

A Visualization System for Hexahedral Mesh Quality Study

Lei Si*
University of Houston

Guoning Chen†
University of Houston

ABSTRACT

In this paper, we introduce a new 3D hex mesh visual analysis system that emphasizes poor-quality areas with an aggregated glyph, highlights overlapping elements, and provides detailed boundary error inspection in three forms. By supporting multi-level analysis through multiple views, our system effectively evaluates various mesh models and compares the performance of mesh generation and optimization algorithms for hexahedral meshes.

Keywords: hex-mesh analysis, mesh quality visualization

Index Terms: Human-centered computing—Visualization—Visualization systems and tools—

1 INTRODUCTION

Hexahedral (hex-) meshes are preferred in many mechanical and (bio-)medical engineering applications due to their desired properties for numerical simulations [4, 11, 43]. It is known that the quality of hex-meshes impacts the accuracy and efficiency of various finite element simulations [11, 48].

Quantitative mesh quality metrics are crucial for mesh generation algorithms. A large number of mesh quality metrics [5, 11, 20] have been designed to measure the deviance of individual elements from their ideal shape (e.g., a regular cube for a hexahedral element). However, existing visualization techniques (see Figure 1(a)(b)) may overlook small, poor-quality elements due to viewer's attention being drawn to large patterns and areas with large elements. Nonetheless, these small and poor-quality elements could lead to the failure of simulations just like the other poor-quality elements. Boundary preservation is another important criterion impacting the accuracy of boundary condition problems, but most tools lack effective boundary error visualization. Overlapping elements, introduced by some mesh quality improvement algorithms, may also be concealed by traditional color coding or volume rendering methods.

The above limitations of the existing tools motivate our work. To address these limitations, a mesh quality visualization system should provide the following.

- G1. Effectively reveal regions with bad quality elements despite the region and element sizes.
- G2. Intuitively highlight overlapping/intersecting elements
- G3. Show the local configuration of bad elements (e.g., angles at the corners of bad elements) to understand their causes
- G4. Display boundary difference/error with respect to the reference surface to highlight places with large differences.

To achieve the above goals, we design and develop a new mesh quality visualization system and make the following contributions.

- We propose a glyph design to highlight places with poor-quality elements independent of the element sizes. This glyph can be aggregated to provide collective information on the quality of elements within a region.
- We explicitly detect and visualize the overlapping elements that may be hidden in the mesh.

*e-mail: lsi@uh.edu

†e-mail: gchen16@uh.edu

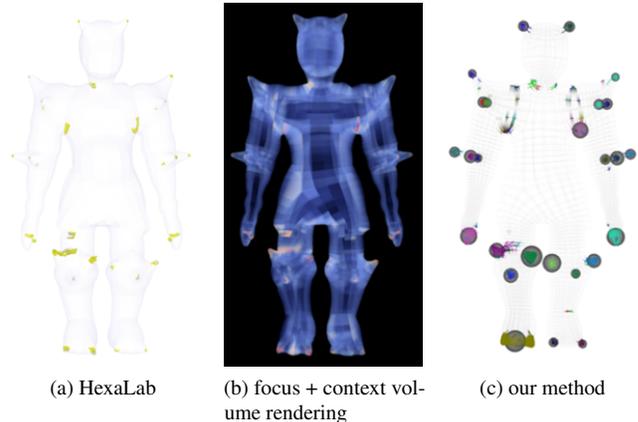


Figure 1: Visualization of the quality of a warrior hex-mesh [26] using the quality filtering by the HexaLab [5] (a), focus+context volume rendering [33] (b), and our glyph-based method (c), respectively. Our method highlights places with bad-quality elements. The larger the glyphs are, the worse the element quality is.

- We design a comprehensive visualization framework for boundary errors to provide both a summary view of the boundary error statistics and the detailed spatial distribution of error. In particular, parametric representations of 3D boundaries and their errors are used to aid the exploration of the boundary errors for hexahedral meshes.
- We incorporate the above new design of visualization into an interactive, multi-linked-view visual analytics system to support an effective and efficient study of the mesh quality.

2 RELATED WORK

In this section, we briefly review the literature on hex mesh generation and optimization techniques and the relevant tools that provide the visualization of the meshes.

