, b_2, b_3, b_4)$, where:

$$b_1 + b_2 + b_3 + b_4 = 1, \quad \text{with} \quad b_i \geq 0, \forall i.$$

Barycentric coordinates allow us to express any point inside a tetrahedron as a weighted sum of the positions of the four vertices, which simplifies interpolation tasks such as field values or deformations. For instance, any point $\mathbf{p}$ inside the tetrahedron can be written as:

$$\mathbf{p} = b_1\mathbf{v}_1 + b_2\mathbf{v}_2 + b_3\mathbf{v}_3 + b_4\mathbf{v}_4$$

where the weights, b_1, b_2, b_3, b_4, are the barycentric coordinates. This condition ensures that the point $\mathbf{p}$ lies inside (or on the surface of) the tetrahedron.

These barycentric coordinates allow for the interpolation of scalar fields or vector fields across the tetrahedron's volume. For instance, if each vertex $\mathbf{v}_i$ has a scalar field value f_i, then the interpolated value at point $\mathbf{p}$ is given by:

$$f(\mathbf{p}) = b_1 f_1 + b_2 f_2 + b_3 f_3 + b_4 f_4.$$

The flexibility of parameterization facilitates various applications, including deformation modeling, physical simulation, and field interpolation within 3D volumes.

Tetrahedral Representation The structure of a tetrahedral mesh is typically represented as a collection of tetrahedral elements, each defined by four vertices. A common data structure to store a tetrahedral mesh is a list

of vertices $V = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ and a list of tetrahedra $T = \{t_1, t_2, \ldots, t_m\}$, where each tetrahedron $t_i = (\mathbf{v}_{i_1}, \mathbf{v}_{i_2}, \mathbf{v}_{i_3}, \mathbf{v}_{i_4})$ is defined by its four corner vertices. The connectivity information between tetrahedra, such as shared faces or edges, is often maintained through adjacency data structures, which enable efficient neighborhood queries, collision detection, and geometric processing. You could find more detailed introduction from [6].

Marching Tetrahedra A pivotal algorithm for creating tetrahedral meshes from scalar fields is the **Marching Tetrahedra** algorithm [3], a variation of the well-known Marching Cubes algorithm. The marching tetrahedra algorithm operates on a grid of tetrahedra, evaluating a scalar field at the vertices of each tetrahedron to determine where the isosurface intersects the tetrahedron's edges. For each tetrahedron, the algorithm classifies the vertices as either inside or outside the surface based on the scalar field values. It then generates triangular faces that approximate the isosurface within the tetrahedron.

2. Bridging 3D Data and Tetrahedral Meshes

Given the fundamental concepts of tetrahedral parameterization, representation, and isosurface extraction through the Marching Tetrahedra algorithm, we now turn our attention to recent advancements in transforming various 3D data forms into tetrahedral meshes and the optimization processes surrounding them.

2.1. From Surface to Tetrahedral Mesh

One prominent research direction involves the transformation of surface meshes into tetrahedral meshes since tetrahedral meshes offer a volumetric representation of 3D objects that allows for efficient physical simulations, including those that involve large deformations and complex geometrical shapes.

One of the key challenges in this domain is the preservation of surface features while ensuring that the interior mesh elements meet strict quality criteria. For example, the method proposed in [17] focuses on generating high-quality tetrahedral elements by optimizing both the interior and boundary regions of the mesh. This technique uses a signed distance function to define the object's geometry and efficiently tiles the space with tetrahedra.

Similarly, Isosurface Stuffing [13] and its improved version [4] are designed to address the issue of dihedral angle quality in tetrahedral meshes. These methods ensure that the generated tetrahedral elements are well-shaped, with tight bounds on dihedral angles, making them suitable for finite element analysis and mechanical simulations.

