A Visualization System for Hexahedral Mesh Quality Study – Supplemental Document

Lei Si[*]
University of Houston

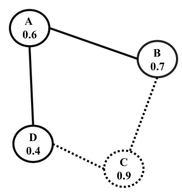
Guoning Chen[†]
University of Houston

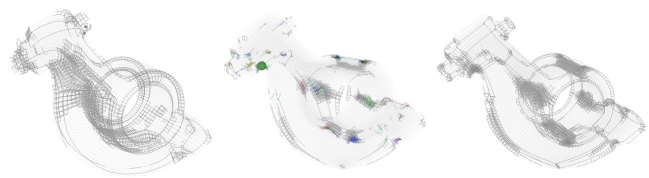
Abstract

The supplemental shows the extension of the proposed system. the proposed system can be also applied to the Quad mesh analysis.

1 Feature Edge Filtering

To visualize the distribution of poor-quality elements, we provide a *feature edge filtering* functionality along with the aggregated glyphs. In particular, we emphasize the edges that connect two vertices whose surrounding elements' quality is lower than a user-specified threshold, while de-emphasizing other edges (see the inset figure to the right for an illustration).





(a) rockerarm-m1 (b) rockerarm-m2 (c) rockerarm-m2

Figure 1: Feature edge based quality summary representation for two rockerarm hex-meshes, i.e., the mesh shown in (a) is different from the mesh in (b) and (c). Also, we set $e_{qmax} = 0.8$ for (a) and (b) whose corresponding feature edges reveal the structure difference between the two meshes. $e_{qmax} = 0.85$ for (c) which shows the same mesh of (b) with different sets of filtered feature edges.

2 Impact of the Parameters

Impact of the parameters. There are a number of important user-controllable parameters that will impact the effect of visual encoding. They are the maximum sphere radius r_{max}, the minimum displayed sphere radius r_{dmin}, and the maximum quality of feature edges e_{qmax}.

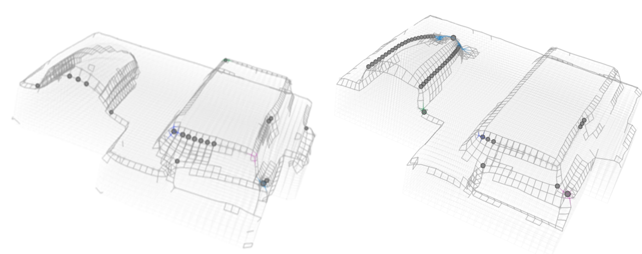
maximum sphere radius. After numerous experiments, we identified an empirical percentage of the maximum sphere radius. The percentage is the 50% of average edge length. If the radius is set too large, the glyph will occupy a large space. But if the radius is too small, glyphs may be too small to highlight regions with bad-quality elements.

minimum displayed sphere radius. In most cases, we set $r_{dmin} = 10\% \times r_{max}$, which is sufficient to filter out the glyphs that are too small (i.e., corresponding to regions with good quality elements).

maximum quality of feature edges. Based on our experiments, we found that $e_{qmax} = 0.2 + q_m$ usually yields good feature edge structure, where q_m is the middle value of all vertex quality values. Figure 1 compares two meshes for the same model with different feature edge thresholds.

2.1 Comparison of different 3D hex mesh generation techniques.

In this section, we conduct three sets of comparisons of different hex-mesh generation techniques with our system using three different models, respectively. The first experiment is shown in Figure 2. By using aggregated glyphs and feature edge filtering, we can see that the two techniques have significantly different results on the circular boundary. In Figure 2 (a), no highlighted glyphs are at the right top plane, but three glyphs are at the front of the circular shape. However, in Figure 2 (b), a large number of glyphs appear at the top of the circular shape. And there is a small cluster on the back end. Then, by comparing the feature edges of those two meshes, we can see that the structure of this model under the current threshold is almost the same. The structure can indicate that those two techniques may have a similar strategy of boundary generation. In conclusion, our comparison of the two meshes using the proposed system indicates that both techniques exhibit similar performance in generating high-quality boundary cells. However, the method employed to produce the mesh shown in Figure 2 (b) is less effective in generating high-quality cells on the circular plane.