Hex mesh generation and optimization. Hexahedral mesh generation has been a research focus due to its importance in various applications. Earlier techniques [21, 29–31, 35, 37, 39–41, 47] generated unstructured hex meshes, while recent geometry processing approaches, such as polycube mapping [9, 13–15, 22, 23, 28], produce semi-structured meshes with limitations in handling complex geometry and topology. 3D parameterization methods [16, 19, 25, 34], utilizing orthogonal and non-orthogonal frame fields, have gained popularity but lack guarantees for generating valid hex-meshes. High-quality all-hex mesh generation remains challenging [4], and post-processing is often required to improve mesh quality [1, 7, 8, 10, 12, 18, 27, 36, 44, 46]. To support mesh optimization, displaying the obtained meshes' quality is crucial for engineers and researchers to identify areas for improvement.

Meshing tools with visualization capability. Due to its importance for various tasks, numerous mesh generation tools have been developed, including generic tools like CUBIT [3] and other specific tools (or libraries) as listed in [38]. Most of them do not provide a visualization front end to allow the users to check the quality of the generated meshes. A separate tool is usually needed, such as MeshLab [6], libigl [17], and Paraview [2], to aid the visual analysis of the

generated meshes. Many finite element simulation softwares [32] also offer some mesh generation and processing functionality with limited visualization capability (e.g., wire-frame representation of the meshes, color coding quality values, and color coding different patches). Recently, two works for the visualization and analysis of hexahedral mesh quality have been reported, i.e., the HexaLab [5] and the hex mesh structure evaluation and visualization [45]. Both works focus on effective visual representation and high-quality rendering to aid the inspection of 3D meshes produced by different methods. However, both tools do not provide split screens to compare two meshes of the same model produced by different techniques. Also, boundary error analysis and overlapping element analysis are not offered by either tool. Most recently, a focus+context volume rendering technique [33] has been introduced for the inspection of hexahedral meshes, which allows the user to focus on regions with bad-quality elements without occlusion or cluttering. However, this technique may not reveal bad elements with small sizes (Figure 1 (b)) and does not address the boundary error analysis. In this work, we present a comprehensive visual analysis system that can support the quality and boundary error analysis of hex-meshes.

3 THE DESIGN OF HQVIEW

HQView, a multi-view visual analysis system (Section 3.3), offers various representations for 3D hexahedral mesh quality information. It computes a quality metric [20] for each element corner as a basis for visualization. Besides traditional color encoding and histograms, HQView provides additional functionality to achieve goals G1-G4 listed in the introduction.

3.1 A Glyph Design for Bad Quality Element (G1)

Element quality is a crucial mesh quality indicator. Traditional color maps work well for 2D meshes but face occlusion issues in 3D hex meshes. Alternative methods like filtering in HexaLab [5], focus+context volume rendering [23], or magic lens techniques have limitations (see Fig. 1). To achieve size-independent visual encoding, we introduce a glyph design mapping quality values to spheres, with radius encoding the quality value (i.e., the scaled Jacobian metric).

Instead of using a quality value to describe the regularity of a 3D cell, we calculate the quality of the corners shared by the one-ring neighborhood of a vertex. The smallest quality value is selected to represent the quality of the vertex. The radius of the sphere glyph at each vertex is then determined by the quality value. We define an inverse relationship between the quality of the vertex and the radius of its sphere glyph such that the vertices with bad-quality elements have prominent sphere glyphs that can be easily identified. The inset to the right illustrates this mapping where J_i represents the scaled Jacobian value at corner i . To support the metric mapping to the radius of the sphere, we map it to the range of $[0, 2]$ as follows.

$$c = 1 - J_m \quad (1)$$

where J_m is the smallest scaled Jacobian of a vertex. c has a reverse relation with J_m . That is, if $J_m = 1$ (the best scaled Jacobian), $c = 0$, while $c = 2$ when $J_m = -1$ (the worst scaled Jacobian). The radius of the sphere centered at the vertex is then controlled by c . In particular, $r = c * r_{max}$, where r_{max} is a user-controllable parameter to enhance the visibility of the sphere glyphs. Using this strategy, the vertices surrounded by bad-quality elements can be highlighted, regardless of their element sizes.

The above glyph-based visualization can lead to clutter and overlap in areas with many small and bad-quality elements as they lead to large spheres packed in a small region (e.g., Fig. 6 (b) and Fig. 2 (a)). In addition, rendering sphere glyphs for all vertices of a large mesh is time-consuming. To address these issues, we introduce a clustering-based glyph aggregating strategy (Fig. 2 (b)).