In the broader scope of tetrahedral mesh generation, tools like TetGen [9] and techniques introduced in Tetrahedral Meshing in the Wild [11] extend the applicability of tetrahedral meshing to more complex and unstructured real-world geometries. TetGen [9], based on Delaunay triangulation [2], is widely used for its ability to generate high-quality tetrahedra even for intricate geometries with boundary constraints. On the other hand, Tetrahedral Meshing in the Wild [11] provides a fully automated solution that works on arbitrary surface triangulations without user intervention, allowing for seamless integration into large-scale applications where geometry may be highly irregular or complex.

2.2. From Raw Points and Images to Tetrahedral Mesh

Similarly, converting point clouds or even image-based data directly into tetrahedral meshes has opened new possibilities for data-driven and learning-based approaches to 3D reconstruction. DefTet [6] provides promising solutions by utilizing neural networks to deform grid-based and tetrahedral representations to align with the input data involving 2d images and 3d points, which demonstrate significant improvements in capturing object boundaries and producing high-quality 3D meshes from 2D and 3D data.

DefGrid [8], which lays the foundation for DefTet [6], focuses on 2D image representation and introduces the idea of deforming a uniform triangular grid to better capture geometric structures in images. By learning to deform the grid based on image content, DefGrid ensures that the edges of the triangles align with important features such as object boundaries.

Building on the success of DefGrid [8], DefTet [6] extends this idea into 3D, where tetrahedral meshes are used to represent volumetric data. DefTet [6] generates a high-quality tetrahedral mesh by initializing a uniform mesh and then deforming the vertices based on input from point clouds or images. This deformation allows the tetrahedral mesh to accurately represent complex 3D shapes, enabling it to be used for both geometric modeling and simulations. One of the most significant advantages of DefTet [6] is its differentiable nature, allowing it to be optimized within a neural network framework. This makes DefTet [6] particularly suitable for converting noisy point clouds into structured meshes or generating meshes from single images.

2.3. Tetrahedral Mesh for High-Resolution 3D Shape

The optimization of tetrahedral meshes for specific applications—such as generating high-quality isosurface representations—has seen significant development as well [21]. Specifically, DMTet [21], a deep 3D conditional generative model which uses simple user guides such as coarse voxels, could synthesize high-resolution 3D shapes. Beyond DefTet [6], which could only generate corresponding tetrahedral mesh from the input point cloud, DMTet [21]

steps further and generates the surface by Marching Tetrahedra.

The fundamental idea of DMTet [21] is to directly optimize for the reconstructed surface instead of regressing the signed distance values, which enables it to synthesize finer geometric details with fewer artifacts. To represent and generate a better surface, it marries the merits of implicit and explicit 3D representations by leveraging a novel hybrid 3D representation, which is composed of a deformable tetrahedral grid that encodes a discretized signed distance function and a differentiable marching tetrahedra layer that converts the implicit signed distance representation to the explicit surface mesh representation. Therefore, DMTet [21] allows joint optimization of the surface geometry and topology as well as generation of the hierarchy of subdivisions using reconstruction and adversarial losses defined explicitly on the surface mesh.

However, despite the advantages of DMTet [21] in optimizing the surface directly and balancing implicit and explicit representations, it still faces several limitations in terms of adaptability and computational efficiency. One key drawback of DMTet [21] is its reliance on a predefined tetrahedral grid, which can constrain the reconstruction process, especially in regions requiring highly adaptive resolution. The rigid subdivision of space into fixed tetrahedra restricts its ability to fully capture intricate details in the input geometry. As a result, DMTet [21] struggles in areas with complex topology or sharp features, often resulting in oversmoothing or requiring significant post-processing to correct artifacts. Additionally, while DMTet performs well for moderate levels of detail, its computational cost scales rapidly with input resolution, making it less efficient when dealing with high-fidelity reconstructions.

