Surface Remeshing: A Systematic Literature Review of Methods and Research Directions

Dawar Khan, Alexander Plopski, Yuichiro Fujimoto, Masayuki Kanbara, Gul Jabeen *(Student member, IEEE),* Yongjie Jessica Zhang, Xiaopeng Zhang, and Hirokazu Kato *(Member, IEEE)*

Abstract—Triangle meshes are used in many important shape-related applications including geometric modeling, animation production, system simulation, and visualization. However, these meshes are typically generated in raw form with several defects and poor-quality elements, obstructing them from practical application. Over the past decades, different surface remeshing techniques have been presented to improve these poor-quality meshes prior to the downstream utilization. A typical surface remeshing algorithm converts an input mesh into a higher quality mesh with consideration of given quality requirements as well as an acceptable approximation to the input mesh. In recent years, surface remeshing has gained significant attention from researchers and engineers, and several remeshing algorithms have been proposed. However, there has been no survey article on remeshing methods in general with a defined search strategy and article selection mechanism covering the recent approaches in surface remeshing domain with a good connection to classical approaches. In this article, we present a survey on surface remeshing techniques, classifying all collected articles in different categories and analyzing specific methods with their advantages, disadvantages, and possible future improvements. Following the systematic literature review methodology, we define step-by-step guidelines throughout the review process, including search strategy, literature inclusion/exclusion criteria, article quality assessment, and data extraction. With the aim of literature collection and classification based on data extraction, we summarized collected articles, considering the key remeshing objectives, the way the mesh quality is defined and improved, and the way their techniques are compared with other previous methods. Remeshing objectives are described by angle range control, feature preservation, error control, valence optimization, and remeshing compatibility. The metrics used in the literature for the evaluation of surface remeshing algorithms are discussed. Meshing techniques are compared with other related methods via a comprehensive table with indices of the method name, the remeshing challenge met and solved, the category the method belongs to, and the year of publication. We expect this survey to be a practical reference for surface remeshing in terms of literature classification, method analysis, and future prospects.

Index Terms—mesh generation, surface remeshing, meshing quality, finite element method, systematic literature review

1 Introduction

Mesh generation plays a vital role in representing three-dimensional (3D) data in mathematical modelling, computer animation, physical simulation, and many other computer graphics applications [1], [2]. Triangle meshes are commonly used in these applications owing to their efficiency, flexibility, and simplicity.

Meshes are typically generated in raw form from different sources, such as 3D scanners, 3D images, and 3D data of, for example, protein structures. These raw meshes mostly suffer from poor-quality elements (such as small angles, large angles, short edges, and irregular vertices) and other defects (such as redundant vertices and self-intersections) [1], [3], [4]. Downstream applications often fail to directly use these poor-quality meshes [1]. Therefore, a remeshing process is commonly applied beforehand to improve the mesh quality. High-quality surface meshes are crucial for many practical applications, including numerical simulation, 3D visualization, animation, tetrahedral generation, mesh segmentation, and mathematical modelling [1], [5], [6]. Research has revealed that mesh quality can significantly improve the performance results. For example, Gutierrez *et al.* [7] showed that simulation efficiency can be improved by at least 30% with their mesh improvement.

Despite the significant research on surface remeshing, there is no comprehensive survey article that follows a standard methodology to ensure its significance, completeness, and unbiasedness. In this article, we systematically review state-of-the-art surface remeshing techniques by following the standard systematic literature review (SLR) methodology (see Section 2). We classify articles into different categories, discuss their results analyses, highlight their pros and cons, and suggest some interesting future research directions.

1.1 Background and motivation

Surface remeshing has gained much attention from researchers, especially in the past two decades. Surface remeshing methods should be either specific to some particular applications, such as CAD models, or should be a general framework that is applicable anywhere. However, each method type has its own quality constraints and mechanism to meet these quality constraints.

Typically, remeshing algorithms consider different aspects to improve mesh quality. These aspects include the removal of

- *D. Khan, A. Plopski, Y. Fujimoto, M. Kanbara, and H. Kato are with IMD Lab, Information Science, Nara Institute of Science and Technology, Nara 630-0192, Japan.*
  E-mail: {dawar, plopski, yfujimoto, kanbara, kato}@is.naist.jp
- *G. Jabeen is with the School of Software Engineering, Tsinghua University, Beijing, China.E-mail: jgl14@mails.tsinghua.edu.cn*
- *Y. J. Zhang is with the Department of Mechanical Engineering at Carnegie Mellon University (CMU), USA.Email: jessicaz@andrew.cmu.edu*
- *X. Zhang is with NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing, China. Email: xiaopeng.zhang@ia.ac.cn*

Manuscript received MM dd, YYYY; revised MM dd, YYYY.

small and large angles [3], [5], [8], improving the regularity of vertices, efficiency, robustness, simplification, removal of short edges, and general defect removal [1], [9], [10]. In addition, a remeshing algorithm must consider the output mesh's degree of approximation from the input. Furthermore, it must preserve sharp features and the topology [11], [12].

With these different remeshing objectives and a number of target applications, research on surface remeshing has increased rapidly. As an example, the work done in the most recent decade is almost double as that of the previous decade. To the best of our knowledge, this is the first SLR in the meshing domain. Although there are a few review articles in the meshing domain, as discussed in Section 1.2, these articles were not SLRs and did not use any other systematic review methodology for all existing articles.

1.2 Related surveys

An SLR [13], used in medical sciences and software engineering, is a review conducted using a predefined methodical series of steps. To the best of our knowledge, no SLR has been conducted in the mesh processing domain. In this section, we describe some review articles in the meshing domain.

Payan *et al.* [14] surveyed various semi-regular (SR) triangle remeshing algorithms developed from 1995 to 2015. They classified the algorithms based on their remeshing goals and input/output meshes. In addition, they highlighted various interesting future directions. Based on their goals, their study [14] is related to ours; however, their review is not clearly defined and is specific to SR remeshing.

There have been some other surveys covering only a specific meshing domain, such as unstructured meshes [15], feature remeshing [16], bio-medical mesh generation [17], and molecular surface meshing [18]. Araújo *et al.* [19] conducted a comprehensive survey on implicit surfaces (IS) and visualization methods. However, the scope of their study is very general in the visualization of IS and contains minimal work on remeshing of surfaces. Heckbert and Garland [20] conducted a comprehensive survey of algorithms for simplifying polygonal surfaces. They classified different methods, summarized different articles, and conducted a comparison of various approaches; however, they did not provide any mechanism for article search and selection. Similarly, Shamir [21] conducted a review of mesh segmentation techniques. He classified state-of-the-art segmentation methods based on their goals and approaches.

To the best of our understanding, a 2008 survey article [2] is the most relevant to our study. However, it may be of less interest now because much new work has been conducted in the past 12 years. Furthermore, it did not use any systematic methodology. Therefore, a new study to provide a comprehensive review of all surface remeshing methods, especially those proposed in the last 10 to 15 years, is required. In summary, to the best of our knowledge, there is no survey in the mesh processing domain that is based on well-defined rules for article searching, selection, and data extraction. Our method has the following distinctions over previous surveys:

1) We conducted a systematic review that covers all the existing literature on surface remeshing.
2) We followed a well-defined strategy for article search, inclusion/exclusion, quality assessment, and data extraction while avoiding any possible research bias.
3) We conducted a detailed analysis of state-of-the-art methods.

1.3 Contributions

The contributions of this review can be summarized as follows:

1) We conducted the first SLR in the surface remeshing domain. We identified a set of 104 primary studies describing different surface remeshing algorithms.
2) We classified all articles, discussed their pros and cons, and highlighted future directions.
3) We identified different parameters used for surface remeshing algorithm analysis.
4) We also identified recent advances in different aspects of quality improvement.

Researchers can use this survey article to extend their knowledge in this domain. The pros and cons of all articles along with future direction suggestions will give researchers a good basis for their research. The identification of quality measurements will help researchers to understand important quality considerations in surface remeshing and how to define a good mesh. Furthermore, depending on their quality requirements, researchers/end-users can easily find a suitable state-of-the-art algorithm for their research/applications. This will also help them in understanding how to conduct algorithm analysis for surface remeshing and compare their results with those of state-of-the-art algorithms.

2 Research Methodology

We imitated the SLR guidelines from the software engineering field [13]. An SLR reviews existing studies by following a set of pre-defined steps for identification, analysis, and interpretation of all state-of-the-art evidences related to a specific research question in an unbiased and (to a degree) repeatable manner [13]. Figure 1 illustrates the overall pipeline of our study. Our methodology is further described in the following subsections.

2.1 Research questions

Prior to article search and selection, we formulated our research questions. Research questions provide a baseline to other phases of the review, including search string design, inclusion/exclusion criterion, and data extraction [13]. With reference to the objectives of our study, we formulated the following research questions:

RQ 1: What are the challenges in remeshing surfaces?
RQ 2: What are the remeshing objectives in high-quality surface remeshing?
RQ 3: What are the techniques for high-quality surface remeshing?
RQ 4: What are the metrics used for surface remeshing algorithm analysis?

2.2 Search strategy

Search strategy plays a vital role in collecting relevant literature on a specific topic in an efficient manner. The search string itself and the libraries used for search are significant. We designed one search string and selected five different libraries for searching related articles. We considered the keywords from our research questions (Section 2.1), followed the SLR guidelines [13], and designed a search string by concatenating the keywords and their synonyms using Boolean operators. We refined and finalized the following search string:

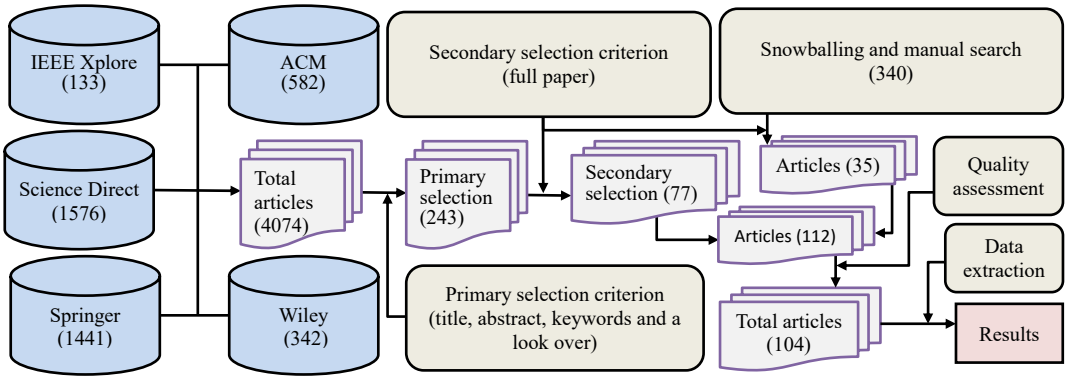


Figure 1. Research methodology, including search strategy, article selection criteria, quality assessment, and data extraction.

((“computer graphics” OR “computational geometry”) AND (“surface mesh*” OR “surface remesh*” OR “mesh improvement” OR “mesh refinement”) AND (quality OR “minimal angle” OR “maximal angle” OR “aspect ratio” OR “feature preserv*” OR “short edge” OR “regular*”) AND (triangle OR triangulation))

The following libraries were searched for articles.

1) IEEE Xplore http://ieeexplore.ieee.org,
2) ACM digital library https://dl.acm.org,
3) ScienceDirect http://www.sciencedirect.com,
4) SpringerLink https://link.springer.com, and
5) Wiley https://onlinelibrary.wiley.com/.

In addition to the above list of libraries, the *Eurographics digital library* http://diglib.eg.org/ was searched manually as it does not support our search string. We also added articles using manual searches of other sources. In addition to SLR and manual search, we also used the snowballing concept [22] to include further studies. We included the relevant articles cited in a given article (forward snow balling) as well as the relevant articles citing a given article (backward snow balling).

2.3 Inclusion and exclusion criteria

In the initial search, we included articles by reading the title, keywords, and abstract of each. In the second phase, we read the entire article and excluded certain articles based on the following inclusion and exclusion criteria (see Figure 1):

1) We included articles on triangular meshing (including surface meshing and 2D meshing) and excluded those related to quad, tetrahedral, and hexahedral meshing.
2) We included articles concerned with mesh quality (such as triangle quality, aspect ratio, minimal angle, maximal angle, regularity, and features preservation) improvement and excluded those with general applications without considering mesh quality.
3) We included articles describing the challenges and objectives of mesh generation or its improvement.
4) We included articles describing an algorithm whose input and output are meshes.
5) We excluded articles presenting non-peer reviewed reports and books.
6) We excluded articles presented in non-English languages.
7) We excluded articles duplicating other articles.
8) We excluded articles based on our quality assessment, as described in Section 2.4.

2.4 Article quality assessment

In addition to the above selection criteria, we used a quantitative measurement to rate the quality of each article and its suitability in the context of our current study. We analyzed each article based on the following questions:

1) Is the article frequently cited (number of citations per year using Google Scholar citation)?
2) Is it published in a relevant and high-quality journal/conference?
3) Is the method novel, appropriate, and helpful to improving a sufficient number of mesh quality metrics?
4) Are the results analyzed appropriately (compared with existing methods clearly and accurately or proved theoretically)?

Each of these questions was answered with "good: 1", "average: 0.5", or "poor: 0". The articles with an accumulative assessment score less than 2 were excluded from the review. For Question 1 (i.e., yearly citations), recently published articles (after 2017) and those with 5 or more yearly citations were marked with 1. The remaining articles were marked with 0.5 if they had ≥ 3 yearly citations and with 0 otherwise. For Question 2, the articles published in relevant high-quality journals/conferences were marked 1, and those published in multi-disciplinary journals/conference were marked 0.5 or 0. Each method was analyzed for its novelty and performance. Similarly, results were assessed for their completeness and relevance to this survey. We excluded 8 articles based on their quality assessment.

2.5 Data extraction

With consideration of our research questions, we extracted the following information from each selected article:

1) The key challenges described in the article, and the proposed solution to overcome these challenges.
2) The main remeshing objectives in the article and the way these objectives were achieved.
3) The main concept, the pros and cons of the proposed algorithm, and possible future directions.
4) The parameters used for measuring mesh quality and method of results analysis.

2.6 Presentation of findings

The literature provides numerous surface remeshing techniques. In total, we selected 104 articles for detailed analysis. We classified these articles into 9 major classes. The extracted results are organized as follows. First, in Section 3, we present different

parameters used for surface remeshing algorithm analysis in the selected articles. The main classification of the articles (see Figure 6) with their pros and cons and possible future directions is presented in Section 4. Recent advances in remeshing objectives are summarized in Section 5. Table 2 presents a brief summary of the selected articles with their categorization. Note that there is no final categorization, and all categories overlap with each other. Therefore, some methods may belong to more than one category. Table 3 lists the pros and cons of each method. Finally, Section 6 provides a discussion on our findings and a summary of future research directions.

3 Surface Remeshing Evaluation Metrics

Meshes are generated from different sources, such as captured images [23], [24], [25], [26], scanned point clouds [27], specified CAD models [28], and other 3D data [29]. A generated mesh is typically refined for quality improvement prior to use in downstream applications. Surface remeshing is a process that converts an input mesh into another optimized output mesh to reach a predefined goal while approximating the input one [2]. The goal of this conversion varies from application to application.

There are many different metrics for meshing quality and analysis of surface remeshing algorithms, including triangle quality, minimal angle, maximal angle, aspect ratio (AR), regular vertices, approximation error, and time complexity [1], [3], [30].

In this section, we summarize different metrics used for the analysis of surface remeshing methods. These metrics help researchers to compare their results with those of state-of-the-art methods. In fact, the literature has different quality measurements for analysis of different remeshing objectives. High-quality remeshing smooths a given mesh and improves quality metrics such as aspect ratio, regular vertices, and maximal and minimal angles. Improvement of these quality measurements leads to efficient and reliable simulations.

Similarly, different methods [30], [31], [32], [33] have their own energy functions reflecting the mesh quality and are used for the analysis of their algorithms' convergence. Nguyen *et al.* [34] proposed several quality measurements for uniform grids in the planar domain. These quality measurements included covariance (COV) λ, mesh ratio γ, regularity, cell volume deviation v, second moment trace τ, second moment determinant d, and normalized standard deviation p. Table 1 lists different metrics used in articles for measuring mesh quality and analyzing remeshing algorithms. Figure 5 plots a histogram to present the frequency of each metric from the articles in our final selection. In the following, we discuss these metrics.

3.1 Triangle quality

The quality $Q(t)$ of a triangle t can be calculated as $Q(t) = \frac{6}{\sqrt{3}} \frac{A_t}{S_t E_t}$, where A_t is the area of triangle t, S_t is its half-perimeter, and E_t is the length of its longest edge [35]. Typically, Q_{min} and $Q_{avg.}$ are used for quality measurement, representing minimal and average quality of a triangle(s), respectively.

AR is another metric used for triangle quality measurement. There are various definitions of the AR of a triangle. For example, $AR = \frac{\sqrt{3}E^2}{4A_t}$, where E is the longest edge and A_t is the area of the triangle [36]. In some articles, it is simply calculated as the ratio of the longest to the shortest edge length of a triangle [37]. Similarly, in some articles, it is defined as the ratio of the circumradius to the in-radius of a triangle, calculated as $AR = \frac{e_1 e_2 e_3}{8(S-e_1)(S-e_2)(S-e_3)}$, where e_1, e_2, and e_3 are the lengths of the triangle's edges and $S = \frac{e_1+e_2+e_3}{2}$ [1]. Despite their minor variations, all definitions have similar meanings. The AR of an equilateral triangle is 1.