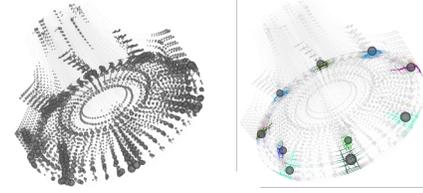
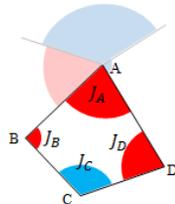


Figure 2: The aggregated glyphs (b) reduce the clutter of non-aggregated glyphs (a).

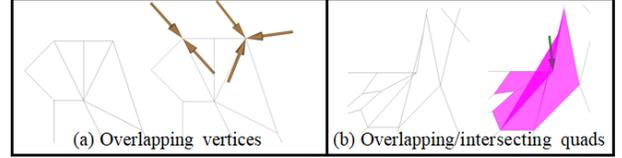


Figure 3: Two different overlapping situations may arise (2D illustration): (a) overlapping vertices caused by degenerated edges, (b) overlapping quads caused by one vertex falling inside another quad. Existing visualization (left image of each pair) cannot effectively highlight the overlapping in either case. In contrast, we use arrow(s) to intuitively direct the viewers toward places with overlap (right image of each pair).

Our clustering strategy groups nearby vertices based on the spatial proximity of their glyphs. Specifically, if two glyphs overlap (i.e., the sum of their radii exceeds the distance between their centers), the corresponding vertices belong to the same cluster. The aggregated glyph is positioned at the vertex in the cluster closest to the concentration of cluster vertices. We also employ different colors for edges within distinct clusters. Figure 4 (view A) displays this aggregated visualization, which effectively directs the user’s attention to regions with poor-quality elements for further analysis (Section 3.3).

3.2 Overlapping Element Highlighting (G2)

Overlapping elements can be hard to identify, as they may have good quality or be small, making them difficult to distinguish from other bad-quality elements. Two types of overlapping elements can arise (1) overlapping vertices and (2) overlapping cells.

Overlapping vertices occur in regions with near-degenerate meshes that don’t exhibit inverted elements, like those with zero-length edges. This degenerate configuration is hard to detect and can cause issues during finite element calculations. *Overlapping cells* are more common and occur when a vertex of a cell moves inside another cell due to mesh operations. These overlapping elements need to be identified and optimized, as they alter calculations in the coordinate system.

Current visualization tools struggle to effectively highlight mesh regions with either overlapping configurations. Our visualization system uses an arrow placement strategy to highlight overlapping vertices (Fig. 3(a)) and transparent shaded rendering for overlapping cells (Fig. 3(b)). In a scenario where a location contains two overlapping vertices, a pair of arrows will be generated, each pointing towards a vertex. If n arrows point to the same vertex, they will be evenly distributed around the vertices at intervals of $360/n$ degrees.

3.3 Multi-level Element Quality Analysis (G3)

Our visual analysis system supports mesh quality inspection at multiple levels. It consists of six views (Fig. 4): A. main view, B. overall vertex quality chart, C. sub-region view, D. local element view, E. selected point quality chart, and F. reference view.

A. Main View visualizes mesh quality using glyphs and highlights overlapping elements. It also displays a quality histogram.

B. Overall Vertex Quality Chart sorts vertices by their quality and supports selection for further analysis.

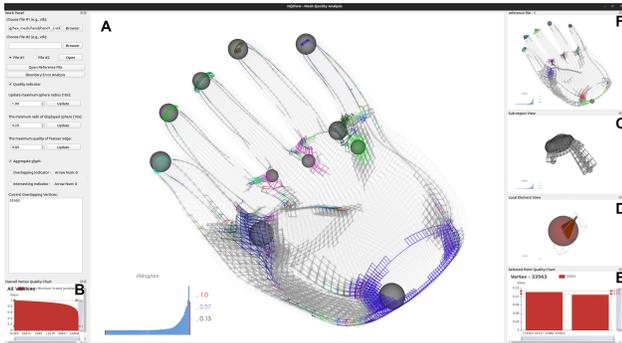


Figure 4: The multi-view interface of our system for the level-of-detail inspection of mesh element quality.