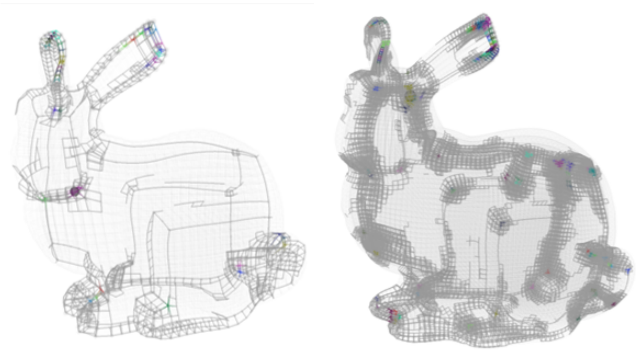
(a) rkm012-1-Fu2016 (b) rkm012-1-proposed

Figure 2: Comparison of two hex-meshes generated using two different techniques. For both visualization, we set $r_{max} = 1$, $r_{dmin} = 0.5$, and $e_{qmax} = 0.85$.

In experiment 2, we compare two polycut techniques using the bunny model. Through the feature edges visualization, we can see that the numbers of cells of the two models are different, but then, through the aggregate glyphs we see that their structures and clustering positions are relatively similar. For example, there are clusters between the ears and the two front legs. However, the number of displayed feature edges shows that the quality of Figure 3 (a) is better than the quality of Figure 3 (b), because fewer feature edges are shown when the threshold values are identical. In the future, technique (b) may be improved by optimizing the small elements along the structure.

In experiment 3, both meshes show a lot of clusters, but since no glyphs are highlighted, we believe that the element quality of the cluster region is good for both meshes. By comparing the two filtered feature edges, we can see that Figure 4 (b) has a clearer edge structure than (a). This indicates that the quality of the mesh in Figure 4 (b) is relatively better than the one in (a). Figure 4 (a) can

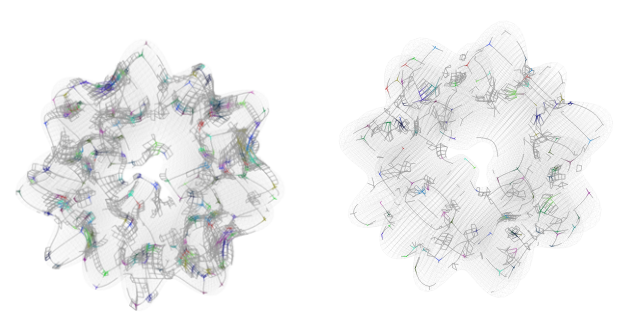
[*]e-mail: lsi@uh.edu
[†]e-mail: gchen16@uh.edu



(a) bunny (b) bunny

Figure 3: The comparison of two different polycut approaches for the bunny model. (a) $r_{max} = 1$, $r_{dmin} = 0.2$, and $e_{qmax} = 0.9$ (b) $r_{max} = 0.01$, $r_{dmin} = 0.002$, and $e_{qmax} = 0.9$.

be improved in areas where the boundaries change drastically.



(a) bumpy torus (b) bumpy torus

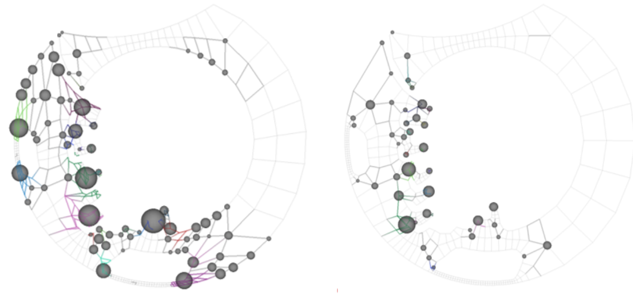
Figure 4: The comparison of two bumpy torus hex-meshes. Both visualizations $r_{max} = 0.01$, $r_{dmin} = 0.04$, and $e_{qmax} = 0.8$.

3 Quad mesh quality visualization

It is known that the quality of quadrilateral meshes impacts the accuracy and efficiency of various finite element simulations [4, 7]. Therefore, a common goal of all quad mesh generation and optimization algorithms is to produce high-quality meshes [2].

3.1 Comparison of 2D quad meshes before and after optimization

We use our system to compare the quality of two quad meshes before and after optimization. Figure 5 shows such an example. In this example, a maze wheel quad mesh is used. The original mesh and its quality are shown in Figure 5 (a), while the optimized mesh using the method by Akram et al. [1] is shown in (b). We can immediately notice the reduction of both the number of aggregated glyphs and the sizes of the individual glyphs in (b) when compared to those in (a). This suggests the improvement of the overall element quality of the entire mesh. To study the boundary error, we turn on the Boundary Error Analysis Window as shown in Figure 6. Since this mesh consists of three closed boundary curves, including the outmost boundary consisting of most boundary vertices and two small holes. The collated error plot shown in the lower right of the figure suggests that most parts of the boundary are preserved well in the optimized mesh, especially the two holes as their error ranges are smaller than the range of the outer boundary (e.g., the red section). Indeed, the places with the largest boundary error are on the outer boundary and located at the concave sharp feature.