Radii ratio is another metric used; it is the ratio of the triangle's incircle to its circumcircle [38]. Higher radii ratios indicate higher quality triangles and vice versa; a zero value indicates a degenerated triangle [38]. Triangle regularity is another metric used, calculated as the ratio of the in-radius r to the length of the longest edge E, i.e., $Triangle\ Regularity = \frac{2\sqrt{3}r}{E}$ [39]. A similar alternative measurement for triangle quality is radius edge ratio $ReR = \frac{R}{e}$ [40], where R is the circumradius and e is the length of the shortest edge of the triangle. For all these metrics, a smaller value indicates a more uniform grid and vice versa.

Similarly, θ_{min} is the minimal angle and θ_{max} is the maximal angle in the output mesh. The minimal angle of a triangle is relevant to the radius edge ratio (ReR): $\sin\theta_{min} = \frac{1}{2ReR}$ [41]. $\overline{\theta}_{min}$ is the average value of the minimal angles in all triangles. Maximal and minimal angle improvements [42] and especially non-obtuse remeshing [3], [5], [43], [44] have received significant attention in recent years.

Chiang *et al.* [45] used another quality measurement for mesh quality:

$$Q(M) = \frac{1}{n} \sum_{\theta_i \in M} \|\theta_i - \theta_{opt.}\|, \quad (1)$$

where θ_i is the interior angle of the triangle or quad, and $\theta_{opt.}$ is the optimal value (i.e., 60^o for a triangle and 90^o for a quad mesh).

3.2 Topology preservation and geometric error

In surface remeshing, while changing the mesh structure and its element positions, one key consideration is the preservation of its shape and topology. In other words, the output mesh must have an acceptable approximation to the input mesh. Some features, such as sharp corner preservation, can be observed visually. However, for geometric approximation, various measurements have been used. Hausdorff distance (d_H) [46], [47], [48], mean distance, and mean square distance (d_{RMS}) [33], [49] have been used to measure the difference between input and output meshes to ensure topology preservation or shape approximation.

The Hausdorff distance d_H between an input surface mesh M with vertex set $\{v_1, v_2, v_3, ..., v_n\}$ and an output surface mesh M' with vertex set $\{v'_1, v'_2, v'_3, ..., v'_n\}$, is calculated as(2)

$$d_H = \max\{d_H(M, M'), d_H(M', M)\}, \quad (2)$$

where

$$d_H(M, M') = \max_{p \in M}\{d(v, M')\}, \quad (3)$$

and

$$d_H(M', M) = \max_{v' \in M'}\{d(v', M)\}, \quad (4)$$

where

$$d(p, M') = \min_{p' \in M'} d(v, v') \quad (5)$$

is the Euclidian distance from vertex v to the nearest vertex in mesh M', and

$$d(v', M) = \min_{v \in M} d(v', v) \quad (6)$$

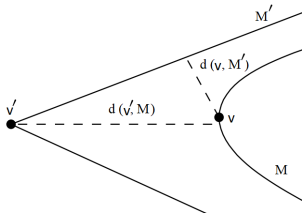


Figure 2. Asymmetric property of Hausdorff distance [50].

is the Euclidian distance from vertex v' to the nearest vertex in mesh M. Note that d_H calculated by Equations (3) and (4) is asymmetric, i.e., $d_H(M,M') \neq d_H(M',M)$ (see Figure 2, where $d_H(M,M') < d_H(M',M)$ because $d(v,M') < d(v',M)$), while Equation (2) is its symmetric counterpart [11], [47], [50].

Mean distance [49] is another metric, which is calculated as the surface integral of the distance divided by the area of the mesh M. We have

$$d_{mean}(M,M') = \frac{1}{|M|}\int_{v\in M} d(v,M')dM, \tag{7}$$

where $|M|$ is the area of the surface mesh M. Similarly, the root mean square distance (d_{RMS}) is also used to measure the geometric approximation [50]. Following Equation (7), d_{RMS} is calculated as

$$d_{RMS}(M,M') = \sqrt{\frac{1}{|M|}\int\int_{v\in M} d(v,M')^2 dM}. \tag{8}$$

3.3 Vertex regularity

The valence of a vertex (number of all adjacent edges to a vertex), called vertex regularity, is another quality metric used in surface remeshing algorithm analysis [1], [30]. Valence-6 vertices are optimal (regular) vertices in the interior of a mesh, whereas valence-4 vertices are optimal vertices at the boundary of a mesh [1], [30]. Vertices with optimal valence (valence 6) or near to optimal valence (valence 5 or 7) are well-suited to surface remeshing. In this regard, $V_{567}\%$ is typically used as a quality measurement; $V_{567}\%$ represents the percentage of vertices with optimal or near optimal valence (i.e., 5, 6, or 7) [5], [11], [30], [51].

3.4 Visual perception

In addition to quantitative metrics, visual perception is also important. Visual results are used to examine different aspects, including sharp feature preservation (see Figure 11), shrinkage, adaptivity, and regularity of the mesh. Furthermore, the ratio of low-quality elements can also be highlighted visually (see Figure 19 for the ratio of obtuse triangles and Figure 23 for vertex regularity). Typically, the surface mesh is given with highlights on different elements to show the visual quality.

3.5 Validity and complexity

The output mesh should be valid. Mesh validity ensures that the mesh is a closed and simple manifold [2]. The mesh complexity (number of mesh elements) has a trade-off with efficiency, memory usage, and accuracy [52], [53]. Complex meshes are more accurate (see Figure 3), whereas simple meshes are more efficient (see Figure 4). Therefore, remeshing algorithms aim to maintain a balance between mesh complexity and efficiency. Some algorithms [3], [54], [55] insert new points for mesh quality improvement. An algorithm that reaches a target quality limit with minimal point insertion is preferred over the algorithm that requires more point insertions. Sifting ratio α [56] is another metric used to represent the percentage reduction of vertices during mesh simplification.

3.6 Execution time

Similar to other algorithm analyses, execution time is also used for analysis of different remeshing algorithms [3], [5], [11], [39], [59], [60]. In addition to other metrics, it is important to consider the execution time of the remeshing algorithm. The impact of the remeshing algorithm on the efficiency of the downstream application is also significant. However, some algorithms only consider quality without considering time, whereas others consider only improving the efficiency of existing methods while achieving the same quality. For example, Figure 4 shows a time comparison among different state-of-the-art methods.

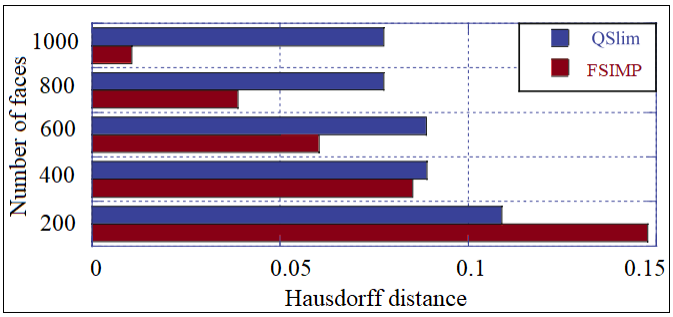


Figure 3. Impact of mesh complexity on accuracy: Hausdorff distance versus number of faces. A comparison of fast simplification (FMSIM) [57] and quadric simplification (QSlim) [58]. For both methods, the Hausdorff distance decreases with increasing number of faces. These results are taken from FMSIM [57].

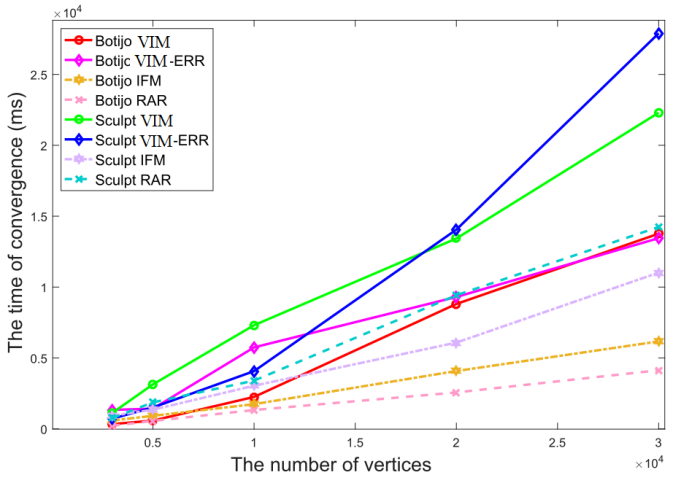


Figure 4. Timing statistics versus mesh complexity. A comparison among the latest methods, including RAR [59], instant field meshing (IFM), and vertex insertion methods (VIM simple and VIM modified) [5]. Note that the convergence time increases with increasing number of vertices. These results are taken from VIM [5].

3.7 Algorithm robustness

Some algorithms work only for a specific domain, while others are more generic and work on arbitrary models. Similarly, some algorithms only work on small models but fail on large models [30], [61], [62]. Robustness is measured by remeshing different models with arbitrary complexity. Occasionally, the output mesh is checked in the downstream application for success or failure [1], [63]. For example, the outputs of SMOPT [63] and another local refinement method [1] have been checked in their downstream application, in this case TetGen [64].

4 Selected Articles with In-depth Analysis

Surface remeshing methods overlap with each other in different aspects, such as having similar remeshing objectives, same

Table 1
Metrics for analysis of surface remeshing algorithms adopted in the reviewed literature, where e is the triangle's shortest edge, R is its circumradius, and r is its in-radius.

Metric	No. of Articles	References
$Q(t) = \frac{6}{\sqrt{3}} \frac{A_t}{S_t E_t}$	29	[1], [3], [5], [11], [12], [30], [32], [33], [44], [60], [65], [66], [67], [68], [69], [70], [71], [72], [73], [74], [75], [76], [77], [78], [79], [80], [81], [82], [83]
θ_{min}	45	[3], [5], [11], [12], [30], [33], [36], [54], [56], [59], [60], [65], [66], [67], [68], [70], [71], [72], [80], [84], [85], [86], [87], [88], [89], [90], [91], [92], [93] [1], [10], [32], [43], [63], [74], [76], [77], [78], [81], [82], [83], [94], [95], [96], [97]
θ_{max}	28	[1], [3], [5], [11], [12], [30], [33], [36], [43], [44], [54], [56], [60], [63], [65], [70], [71], [72], [74], [77], [78], [82], [84], [85], [87], [92], [96], [97]
$\bar{\theta}_{min}$	21	[1], [3], [5], [12], [30], [33], [43], [59], [66], [67], [70], [76], [77], [78], [80], [81], [82], [86], [88], [89], [94]
Aspect ratio (AR)	15	[1], [3], [12], [30], [36], [45], [74], [88], [95], [98], [99], [100], [101], [102], [103]
d_H (Hausdorff distance)	37	[5], [6], [9], [11], [30], [33], [36], [53], [56], [58], [59], [65], [70], [71], [72], [74], [75], [77], [78], [84], [87], [88], [89], [94], [95], [100], [104], [105] [32], [38], [48], [82], [83], [99], [106], [107], [108]
$V_{567}\%$	9	[5], [6], [11], [30], [70], [71], [72], [77], [78]
Vertices Regularity	9	[1], [5], [6], [11], [12], [74], [84], [86], [89]
d_{RMS}	19	[5], [11], [30], [33], [45], [48], [53], [56], [58], [59], [60], [67], [70], [72], [74], [78], [80], [106], [109]
radii ratio	4	[38], [101], [102], [103]
Execution time	73	[3], [5], [11], [31], [39], [45], [59], [60], [61], [65], [71], [72], [77], [79], [80], [84], [88], [90], [91], [92], [93], [101], [107], [110], [111], [112], [113], [114] [9], [10], [32], [33], [36], [44], [56], [70], [74], [75], [76], [78], [86], [89], [99], [100], [103], [105], [106], [109], [115], [116], [117], [118], [119], [120], [121] [43], [48], [63], [81], [82], [83], [94], [97], [104], [108], [122], [123], [124], [125], [126], [127], [128], [129]
Radius edge ratio $= \frac{R}{e}$	8	[9], [40], [74], [85], [115], [122], [123], [128]
Sharp features	16	[11], [33], [45], [52], [74], [75], [92], [96], [100], [102], [106], [107], [112], [116], [119], [125]
Genus	7	[1], [39], [61], [63], [66], [99], [113]
Standard deviation	4	[3], [61], [82], [94]

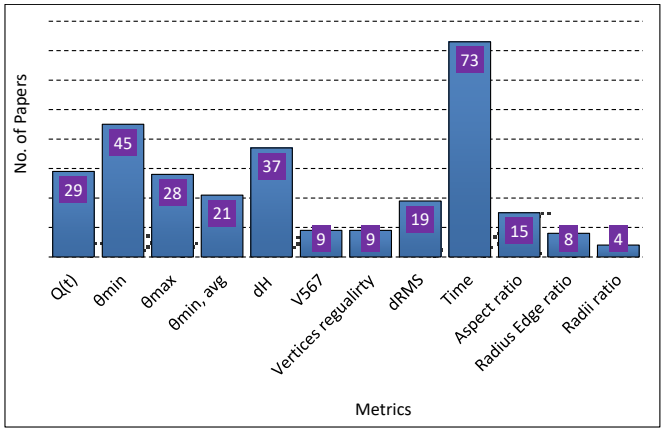


Figure 5. Histogram of the metrics used for analysis of surface remeshing methods. The frequency is calculated as the number of articles from our selection that use each particular metric.

methodology, or same specific domain. We classified the selected articles into 9 different categories based on their remeshing methodologies. Table 2 indicates how the articles overlap in different categories. However, a detailed description of each article is provided under its most relevant category. Figure 6 shows our classification of these articles. The articles in each category are described in the following subsections. Figure 7 shows the year-wise distribution of the selected articles. Figure 8 shows the quality assessment scores for all selected articles.

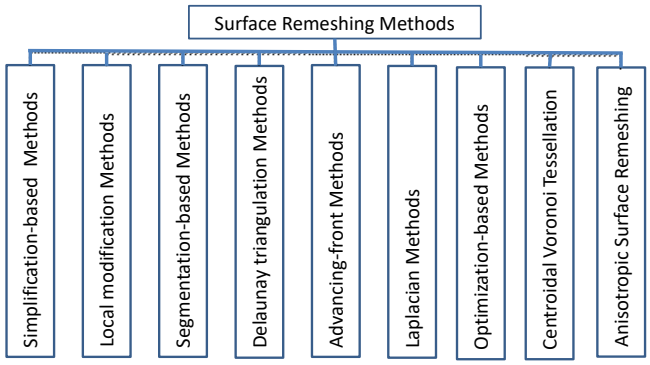


Figure 6. Classification of selected articles.

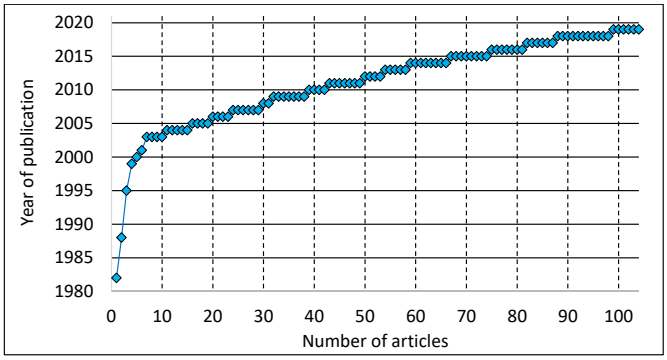


Figure 7. Year-wise distribution of selected articles.

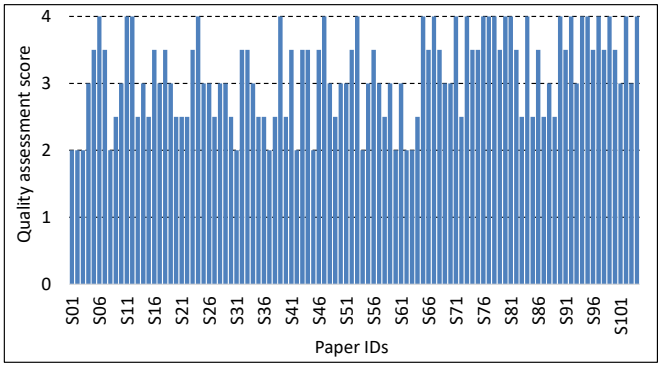


Figure 8. Histogram of quality assessment, indicating the quality of each article calculated based on its citations, quality of publication venue, proposed method, results, and relevance to this survey.

4.1 Remeshing with simplification

Algorithms generate meshes in complex form (with greater details) for a better approximation of the surface. However, mesh complexity affects efficiency in downstream applications. Therefore, mesh simplification is used to minimize the input mesh complexity for efficient computation and optimal memory consumption [20], [53]. To balance the tradeoff between accuracy and efficiency, the mesh must be simplified to an optimal level of detail.

Various methods have been proposed for mesh simplification. Some of these methods, such as that in [130], have improved efficiency but consume more memory. Other methods improve efficiency and use optimal memory. For example, the half-edge

Figure 9. Results from recent methods of mesh simplification. Note that the method of Yi *et al.* [104] is better in Hausdorff distance (in terms of minimal error) than other methods, such as the methods of Liu *et al.* [105] and Dyer *et al.* [87], in geometry-altering (GA) and geometry-preserving (GP) mode even though there is no significant improvement in the visual results. These results are taken from Yi *et al.* [104].

collapsing scheme [58], which collapses edges without introducing new vertices, improves efficiency with optimal memory usage. Normal field deviation [57], which uses geometric fidelity, is two times faster than [130] while also consuming less memory.

Distance measurement is used to compute geometric distortion, which is combined with field deviation strategy [57] to reach an optimal mesh decimation [117]. This method [117] maintains an optimal trade-off among accuracy, efficiency, and memory usage. However, there is no significant improvement in mesh quality. The half-edge collapsing scheme was recently extended to consider edge length during simplification for further efficient processing [53]. This method [53] uses Euclidean distance to ensure decimation validity. The results show that this method performs better than previous methods [57], [58], [117] in terms of efficiency and approximation error control. However, there is no significant improvement in mesh quality rather than simplification.