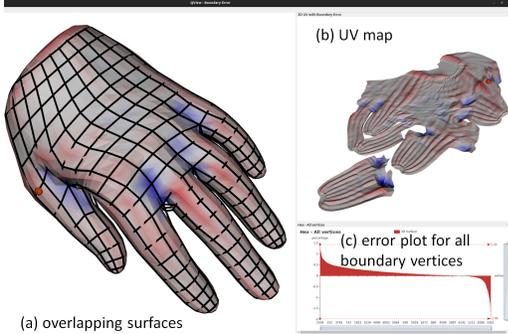


Figure 5: A multi-view interface for the boundary error visualization of a hand hex-meshing.

C. Sub-region View shows individual elements within clusters for a more detailed study.

D. Local Element View displays the one-ring neighborhood of a selected vertex, offering a thorough quality analysis.

E. Selected Point Quality Chart provides quality values of all corners shared by the selected vertex, helping to identify local configuration issues.

F. Reference View enables comparison of the main mesh with a ground truth or other algorithms' results.

3.4 Multi-Dimension Views for Boundary Difference Visualization(G4)

Analyzing the boundary error of 3D meshes is difficult due to occlusion, requiring users to choose different views around the mesh. To efficiently analyze global boundary error information without changing viewpoints, Our system offers a comprehensive boundary error analysis supported by a multi-view visualization for 3D meshes.

The boundary error is calculated for each surface point based on the closest point on the original mesh. Positive boundary error values (red) indicate modified mesh points lying outside the original mesh, while negative values (blue) indicate points inside the original mesh(Figure 5 (a)). To provide a holistic visualization of the boundary error without requiring the user to rotate the model, we use a UV map of the surface that is extracted from original mesh using the OptCuts [24]. The UV map unfolds the curve surface to a planar representation.

In the example shown in Figure 5 (b), positive errors correspond to surface ridges, while negative errors are at concave areas.

A collated percentage graph is provided for all modified mesh surface vertices, offering an overview of the surface areas inside or outside the original mesh. Users can select individual vertices from the graph for further analysis(Figure 5 (c)).

4 EVALUATION

We integrate the above comprehensive visualization techniques into one unified system with two separate windows, i.e., the Mesh Quality Analysis Window, and the Boundary Error Analysis Window. We

applied our system to analyze the quality of the hex-meshes included in the database of HexaLab [5] and from the hex-mesh structure simplification work [12].

The process of using our system to analyze a hex-mesh is as follows. After loading a hex-mesh, the Mesh Quality Analysis window displays the wire-frame of the mesh, the distribution of individual element quality, and an interactive bar chart ranking vertex quality (Figure 6 (a)). To locate poor-quality elements, glyph representation is activated, using aggregated glyphs to prevent clutter (Figure 6(b)). By focusing on the regions with poor quality, the user can select a large aggregated glyph corresponding to a small cluster and inspect the area in the sub-region view (Figure 6(c), top). A bad-quality vertex can be chosen for detailed inspection, revealing its Jacobian configurations and connectivity configuration (Figure 6(c), middle and bottom). This multi-level, multi-perspective mesh quality analysis process efficiently identifies and analyzes mesh quality issues.

Element quality analysis. To evaluate the effectiveness of our proposed visual encoding, we compare our visualizations of a few meshes with those shown by HexaLab [5] and the focus+context volume rendering [33], respectively. Since the other two approaches do not explicitly support the boundary error visualization, our comparison focuses on the element quality analysis and the effective revelation of low-quality elements with different sizes. Figure 1 compares the quality visualization of a warrior hex-mesh using the three approaches. Both HexaLab and the focus+context volume rendering cannot effectively reveal the bad elements at the tips of those protruded features. In contrast, our aggregated glyphs not only highlight those places but also convey how severe the element quality is in those regions via the sizes of the glyphs.

Figure 7 compares the quality visualization of a kitten hex-mesh using the three methods, respectively. From this comparison, we see that among the three approaches, the focus+context volume rendering can provide the smoothest visual representation of the mesh quality. However, if the two areas have a similar quality, the volume rendering will not effectively distinguish their difference as humans cannot accurately tell the difference between similar colors if they are not next to each other (e.g., the back and the left ear of the kitten). Also, one may think the quality of the elements in the left ear of the kitten is worse than those in the right ear because they look more prominent. In contrast, HexaLab can provide a more accurate reading on the element quality by filtering. That is, the remaining elements after filtering all have quality lower than a user-specified threshold. However, the difference among these remaining elements is hard to discern (e.g., it is hard to decide which elements are worse than the other in Figure 7(a)). Also, depending on the threshold, other less optimal areas may not be captured (e.g., the top of the kitten and the middle of the tail of the kitten). As a comparison, our aggregated glyph visualization retains most of the areas with bad-quality elements and allows a more effective differentiation of the element quality. For example, the quality of the elements at the back of the neck of the kitten (represented by a big green glyph) is worse than those at the back of the kitty that are prominent in both HexaLab and volume-rendering visualization. Similarly, there is a ring of bad-quality elements at the top of the head of the kitten that is not emphasized by the other two methods due to their small sizes.