(a) before optimization (b) after optimization

Figure 5: The aggregated glyph visualizations for a maze wheel quad mesh before (a) and after optimization (b).

3.2 Boundary error analysis for 2D meshes

For 2D meshes with boundaries, boundaries need to be extracted before calculating the boundary difference between the original (i.e., the output of some re-meshing or optimization algorithms) and the reference (i.e., the input mesh whose boundary needs to be preserved) mesh models. After boundary extraction, for each point on the extracted boundaries of the original mesh, the boundary error based on the closest point on the extracted boundaries of the reference mesh is calculated as:

$$b_{error} = \frac{dist(\mathbf{p}_{original}, \mathbf{p}_{reference})}{diag_{reference}} \quad (1)$$

where $diag_{reference}$ denotes the diagonal of the bounding box of the reference mesh.

Our method visualizes the extracted boundaries of the original mesh with color coding and a corresponding boundary error plot for each closed boundary. Note that multiple boundary curves may exist for models with holes. Each value on the boundary error plot belongs to $\mathbb{R}$, where positive values indicate that the boundary point is outside the reference mesh while negative values identify boundary points that are inside the reference mesh as shown in Figure 6.

In addition, a collated percentage graph is visualized for all the boundary vertices in the original mesh as illustrated in Figure 6 (bottom right plot). This enables further insight into boundary error by providing an overall representation of the boundary difference between the original and reference meshes. Thus, boundary areas with a maximum difference between original and reference meshes can be quickly identified.

4 User Feedback

To evaluate our proposed system, we created an online user feedback survey using Typeform, a platform providing various features that support the survey's credibility and feasibility. The reasons for conducting an online survey using Typeform are as follows.

Firstly, an online survey platform offers a highly convenient means of data collection, eliminating the need to consider geographical locations and enabling data collection from individuals across various disciplines.

Secondly, the ranking and picture selection features of Typeform reduce the necessity of pre-setting permutations, combinations, and answers. Consequently, this minimizes the potential influence of the designer on participants. For instance, rather than explicitly suggesting where each method needs attention, users can make selections based on their personal judgments.

Thirdly, the one-question-at-a-time display mitigates the issue of users referencing images from other questions while answering the current one.

Lastly, Typeform provides an engaging and visually appealing interface for the survey.

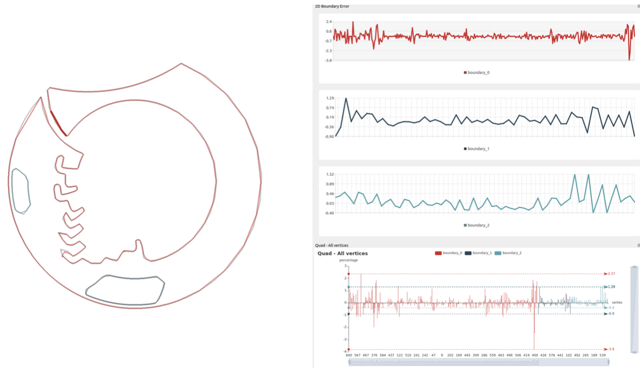


Figure 6: Interface for the 2D boundary error visualization. The main view (left) shows the overlapping boundaries of the two meshes [1]. The right views show the error plots for the individual boundary curves. The vertices on each boundary curve are sorted based on their positions on the curve. The bottom right plot shows the error for all boundary vertices, which supports user interaction such as zoom-in, panning, and selection. In this case, a selected boundary section is highlighted in the left main view.

Capitalizing on these features of Typeform, we designed 13 questions for users to evaluate our system. The survey attracted 185 views and received 38 valid submissions. The questions and corresponding results are discussed in the following three sections.

4.1 Introduction to Hexahedral Meshing

Given that both the proposed system and the survey cater to more than just mesh experts, we have included a brief introduction prior to the system-related questions. This ensures that participants acquire some foundational knowledge before delving into the main survey questions.

Hexahedral Meshing Introduction is the first part of this section, which contains two parts. The first part provides a brief explanation of hexahedral meshing. The second part asks participants to indicate whether they are mesh experts.

From the responses, we found that 11 participants identified themselves as mesh experts, while the remaining 27 indicated they did not possess expertise in the field.

Hex Element Shape Analysis seeks to educate users that an ideal hexahedral element has a regular hexahedral shape (i.e., a regular cube with six equal quad-faces and 90-degree angles between faces). Deviations from this shape, such as highly distorted or skewed elements, can have adverse effects on mesh quality.