Feature preservation is also important in mesh simplification. In this regard, Wei and Lou [108] proposed a new approach for mesh simplification with feature preservation via a feature sensitive metric. However, it is slower than other quadratic error metric (QEM) simplifications [130], which can be improved in the future. Similarly, Cai *et al.* [131] used optimal geometric partition for surface approximation. They used principle component analysis (PCA)-based energy, which guarantees polygonal approximation and improves efficiency by avoiding connection information during optimization. However, the triangular approximation is lacking optimality, which is a future challenge.

Constraint re-sampling [56] is another method that uses different re-sampling operators (including vertex translation, vertex removal, and vertex insertion in a feasible way) to generate high-quality meshes. The advantage of constraint re-sampling is that it can simplify Delaunay meshes with preservation of element quality and elimination of obtuse angles and short edges. However, its failure in obtuse angle elimination from complex input is its main limitation, while it also fails in sharp feature preservation.

Dyer *et al.* [87] presented an algorithm for Delaunay mesh generation from a typical triangle mesh using edge-based operations. Their method is based on non-locally Delaunay (NLD) edges. An edge is locally Delaunay if the sum of its opposite angles is 180^o; otherwise, it is an NLD edge. Their method flips NLD edges if the dihedral angle is zero; otherwise, they handle it with an edge split. They also proposed a mesh decimation scheme to simplify Delaunay triangulation (DT) to a given level of detail. This method can generate geometry-preserving Delaunay meshes with simplification. However, the mesh quality has no significant improvement (e.g., the minimal angle is still $< 2^o$). Liu *et al.* [105] proposed another NLD edge refinement strategy for simplification of Delaunay meshes using QEM [130]. Their method is simple, easy to implement, and efficient. However, its space complexity is $O(Kn)$, where n is the number of vertices and K is a model-dependent constant. Minimization of this complexity to $O(n)$ is a possible future work.

Recently, a differential evolution-based method (DESIMP) [104] that gives a Delaunay mesh after its simplification was proposed. DESIMP gives better approximation than previous methods [87], [105] and is good for CAD and man-made models with sharp features. However, it is limited to 2-manifold meshes, and it does not significantly improve visual effects (see Figure 9). Furthermore, it is 10 to 100 times slower than the method proposed by Liu *et al.* [105]. Ozaki *et al.* [109] proposed another QEM-based method that divides large meshes into patches and then simplifies them. The method is efficient and can handle large models; however, parallel processing of the patches can further improve the efficiency. Another limitation is that the size of patches is not same, which is another future direction.

Furthermore, a CVT-based method [65] that leads to uniform coarsening and simplification of input meshes was proposed. This method [65] uses CVT [132] to simplify a complex mesh into a mesh with fewer elements. The method [65] is efficient and easy to implement with low memory requirements. However, it fails with sharp features and adaptive meshes. An extended version of this algorithm has already been proposed for anisotropic discrete centroidal Voronoi diagrams (DCVDs) [33]. DCVD [33] is an extension of the uniform CVT [65] to non-uniform and anisotropic meshing. DCVD also simplifies a mesh into a user-given number of elements. It is able to process complex meshes up to several millions of triangles, but it is computationally slow.

Triangle contraction is another approach to mesh simplification; it uses the Hausdorff distance to ensure geometric approximation [48]. Yue *et al.* [84] presented a method for mesh quality improvement of CAD models that uses an edge-based operation for vertex sampling, weighted CVT for regularity, and signal processing modules for mesh de-noising. In addition, it has an optimization module for vertex connectivity. This method [84] is robust, preserves features, and avoids shape deviations. However, it is specific to CAD models. Furthermore, the angle quality is still very low. For example, the results still have a minimal angle of 10.2^o.

The main challenge of mesh simplification is to find an optimal balance among accuracy, efficiency, and memory management.

DESIMP [104] is the latest method that preserves sharp features and gives minimal error, but its visual effects and efficiency are still lower than those of a previous method [105]. Similarly, the method of Liu *et al.* [105] also yields good visual results with satisfactory control over the approximation error (see Figure 9). Further investigation is required to improve efficiency, and more semantics can be utilized to highlight important parts of the input mesh as priority regions.

4.2 Remeshing with local modification

Local modifications change only some parts of the mesh with local operators, such as edge flipping, edge collapsing, edge splitting, and vertex translation [11], [58], [59], [84], [87], [93]. Figure 10 illustrates local operators and how they improve θ_{min}.

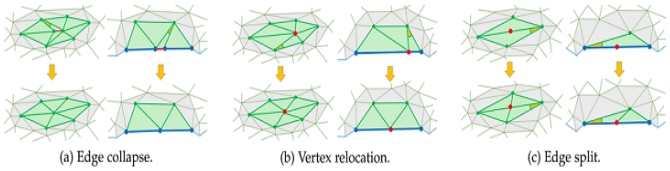


Figure 10. Local Operators. The top and bottom rows show local regions before and after applying local operators, respectively. In each sub-figure, the left side shows local regions on the inner, while right shows local regions on the boundary. The green faces are directly affected by local operations. The yellow color represents θ_{min} [11].

Among recent works, real-time adaptive remeshing (RAR) [59] is an efficient method that works with local operators. In addition to its efficiency, RAR is easy to control and simple to implement. The efficiency of RAR makes it suitable for real-time applications. This method uses an adaptive sizing function to compute the edge length $L(e_i)$ for an edge e_i with vertices v_s and v_t at its two ends. We have

$$L(e_i) = min\{L(v_s), L(v_t)\}, \tag{9}$$

where the sizing field $L(v_i)$ is calculated for vertex v_i as

$$L(v_i) = \sqrt{6\varepsilon/\kappa_i - 3\varepsilon^2}, \tag{10}$$

where ε is an error tolerance and κ_i is the maximum absolute curvature of a vertex. An edge with an actual length shorter than $\frac{4}{5}L(e_i)$ is collapsed, and if its actual length is longer than $\frac{4}{3}L(e_i)$, it is split. In addition to edge splitting and collapsing, RAR attempts to optimize vertex valence through edge flipping. Valence 6 for interior vertices and 4 for boundary vertices are considered optimal. Furthermore, RAR translates each vertex v_i to its new position. The new position is computed using the weighted average c_i of its one-ring neighborhood with net weight $\sum_{j=1}^{N} w_j$ with N vertices projected onto the plane (v_i, n_i) as

$$c_i = \frac{\sum_{j=1}^{N} w_j v_j}{\sum_{j=1}^{N} w_j}, v_i \leftarrow c_i + nn^T(v_i - c_i). \tag{11}$$

RAR is efficient and can be used in real-time applications; however, it fails to handle raw and defective input meshes, such as molecular surface meshes. Another drawback of RAR is that the approximation error tolerance ε is not necessarily satisfied due to the discrete nature of curvature and computation of the sizing field. Furthermore, it has minimal improvement in angle quality.

Recently, Cheng *et al.* [94] used RAR [59] as part of their method for error-bounded remeshing. They proposed a robust and efficient method that can remesh an input model within a given error bound. They added new vertices during remeshing to remesh the input while satisfying the input approximation. Although they achieved robustness, efficiency, and error control, there are still a few open challenges for the future. First, they added new vertices for error bounding, increasing the mesh complexity. Secondly, like RAR, it has no significant improvement in mesh quality, as the minimal angle is still less than 20^o. Thirdly, the method [94] is limited to isotropic remeshing and has no theoretical proof; therefore, extension to anisotropic meshes and theory-based study are interesting future directions. Lastly, it can also be extended to compatible and all-hex meshing.

Recently, Khan *et al.* [1] proposed a method that begins with an initialization using RAR [59], followed by aspect ratio enhancement and a cut-and-fill module to eliminate invalid regions and small triangles. These selected regions are further improved with edge splitting, edge collapsing, and vertex translation. This method is capable of generating a good-quality mesh with an angle bound of $[30^o, 120^o]$. However, its efficiency and robustness still require improvement. Furthermore, minimizing the maximal angle up to 90^o is also a future direction. In addition, it is specific to molecular surface remeshing [1].

Explicit surface remeshing (ESR) [89] is another mesh smoothing method that uses local operators to improve mesh quality. To deal with models with arbitrary genus, this method uses patch-wise parametrization. Unlike other typical remeshing methods, this method avoids global operations and 3D mesh optimization. This method uses local modifications; hence, it is comparatively more efficient than other methods [133], [134]. In addition to efficiency, this method improves the mesh quality in term of the minimal angle and regularity, and it can handle arbitrary complex meshes. However, ESR does not consider maximal angle improvement, and minimal angle is also below the limit of recent methods (i.e., 30^o), with some triangles with angles of $< 10^o$. Furthermore, area-based smoothing methods cannot improve vertex sampling beyond the boundaries defined by feature edges [89]. This can be considered as a future direction. There are two possible improvements. One is to use post-processing, which restores the feature edges after the smoothing process using an existing approach [135]. Another solution is to lock the feature edges during the smoothing process; for example, the line drawing scheme [12] locks the sharp features during remeshing.

Guo *et al.* [70] proposed an efficient algorithm for high-quality remeshing of CAD models. Their main goal was to improve efficiency without using the two approaches in typical remeshing algorithms: parametric-based remeshing and direct remeshing in 3D space. Alternatively, this algorithm [70] uses scalable locally injective mappings (SLIM) parametrization [136] to map different patches of the input mesh into a 2D domain and improves this using constrained DT (CDT). The 2D projection is mapped back to 3D space, and the mesh regularity and angle quality are further improved with modified global isotropic remeshing. The algorithm is robust, with consideration of degeneracies, sharp boundaries, and artifacts. The advantages of the algorithm are simplicity, efficiency, and robustness for high-quality automatic mesh generation. Its limitation is that there is no mechanism to handle models with geometric errors, such as self-intersection. Zangeneh and Ollivier-Gooch presented a method of multi-threading and parallel processing of element-based operators, including vertex insertions and face and edge swapping [114]. The authors concluded that the efficiency can be improved by at least 50% for parallel execution

on 24 threads. However, the study was mostly concerned with efficiency; therefore, there was no significant improvement in mesh quality. The same mechanism can be applied to high-quality surface remeshing, such as with an angle bound of $> 30^o$ or non-obtuse remeshing.

Dassi *et al.* [101] proposed a curvature adaptive mesh optimization method for CAD models. Inspired by a previous work [137], which embedded the surface in higher-dimensional space, the surface is remeshed uniformly. They use surface normal and embed the surface into higher-dimensional space $\mathbb{R}^6$. This directly optimizes the mesh in 3D space and uses the embedding space $\mathbb{R}^6$ for mesh size and quality evaluations. For mesh refinement, this method uses local modification operators, including edge flipping, edge collapsing, and vertex insertion and translation. The method is curvature adaptive with sharp feature preservation and gives nicely shaped meshes. However, this method does not consider the mesh quality (such as the minimal and maximal angles or AR) in $\mathbb{R}^3$. Another improvement in higher dimensional embedding (HDE) is coupling surface reconstruction via radial basis functions [102]. Here, the only input is an initial mesh, taken from discrete surface data sets instead of CAD models, like in the previous method [101]. The mesh improvements are decided based on the target edge length. The improvement is either simplification of input, remeshing, or a fill-in. The method gives an acceptable trade-off between complexity and accuracy. The main limitation of this method is the smoothing of sharp edges for complex inputs [102].

Local operators are efficient and simple to use. They allow different parts/regions of the model to be remeshed independently of the remaining parts/regions. However, it is challenging to ensure that each local operator will not affect the topology and geometry of the input model. Sharp feature preservation is also difficult with local operators (see Figure 11). Additional constraint and segmentation lines [12] can preserve sharp features but degrade the efficiency. Therefore, further research is required to minimize the approximation error and to preserve sharp features with the same level of efficiency, similar to that of RAR [59].

4.3 Remeshing with segmentation

Local operators (Section 4.2) may process some regions in the mesh while keeping other regions unchanged. However, completely independent remeshing cannot be achieved. To deal with this issue, mesh segmentation is used to subdivide the input mesh into segments, where the segments can be remeshed independently and then stitched back together.

There are two main categories of segmentation-based remeshing. One category [45], [95], [118] first defines a coarse mesh (or base mesh) over the input mesh by mesh simplification. Then, the base mesh is mapped back to the original mesh and further subdivided to form a semi-regular output mesh. Another category [12], [67], [125], [138], [139], [140] starts with segmentation of the input mesh. Then, each mesh segment is remeshed individually, and finally, all individual segments are combined. In the following, we summarize some state-of-the-art methods of segmentation-based surface remeshing.

The unified subdivision method [118] approximates an arbitrary surface by a displaced subdivision surface. In addition to surface remeshing, this subdivision scheme can be used in many other applications, including geometry compression, animation, editing, scalability, and adaptive rendering. This scheme is simple and efficient for evaluation of surface properties. However, this method loses sharp features and sometime suffers from distortion error. Mansouri and Ebrahimnezhad [95] presented an alternative method of curvature-adapted subdivision, which achieves lower distortion error and higher AR than the subdivision method [118]. However, the method [95] is based on semi-regular remeshing, which cannot modify the mesh connectivity arbitrarily, causing distortion in highly curved regions.

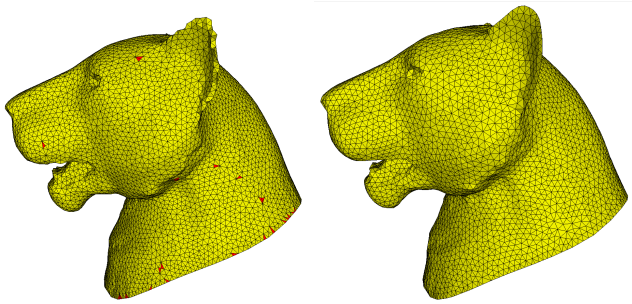


Figure 11. Remeshing of the lion-head model. Left: RAR method [59] (without feature preservation). Right: With sharp feature preservation [12]. Red color shows angles $< 30^o$.

Edwards *et al.* [67] used *variational shape approximation* (VSA) [141] for mesh segmentation and CVT [80] for remeshing. Many other segmentation methods are also used to define the feature skeleton of input meshes, such as IsoChart [142], Exoskeleton [143], and LiveWire [144]. Because most segmentation algorithms use triangles as their basic primitives for clustering, the segmentation boundaries are not smooth enough, especially for inputs with badly-shaped triangles. Such boundaries lead to low triangle quality in the output mesh. Peyre and Cohen [52] proposed a geodesic extension of the Lloyd algorithm for construction of geodesic centroidal tessellation. The method starts with mesh segmentation using centroidal tessellation, followed by refinement. This method solves some existing limitations, including topological errors and remeshing large models. However, the method is slow, which is a possible future improvement.

Harmonic map-based surface remeshing is another technique [99] to obtain a high-quality mesh from an input mesh with arbitrary complexity. It depends on the parametrization of a genus zero surface mesh with a harmonic map. The authors used a cavity check strategy to ensure one-to-one mapping. Their method outputs higher-quality meshes compared to the convex combination map of Floater [145]. Furthermore, the method is easy to implement, robust, and efficient. The main drawback is its applicability to only genus zero surfaces. For arbitrary genus surfaces, it might be extended with further modules.

Recently, Khan *et al.* [12] proposed a segmentation-based method for preservation of sharp features during surface remeshing. This method starts with line drawing (using LiveWire [144]) over the mesh to identify segments. After mesh segmentation, it applies segment-based surface remeshing. The lines are used to preserve sharp features (see Figure 11). This method is able to preserve sharp features, but the sequential order of segment remeshing is not efficient. Parallel remeshing of all the segments could improve the efficiency [12]. Segmentation-based methods are able to remesh large models in a divide-and-conquer manner. This helps to provide a feature skeleton that reduces the approximation error. However, boundaries between the segments are difficult to improve. Moreover, some methods are slow due to segmentation

time [12]. An interesting future work is the parallel remeshing of segments to improve efficiency. We refer the reader to the survey article [21] for more details on mesh segmentation.

4.4 Remeshing with Delaunay triangulation

The Delaunay criterion [147] plays a vital role in mesh processing and computation geometry. This criterion ensures the empty sphere/circle property, which means that the circumscribed circle/sphere of any triangle/tetrahedron in a mesh will have no node inside it (see Figure 12) [15]. Insertion algorithms insert new points into a DT while keeping the Delaunay criterion [148]. Triangle removal is another strategy [106] for constructing DT while handling noise and sharp features. Sink-insertion [122] can be used to improve efficiency while maintaining the mesh quality of DT.

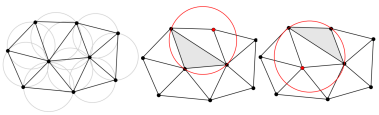


Figure 12. Delaunay criterion. Left side is Delaunay mesh, where circumcircle of each triangle does not contain any node. The middle and right side meshes violate the Delaunay criterion [146].

Chen *et al.* [149] proposed optimal DT (ODT). They use several local mesh smoothing and global mesh optimization schemes based on minimizing energies related to a weighted interpolation error. Although this method can produce well-shaped triangulations with considerably high quality, it still cannot guarantee that all bad triangles are eliminated from the mesh. Furthermore, because it does not modify the local connections, if the initial triangulation is not regular enough, the results will be poor. Similarly, weighted triangulation [150] is another approach; it is equipped with a scalar weight per vertex, and it generalizes the DT to surface meshes. This approach uses the limited memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) method [151] to optimize vertex positions for well-centered triangulation. The approach is efficient, robust, and can generate well-centered meshes, self-supporting surfaces, and sphere packing. However, it is more general to geometric processing rather than specific to remeshing; therefore, it is missing other mesh quality considerations.