Overlapping cells analysis. Fig. 8 shows an example of overlapping and intersection elements in a fandisk mesh. The overlapping vertices Fig. 8(a) are difficult to notice without the additional arrows. A similar observation can be made for the intersection elements Fig. 8(b). In addition, the cells involved in the intersection are highlighted, with each element colored distinctly, and an arrow is placed to indicate each intersection vertex.

Boundary error analysis. To analyze the boundary error of a mesh, the system loads the modified and original meshes into the Boundary Error Analysis Window, as shown in Figure 6 (d). In the window, the main view displays overlapping surfaces, but occlusion hides

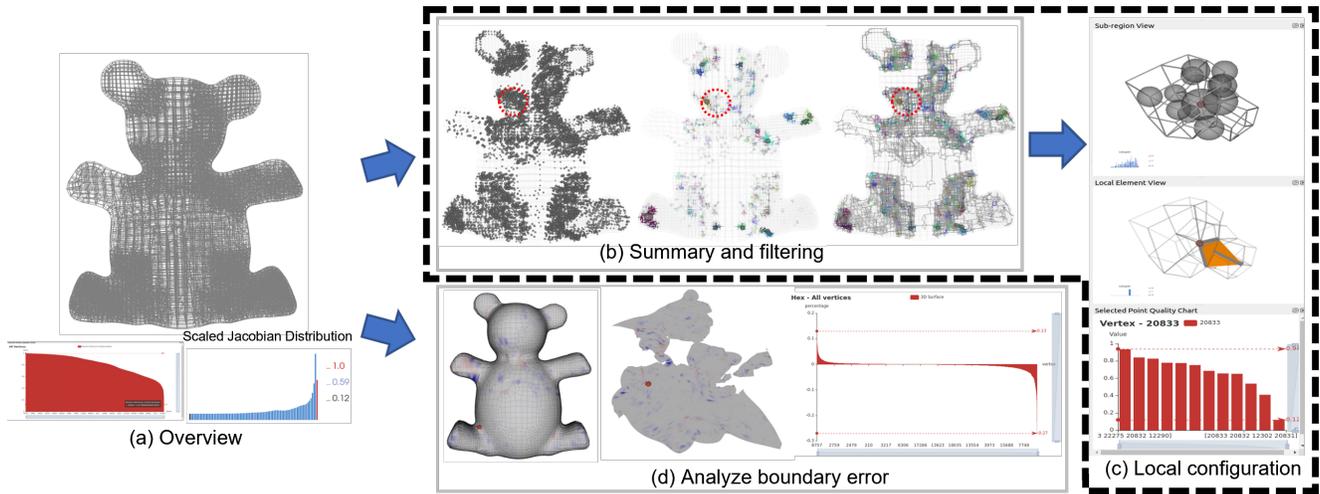


Figure 6: A use case of our system for the analysis of a hex-mesh quality.

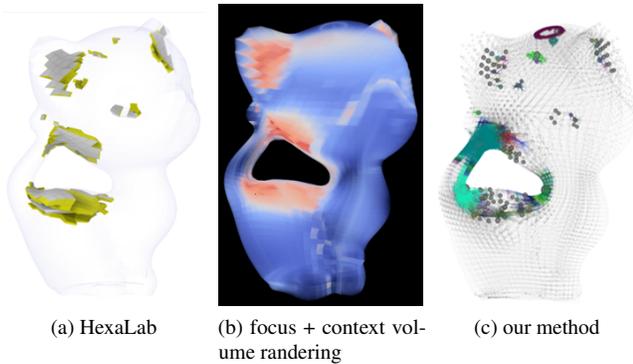


Figure 7: Comparison of the quality visualization using HexaLab (a), focus+context volume rendering (b), and our methods (c) for a teddy bear hex-mesh (top row) and a kitten hex-mesh (bottom), respectively.