Once users have grasped these basic concepts, they are asked to select the element of the worst quality from three options varying from ideal to 45 degrees and finally to a flat corner.

In the responses collected, 73% of users were able to correctly identify the worst shape.

4.2 Comparative Analysis: Evaluating Three Different Tools

This section focuses on the evaluation of three different methods used to analyze hex mesh quality.

Region Ranking In order to evaluate whether our proposed system can assist users in prioritizing different regions, we applied three methods to the same model. Given that the survey does not allow users to manually mark regions, we pre-identified three regions and labeled them for users to rank.

Based on the reference image Fig. 7, which is created by the proposed method, Region 1 contains the poorest quality elements and the globally lowest mesh quality. And Region 2 and Region 3 contain a considerable number of poor-quality elements.

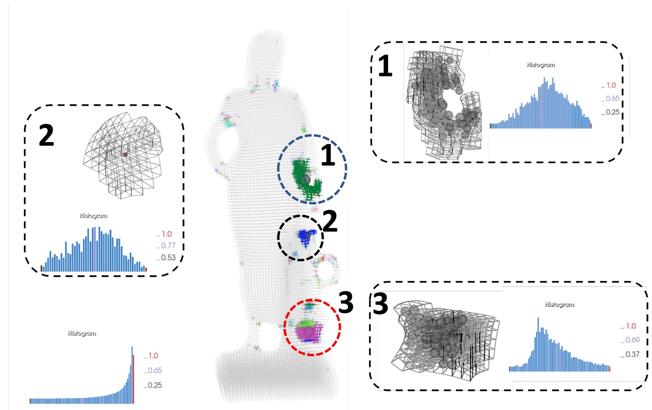


Figure 7: A Greek Sculpture model from [3] is analyzed by the proposed HQView system. Region 1 contains the lowest mesh quality, 0.25. The mesh quality in Region 2 is between 0.53 to 0.77, but since the region is very crowded, hence the region is also been noticed. Region 3 contains elements with quality from 0.37 to 1.0 since the quality of most elements in this region is lower than 0.69, hence the region is noticeable.

From the survey results, we observed that more than 50% of the respondents ranked Region 1 as the highest priority for all three methods. However, in Hexalab, 50% of users ranked Region 3 as the lowest priority. For volumetric context rendering and HQView, the respondents nearly equally divided Region 3 between the second and third rankings. The results indicate that Region 2 can successfully draw the attention of users, despite the mesh over this area being relatively small.

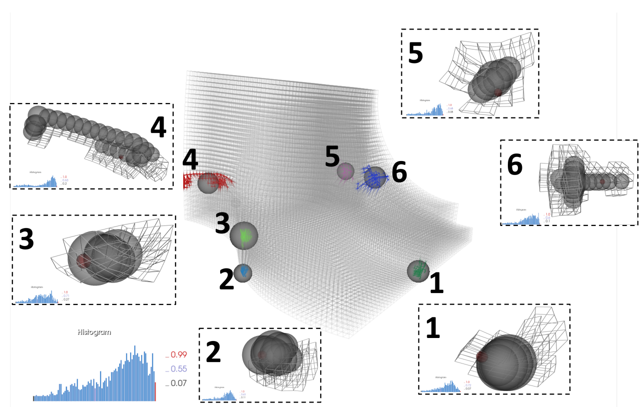


Figure 8: A fandisk model from [6] is analyzed by the proposed HQView system. This model contains 6 regions characterized by elements of poor quality. With the detail views, each of these regions contains at least one element with a quality value of less than 0.3

Region Counting In order to assess the effectiveness of identifying small-sized elements of poor quality, the survey asked participants to identify all regions containing such elements. A reference image, as displayed in Fig. 8, reveals that this model includes 6 distinct regions marked by elements of their quality. Upon the detailed sub-region reviews, it is observed that each of these regions has at least one element that falls below a quality score of 0.3.

The results indicate that Hexalab struggles to clearly depict all regions, with over 78% of responses ranging between 2 to 4 identified regions.

The focus+content volume rendering method, on the other hand, can display these regions, but it might lead to confusion among users. The range of regions identified from the focus+context volume rendering ranges from 3 to 9, and no single number of regions was selected by more than 50% of users.

Conversely, the proposed method can depict the regions more accurately. The number of regions identified ranged from 3 to 8. More specifically, 50% of respondents identified 6 regions, and 23.7% of respondents identified 7 regions.

Region Highlighting The main focus of our proposed method is to highlight regions or elements where the size of the element is small, yet the quality is poor.

To assess the effectiveness of this approach, we included two questions in this section to determine if participants find our method better at identifying such regions or elements.