Cheng *et al.* [40] proposed another method for the sampling and triangulating smooth surfaces. This method ensures bounded aspect ratio, size optimization, and smoothness of the output mesh. The main advantage of this method is that it does not need to compute local feature sizes to generate sample points, which was a major bottleneck in previous methods [152], [153], [154]. The cost of critical point computations is the limitation of the algorithm. The off-center method [54] is another Delaunay refinement scheme that generates size-optimal meshes with guaranteed quality. It has the same level of guarantee as Ruppert's method [155]. The advantage of the off-center method [54] is that it inserts fewer Steiner points than other methods. For example, for a minimal angle of 20^o, the off-center method inserts 40% fewer Steiner points than circumcenter insertion algorithms [55]. However, this method is limited to the 2D domain only.

Efficient DT [85] is another method for refinement of DT in optimal time. For Steiner points, this method adopts the off-center method [54]. However, this method [85] avoids computing very skinny Delaunay triangles, and instead it uses a scaffold quadtree structure to compute, locate, and insert the off-center Steiner points in an efficient manner. In this way, the time-optimality is enhanced. However, it is still limited to the 2D domain.

Chen *et al.* [10] proposed a GPU-based algorithm for 2D Delaunay refinement. They insert Steiner points as input to a planar straight line graph (PSLG) to generate a constraint DT with minimal angle improvement up to a given threshold. The method is efficient, robust, and controls the degeneracy. However, for some models, such as synthetic data, the speed is not satisfactory. Boubekeur and Schlick previously used the GPU for adaptive mesh refinement [156]. This method uses the CPU for global operations (such as depth tagging for vertex attribute) and the GPU for local fast refinement. Though there is no specific consideration of mesh quality, this method is flexible with access to adaptive level of detail, allows crack-free adaptive multi-resolution mesh refinement, and uses single-pass vertex tessellation. Therefore, the method [156] can be used to achieve efficiency in recent high-quality meshing algorithms.

Sparse Voronoi refinement (SVR) [123] is another work that maintains the Delaunay criterion and an optimal radius-edge ratio throughout the remeshing process. SVR keeps the degree of most vertices constant, thereby achieving a constant processing time. However, SVR is still slow because it works sequentially. This issue has been solved by another variation of DT refinement called parallel SVR [9], which is an extension SVR [123]. Parallel SVR [9] improves efficiency by parallel instead of sequential processing. Parallel SVR also considers feature preservation. However, there is no specific attempt towards mesh quality improvement, such as minimal/maximal angle, and valence optimization.

Ma *et al.* [68] used the uniformization concept via dynamic discrete Yamabe flow and Delaunay refinement that can handle complex models without any partitioning. This method converts the 3D surface into a 2D plane, making it independent of dimensions. Furthermore, it is robust with a theoretical proof. Experimental results revealed a significant improvement in mesh quality and a minimal angle above 30^o. However, there is no consideration of the maximal angle and sharp features. A similar parametrization-based method that also uses DT for mesh refinement was recently proposed by Su et al. [107].

Similarly, maximal Poisson disk sampling (MPS) [72] was used for surface remeshing generalized from a 2D algorithm [157]. This study [72] considered maximal Poisson-disk sets with variable radii and their geometric analysis. An algorithm was proposed for gap detection considering dynamic updates with changes in radii or disk insertion/deletion/movement. Gaps were updated in the context of regular triangulations and the power diagram [72]. The results reveal that the mesh quality (2D mesh and surface mesh) is improved in terms of minimal angles, maximal angles, vertices regularity, triangle quality, and other quality metrics. The main limitations of the method are that it fails to deal with sharp features and self-intersections. These issues were addressed in a later study [71]. Yan *et al.* [78] generalized the MPS method to unbiased isosurface sampling and meshing. However, it is not efficient and fails with thin features. Guo *et al.* [71] used MPS for surface remeshing to improve quality with consideration of efficiency and memory usage. Unlike previous methods [72], this one can handle sharp features and self-intersections. However, the input model must have connectivity information, as it fails to handle triangle soups and noisy data. This is an open future challenge.

Boltcheva *et al.* [119] proposed a feature-preserved DT method for multi-material 3D segmented images. It detects corners and edges in input images. These corners and edges are used as constraints

for DT in the next step. The method is efficient enough to be used for realistic visualization. However, the method is limited to meshes generated from segmented images, and extending it to generic surface remeshing is a future consideration. DT also works for compatible meshing [90], [91].

DT-based methods are easy to implement with a strong theoretical baseline. They have been used in many surface remeshing methods. However, in some cases, their efficiency is not sufficient, while in other cases, DT methods fail to preserve sharp features. Furthermore, these methods often fail with noisy data and defective inputs or require special care to handle such defects. A number of methods are available for significant improvement in DT given additional special care. For example, MPS [71] can be improved in efficiency and memory management, whereas suboptimal DT [92] can preserve sharp features. The GPU implementation of DT [10] is a recent attempt towards efficiency enhancement. Efficient DT [85] is in an optimal method for 2D DT that can be extended to surface remeshing.

4.5 Remeshing with advancing-front

Figure 13 illustrates the process of mesh generation with the advancing-front method. This method starts with an un-meshed domain front, meshing it element-by-element, and connecting new elements with a mesh in a sequential order. The process is repeated until all elements are connected with the mesh [158]. This method was introduced for the first time in 1971 [159]. Researchers have used the advancing-front method for different mesh processing objectives. It uses the metric map to handle changes in parametrization of the surface [160].

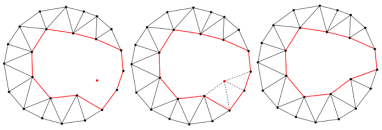


Figure 13. Advancing-front strategy. Left: Placing a new node. Middle: Connecting the new node with the front. Right: Adapting the front [146].

Schreiner *et al.* [100] used the advancing-front method for a complete mesh as well as to remesh some portions locally. Their method is efficient and suitable for real-time applications. However, the approximation error for some meshes is higher than the theoretical bound. Similarly, other methods [103], [161], [162], [163] have also used advancing-front for surface remeshing. The advancing-front method is comparatively more robust in extremely anisotropic elements, such as in fluid mechanics applications [148].

Dietrich *et al.* [38] proposed a method called marching cubes using edge transformations (Macet). Macet eliminates poor-quality triangles by edge transformation in MC [164]. The edge transformation is performed such that the geometry of the input mesh is preserved. Macet is simple, fast, open source, and able to eliminate poor-quality elements. The main limitation of Macet is that it is not applicable to sharp corners and adaptive versions of MC [165], [166], which is highlighted as a future research direction.

The loss of important details in surface mesh generation is another issue with previous methods, such as marching cube (MC) [164]. Schreiner *et al.* [103] proposed an algorithm to solve this issue. They generated triangular surface meshes from regular and irregular volumetric data in an advancing-front manner. They extended a previous advancing-front method [100] to design a new algorithm for isosurface generation. Triangle quality, adaptivity, fidelity, and direct downstream applicability of the mesh were the key meshing objectives. Generalization for processing gigantic data and to give triangle shape guarantees for the meshes generated by the algorithm [103] are the highlighted future directions.

Liu *et al.* [39] proposed a new method for triangulation of implicit surfaces. The algorithm starts by uniformly sampling the implicit surface, followed by reconstruction of triangular meshes from the sample points using the ball pivoting algorithm (BPA). A *1-to-4* subdivision approach is then applied to obtain a high-quality implicit surface mesh. The main advantages of this method [39] are as follows: (1) The method works for algebraic surfaces and dynamic implicit surfaces. (2) The generated triangles are close to equilateral triangles. (3) A user-defined resolution is used to approximate the implicit surface. (4) The LOD representation of an implicit surface is taken automatically. The experimental results are shown with simple models, and it is not clear how this method will work with complex models and sharp edges.

In 2013, Lo [79] proposed a dynamic grid scheme for adaptive meshing with the advancing-front method. This method [79] reduces search time over the generation front by partitioning the domain. Experimental results revealed that the efficiency can be improved fivefold with the use of a partition grid compared to other conventional methods (those without a background grid). The limitation of this method is that it is limited to the 2D domain; therefore, it is a future direction to extend it to surface meshes and higher dimensions. The main focus is on efficiency, and the mesh quality has not yet been properly considered.

The advancing-front method is robust and able to generate well-shaped and geometry-aware meshes of different types including triangular, quadrilateral, tetrahedral, and hexahedral. However, the main problem with these methods is low efficiency. Specifically, handling a large mesh and ensuring avoidance of self-intersection are time-consuming [79]. Local region processing [100] can improve the efficiency, but it fails to control the approximation errors. An extension of Macet [38] to handle sharp features and adaptive density [165], [166] can be considered as a future work. In summary, the advancing-front method is good for mesh generation. However, for surface remeshing, it fails to find an optimal balance between efficiency and approximation error.

4.6 Laplacian smoothing

Laplacian smoothing [167], [168], [169] is a well-known and the simplest method for mesh smoothing; it moves each vertex to the central position of its neighbors. A simple example is shown in Figure 14. Here, the new position p_i of a free vertex v_i is calculated as the mean of all of its neighbors' positions q_1, q_2, q_3, ..., q_n:

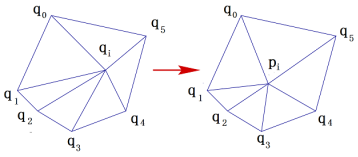


Figure 14. Example of Laplacian smoothing [110].

$$p_i = \frac{1}{n}\sum_{j=0}^{n-1} q_j, \tag{12}$$

where n is the number of vertices adjacent to v_i.

Although the Laplacian method is attractive for its simplicity, it is not guaranteed to converge, and often only the first few sweeps are beneficial. Furthermore, it can produce inverted elements (i.e., negatively signed area for triangles) [167]. Some of these issues have been addressed in later research on Laplacian smoothing.

Another variation of Laplacian smoothing uses weighted Laplacian and discrete mean curvature normal [75]. The weighted Laplacian [75] is an efficient method that improves mesh quality with feature preservations. However, the weight and feature selection are set manually. Automatic feature selection could further improve its efficiency.

Vollmer *et al.* [110] addressed the issue of deformation and shrinkage. Unlike the Laplacian method [167], this method [110] has a better preservation of shape and size of the input model. The algorithm [110] is faster than the Laplacian method while achieving the same degree of smoothing. It can be applied to noisy point sets or meshes. However, it cannot significantly improve mesh quality. Another aspect is GPU implementation [111] to improve the efficiency of the Laplacian method.

Similarly, Liu *et al.* [112] proposed a global optimization scheme that works with both a global Laplacian operator for mesh fairing and constraints to ensure mesh fidelity. The target applications for this global optimization are noisy mesh smoothing, improving a simplified mesh, and subdivision connectivity-based geometric modeling. Although global optimization is slower, this method is still efficient in that it is non-iterative. It smooths the mesh without any shrinkage or distortion. However, there is neither any quantitative improvements for mesh quality nor thresholds (such as minimal and maximal angle or error-bound) to ensure mesh quality. Furthermore, Chen *et al.* [98] proposed an improved version of the Laplacian algorithm. This method checks feature nodes, and the mesh is optimized with a quadratic penalty scheme that handles the issues of shrinkage and inverted elements. However, this method fails for complex surface meshes. Furthermore, there is very little improvement in mesh quality.

ISO2mesh [62] is a free MATLAB/Octave-based toolbox used for mesh generation and processing. It is used to create a tetrahedral mesh from a surface mesh, 3D binary, and gray-scale volumetric images, such as segmented MRI/CT scans. ISO2mesh is also used for mesh smoothing [1], [63], but it fails to handle self-intersecting triangle pairs and small-angle triangles. SMOPT [63] is another improvement of Laplacian smoothing that improves the mesh quality with elimination of redundant vertices and self-intersecting faces from raw molecular surface meshes generated by TMSmesh [170]. The improved Laplacian smoothing used by SMOPT is given in Equation (13).

$$p' = (1-\beta)q_i + \frac{\beta}{N_i}\sum_{j=1}^{N_i} q_j, \tag{13}$$

where $\beta \in [0,\ 1]$ is a metric to control the rate of smoothing, p' is the new position of a vertex, N_i is the number of vertices in one ring, and q_j represents the position of the j^{th} adjacent vertex in the one-ring of the i^{th} vertex. SMOPT's results show a significant improvement in mesh quality. There still exist very small angles that destroy the quality of triangles. The main drawback of SMOPT is that the mesh quality, including the minimal and maximal angles and other quality metrics, are still very low. For example, in some cases, the minimal angle is less than 1^o.

Laplacian-based methods are simple and easy to implement. These methods give good results with few iterations. However, higher approximation error has been reported, especially in the case of more iterations, and sometimes inverted elements are introduced [167]. Topology distortion, deformation, and shrinkage are its other limitations. Similarly, feature preservation also requires special care [75].

4.7 Remeshing with optimization

Optimization-based methods are available with either local operations [59], [135], [171], [172] or global energy minimization. The global optimization approaches can be further classified as parametrization-based [76], [173], [174], [175], [176], [177], discrete clustering [33], and direct 3D optimization [30], [66], [80], [81], [86], [88], [178], [179].

The Taubin method [180] uses a signal processing approach that combines two Laplace-like filters, one with positive weight and another with negative weight. It finds new position p' for each vertex calculated from previous position p in (14):

$$p' = p + \lambda \sum_{j=1}^{n} \omega(q_j - p), \tag{14}$$

where ω is the weighting factor, which is commonly given as $\omega = \frac{1}{n} \cdot \lambda$. The weighting factor is dynamically replaced with another factor $\mu = -(\lambda + \varepsilon)$, where $\varepsilon = 0.02$ is a small value. ε is used to set the value of μ to be slightly smaller than $-\lambda$. λ and μ are alternatively applied for backwards translation [180]. The Taubin method [180] provides a significant improvement, even for complex meshes. However, there is no specific threshold for mesh quality, such as minimal and maximal angles. Xing *et al.* [124] introduced a semi-automatic algorithm, where users set the front-view of the input model manually, which is used for mesh extraction by gaining five more orientations automatically. They used parallel remeshing to achieve efficiency and extract perceptual features (from the input model's image) to control the geometric error. One possible future direction is to fully automate this algorithm [124].

Hierarchical Poisson Disk sampling multi-triangulation (HPDS-MT) [36] combines HPDS and MT. The key objective is to generate a high-quality adaptive mesh. High accuracy, theoretical guarantee, and topology preservation are the advantages of HPDS-MT. Its results [36] show a minimal angle of $> 30^o$ and maximal angle of $< 120^o$. The mesh's geometry is preserved by approximating the distribution of the Poisson disk radii over the surface. This method [36] works for meshes with an empty boundary, and representation of a surface with a boundary is a future work.

ODT [181] minimizes the interpolation error of DT with the same number of vertices. Chen and Xu introduced ODT in a previous research [182]. Their method [181] improves DT by considering interpolation error as mesh quality, and it translates vertices to their new positions to reduce interpolation error. Several mesh smoothing schemes have been used in ODT, including Laplacian smoothing as a special case. ODT is applicable to both isotropic and anisotropic cases. ODT gives better results than Laplacian and CVT in terms of efficiency, triangulation shape, and interpolation error. ODT improves the quality dramatically in a simple and efficient manner. However, if the initial triangulation is irregular, it will give poor results as there is no mechanism for modification of the local connections. The method is also limited to the 2D domain, but it was extended to surface meshing in later research [88], [149].

Chen and Holst proposed various smoothing approaches (local and global) with ODT and centroidal patch triangulation (CPT), applicable in both isotropic and anisotropic cases [149]. CPT is triangulation in which each vertex is the centroid of a patch with respect to the assigned density. This method [149] has similar remeshing objectives to a previous 2D method [181], with a few improvements, including non-uniform density function, avoiding

degeneration of elements near the boundary, global mesh optimization, and 3D numerical results. The authors plan to combine 3D mesh optimization and surface mesh optimization in future.

Chen *et al.* [88] generalized the CPT method to surface meshes. They proposed a parametrization-free method for constrained centroidal Delaunay meshing. They used a vertex relocation strategy, which computes a new position for each vertex as the centroid of the 1-ring neighborhood. They generalized CPT [149] from the 2D domain to surface meshes. This method is efficient in that it is parametrization-free and avoids geodesic information. The output mesh is well-shaped and of considerable quality. However, further improvement in the mesh quality is desired; for example, the minimal angle is still $< 20^o$ in some cases. Furthermore, there is no theoretical proof of the optimization scheme.

Ruiz-Girones *et al.* [73] proposed an algorithm to optimize a curved high-order mesh with optimal balance between mesh quality and geometric approximation. This method [73] combines two objective functions: element quality and L_2-disparity. These objective functions are optimized with a penalty method to achieve the desired balance between mesh quality and geometric approximation. The final mesh is non-interpolative, making it more accurate. The main drawback is that it generates ill-conditioned Hessian matrices when the penalty parameter increases, which may reduce the mesh quality. Farthest-point optimization (FPO) [183] uniformly distributes the points in the plane with optimal blue noise properties. *Yan et al.* [77] generalized FPO to non-uniform sampling and blue-noise remeshing. Non-uniform FPO [77] is a simple yet efficient method. However, it fails with noisy meshes.

A point insertion method (PIM) [5] was recently proposed for obtuse triangle removal and minimal angle improvement. PIM improves the surface mesh to an angle bound of $[30^o, 90^o]$ while also improving quality measurements such as efficiency, valence optimization, and feature preservation. It splits the long edge of obtuse angles and then flips an edge near the new vertex (the vertex created with the edge split) for valence optimization. Later, smoothing and other operations are also applied for further quality improvement. Figure 19 compares different methods, demonstrating that PIM is the only method to eliminate all obtuse angles from the input mesh. It can handle complex models and near sharp edges that other previous methods [30] fail to remesh. However, for more complex meshes, such as molecular meshes [1], the PIM method [5] may not succeed to eliminate all obtuse angles. Therefore, non-obtuse remeshing for molecular meshes is an interesting future direction. Furthermore, GPU implementation [184] and an extension to anisotropic remeshing are also interesting directions for future research.