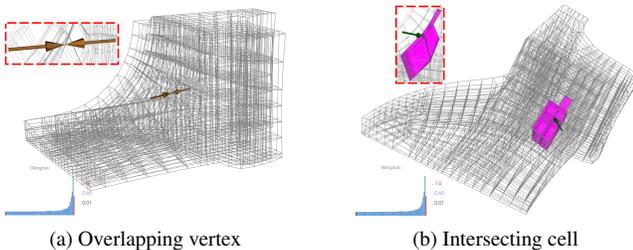


Figure 8: Overlapping vertices and intersecting cells in a 3D view are difficult to be distinguished, as the lines obstruct their visibility.

some error distributions. The UV map provides a comprehensive overview without requiring the user to select different viewpoints, though it lacks 3D context like sharp features and corners. The boundary error distribution doesn’t show a consistent pattern near specific feature types. For instance, concave areas corresponding to limb-torso junctions can exhibit both positive and negative errors. Larger errors typically occur where the surface has a large curvature, as these areas are challenging to preserve. The sorted error value bar chart enables quick identification of vertices with large boundary errors and their nature.

User feedback. We designed an online survey to gather unofficial user feedback on our system. Among the 38 responses received, 11 participants identified themselves as mesh experts. The survey

consists of 13 questions, 9 of which are designed for mesh element quality analysis, serving three different objectives. The first category of questions requests participants to rank regions by applying their judgment of mesh quality across three different methods. The second category of questions prompts users to identify all problematic areas. The third group of tasks requires users to select the most effective method for highlighting poor-quality regions. Responses to the questions suggest that our methodology is effective in helping users identify regions of poor quality, particularly when dealing with small mesh sizes. Detailed feedback of the survey can be found in the supplemental document.

5 CONCLUSION

We present a new visual analysis system for the study of the quality of 3D hexahedral meshes. Our system offers simple but effective visual encoding techniques and a multi-view capability to help reveal small elements with low-quality and overlapping configurations and support the inspection of boundary errors. The evaluation shows that our system outperforms the existing tools in the tasks of locating small elements with low quality, finding overlapping elements in the mesh, and studying boundary error configurations.

To improve our system, we will address the following limitations of the system. First, our aggregation glyph construction requires performing collision detection among nearby spheres. Our current implementation using traversal has a complexity of $O(n^2)$. To accelerate, we will adopt a pre-computed tree-like structure, such as a union-find data structure [42]. Second, our system does not suggest the ideal configuration for a comparative study of the bad-quality elements. Third, our arrow placement strategy for highlighting overlapping elements may still produce cluttered arrows in small regions with many overlapping elements. Nonetheless, the cluttered arrows may help draw the attention of the viewers. To reduce clutter, a view-dependent placement of arrows may be explored. Fourth, some models may not be successfully unfolded for boundary error visualization due to the limitation of the used UV unfolding algorithm. Finally, the current user evaluation is rather informal and incomplete. Future work will focus on addressing these limitations while considering other element quality measures [11] and incorporating the visualization of the simulation results run on the respective meshes to provide new insights into the mesh quality and its impact on the downstream tasks.

ACKNOWLEDGMENTS

We wish to thank the anonymous reviews for their constructive feedback to help improve this work.