To address these limitations, Flexicubes [22] introduces a more flexible and adaptive representation for reconstructing surfaces from coarse 3D inputs. Unlike DMTet [21], which operates within a fixed tetrahedral grid, Flexicubes [22] leverages deformable cubic structures that can dynamically adjust to the input geometry. This allows the algorithm to allocate computational resources more efficiently, focusing on regions with fine geometric details while maintaining coarser representations in simpler areas. The adaptability of Flexicubes [22] not only improves the quality of the reconstructed surfaces—especially around edges and complex topological features—but also reduces the overall computational burden by employing a more efficient and localized subdivision of space. As a result, Flexicubes [22] can generate high-quality surfaces with fewer artifacts and less need for post-processing, providing a more scalable solution for high-resolution surface reconstruction.

Flexicubes [22] also improves upon DMTet [21] 's optimization process by allowing for finer control over the surface geometry and topology. Its deformable cubic grid structure enables better handling of sharp edges and intricate details that DMTet [21] often smooths out due to its fixed tetrahedral grid. This flexibility allows Flexicubes [22] to achieve higher accuracy in reconstructing complex shapes, and its ability to deform the cubic structures ensures that the resulting surfaces align more closely with the input data. Additionally, Flexicubes [22] reduces the need for heavy post-processing by producing more accurate surface representations during the optimization process itself. This makes Flexicubes [22] an essential advancement over DMTet [21], especially for applications that require both precision and computational efficiency, such as high-fidelity 3D modeling and simulation.

3. Impact and Future

With the high quality 3D mesh provided by the aforementioned works, many novel systems and applications are proposed. Meanwhile, there are also some works that try to improve the results or overcome the problems in previous methods. In this section, we will particularly focus on the extended applications and potential counterwork/improvement of DMTet [21] and Flexicubes [22].

3.1. Extended Applications

Textured meshes and Rendering Other than the generic geometry and topology information provided by tetrahedral mesh, a natural extension is whether we can add rendering related properties to these 3D shapes, such as textures and lightings. In Nvdiffrec [18], this is done by leveraging coordinate-based networks in differentiable rendering to compactly represent volumetric texturing, alongside with DMTet [21] as the mesh extractor to enable gradient-based optimization directly on the surface mesh. The supervision is posed in a 2D fashion by using the rendered image. Similar idea is also used in Get3D [7], where the the generation process is split into a geometry branch and a texture branch, and still, DMTet [21] is utilized in the geometry branch to extract a 3D surface mesh from the SDF and query the texture field from the texture branch at surface points. Nvdiffrast [14] is used for rendering and the supervision is also posed in 2D images. Nvdiffrecmc [10] further improve Nvdiffrec's line of work by introducing a differentiable Monte Carlo renderer for direct illumination and differentiable denoiser to reduce variance, while the core geometry part again utilizes DMTet [21].

Text-to-3D Generation Another important application of DMTet [21] and Flexicbues [22] lies in the Text-to-3D generation field. In these works, they are primarily used as a basic geometry representation/module in all the generation pipeline due to their differentiable feature, making them easy to be integrated in an end-to-end manner. In

Fantasia3D [1], DMTet [21] is used as a 3D geometry representation for disentangled learning of geometry and appearance models. Sweetdreamer [15] further improve Fantasia3D by generating viewpoint-conditioned canonical coordinates maps, enabling the alignment of the 2D geometric priors in diffusion models with 3D shapes represented by DMTet [21]. Magic3D [16] used DMTet [21] after the course neural field representations in order to differentiably extract high-resolution textured 3D mesh.

Two-stage Refinement As is mentioned before, DMTet [21] and Flexicbues [22] can produce high-quality 3D meshes with fine geometric fidelity. Therefore, in many 3D generation pipelines, a two-stage coarse-to-fine framework is often used, and DMTet [21] and Flexicbues [22] can act as a refinement post processor in the latter stage. Magic3D [16] mentioned above is a typical example that adopt such pipeline, and concurrent work [23] reports that such kind of approach helps in preventing multi-view inconsistency. Similarly, Magic123 [19] also propose to first optimize an Instant-NGP neural radiance field to reconstruct a coarse geometry, and then initialize a DMTet [21] mesh from the radiance field output and optimize a high-resolution mesh. It is observed that the use of DMTet [21] representation enables higher quality 3D content that fits the objective and produces more compelling and higher resolution 3D consistent visuals.