Field-based methods: The field-aligned method starts with designing a frame field to guide the interior and boundary layout of the parametrization. Mesh generation and parametrization are consecutively performed in alignment with the given frame field [185]. In this context, Jakob *et al.* [97] proposed a field-aligned meshing method for isotropic triangular/quad-dominant remeshing with a local smoothing operator and sharp features control. However, the approximation error and element quality saw no significant improvements. Lai *et al.* [127] proposed an automatic N-RoSy field design scheme for arbitrary surfaces. The method [127] gives the user full control over the singularities (with global constraints) and the iterations. This method reduces the complexity of the design via the flat cone metric. It aims to simplify the Riemannian metric. The main drawback is that the computation of the flat cone metric is based on Ricci flow. Furthermore, it is a nonlinear method, making it slower than linear methods.

In addition to previous studies [97], [127], [185], a recent article [186] proposed a hex-dominant mesh generation with a field-aligned strategy. Field-aligned methods can generate high-quality isotropic and anisotropic tetrahedral meshes [187]. These methods are closely related to anisotropic remeshing. Some methods [82], [83] focus on Riemannian distance, while others [60], [97] focus on the control of direction.

Optimization-based methods attempt to minimize the energy and provide an optimal balance between interpolation error, mesh complexity, and other quality metrics. ODT [181] and its improved version [149] are among the significant methods for mesh optimization. The error-bounded meshing proposed by Hu *et al.* [11] gives better results by providing an optimal balance between geometric error, mesh complexity, and element quality. Similarly, among recent studies, non-obtuse remeshing methods [5], [30] also give significant improvements in mesh quality. The generalization of these methods to complex and defective meshes, such as molecular surface meshes, is still challenging and may be a future research direction.

4.8 Centroidal Voronoi tessellation (CVT)

CVT [132] makes a great contribution to surface remeshing. The Voronoi diagram (Voronoi tessellation) and DT are very closely related (see Figure 15); a Voronoi tessellation is called CVT when the generating point of each Voronoi cell is also its mean (center of mass). For uniform CVT, the density $\rho(x)$ is kept constant [132]. Figure 16 shows CVT from 10 random points. CVT of n distinct vertices (seeds) $V = \{v_i\}_{i=1}^n$ in R^3 is a special type of Voronoi diagram (VD) that minimizes the following energy function:

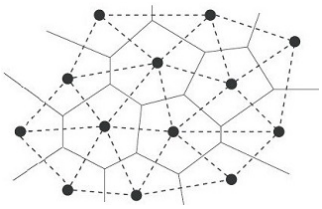


Figure 15. DT (dotted lines) and corresponding CVT (solid lines) [188].

$$F_{CVT}(V) = \sum_{i=1}^{n} \int_{\Omega} \rho(v) \|v - v_i\|^2 d\sigma, \tag{15}$$

where $\Omega_i = \{v \in R^3 \,|\, \|v - v_i\| \leq \|v - v_j\|, \forall j \neq i \}$ is the Voronoi cell of vertex v_i, and $\rho(v) \geq 0$ is a density function defined over the domain. In CVT, each vertex v_i coincides with v_i^*, the mass center of its Voronoi region Ω_i, which is defined as

$$v_i^* = \frac{\int_{\Omega_i} \rho(v) v d\sigma}{\int_{\Omega_i} \rho(v) d\sigma}. \tag{16}$$

The Lloyd method [189] and quasi-Newton-like solver optimization [31] are the most popular implementations of CVT. The Lloyd method [189] is very straightforward; it moves each vertex to the centroid of the corresponding Voronoi cell. Newton optimization [31] uses a quasi-Newton-like solver, such as L-BFGS [151], [190]. The Lloyd method [189] is slower, with linear convergence, compared to Newton optimization [31], with quadratic convergence. Figure 17 shows an analysis of the rates of convergence of the Lloyd and L-BFGS (Newton optimization) methods. Although CVT is a well-known method, it is still not sufficient for mesh quality improvement such as to eliminate all obtuse angles, especially in adaptive density meshes.

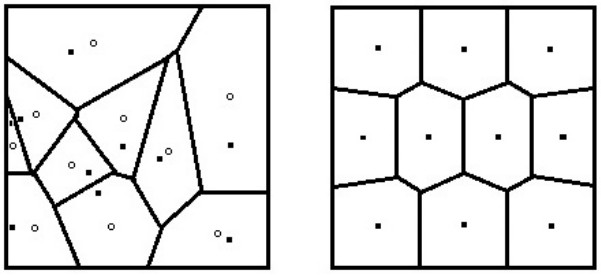


Figure 16. CVT example. Left: VD with 10 random points. The dots are the Voronoi generators, and the circles are the centroids of the Voronoi cells. Right: Corresponding CVT [132].

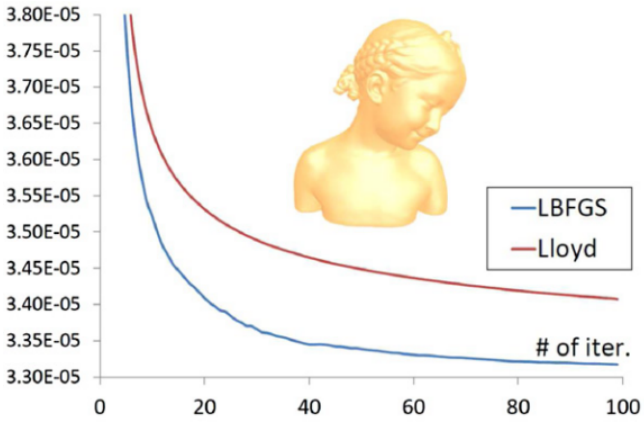


Figure 17. Comparison of rate of convergence between Lloyd [189] and L-BFGS (Newton optimization) [31] methods: Bimba model with 5000 sites [66].

CVT extensions: A number of methods have used/improved the CVT energy computation for various surface remeshing methods, and a few of them are discussed here. Yan *et al.* [30] improved the CVT energy function by adding a penalty term to avoid short edges that cause obtuse triangles (see Figure 18). In this way, they proposed a new CVT-based method for quality surface remeshing, avoiding small and obtuse angles. The improved energy function is given in Equations (17) and (18). The angle bound is $[30^o, 90^o]$, and the vertex regularity for $V_{567\%}$ is near to 100. However, it fails with models having sharp edges or noise. It also fails to eliminate all obtuse triangles in complex models. A recent method [5] that can be used directly or as post-processing with other methods (such as CVT) solves these limitations (as described in Section 4.7). We have

$$F(V) = F_{CVT}(V) + \lambda R(V), \tag{17}$$

where

$$R(V) = \sum_{i=1}^{n} || \sum_{(v_i,v_j)\in RDT} w_{i,j}(v_i - v_j)||^2 \tag{18}$$

is the penalty term used to avoid short Voronoi edges. Furthermore, $w_{i,j} = \frac{||(v_i,v_j)||}{||dual(v_i,v_j)||+\varepsilon}$, where ε is a small value for avoiding a zero denominator. Here, (v_i, v_j) is a triangle edge, and $dual(v_i, v_j)$ is its dual (i.e., Voronoi edge) [30]. This method gives a significant improvement in the mesh quality, but for complex models and sharp features, the results are not satisfactory (see Figure 19).

Du *et al.* [191] proposed constraint centroidal Voronoi tessellation (CCVT) on a surface, which is given as

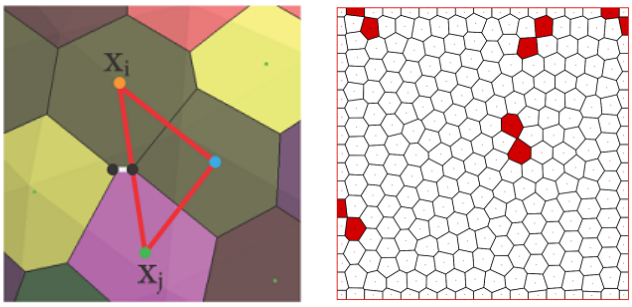


Figure 18. Illustration of short edges and obtuse angles. Left: Short Voronoi edge (white color) and its relation with an obtuse triangle (red color). The largest edge of the obtuse triangle is dual to the short Voronoi edge [30]. Right: CVT optimized VD and short Voronoi edges. Red color shows the Voronoi cells incident to an edge shorter than 5% of the given threshold length of a Voronoi edge [121].

$$F_{CCVT}(V) = \sum_{i=1}^{n} \int_{R_i} \rho(v)\|v - v_i\|^2 d\sigma. \tag{19}$$

In CCVT, each vertex $v_i \in M$ is the constrained centroid of R_i, which is defined as

$$v_i^* = argmin_{y\in M} \int_{R_i} \rho(v)\|y - v\|^2 d\sigma. \tag{20}$$

CCVT performs better than other previous methods [192]; however, the local minimum problem of the energy function is an open challenge. Similar to a previous CCVT method [191], Nguyen *et al.* [34] proposed two CVT variants for a general planar domain. They use an extended domain Ω_ε instead of the domain Ω itself for computing the centroids. The extended domain Ω_ε avoids centroids to be located near the boundary. Furthermore, their study [34] presented several quality metrics for uniform grids. However, the study was limited to the planar domain only.

Yan *et al.* [80] presented another CVT-based method for computing the exact restricted VD by using an efficient quasi-Newton method. This method is efficient, robust, and can remesh large models. However, the computation of an exact restricted VD in each iteration is time consuming. Another limitation is the lack of termination guarantees for more complex models. Recently, Du *et al.* [60] further extended their model to CVT for field-aligned surface remeshing (see Equations (21) and (22)). We have

$$F_2(V) = F_{CVT}(V) + \lambda E_2(V), \tag{21}$$

$$E_2(V) = \sum_{i=1}^{n} \sum_{j\in N(i)} w_{ij} \cdot D_{ij}, \tag{22}$$

where D_{ij} represents directional distance of edge e_{ij}, $N(i)$ represents the set of its adjacent points, and w_{ij} represents its weight. This method shows a significant improvement in meshing quality over CVT and other state-of-the-art methods. However, due to the conflict of density function and field alignment constraints, the adaptive remeshing results are inferior to those of CVT [80]. Field interpolation is also time-consuming in this method.

Khan *et al.* [3] applied the point insertion method to the local region around each obtuse angle to remove obtuse angles and then applied CVT and other smoothing operations to achieve a high-quality mesh with an angle bound of $[30^o, 90^o]$. However,

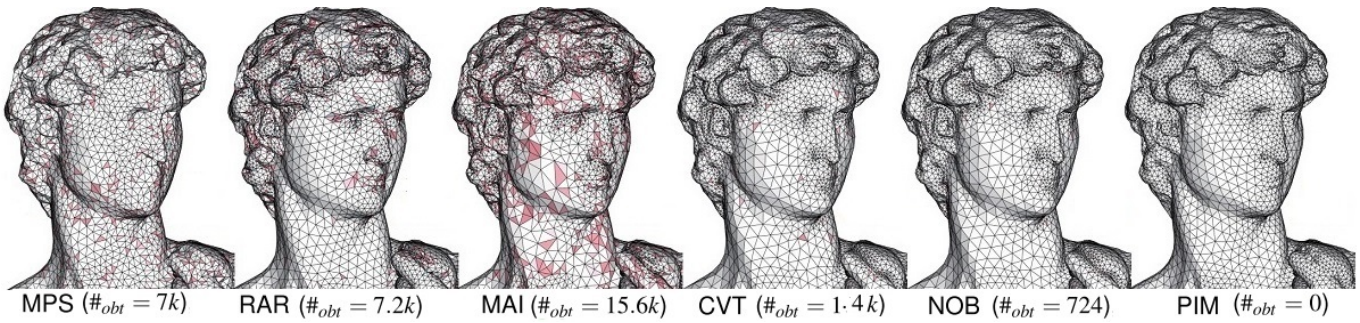


Figure 19. Comparison of different remeshing approaches. The input David's head model (60k vertices and 120k triangles) is remeshed with approximately 60k faces using each method. From left to right are the results of MPS [72], RAR [59], minimal angle improvement (MAI) [11], CVT [80], non-obtuse remeshing (NOB) [30], and PIM by Wang *et al.* [5]. Note that PIM [5] is the only one that completely removed the obtuse triangles colored in pink. Here, $\#_{obt}$ indicates the number of obtuse triangles after remeshing. This figure is taken from the article [172].

this method is limited to 2D mesh generation. Lp CVT [96] is a feature-sensitive CVT that injects normal anisotropy into the CVT energy function. Lp CVT is a generalization of CVT to anisotropic and quad/hex-dominant meshing. The method is able to preserve sharp features during remeshing. However, it fails in complex and defective models. For example, it misses triangles when there are gaps in the input model.

Edwards *et al.* [67] proposed another variation of CVT called k-CVT, which divides the input mesh into segments for segment-wise remeshing. The method is independent of local feature size and can achieve topological correctness in all flat regions. Although this method outperforms several CVT methods, especially in terms of error control and high-quality remeshing with fewer points, the mesh quality is still very low. For example, in some models, the minimal angle is below 10^o. Feature preservation is also not considered. A recent study [193] extended CVT to a practical and efficient resampling method for point cloud datasets. The method [193] is robust and efficient; however, it fails to preserve sharp features. Alliez *et al.* [120] proposed a method for isotropic remeshing. This method specifies a density function over the surface, which is partitioned into samples of the same or near to the same density. It uses weighted CVT [132], where the calculated density is used as weighting. The main limitation of this method is the sampling of $genus > 0$ surfaces. This issue can be solved by avoiding the global parametrization in [174].

Similarly, You *et al.* [69] proposed an adaptive meshing framework that generates meshes for heterogeneous materials. It uses a specific density function depending on the heterogeneity of material and material variations. Finally, CVT and mesh subdivision are applied. Experimental results show that this method is able to deal with complex heterogeneous materials models, minimize mesh complexity, and generate high-quality meshes for heterogeneous materials by maintaining a balance between accuracy and mesh complexity. However, it is limited to only 2D heterogeneous objects; it can be extended to 3D in the future.

Rong *et al.* [113] extended CVT from Euclidean space to spherical and hyperbolic space. They defined CVT energy functions in these spaces and provided their theoretical proofs. The study [113] revealed the convergence of the Lloyd algorithm in spherical and hyperbolic spaces. The results show that spherical and hyperbolic CVTs have the same properties as Euclidean CVT. Furthermore, it combines spherical and hyperbolic CVTs into a single framework and applied uniform partitions and achieved high-quality remeshing for surfaces with arbitrary genus. However, the method is slow because it computes the VD on the universal covering space with all points, including initial sites and the points in their neighbor domains. GPU-assisted CVT [61] is another variation that considers efficiency. Nguyen *et al.* [34] conducted a comparison of several CVT-based methods with different quality metrics.

Hu and Zhang [140] used CVT-based hexahedral mesh generation for automatic polycube construction. Similarly, Hu *et al.* [139] used CVT for segmentation-based remeshing of tetrahedral/hexahedral meshes. These methods [139], [140] are used for tetrahedral/hexahedral meshes, and their extension to high-quality surface remeshing is an interesting future direction. The optimized VD [121] is a 2D CVT-based method that removes short edges (which are the main cause of numerical instabilities). It attempts to minimize the distance between the circumcenter and in-center of a triangle. However, the OVD method is not considerably accurate in approximation. CVT has also been generalized for anisotropic meshes, called anisotropic CVT (ACVT) [126], [128], [194].

The two most popular implementations of CVT are the Lloyd method [189] and Newton optimization [31]. Newton optimization is more efficient than the Lloyd method (see Figure 17). The CVT energy function has been extended with additional modules in various surface remeshing methods. There are a number of CVT-based methods [3], [69] in 2D that can be extended to surface meshes. Other methods are limited to specific applications [43], [193] and can be generalized in the future. Efficiency improvement is also a future direction for various CVT implementations [113], [120]. VoroCrust [74] is another recent and powerful implementation for Voronoi meshing that can robustly handle meshes with arbitrary curved boundaries and sharp features. CVT-based methods have also been used in anisotropic meshing [126]. Further research is required for anisotropic CVT to deal with highly anisotropic regions.

Geodesic-based remeshing: Geodesic-based remeshing [52], [66], [86], [144], [195], [196], [197] is also an extension of CVT. This method compute geodesic distance for quality improvement; however, it is slow due to geodesic path computation during the remeshing process.

Wang *et al.* [66] proposed two intrinsic algorithms to compute the center of mass for any geodesic VD efficiently. The first method is based on the Lloyd iterations, where the Riemannian center and center of mass of any GVD are computed and the corresponding point (seed/generator) is translated. The second is the L-BFGS method, which improves the efficiency of intrinsic CVT (see Figure 17). Unlike previous CVT methods, this method [66] is independent from the embedding space and does not require global parametrization; therefore, it can work with complex models.

However, there is no significant improvement in quality. For example, the minimal angle is smaller than 30^o.

Fu and Zhou [86] extended the 2D Poisson disk sampling method [198] for surface remeshing with consideration of geodesic distance as a distance metric and the Lloyd relaxation for smoothing sampling points. The method gave a significant improvement in the minimal angle and vertex regularity. However, due to geodesic isoline computation, this method is not efficient. Liu *et al.* [81] used manifold differential evolution (MDE) for computing globally optimal geodesic CVT. However, the method is slow due to geodesic computation. Further CVT-based methods [33], [65], [84], [199] are discussed in Sections 3 and 4.2.