REFERENCES

- [1] M. N. Akram, L. Si, and G. Chen. An embedded polygon strategy for quality improvement of 2d quadrilateral meshes with boundaries. In *VISIGRAPP (1: GRAPP)*, pp. 177–184, 2021.
- [2] U. Ayachit. *The paraview guide: a parallel visualization application*. Kitware, Inc., 2015.
- [3] T. D. Blacker, W. J. Bohnhoff, and T. L. Edwards. Cubit mesh generation environment. volume 1: Users manual. Technical report, Sandia National Labs., Albuquerque, NM (United States), 1994.
- [4] D. Bommers, B. Lévy, N. Pietroni, E. Puppo, C. Silva, M. Tarini, and D. Zorin. Quad-mesh generation and processing: A survey. *32(6):51–76*, 2013.
- [5] M. Bracci, M. Tarini, N. Pietroni, M. Livesu, and P. Cignoni. Hexalab.net: An online viewer for hexahedral meshes. *Computer-Aided Design*, 110:24–36, 2019.
- [6] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. Meshlab: an open-source mesh processing tool. In *Eurographics Italian chapter conference*, vol. 2008, pp. 129–136. Salerno, Italy, 2008.
- [7] J. Daniels, C. T. Silva, and E. Cohen. Localized quadrilateral coarsening. *28(5):1437–1444*, 2009.
- [8] J. Daniels, C. T. Silva, J. Shepherd, and E. Cohen. Quadrilateral mesh simplification. *ACM Trans. Graph.*, 27(5):1–9, 2008.
- [9] X. Fang, W. Xu, H. Bao, and J. Huang. All-hex meshing using closed-form induced polycube. *ACM Trans. Graph.*, 35(4):1–9, 2016.
- [10] X. Gao, Z. Deng, and G. Chen. Hexahedral mesh re-parameterization from aligned base-complex. *ACM Trans. Graph.*, 34(4):1–10, 2015.
- [11] X. Gao, J. Huang, K. Xu, Z. Pan, Z. Deng, and G. Chen. Evaluating hex-mesh quality metrics via correlation analysis. *Computer Graphics Forum*, 36(5):105–116, 2017.
- [12] X. Gao, D. Panozzo, W. Wang, Z. Deng, and G. Chen. Robust structure simplification for hex re-meshing. *ACM Trans. Graph.*, 36(6):1–13, 2017.
- [13] J. Gregson, A. Sheffer, and E. Zhang. All-hex mesh generation via volumetric polycube deformation. *30(5):1407–1416*, 2011.
- [14] H.-X. Guo, X. Liu, D.-M. Yan, and Y. Liu. Cut-enhanced polycubemaps for feature-aware all-hex meshing. *ACM Trans. Graph.*, 39(4):106–1, 2020.
- [15] J. Huang, T. Jiang, Z. Shi, Y. Tong, H. Bao, and M. Desbrun. ℓ_1 -based construction of polycube maps from complex shapes. *ACM Trans. Graph.*, 33(3):1–11, 2014.
- [16] J. Huang, Y. Tong, H. Wei, and H. Bao. Boundary aligned smooth 3d cross-frame field. *ACM Trans. Graph.*, 30(6):1–8, 2011.
- [17] A. Jacobson and D. Panozzo. libigl: prototyping geometry processing research in c++, 2017.
- [18] Z. Ji, L. Liu, and G. Wang. A global laplacian smoothing approach with feature preservation. In *Ninth International Conference on Computer Aided Design and Computer Graphics (CAD-CG'05)*, pp. 6–pp. IEEE, 2005.
- [19] T. Jiang, J. Huang, Y. Wang, Y. Tong, and H. Bao. Frame field singularity correction for automatic hexahedralization. *IEEE Transactions on Visualization and Computer Graphics*, 20(8):1189–1199, 2013.
- [20] P. M. Knupp, C. Ernst, D. C. Thompson, C. Stimpson, and P. P. Pebay. The verdict geometric quality library. Technical report, Sandia National Laboratories, 2006.
- [21] B. Lévy and Y. Liu. \mathcal{L}_p centroidal voronoi tessellation and its applications. *ACM Trans. Graph.*, 29(4):1–11, 2010.
- [22] B. Li, X. Li, K. Wang, and H. Qin. Surface mesh to volumetric spline conversion with generalized polycubes. *IEEE Transactions on Visualization and Computer Graphics*, 19(9):1539–1551, 2012.
- [23] L. Li, P. Zhang, D. Smirnov, S. M. Abulnaga, and J. Solomon. Interactive all-hex meshing via cuboid decomposition. *ACM Trans. Graph.*, 40(6):1–17, 2021.
- [24] M. Li, D. M. Kaufman, V. G. Kim, J. Solomon, and A. Sheffer. Optcuts: Joint optimization of surface cuts and parameterization. *ACM Trans. Graph.*, 37(6):1–13, 2018.
- [25] Y. Li, Y. Liu, W. Xu, W. Wang, and B. Guo. All-hex meshing using singularity-restricted field. *ACM Trans. Graph.*, 31(6):1–11, 2012.