3.2. Concurrent Counter-work or Improvements

QUADify In the work QUADify [5], it is reported that the mesh surface generated by DMTet [21] and Flexicubes [22] often contain many skinny, sliver triangles, making them unsuitable for further processing. Therefore, this work proposes to directly extract regular quad-dominant meshes instead of triangular meshes. In addition, with the quad-dominant regular meshes, the result would not suffer from artifacts when performing Catmull-Clark subdivision and dispacing, while for triangular meshes, the subidivision would creates irregularly sized faces due to skinny triangles visible as shading artifacts. Though displacing the subdivision surface of the triangular meshes removes some artifacts, it still fails to accurately reconstruct the reflected highlights of the reference. The results presented in this work surpass DMTet [21] and Flexicubes [22] in geometric accuracy and enabling pixel-level details extractions, which is only possible at excessive voxel resolutions for DMTet [21] and Flexicubes [22].

DMesh DMesh [26] propose an explicit shape representation that encodes every information into point cloud. The meshes are extracted by determining the existence probability of faces based on the point cloud. Therefore, the point attributes are optimized to reconstruct the target 3D mesh. In this work, it is shown that for closed meshes, in general, approaches like Flexicubes [22], capture fine details well. However, when it comes to open or mixed mesh models, which are more ubiquitous in real application, it usually suffers. The main reason lies in that Flexicubes [22] are hard to prevent false internal structures or self-intersecting faces. DMesh [26] avoids such problem by adding more constraint in defining the face existence as a post-processing step, and thus it can represent open surfaces better without self-intersecting faces.

Sur^{2}f One key feature of DMTet [21] and Flexicubes [22] is the hybrid representation that combines implicit and explicit surface representations. However, Sur^{2}f [12] points out that implicit functions used in the above works are discretely defined on tetrahedral grid vertices, thus making the key benefits of optimization in continuous 3D spaces from implicit field representations only be enjoyed partially. In addition, such kind of setting does not support rendering from its implicit functions and thus cannot directly receive supervisions from the image observations. Instead, this work proposes to improve the hybrid representation to make a better use of the explicit and implicit surface representations. It learns two parallel streams of an implicit SDF and an explicit, so-called "surrogate" surface mesh, and the implicit part is continuously defined and can be directly rendered to receive supervision from images. Sur^{2}f is tested by directly replacing DMTet [21] and Flexicubes [22] in Nvdiffrec [18], and the results shows that it promotes better quality and efficiency and also enables many downstream applications with an existing 3D content creation tool Blender.

4. Conclusion

In conclusion, this survey targets on an important topic in 3D reconstruction: Mesh and Tetrahedron and provides a roadmap of it. Specifically, we first introduce the four basic concepts in tetrahedral meshing and corresponding mathematical description. Then we discuss in depth the recent advances on bridging the 3D data and tetrahedral meshes, and categorize the related works into three main classes. We show the evolution of a series works and in the end put most emphasis on the state-of-the-art works DMTet [21] and Flexicbues [22]. Later, focusing on the above two works, we show several systems and applications enabled by them including rendering, simulation, text-to-3D generation and geometry refinement. Last but not least, we also point out the drawbacks of these approaches and some concurrent works that try to propose new methodology to improve and overcome the problems in existing works.