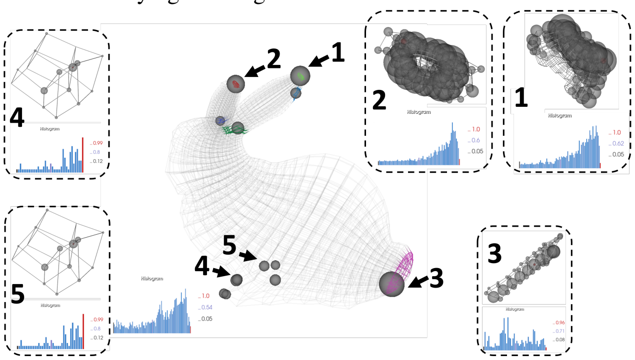


Figure 9: The bunny model is obtained from [5]. In this model, the areas with the poorest mesh quality are found at the tip of each ear, where the size of the mesh elements is also particularly small.

The first question revolved around a 3D bunny model. Within this model, certain elements, specifically located at the tip of the ears, were identified as being of low quality. Notably, these elements were small in size. The results demonstrate over 55.3% of respondents affirmed that our method was superior in detecting these small, low-quality regions within the bunny ears.

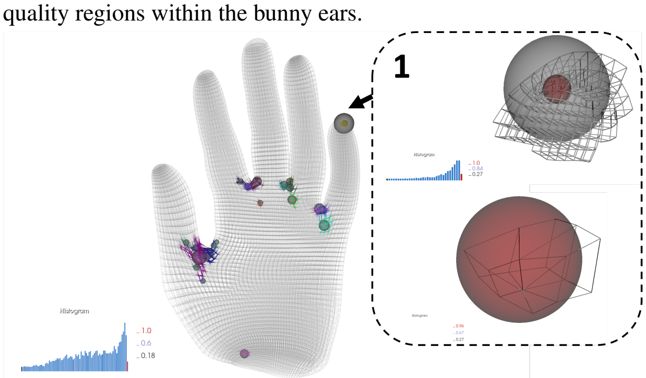


Figure 10: The hand model is obtained from [5]. Within this model, the majority of the mesh components exhibit superior quality. However, a solitary and exceptionally small element located at the tip of the little finger is inconspicuous. Our proposed system emphasizes this region by accentuating the problematic element with a prominently displayed, large glyph.

The second question pertains to a hand model. Within this model, an inconspicuous element located on the little finger exhibits poor quality, registering a score of 0.27 compared to the lowest mesh quality score of 0.18. The size of the element makes the element challenging to discern when using the other two methodologies. Consequently, our tool was chosen by over 50% of respondents as the optimal solution for this task.

Method Selection The last question in this section prompted participants to choose from multiple tools they think are effective at showing degenerate mesh elements.

Each of the three methods gets comparable responses. However, it's crucial to underline that the first option actually fails to accurately represent where the degenerate mesh elements are hidden by an edge.

The other three options have the ability to locate the regions that contain degenerate mesh elements.

4.3 Sub-System and Component Evolution

The proposed methodology, which incorporates a multi-view boundary error analysis subsystem and a component dedicated to highlighting overlapping elements, is further evaluated by presenting two questions to the users. These questions aim to gather users' ratings regarding the utility of these components.

Boundary Analysis Sub-system Over 54% of the responses rated the introduced components as 3 or higher (on a scale where 3 is neutral), indicating agreement with the usefulness of these additions.

Marks of Overlapping Elements Over 55% of the responses rated the overlapping indicator components as 3 or higher (where 3 is neutral), suggesting agreement with the stated usefulness of these features.

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] D. Bommes, B. Lévy, N. Pietroni, E. Puppo, C. T. Silva, M. Tarini, and D. Zorin. Quad-Mesh Generation and Processing: A Survey. *Computer Graphics Forum*, 32(6):51–76, 2013.

[3] X.-M. Fu, C.-Y. Bai, and Y. Liu. Efficient volumetric polycube-map construction. In *Computer Graphics Forum*, vol. 35, pp. 97–106. Wiley Online Library, 2016.

[4] 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.

[5] X. Gao, T. Martin, S. Deng, E. Cohen, Z. Deng, and G. Chen. Structured volume decomposition via generalized sweeping. *IEEE Transactions on Visualization and Computer Graphics*, 22(7):1899–1911, 2016. doi: 10.1109/TVCG.2015.2473835

[6] H.-X. Guo, X. Liu, D.-M. Yan, and Y. Liu. Cut-enhanced polycube-maps for feature-aware all-hex meshing. *ACM Trans. Graph.*, 39(4):106–1, 2020.

[7] 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.