4.9 Anisotropic surface remeshing

Anisotropic surface remeshing is typically found with different meshing goals from isotropic meshes. However, in some cases, similar meshing goals are also achieved via different approaches in both categories. For example, non-obtuse remeshing in both anisotropic [43], [44] and isotropic remeshing [3], [5], [30] have similar goals. Isotropic meshing generally focuses on regular face shape, i.e., equilateral triangles and optimal vertex valence, whereas anisotropic meshing is adaptive to local surface anisotropy [44]. Unlike isotropic Euclidean metrics, anisotropic meshes are typically defined with some anisotropic metrics, such as Riemannian metrics (either specified by users or from surface curvature tensors) [82], [200], [201]. The evaluation of mesh quality in the anisotropic case requires triangles to be transformed back to Euclidean space by the local Riemannian metric at each triangle. The anisotropy of these meshes is typically related to the optimal approximation of a given surface [202], [203]. Anisotropic remeshing is related to field-aligned remeshing, but they are not exactly the same. Field-alignment includes the alignment of both Riemannian distances and directions. The control of both Riemannian distances and directions can provide better quality for anisotropic triangular meshing [187].

In its early years, local connectivity and optimization of the vertices [204] were used for anisotropic meshing, but these methods were not efficient. In later years, anisotropic meshing was given its own implementations of well-known methods, such as CVT [126], particle-based optimization [83], DT [128], [197], [205], ODT [129], and discrete VD [33], [41]. ACVT [126] is an anisotropic extension of the Lloyd [189] method that uses a given Riemann metric to handle the anisotropy. Further investigation is required in various directions, including efficiency, application to numerical PDEs, and theoretical study of the duality of ACVT and anisotropic DT. Particle-based optimization [83] gives a metric-adapted mesh from a given input mesh and specified number of vertices. Here, the anisotropic space is mapped to an embedding space (i.e., a higher-dimensional isotropic space) [206] and then mapped back to the original space after processing. The processing includes uniform sampling and optimization with the quasi-Newton algorithm [31]. Although it gives better results than other previous methods, its efficiency and approximation still require further improvement. Anisotropic VD [41] is another work as an extension of the weighted VD to ensure high-quality anisotropic meshing.

Anisotropic ODT [82], [129] is a generalization of CVT that employs first-type Bregman diagrams [207]. It refines anisotropic meshes with controllable density and small approximation error. However, this anisotropic ODT [129] fails to handle the possible conflict between anisotropy and local geometry, i.e., to generate a well-adapted mesh with anisotropy. Fu *et al.* [82] previously addressed this issue by preventing further refinement if the edge length in the local metric of the anisotropic meshes becomes too small. However, this cannot conform to the Hessian matrix of the input function. Furthermore, even if this method [82] is efficient and can be bound to geometric error when the anisotropy is derived from the curvature, it still fails to achieve the desired geometric approximation if the anisotropy is otherwise specified (i.e., not related to the domain's curvature).

Furthermore, Delaunay refinements [128], [200] utilize local vertices' connectivity, which is derived from the Delaunay criterion. However, this requires a significant number of refinement operations. Anisotropic DT [128] is a simple and generic algorithm that can handle various types of meshes with complex shapes and anisotropic metric fields. However, it may fail to preserve sharp features. Zhong *et al.* [76] used a global conformal parametrization on a surface with simple topology, followed by remeshing with CVT in 2D. This is an extension of isotropic CVT [113], [208] to ACVT. The CVT is computed in a 2D parametric domain, which makes it efficient. However, this method may lead to an illegal triangulation if the input mesh density is not sufficient to catchup the desired anisotropic metric. This causes failure of the conformal embedding formulation.

A *hexagonal Minkowski metric* was used to define an anisotropic CVT, which significantly reduced the number of obtuse triangles [43] . However, it is very slow and cannot eliminate all obtuse triangles. A recent method [44] addressed this issue and can efficiently eliminate all obtuse triangles. It is a generic method, i.e., not specific to a certain type of anisotropic mesh. However, face anisotropy is difficult to maintain and evaluate after applying this method. Perhaps this method [44] works better than all previous methods for angle improvement in anisotropic meshing.

Mapping the input mesh to high-dimensional space [209] is another approach to anisotropic meshing. Recently, Zhong *et al.* [32] proposed an algorithm that computes anisotropic VDs. It computes Euclidean embedding without any self-intersection used for VDs and conducts remeshing with Riemannian metrics. Being the latest work and a pioneering algorithm towards self-intersection-free Euclidean embedding [32], it has several advantages over previous anisotropic meshing methods. Further work could possibly robustly deal with discontinuities in the input metric.

5 Recent Advances in Remeshing Objectives

In this section, we summarize various surface remeshing objectives, recent achievements, and possible further improvements. Figure 20 presents our classification of surface remeshing objectives.

5.1 Angle improvement and non-obtuse remeshing

Concerning its quality improvements, the angles in triangulation have gained attention from researchers [210], [211]. Angle improvement, including maximal and minimal angle improvement, are important mesh quality metrics. It is necessary to eliminate all small and large angles from the input meshes. Typically, researcher uses 30^o as a threshold for the minimal angle and 120^o or 90^o for the maximal angle. Besides the fact that the minimal angle (θ_{min}) and maximal angle (θ_{max}) are closely related in triangulation, the minimal angle has received comparatively more attention in surface remeshing (see Table 1). In this regard, a

Table 2
A brief summary of all articles included after final selection.

Article ID	Method name /short title	Remeshing Challenges	Category	Year
S01 [58]	Surface simplification using Quadric error metric	Fast simplification	Simplification	2003
S02 [117]	Efficient simplifications for generating high quality meshes	Efficient with less memory	Simplification	2009
S03 [53]	Simplification of 3D mesh for level of detail computation	Simplification, efficiency	Simplification	2014
S04 [56]	Constraint re-sampling method	Simplify DT; improve quality; eliminate obtuse triangles	Simplification, DT, Non-obtuse	2017
S05 [87]	Delaunay Mesh Construction	Delaunay Generation	DT, Simplification	2007
S06 [105]	Efficient construction and simplification of Delaunay meshes	Arbitrary mesh simplification	Simplification, DT	2015
S07 [65]	CVT for Uniform Polygonal Mesh Coarsening	Reducing the complexity of the mesh	Simplification, CVT extensions	2004
S08 [48]	Mesh Simplification Using Hausdorff Distance	Simplification with geometric approximation	Simplification	2012
S09 [109]	Out-of-core Framework for QEM-based Mesh Simplification	Simplification of large meshes	Simplification, Segmentation	2015
S10 [108]	Feature preserving mesh simplification using FS metric	Feature sensitive metric	Simplification, Features preservation	2010
S11 [104]	Delaunay mesh simplification with differential evolution	Reduction of approximation error	Simplification, DT	2018
S12 [33]	Remeshing with Metric-Dependent Discrete VD	Complex meshes to an arbitrary number of elements	Simplification, CVT, Isotropic/anisotropic	2008
S13 [84]	Triangular Mesh Optimization for CAD	Optimizing the raw mesh generated from CAD models	Simplification, optimization, CVT	2007
S14 [101]	Curvature-Adapted Remeshing of CAD Surfaces	Features and topology preservation	Local modification, Anisotropic	2014
S15 [102]	HDE with Radial Basis Functions	Optimization between complexity and accuracy	Local modification, Anisotropic	2016
S16 [59]	Real-time adaptive remeshing (RAR)	Real-time remeshing	Local modification	2013
S17 [89]	Explicit Surface Remeshing	Avoiding global operations and optimization	Local modification	2003
S18 [70]	Automatic and high quality CAD meshing	Efficiency, automation, avoiding parametric remeshing	Local modification, CAD meshing	2019
S19 [114]	Thread-parallel mesh improvement	Efficiency and optimal memory usage	Local modification, efficient remeshing	2018
S20 [93]	High quality compatible triangulations	Compatible triangulations of planar polygons	Compatible remeshing, Local operators	2004
S21 [116]	Dynamic Remeshing and Applications	Avoiding global parametrization	Local modification, efficiency	2003
S22 [6]	5-6-7 Remeshing	Regularity improvement	Local operators, valence optimization	2012
S23 [1]	Molecular surface remeshing with local regions refinement	Angle improvements in molecular meshes	Local modification, Molecular surfaces	2018
S24 [94]	Practical error-bounded remeshing by adaptive refinement	Efficiency and error-bound	Local operators, features and error-bound	2019
S25 [118]	Displaced Subdivision Surfaces	Mesh subdivision	Segmentation-based remeshing	2000
S26 [95]	Segmentation-based semi-regular remeshing	distortion error control	Segmentation-based remeshing	2016
S27 [52]	Segmentation Geodesic Centroidal Tessellation	Large models, topological error	CVT, Segmentation, Geodesic	2004
S28 [12]	Remeshing with robust user-guided segmentation	Preservation of sharp features	Segmentation-based remeshing	2018
S29 [99]	High-quality surface remeshing using harmonic maps	Efficiency and robustness	Segmentation	2011
S30 [45]	Feature-preserving semi-regular remeshing	Bypassing parametrization and preserving features	Feature, CVT, segmentation	2011
S31 [125]	A Parallel Remeshing Method	Efficiency	Parallel remeshing, Segmentation	2011
S32 [150]	Weighted Triangulations for Geometry Processing	Well-center triangulation	Weighted Triangulations	2014
S33 [40]	Sampling and Meshing with Guaranteed Topology	Avoiding local feature size computation	DT	2007
S34 [54]	Off-centers Steiner points for DT refinement	Fewer Steiner points	Delaunay insertion	2004
S35 [85]	Time-Optimal Delaunay Refinement	Avoid computing very skinny DT	Delaunay insertion	2005
S36 [10]	Delaunay Refinement Using the GPU	GPU implementation of DT	DT, GPU-based remeshing	2017
S37 [9]	Parallel Sparse Voronoi Refinement	Simultaneous processing of the operations in SVR	Delaunay insertion	2007
S38 [68]	Surface remeshing based on uniformization	quality enhancement without partitioning	Delaunay refinement	2017
S39 [72]	Gap Processing for Adaptive MPS	MPS for surface and quality remeshing	MPS, Sampling	2013
S40 [119]	Feature preserving DT from 3D images	DT with sharp features	DT, segmentation	2009
S41 [149]	Efficient mesh optimization with ODT	Efficient optimization	Optimization, DT, Anisotropic/Isotropic	2011
S42 [106]	A simple triangle removal approach	Feature preservation, surface reconstruction	DT, Sharp Feature preservation	2017
S43 [107]	Curvature adaptive surface remeshing	Feature preservation, efficiency	Local modification, DT, Feature preservation	2019
S44 [92]	Feature-preserving mesh smoothing via suboptimal DT	Feature-preserved ODT	ODT, Feature-preservations	2013
S45 [90]	High Quality Compatible Triangulations for 2D Shape Morphing	Improve minimal angle with fewer Steiner points	DT, Compatible meshing	2015
S46 [91]	High quality compatible triangulations and interactive animation	To improve minimal angle with fewer Steiner points	Compatible meshing	2018
S47 [71]	Efficient MPS and Remeshing on surfaces	Memory usage and efficiency in MPS	MPS, Sampling	2015
S48 [115]	Point Cloud Meshing (PCMesh)	Efficiency improvement	2D, PCD, DT	2018
S49 [122]	Sink-insertion for mesh improvement	Efficiency	Delaunay triangulations	2001
S50 [123]	Sparse Voronoi Refinement (SVR)	DT in arbitrary dimension in optimal time	DT	2006
S51 [78]	Unbiased sampling and meshing of Isosurfaces	Generalization of MPS	DT, MPS	2014
S52 [103]	High-Quality Extraction of Isosurfaces	High quality and fidelity	Advancing-front technique	2006
S53 [38]	Marching Cubes using Edge Transformations (Macet)	Poor quality and degenerated triangles of MC	Edge based operation, MC	2009
S54 [39]	High quality triangulation of implicit surfaces	Dynamic implicit surfaces and algebraic surfaces	subdivision	2005
S55 [79]	Dynamic grid for mesh generation by the advancing-front method	Efficiency	Advancing-front	2013
S56 [100]	Direct (re)meshing for efficient surface processing	Avoid parametrization	Advancing-front, surface-reconstruction	2006
S57 [167]	Laplacian smoothing	Mesh smoothing	Laplacian, DT	1988
S58 [110]	Improved Laplacian smoothing for noisy surface meshes	Shrinkage and deformation	Laplacian, DT	1999
S59 [112]	Non-iterative approach for global mesh optimization	Fairness and fidelity	optimization	2007
S60 [98]	Improved Laplacian Smoothing	To avoid shrinkage and inverted elements	Laplacian, Optimization	2007
S61 [75]	Laplacian Mesh Optimization	Feature preserving Laplacian smoothing	Laplacian, Optimization, Features	2006
S62 [111]	3D Laplacian smoothing using the GPU	Improving efficiency with GPU	Laplacian, DT, GPU	2019
S63 [63]	Quality improvement and molecular surface mesh optimization	Improve raw molecular meshes	Molecular surface remeshing	2017
S64 [62]	ISO2mesh	Meshing 3D binary and gray-scale volumetric images	Medical/Molecular meshing	2009
S65 [5]	Remeshing without Large and Small Angles	Obtuse/small angles removal, even in complex models	Optimization, Non-Obtuse	2018
S66 [180]	Taubin method	Surface fairing using signal processing	High quality meshing	1995
S67 [36]	HPDS-MT	Efficiency, quality and fidelity	HPDS, MT	2018
S68 [181]	Optimal Delaunay Triangulation (ODT)	Minimizing interpolation error	Optimization, DT, 2D, Anisotropic/Isotropic	2004
S69 [88]	Constrained Centroidal Delaunay Meshing	Efficient optimization; CPT for surface mesh	Optimization, DT	2012
S70 [73]	High-order meshes with optimal quality and geometric accuracy	Quality and geometric accuracy	Optimization	2016
S71 [11]	Error-Bounded and Feature Preserving Remeshing	Balance among error-bound, quality and complexity	optimization, error-bounded	2016
S72 [124]	Parallel Algorithms for Visual Perception Guided Remeshing	Efficiency, Visual fidelity	Optimization	2014

A brief summary of all articles included after final selection (Table 2 continued).

Article ID	Method name /short title	Remeshing Challenges	Category	Year
S73 [97]	Instant Field-Aligned Meshes	To handle large meshes	Field-Aligned meshes	2015
S74 [127]	Metric-Driven RoSy Field Design and Remeshing	To simplify the Riemannian metric	Field-Aligned meshes, Optimization	2010
S75 [77]	Blue-noise remeshing with farthest point optimization	Generalization of the FPO	Optimization	2014
S76 [189]	Lloyd CVT	CVT implementation	CVT	1982
S77 [31]	Newton optimization for CVT	Quick convergence of the CVT	CVT	2009
S78 [30]	Non-Obtuse Remeshing with CVT	Obtuse removal	Non-Obtuse, CVT extension	2016
S79 [191]	Constrained CVT	centriod out of surface	CVT extensions	2003
S80 [80]	Remeshing with Restricted Voronoi Diagram	Efficient and exact computation of RVD	CVT extensions	2009
S81 [60]	Field-aligned CVT (FCVT)	Field-aligned remeshing	CVT extensions, field-aligned	2018
S82 [3]	High quality 2D non-obtuse remeshing	Elimination all obtuse triangles in 2D	CVT extension, non-obtuse, 2D	2017
S83 [199]	Valence optimization for CVT	Valence optimization	CVT extension	2018
S84 [96]	Lp CVT	CVT Generalization to anisotropic, and Quad/Hex	CVT extension	2010
S85 [67]	k-CVT	Fewer points, avoiding small local feature size	CVT, segmentation, sampling	2012
S86 [120]	CVT for isotropic surface remeshing	user specified density for sampling and weighted CVT	Parameterizations-based, CVT	2005
S87 [69]	Adaptive Meshing for FEA of Heterogeneous Materials	Meshing for heterogeneous materials	heterogeneous materials, 2D, CVT	2015
S88 [113]	CVT in universal covering space of manifold surfaces	CVT for spherical and hyperbolic spaces	CVT extension	2011
S89 [34]	CCVT (modified) and comparison of mesh generators	Avoiding centroid near boundary in CVT	CVT extension	2009
S90 [61]	GPU-assisted CVT	GPU implementation to speed up the CVT	CVT extension, GPU-based meshing	2011
S91 [86]	Direct sampling for high quality remeshing	High quality remeshing	CVT extension, Geodesic	2008
S92 [74]	VoroCrust: Voronoi Meshing Without Clipping	Robust polyhedral meshing	Feature preservation, VD	2019
S93 [121]	Optimized Voronoi Diagram (OVD)	Elimination of short edges	CVT extension, 2D	2010
S94 [66]	Intrinsic Geodesic CVT	Independence from embedding space	CVT extension, Geodesic	2015
S95 [81]	Manifold Differential Evolution (MDE) for Geodesic CVT	Computing globally optimal geodesic	CVT, Geodesic, Optimization	2016
S96 [82]	Anisotropic Simplicial Meshing Using Local Convex Functions	Mapping Riemannian metric locally	Anisotropic meshing	2014
S97 [44]	Non-obtuse Anisotropic Surface Remeshing	Obtuse triangles elimination	Anisotropic meshing	2019
S98 [126]	Anisotropic CVT	Generalization of the standard CVT	Anisotropic remeshing, CVT	2005
S99 [128]	Anisotropic Delaunay meshes of surfaces	DT for Anisotropic meshing	Anisotropic meshing, DT	2015
S100 [129]	Optimal Voronoi tessellations with Hessian-based anisotropy	Anisotropic ODT and its dual of ACVT	Anisotropic meshing, CVT	2016
S101 [76]	Anisotropic surface meshing with conformal embedding	Remeshing via embedding in a 2D parametric domain	Anisotropic mesh, Parametrization, CVT	2014
S102 [83]	Particle-based anisotropic surface meshing	Mapping anisotropic space into a higher dimension space	Anisotropic meshing	2013
S103 [43]	Obtuse triangle suppression in anisotropic meshes	Obtuse removal from anisotropic meshes	Anisotropic, CVT, Optimization	2011
S104 [32]	High-dimensional Euclidean embedding from Riemannian Metric	Avoiding self-intersection	Anisotropic meshing	2018

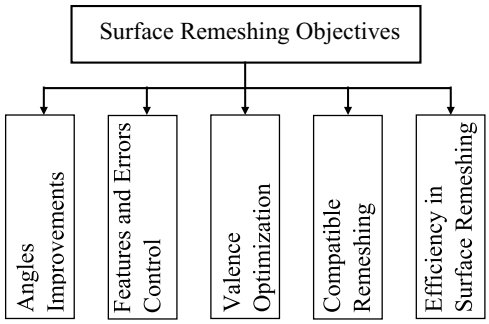


Figure 20. Classification of surface remeshing objectives.

number of methods (such as [1], [11], [12]) should be considered for further improvements of the maximal angle (preferably to an angle bound of $[30^o, 90^o]$) as a future research direction. Angle-bounded remeshing has also received strong attention for quad and tetrahedral meshing [4], [8].