- [26] M. Livesu, A. Muntoni, E. Puppo, and R. Scateni. Skeleton-driven adaptive hexahedral meshing of tubular shapes. In *Computer Graphics Forum*, vol. 35, pp. 237–246. Wiley Online Library, 2016.
- [27] M. Livesu, A. Sheffer, N. Vining, and M. Tarini. Practical hex-mesh optimization via edge-cone rectification. *ACM Trans. Graph.*, 34(4):1–11, 2015.
- [28] M. Livesu, N. Vining, A. Sheffer, J. Gregson, and R. Scateni. Polycut: Monotone graph-cuts for polycube base-complex construction. *ACM Trans. Graph.*, 32(6):1–12, 2013.
- [29] J. H.-C. Lu, I. Song, W. R. Quadros, and K. Shimada. Volumetric decomposition via medial object and pen-based user interface for hexahedral mesh generation. In *Proceedings of the 20th International Meshing Roundtable*, pp. 179–196. Springer, 2012.
- [30] Y. Lu, R. Gadh, and T. J. Tautges. Feature based hex meshing methodology: feature recognition and volume decomposition. *Computer-Aided Design*, 33(3):221–232, 2001.
- [31] L. Maréchal. Advances in octree-based all-hexahedral mesh generation: handling sharp features. In *Proceedings of the 18th International Meshing Roundtable*, pp. 65–84. Springer, 2009.
- [32] D. Marinkovic and M. Zehn. Survey of finite element method-based real-time simulations. *Applied Sciences*, 9(14):2775, 2019.
- [33] C. Neuhauser, J. Wang, and R. Westermann. Interactive focus+ context rendering for hexahedral mesh inspection. *IEEE Transactions on Visualization and Computer Graphics*, 27(8):3505–3518, 2021.
- [34] M. Nieser, U. Reitebuch, and K. Polthier. Cubecover—parameterization of 3d volumes. *30(5):1397–1406*, 2011.
- [35] S. J. Owen and S. Saigal. H-morph: an indirect approach to advancing front hex meshing. *International Journal for Numerical Methods in Engineering*, 49(1-2):289–312, 2000.
- [36] C.-H. Peng, E. Zhang, Y. Kobayashi, and P. Wonka. Connectivity editing for quadrilateral meshes. pp. 1–12, 2011.
- [37] J. Sarrate Ramos, E. Ruiz-Gironés, and X. Roca Navarro. Unstructured and semi-structured hexahedral mesh generation methods. *Computational Technology Reviews*, 10:35–64, 2014.
- [38] R. Schneiders. Mesh generation software. <http://www.robertschneiders.de/meshgeneration/software.html>.
- [39] J. F. Shepherd and C. R. Johnson. Hexahedral mesh generation constraints. *Engineering with Computers*, 24(3):195–213, 2008.
- [40] M. L. Staten, R. A. Kerr, S. J. Owen, T. D. Blacker, M. Stupazzini, and K. Shimada. Unconstrained plastering hexahedral mesh generation via advancing-front geometry decomposition. *International Journal for Numerical Methods in Engineering*, 81(2):135–171, 2010.
- [41] M. L. Staten, S. J. Owen, and T. D. Blacker. Unconstrained paving & plastering: A new idea for all hexahedral mesh generation. In *proceedings of the 14th International Meshing Roundtable*, pp. 399–416. Springer, 2005.
- [42] R. E. Tarjan and J. van Leeuwen. Worst-case analysis of set union algorithms. *J. ACM*, 31(2):245–281, mar 1984. doi: 10.1145/62.2160
- [43] T. J. Tautges. The generation of hexahedral meshes for assembly geometry: survey and progress. *International Journal for Numerical Methods in Engineering*, 50(12):2617–2642, 2001.
- [44] K. Xu, M. N. Akram, and G. Chen. Semi-global quad mesh structure simplification via separatrix operations. In *SIGGRAPH Asia 2020 Technical Communications*, pp. 1–4. 2020.
- [45] K. Xu and G. Chen. Hexahedral mesh structure visualization and evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):1173–1182, 2018.
- [46] K. Xu, X. Gao, and G. Chen. Hexahedral mesh quality improvement via edge-angle optimization. *Computers & Graphics*, 70:17–27, 2018.
- [47] H. Zhang, G. Zhao, and X. Ma. Adaptive generation of hexahedral element mesh using an improved grid-based method. *Computer-Aided Design*, 39(10):914–928, 2007.
- [48] Y. Zhang, C. Bajaj, and B.-S. Sohn. 3d finite element meshing from imaging data. *Computer Methods in Applied Mechanics and Engineering*, 194(48-49):5083–5106, 2005.