References

[1] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22246–22256, October 2023. 4

[2] B. Delaunay. Sur la sphère vide. *Bulletin de l'Academie des Sciences de l'URSS. Classe des sciences mathematiques et na*, 1934(6):793–800, 1934. 2

[3] Akio Doi and Akio Koide. An efficient method of triangulating equi-valued surfaces by using tetrahedral cells. *IEICE TRANSACTIONS on Information and Systems*, 74(1):214–224, 1991. 2

[4] Crawford Doran, Athena Chang, and Robert Bridson. Isosurface stuffing improved: acute lattices and feature matching. In *ACM SIGGRAPH 2013 Talks*, pages 1–1. 2013. 1, 2

[5] M. Fruhauf, H. Riemenschneider, M. Gross, and C. Schroers. Quadify: Extracting meshes with pixel-level details and materials from images. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4661–4670, Los Alamitos, CA, USA, jun 2024. IEEE Computer Society. 4

[6] Jun Gao, Wenzheng Chen, Tommy Xiang, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Learning deformable tetrahedral meshes for 3d reconstruction. *Advances in neural information processing systems*, 33:9936–9947, 2020. 2

[7] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. In *Advances In Neural Information Processing Systems*, 2022. 3

[8] Jun Gao, Zian Wang, Jinchen Xuan, and Sanja Fidler. Beyond fixed grid: Learning geometric image representation with a deformable grid. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 108–125. Springer, 2020. 2

[9] Si Hang. Tetgen, a delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Softw*, 41(2):11, 2015. 1, 2

[10] Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. Shape, light, and material decomposition from images using monte carlo rendering and denoising. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 22856–22869. Curran Associates, Inc., 2022. 3

[11] Yixin Hu, Qingnan Zhou, Xifeng Gao, Alec Jacobson, Denis Zorin, and Daniele Panozzo. Tetrahedral meshing in the wild. *ACM Trans. Graph.*, 37(4):60, 2018. 1, 2

[12] Zhangjin Huang, Zhihao Liang, and Kui Jia. Sur2f: A hybrid representation for high-quality and efficient surface reconstruction from multi-view images. *arXiv preprint arXiv:2401.03704*, 2024. 4

[13] François Labelle and Jonathan Richard Shewchuk. Isosurface stuffing: fast tetrahedral meshes with good dihedral angles. In *ACM SIGGRAPH 2007 papers*, pages 57–es. 2007. 2

[14] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 39(6), 2020. 3

[15] Weiyu Li, Rui Chen, Xuelin Chen, and Ping Tan. Sweetdreamer: Aligning geometric priors in 2d diffusion for consistent text-to-3d. In *The Twelfth International Conference on Learning Representations*, 2024. 4

[16] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 4

[17] Neil Molino, Robert Bridson, and Ronald Fedkiw. Tetrahedral mesh generation for deformable bodies. In *Proc. Symposium on Computer Animation*, volume 8, 2003. 2

[18] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting Triangular 3D Models, Materials, and Lighting From Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8280–8290, June 2022. 3, 4

[19] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, and Bernard Ghanem. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. 4

[20] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 1

[21] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2, 3, 4

[22] Tianchang Shen, Jacob Munkberg, Jon Hasselgren, Kangxue Yin, Zian Wang, Wenzheng Chen, Zan Gojcic, Sanja Fidler, Nicholas Sharp, and Jun Gao. Flexible isosurface extraction for gradient-based mesh optimization. *ACM Trans. Graph.*, 42(4), jul 2023. 3, 4

[23] Yichun Shi, Peng Wang, Jianglong Ye, Long Mai, Kejie Li, and Xiao Yang. MVDream: Multi-view diffusion for 3d generation. In *The Twelfth International Conference on Learning Representations*, 2024. 4

[24] Eftychios Sifakis and Jernej Barbic. Fem simulation of 3d deformable solids: a practitioner's guide to theory, discretization and model reduction. In *Acm siggraph 2012 courses*, pages 1–50. 2012. 1

[25] Randall C Smith and Peter Cheeseman. On the representation and estimation of spatial uncertainty. *The international journal of Robotics Research*, 5(4):56–68, 1986. 1

[26] Sanghyun Son, Matheus Gadelha, Yang Zhou, Zexiang Xu, Ming C. Lin, and Yi Zhou. Dmesh: A differentiable representation for general meshes, 2024. 4