Non-obtuse triangulation, which eliminates all obtuse angle triangles ($\geq 90^o$) from the input mesh, has also gained researchers' attentions in recent years. Non-obtuse remeshing has been used since at least 1988 [212]. There are many works in this field, but we will mention only a few latest methods. Yan *et al.* [30] found that short edges cause obtuse angles in triangulations (see Figure 18 (left)). They added a penalty extension to CVT for elimination of obtuse angles and achieved non-obtuse results for most of the input meshes. However, for some cases, it still failed to eliminate all obtuse angles (e.g., see Figure 19). Khan *et al.* [3] proposed a method of point insertion near obtuse triangles to eliminate them, followed by point removal and selective smoothing operators. This method is efficient and able to generate high-quality well-shaped meshes with an angle bound of $[30^o, 90^o]$. However, it has no theoretical guarantee. In addition, the current implementation is limited to the 2D domain only. A similar method [172] eliminates obtuse angles from a surface mesh. To the best of our knowledge, the most recent method was proposed by Wang *et al.* [5]. This method [5] is similar to the 2D method [5] with further constraints for surface meshes. Figure 19 compares different methods, where only this method can eliminate all obtuse angles from the input mesh. The main drawback of this method is that it fails to eliminate all obtuse angles from complex meshes, such as molecular surface meshing. Therefore, non-obtuse remeshing for complex models, such as molecular meshes, is still s challenge for future research. The two methods [1], [5] can be merged with some special optimization for possible non-obtuse remeshing of molecular surfaces.

5.2 Feature preservation and error control

Feature-preserved remeshing generates an output mesh with minimal distortion in features, such as sharp edges, corners, and geometry of the input mesh. In the following, we describe a few articles in terms of feature preservations and error-bounded surface remeshing.

A number of studies [11], [213], [214], [215], [216], [217], [218] have proposed various remeshing objectives with feature preservation. The simplest approach to preserving remeshing features is automatically predefining feature curves, either by a user or by algorithms (e.g., using dihedral angles) [80], [219]. Such a scheme functions well for models with sharp features, such as CAD models and man-made objects. However, this scheme cannot be applied naturally to free-form objects. Various solutions

Table 3
Advantages and limitations of state-of-the-art methods. *Q.A* score is the quality assessment score calculated for each article based on its number of citations, quality of publication venue, remeshing method, and results (see Section 2.4). The score is calculated based on the relevance to this survey, which means that some good-quality articles might still have lower scores.

Article ID	Advantages	Limitations	*Q.A* score
S01 [58]	Efficient	Consumes more memory	2.0
S02 [117]	Efficient and consumes less memory	Minimal quality considerations	2.0
S03 [53]	Efficient with good approximation of input	Minimal quality considerations	2.0
S04 [56]	Multiple objective at same time: Simplification, quality, obtuse removal	Fails in complex models and sharp features	3.0
S05 [87]	DT for any arbitrary triangulations; simplification to multiple levels of detail	Minimal Quality improvement	3.5
S06 [105]	Fast, simple and easy to use	Complexity is still above $O(n)$	4.0
S07 [65]	Efficient, low memory and easy to implement	Not applicable for sharp features.	3.5
S08 [48]	Geometric approximation	Minimal improvements in angles quality	2.0
S09 [109]	Can simplify large meshes, efficient	Sequential, patches are different in size	2.5
S10 [108]	Feature preservations	Slower than existing method [130]	3.0
S11 [104]	Fully automatic, effective for CAD, better approximation, preserves sharp features	Slow and poor in visual results	4.0
S12 [33]	Can handle complex models with several million triangles	Lower approximation	4.0
S13 [84]	Robust, with feature preservation, and avoids shape deviations	Specific to CAD models	2.5
S14 [101]	Nicely shaped output with topology and features preservations	No quality consideration in 3D space	3.0
S15 [102]	Acceptable trade-off between complexity and accuracy	Destroys sharp features	2.5
S16 [59]	Simple, efficient remeshing in real time	Fails to handle raw and defective input meshes	3.5
S17 [89]	Efficient being using local operators	Mesh quality is still low	3.0
S18 [70]	Fast, robust, high quality	Fails with geometric errors	3.5
S19 [114]	Efficient with optimal memory usage	Improvement in mesh quality is not significant	3.0
S20 [93]	Robust and fast	No specific limit on the number of Steiner points	2.5
S21 [116]	Efficient	Minimal quality considerations	2.5
S22 [6]	Regular meshes with feature preservation	Slow	2.5
S23 [1]	High quality meshing with minimal angle $> 30^o$	Slow; Maximal angle is still not good (120^o)	3.5
S24 [94]	Robust, efficient with error-bound	Increases complexity, minor improvement in quality, no theoretical proof	4.0
S25 [118]	Simple and efficient	Failure in sharp edges; no specific improvement in mesh quality	3.0
S26 [95]	Lower distortion error and higher aspect ratio	fails in highly curved regions	3.0
S27 [52]	Remeshing of complex models, minimize topological error	Low speed performance	2.5
S28 [12]	Robust segmentations avoiding small triangles; sharp feature preservations	Slower due to sequential execution	3.0
S29 [99]	Simple, robust and efficient	Works only on genus zero surfaces	3.0
S30 [45]	AR improvement with preservation of sharp features	Maximal and minimal angles are not considered	2.5
S31 [125]	Efficient, and well-shaped remeshing	Intersections need special care	2.0
S32 [150]	Efficient, robust and can generate well-centered meshes	Very General, not specific to remeshing	3.5
S33 [40]	It avoids to compute local feature size for generating the sample points	The cost of critical point computations.	3.5
S34 [54]	Fast being fewer Steiner points, high quality and size optimality	Only for 2D domain	3.0
S35 [85]	Efficient being avoiding very skinny DT computation	Only for 2D domain	2.5
S36 [10]	Efficient, robust and handle degeneracy	Losses efficiency for some data	2.5
S37 [9]	More efficient than sequential SVR	No quality consideration	2.0
S38 [68]	Efficient, robust, can handle large models without partitioning	Maximal angle and sharp features are not considered	2.5
S39 [72]	First theoretical analysis of MPS for surface and quality remeshing	Fails in sharp edges and complex input	4.0
S40 [119]	Real-time visualization with sharp features	Low quality in sharp edges	2.5
S41 [149]	Efficient, with consideration of connectivity, regularity and density	Fails in complex surface meshes	3.5
S42 [106]	Sharp feature preservation	Parametric algorithm, based on trial and error	2.0
S43 [107]	Better feature preservation and Efficient	Lack of other quality metrics	3.5
S44 [92]	Noise removal, and quality improvement with sharp feature preservations	Fails to improve angle quality in sharp features	3.5
S45 [90]	Robust, efficient, and uses fewer Steiner points to improve minimal angle	Fails with holes and limited to 2D only	2.0
S46 [91]	Simple, robust, works efficiently in real-time	Fails with holes and limited to 2D only	3.5
S47 [71]	Efficient, optimal memory usage, quality improvement	Fails with noisy inputs	4.0
S48 [115]	Improved efficiency with bypassing the curve reconstruction	Fails in curves with singularities, limited to 2D	3.0
S49 [122]	Simple parallelization, efficient	Lack of boundary refinement and slivers removal	2.5
S50 [123]	Achieves DT in arbitrary dimension in optimal time	A sequential method; that is improved in parallel SVR [9]	3.0
S51 [78]	Generalized MPS and high quality meshing with angle bound	Failure with sharp features	3.0
S52 [103]	High quality mesh, fidelity, efficient, improved memory usage and curvature adaptive	no triangle shape guarantees	3.5
S53 [38]	Fast, simple, high quality, and topology preservation	Fails in sharp coroners and adaptive versions of MC [165], [166]	4.0
S54 [39]	High quality and auto multi-resolution	It might fail with complex meshes	2.0
S55 [79]	Fast and robust	Limited to 2D only	3.0
S56 [100]	A fast, robust, and simple method	High approximation error	3.5
S57 [167]	Simple and easy to implement	Only first few iterations are good; deformation and shrinkage	3.0
S58 [110]	Faster; avoid shrinkage and deformation	Mesh quality is still low	2.5
S59 [112]	Mesh smoothing without any shrinkage or distortion	No qualitative improvement in mesh quality	3.0
S60 [98]	Simple and controls inverted elements and shrinkage	Minimal quality improvement	2.0
S61 [75]	Fast, well-shaped with features preservation	No automatic feature selection	3.0
S62 [111]	Efficient	GPU cost; same limitations as Laplacian	2.0
S63 [63]	Avoiding redundant elements, and intersections	Still have very small angles	2.0
S64 [62]	Open source toolbox	It fails to handle self-intersections and small angles	2.5
S65 [5]	Efficient and can eliminate obtuse and small angles	Fails with more complex models like molecular meshes	4.0
S66 [180]	Robust, fairing very large surfaces	Minimal quality improvement	3.5
S67 [36]	High accuracy, theoretical guarantee, and topology preservation	Works only with empty boundary surfaces	4.0
S68 [181]	Efficient, with minimal interpolation error	Limited to 2D domain	3.5
S69 [88]	Efficient, well-shaped, high quality surface remeshing with preserved topology	No theoretical proof	3.0
S70 [73]	Balance between quality and geometric accuracy	Ill-conditioned Hessian matrices when penalty parameter increases	3.0
S71 [11]	Minimal angle improvement up to 35^o, with error-bound	No significant improvement in θ_{max} and failure in noisy and complex meshes	4.0
S72 [124]	Efficient, and high visual fidelity	Semi-automatic, i.e., need user guidance for input models	2.5

Advantages and limitations of state-of-the-art methods (Table 3 continued).

Article ID	Advantages	Limitations	*Q.A* score
S73 [97]	Robust and can handle large meshes	No significant improvement in approximation error and quality	4.0
S74 [127]	Simple and flexible, Mesh simplification with user control	Slow convergence	3.5
S75 [77]	Simple, efficient, and generalized	Fails with noisy models	3.5
S76 [189]	Simple and straightforward	not efficient/ Linear convergence	4.0
S77 [31]	Locally convergent at a quadratic rate	Still does not reach high quality like obtuse removal	4.0
S78 [30]	Eliminates all obtuse triangles and small angles with topology preservation	Fails with noise, sharp edges and complex models	4.0
S79 [191]	Constrained CVT to be on surface	Global minimization is still desired	3.5
S80 [80]	Efficient, and robust	Lack of termination guarantee for complex models	4.0
S81 [60]	simple yet able for high quality remeshing	no significant improvement in adaptive remeshing	4.0
S82 [3]	Eliminates all obtuse angles, and generate well-shaped and high quality 2D mesh	No theoretical guarantee, limited to 2D	3.5
S83 [199]	Generates highly regular meshes	Limited to uniform sampling/meshing	2.5
S84 [96]	Sharp feature preservation	Fails with defects in input models	4.0
S85 [67]	Topological correctness regardless of local feature size, remeshing with fewer points	Mesh quality is still low	2.5
S86 [120]	High quality meshing, weighted CVT with user given density	sampling to $genus > 0$ surfaces	3.5
S87 [69]	Balance between accuracy and mesh complexity	Limited to 2D meshes	2.5
S88 [113]	Spherical CVT and hyperbolic CVT with theoretical proofs	Slow convergence	3.0
S89 [34]	Triangulation of planner region and detailed quality analysis	Limited to planner and strongly dependent on previous methods	2.5
S90 [61]	Efficient CVT using the GPU	Fails with complex shapes and larger number of sites	4.0
S91 [86]	High quality remeshing, valence optimization	Slow	3.5
S92 [74]	High quality meshing with sharp features	Fails to eliminate short edges and hole filling	4.0
S93 [121]	Simple , efficiently remove short edges from CVT, preserves shape	Lower approximation accuracy, 2D only]	3.0
S94 [66]	Independence from embedding space can handle Large models	Quality can be further improved	4.0
S95 [81]	Convergence to the global optimum	Slow due to computational cost of Geodesic	4.0
S96 [82]	Efficient and maintains anisotropy with bound geometric error	No theoretical guarantee, may fail if we change the anisotropy	3.5
S97 [44]	Efficient and eliminates all obtuse triangles	Fails to preserve face anisotropy	4.0
S98 [126]	Generalization of the CVT	Anisotropic DT with Riemannian metric is time-consuming	3.5
S99 [128]	Theoretical guarantees, Generic, Simple	Fails for sharp features	4.0
S100 [129]	Simple yet powerful extension of CVT, Minimal geometric error	Conflict between anisotropy and geometry	3.5
S101 [76]	Efficient as CVT is computed in 2D	It requires sufficiently dense input to compute conformal embedding space	3.0
S102 [83]	Efficient energy optimization	The mapping back is still a speed bottleneck	4.0
S103 [43]	Minimizes the number of obtuse triangles	Slow convergence, Cannot eliminate all obtuse angels	3.0
S104 [32]	Computes self-intersection free Euclidean embedding	Failure with discontinuities in input metric	4.0

(e.g., feature-sensitive remeshing [96] and implicit feature preservation [11], [33]) have been proposed to preserve the features of general objects. However, these solutions cannot successfully handle thin and sharp features, such as the ear of the lion-head model (see Figure 11).

Error-driven remeshing attempts to optimize the tradeoff between mesh complexity (number of elements) and accuracy (geometric fidelity or approximation error). Typically, remeshing algorithms aim to find a satisfactory balance between three factors: geometric fidelity (the output mesh should be a good approximation of the input), mesh quality, and mesh complexity (number of mesh elements) [11]. Geometric fidelity ensures topology and shape preservation. It is typically measured with an approximation error or Hausdorff distance. With the improvement of mesh quality, it is desired to minimize the approximation error to an acceptable threshold [11]. Similarly, although some methods maintain an acceptable approximation between the input and output meshes, they fail to preserve sharp features [12]. For example, see the ear of the lion-head model in Figure 11(left) generated with the RAR [59].

VoroCrust [74] is an algorithm for conforming Voronoi meshing used for quality improvement of surface and volume meshes with the preservation of sharp features. However, there are a number of limitations for future consideration, such as the presence of short Voronoi edges in output and failure in hole filling and handling undesirable cracks. Hu *et al.* [11] proposed a surface remeshing algorithm that takes a user-given minimal angle threshold and error bound. The algorithm improves the mesh quality by local operators (including edge flipping, edge splitting, edge collapsing, and vertex translation) while considering the given input thresholds. In this manner, the algorithm is able to give a high-quality mesh above the minimal angle threshold within the error bound. It uses the Hausdorff distance to measure the distance between the input and output meshes, which controls the error bound. Prior to applying any local operation, the Hausdorff distance is calculated to ensure that it does not go beyond the error bound; if it does, the local operation is skipped. This method gives a good optimization in terms of three remeshing objectives: error bounding (geometric fidelity), mesh quality, and mesh complexity. The method gives minimal angle improvement up to 35^o and also improves other quality metrics with error bounding and feature preservation. However, there is no significant improvement in the maximal angle, and it only tackles two-manifold meshes. Furthermore, it also fails for noisy and complex models. A triangle removal algorithm [106] can be used to preserve sharp features during surface reconstruction. However, it is a parametric algorithm, requiring a trial and error mechanism for its determination.

Su *et al.* [107] proposed a curvature adaptive meshing algorithm by extending the conformal parametrization technique. Unlike other typical methods, this method samples the normal cycle of the surface instead of the surface itself. The main steps of this method are as follows: (1) Conformal parametrization (CFP) of the surface using dynamic discrete surface Yamabe flow [220]. (2) Curvature adaptive parametrization of the normal cycle (CAP). (3) Sampling the n points of (P) uniformly in the CAP domain. (4) Mapping (P) to CFP. (5) Compute DT after mapping, and fullback DT to the original mesh to get the final output. Its results reveal that remeshing with curvature adaptive sampling better preserves the geometry than that with uniform sampling (see Figure 21). Furthermore, this method [107] is based on solid theoretic foundations and is faster than other existing methods. However, there is no quantitative analysis of mesh quality to

examine the improvement in mesh quality. The authors plan to combine this method with CVT in their future work.

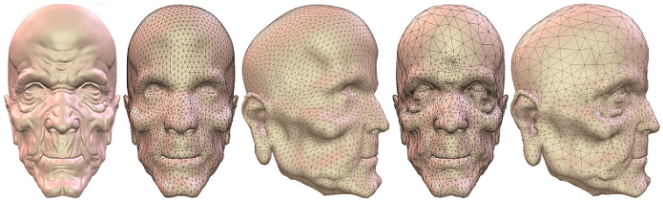


Figure 21. Superiority of curvature adaptive sampling over uniform sampling. From left to right: input, two uniform samplings, and two models with remeshing results for curvature adaptive sampling [107].

Feature-preserved remeshing with DT [92] is another remeshing method that uses the previously proposed ODT [149], [181], [182] with additional constraints for feature preservation. A quadratic optimization scheme is used for mesh quality refinement and minimization of interpolation error between a parabolic function and its piecewise linear interpolation on the surface mesh. A suboptimal DT is used to improve efficiency with minimal loss in accuracy. The main advantages include efficiency, quality improvement with sharp feature preservation, and removal of noise. However, the maximal and minimal angles have not been significantly improved. Even in some inputs with a sharp-featured model, the minimal angle approaches zero. The same failure of quality improvement occurs when dealing with self-intersections. A user-guided method [12] has addressed this issue by improving the minimal angle with preservation of sharp features. However, automatic remeshing is still a challenge for future research.

Chiang *et al.* [45] proposed a parametrization-free remeshing algorithm for semi-regular meshes. This algorithm [45] uses two-step segmentation (normal-based and CVT-based) to reconstruct a high-quality base mesh. Furthermore, this method improves the mesh quality by quadratic error-based mesh relaxation. The authors compared their results in terms of AR, approximation error, and triangle quality (Equation (1)); they found a significant improvement. Their method is able to handle sharp features and self-intersections in 3D models. However, there is no consideration of angle quality. It is not clear whether the sharp edges will have small triangles or not. Furthermore, the method is for semi-regular remeshing and requires generalization in the future.

5.3 Valence optimization

The number of adjacent edges to a vertex is called its valence. Valence 6 is the optimal valence for a vertex [6], [31], [84]. Valence optimization makes remeshing algorithms converge faster. Different researchers [31], [84] have used valence optimization as a submodule in their remeshing algorithms. Other researchers [6], [199] specifically worked on valence optimization.

Valence optimization (i.e., regular remeshing or structured remeshing) generates a structured mesh from a given unstructured input mesh. In a structured mesh, all vertices have a constant number of adjacent edges. Structured meshes (also called regular meshes) can be further classified into semi-regular and highly regular meshes. Semi-regular remeshing works with regular subdivision of the input irregular mesh to generate a mesh where most of the vertices are regular. In highly regular remeshing, a regular mesh is often generated without the subdivision. A number of techniques, such as parametrization [221], [222] and shrink wrapping [223], have been used for structured remeshing as well as unstructured meshing [15].

Aghdaii *et al.* [6] proposed a new method for generating regular meshes called 5-6-7 meshes. We know that valence-6 vertices are called regular vertices [6], [31], [84]. However, it is not easy to ensure all vertices have valence 6. Therefore, 5-6-7 meshes attempt to make each vertex regular (valence 6) or close to regular (valence 5 or 7). In the first phase, a 5-6-7 mesh is achieved with geometric preservation. However, this phase increases the number of elements in the input mesh by a factor of 10. Therefore, a simplification module is applied in the second phase to minimize the number of elements while preserving the 5-6-7 mesh. This method is able to convert an arbitrary mesh into a 5-6-7 mesh. It preserves sharp features with a considerable geometrical approximation. However, the mesh simplification phase is slow.

A recent method [199] attempted CVT valence optimization by merging and splitting operations over Voronoi cells. This valence optimization [199] finds three different types of local clusters and applies local operations to each local cluster. Figure 22 shows three atomic operations applied to different types of local clusters. A local cluster with four adjacent Voronoi cells (two with valence 5 and two with valence 7) is treated by merging the two valence-5 cells using edge collapsing. A similar edge collapsing is also applied when these two cells are valence 7 or one is valence 6 and the other is valence 5 ($V757$). The third type of cluster ($V575$) is treated with vertex insertion. Regular vertices are easier to improve than irregular vertices. Several methods have been used for valence optimization in CVT [31], [84] and remeshing [11], [59]. Valence optimization can be achieved with cell-based operations, including splitting and merging operations [224]. Global Monte-Carlo optimization [225] and hierarchical optimization [226] are among the other methods used for valence optimization. Figure 23 shows valence optimization results using different state-of-the-art methods.

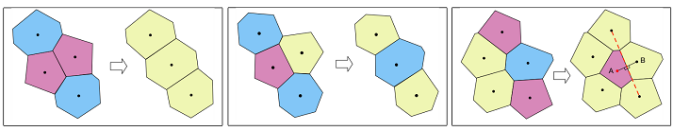
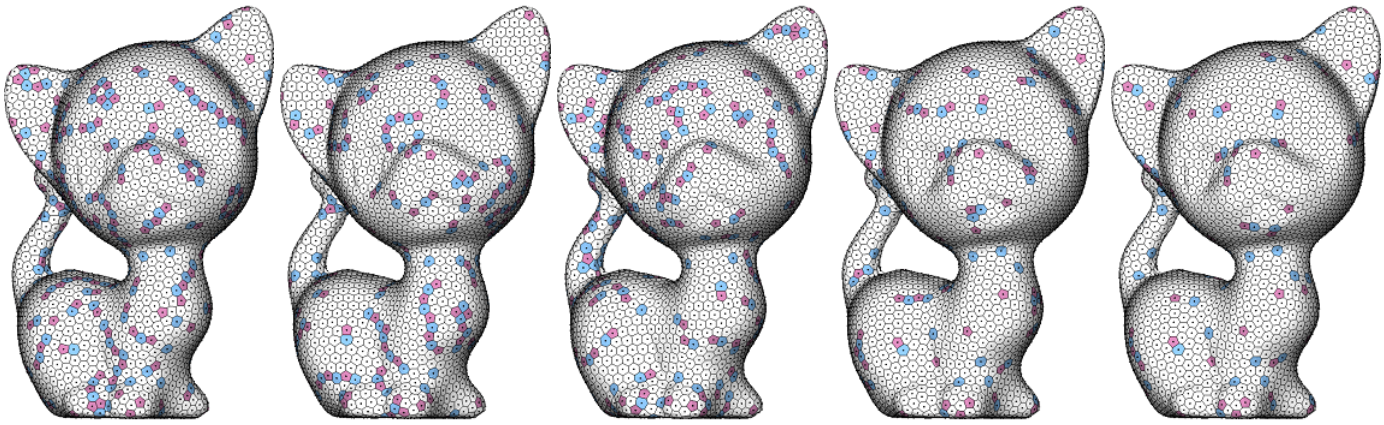


Figure 22. Three local atomic operations for valence optimization with Men *et al.*'s method [199]. From left to right: $V5577$, $V757$, and $V575$.

5.4 Compatible remeshing

Compatible remeshing uses a given set of meshes to generate a new set of meshes with well-shaped elements with common connectivity, good approximation of the input, and proper correspondence. Compatible remeshing is typically desired in applications where element connectivity is more important than quality. A typical example of compatible remeshing is the joint parametrization method [227].

Liu *et al.* [90] proposed an algorithm for high-quality compatible meshing of two polygons to find a smooth transformation between them. This method works in three steps: (1) The source and target polygons are decomposed into sub-polygons, where each sub-polygon is triangulated. (2) Triangulation of the source sub-polygon is mapped onto that of the target sub-polygon, which generates compatible meshes between source and target. (3) The compatible meshes are refined for high-quality planar shape morphing with textures. This method [90] improves the results by



(a) 80.99% (b) 83.33% (c) 85.26% (d) 90.28% (e) 92.95%

Figure 23. Comparison of different valence optimization methods computing the CVT. Kitten model (3,000 points) with % values of v_6: (a) Quasi-Newton solver for CVT [31] (90 iterations), (b) Monte-Carlo optimization [225] (1,500 iterations), (c) Hierarchal method [226] (90 iterations), (d) and (e) Men *et al.*'s method [199] with 90 and 270 iterations, respectively. Magenta color represents valence 5, and blue color represents valence 7. The others are valence 6, i.e., regular cells. This figure is taken from [199].

minimizing the numbers of small triangles and Steiner points. The method is limited to the 2D domain and also fails to handle polygons with holes. In addition, the method considers only the minimal angle as a quality measurement without other considerations, such as the maximal angle or regularity of vertices. It only considers the balance angle partition for target polygons; therefore, the source polygon may still have small triangles. This method was recently extended [91] to be more flexible, use fewer Steiner points, and achieve better mesh quality. The extended version [91] can better improve the minimal angle, considering both source and target polygons. Furthermore, it is simple and fast enough for real-time morphing. However, it still suffers from the abovementioned limitations, including failure for holes in the input and restriction to the 2D domain.

Surazhsky and Gotsman proposed a compatible triangulation method for a planar polygon [93]. They added Steiner points in a compatible way to improve the mesh quality. The method is efficient and works robustly to produce valid results. In addition to compatible meshing, this method [93] uses area-based remeshing to improve the spatial distribution of vertices over the total mesh area. This method is used for planner inputs and can be generalized to other inputs. Although this method uses few Steiner points, there is still no limit to how many Steiner points are sufficient for remeshing with a particular complexity.

5.5 Efficiency in surface remeshing

It is evident that mesh quality improvement leads to efficiency improvement of downstream applications, such as simulations and modelling. However, the time consumption of mesh improvement itself is an issue. Therefore, most applications prefer faster processing of the input mesh for possible quality refinement. In this subsection, we describe the efficiency of the remeshing algorithm itself. Researchers aim to improve the efficiency of remeshing algorithms.

RAR [59] (described in Section 4.2) is a local operation-based method that is efficient enough to be used for real-time applications. Similarly, among the CVT methods, the latest Newton optimization (L-BFGS) [31] improves the efficiency over the previous implementation of the Lloyd method [189] (see Figure 17). Another attempt to improve the efficiency of CVT was the use of GPU-assisted CVT [61]. This method [61] stores geometric images in the GPU for surface representation and computing VD on the surface. It works for 2D CVT as well as surface mesh CVT. The method implements two common CVT algorithms on the GPU: the Lloyd method and L-BFGS. The Lloyd method is fully implemented on the GPU, while L-BFGS is partially implemented using only the major computational parts. The results show a significant improvement over those of the CPU implementation. However, the current implementation [61] works well only for a limited number of sites (up to 10,000). If the number of sites is increased, this method leads to poorer quality due to its large approximation error. It also fails with complex-shaped models. Another study [228] used both CPU and GPU systems for efficient parallel processing capable of handling large meshes; however, it is only applicable to tetrahedral meshes. We suggest a similar work for surface meshing as a future work.

Several algorithms for parallel remeshing [124], [125], [229] are available with interesting future improvements. For example, Nunes *et al.* [125] used parallel remeshing by decomposing the input mesh into different sections. However, the interfaces between two sections require special care. In the future, these interfaces can be treated by local modification, as used by Khan *et al.* [12]. A recent study [111] improved the efficiency of the Laplacian method by using GPU implementation. Although the cost of the GPU is a limitation, GPU implementation of the other surface remeshing techniques might still be considered as an interesting future direction. Another improvement [63] is discussed in a later section. Chen *et al.* [10] implemented 2D Delaunay refinement with a GPU to improve efficiency. Parallel SVR [9] also uses the GPU and a parallelism mechanism to improve efficiency.

Guo *et al.* [71] used MPS to improve mesh quality in an efficient and memory-optimal manner. They considered two main issues of MPS: conflict checking and detection of empty regions and their filling for maximal sampling. The method [71] uses a local region-growing-based approach for conflict checking instead of a global grid, avoiding possible misclassification. Secondly, it improves efficiency by generalizing an efficient 2D algorithm [157], which avoids computation of the restricted power diagram in a similar way to Yan and Wonka [72]. Unlike the previous method [72], this method [71] can deal with sharp features and avoid self-

intersections. Efficiency, optimal memory usage, quality improvement with valid topology, avoiding self-intersection, and handling sharp sheets are the key advantages of this method over other state-of-the-art methods. The main limitation to be considered as future work is the failure to handle triangle soups or noisy models by losing the connectivity of the input mesh. Further considerations for future work are GPU implementation for efficiency and further generalization to volumetric meshing.

Singh *et al.* [115] proposed an algorithm called point cloud meshing (PCM) to generate an unstructured Delaunay mesh of a 2D domain from a given boundary specified by point cloud data (PCD). In general, such algorithms have two main steps: extracting the model definition from PCD and unstructured mesh generation using this model definition. This involves a curve reconstruction phase, which is dependent on the size of the given PCD. Therefore, for densely sampled PCD, the algorithm becomes time-consuming. This method [115] bypasses the curve reconstruction phase by accessing the model information implicitly from well-sampled PCD. In this way, the time consumption of boundary computation is avoided. The reconstruction at local level and intersection calculation using the Voronoi edges are also advantageous. The limitations of this method are that it fails for curves with singularities and it is limited to the 2D domain. Furthermore, there are fewer quality considerations, i.e., circumradius to edge ratio.

Vorsatz *et al.* [116] proposed an optimization algorithm that avoids global parametrization (which slows down the processing) and uses local parametrization. In this way, the efficiency of the remeshing algorithm is improved. This method is implemented in a framework with both interactive and automatic remeshing with quality constraints in terms of short edge, long edge, feature, and valence optimization. However, there is no specific focus on the minimal angle and triangle quality.

6 Discussion and Conclusion

In this section, we present a general discussion on our findings with a summary of open challenges for future research. We also describe the limitations of our survey, our future plans, and concluding remarks.

6.1 Surface remeshing techniques

Surface remeshing techniques improve mesh quality by rearranging the mesh elements. Typically, elimination of the large and small angles has remained a key focus of researchers. Bad triangles (small and large) make the algebraic system of equations ill-conditioned [125]. The angle improvement is related to the AR [121] of triangles as well as short Voronoi edges [30]. Non-obtuse remeshing methods with a typical angle range of $[30^o, 90^o]$ have been achieved in recent works [3], [5]. Similarly, feature preservation [11], [12] and error-bounded remeshing [11], [94] have potential significant improvements.

There are different categories of surface remeshing methods to improve the mesh quality. Mesh simplification aims to minimize the number of vertices to make the mesh processing operation efficient and memory-optimal. However, in terms of mesh complexity (number of vertices), there is a tradeoff between accuracy and efficiency; complex meshes are more accurate, while simple meshes are more efficient. Therefore, an optimal balance is desired between accuracy and efficiency. Local modifications [53], [56], [58] for surface remeshing are efficient, but they do not lead to significant improvement in quality.

Segmentation-based remeshing can be performed in two different ways. It can generate a base mesh that is mapped to the original mesh later. Alternatively, the input mesh can be segmented and each segment remeshed independently [12], [67]. Similarly, DT-based methods [10], [40], [54], [85], [150] depend on the empty circle property of DT. Furthermore, advancing-front methods [103], [158], [159], [162], [163], Laplacian methods [75], [110], [167], [167], CVT [31] and their variants [66], [72], [121], [149], [225], [230] have also seen growing attention in the past two decades.

Surface remeshing with optimization, which includes parametrization-based methods [76], [173], [174], [175], [177], discrete clustering [33], and direct 3D optimization [30], [66], [80], [81], [86], [88], [178], [179], is also an interesting direction. Similarly, blue-noise remeshing [77] and curved optimal DT [231] have also been proposed.

In terms of efficiency, there are two concerns: (1) efficiency of downstream applications, which is typically achieved with mesh simplification and quality refinement, and (2) efficiency of the surface remeshing algorithm itself. In this regard, GPU-based meshing [10], parallel remeshing [124], [125], [229], and local operators [59] have been found to be comparatively more efficient for surface remeshing.

Valence optimization [199], [232] is important in improving efficiency. Note that valence 6 is the optimal value for an inner vertex, while valence 4 is the optimal valence for a boundary vertex. Different methods [11], [59] have used edge flipping operations for vertex valence optimization during their surface remeshing pipelines. There are some algorithms specially designed for valence optimization. For example, a recent method proposed by Men *et al.* [199] is used for regularity improvement of CVT results.

Application-specific remeshing methods, such as CAD model remeshing [70], molecular surface remeshing [1], [63], [105], [170], and modeling and simulation [233], [234], [235], have their own additional challenges. Similarly, anisotropic meshing is also different as it considers the anisotropy, which brings about further challenges in remeshing [82], [200], [201]. For future directions, we highlighted the limitations of each article. Some interesting challenges for future research are described as follows.

6.2 Summary of open challenges

In this review, we have described the limitations of each article included in our final selection. These limitations are open challenges for future research. Here, we summarize a few of them.

(1) Efficiency: The efficiency of different algorithms [1], [6], [11], [12], [33], [81], [86], [113], [127] can be considered for improvement in future research. The use of GPU computing is a feasible solution. Similarly, there are different methods that divide the input mesh into segments; however, all the segments are executed in a sequential order [12], [109], [123]. Therefore, parallel execution of these segments (like in parallel SVR [9]) could improve the efficiency. Avoiding global parametrization [100] is another approach for efficient remeshing.

(2) Generic non-obtuse surface remeshing: Non-obtuse meshing in 2D [3], [212], isotropic surface remeshing [5], [30], and anisotropic surface remeshing [43], [44] have been proposed. However, these methods do not work in complex meshes, such as molecular surface meshes. Therefore, a non-obtuse mesh generation generic for all meshes with arbitrary complexity or only

for molecular surface meshes is in demand.

(3) Molecular surface remeshing: For molecular surface remeshing [1], [63], pattern analysis-based mesh quality enhancement is also an interesting direction. Molecular models have sphere-like structures, meaning that patterns can be identified for segmenting the model and applying segment-based remeshing. In addition, a good mesh generation alternative to TMSmesh [170] is also an important direction to generate surface meshes from Protein Data Bank (PDB) files (https://www.rcsb.org/).

(4) Sharp feature preservation: Feature-preserved remeshing [11], [12] is a considerable achievement in preserving the features and topology of the input model. However, preserving sharp edges/corners is still challenging. Some user-guided methods [12], [56], [65], [75] have the ability to preserve sharp features; however, there is no automatic feature selection for sharp edges. This can be considered as another future work. Similarly, anisotropic meshing [44] also requires extension for sharp feature preservation.

(5) Optimal memory usage: Memory consumption is another major challenge in surface remeshing [58]. Although there are some methods [53], [65], [114], [117] with optimal memory usage, these methods still suffer from either lower efficiency, minimal quality improvements, or failure in sharp feature preservation. Therefore, a new method to provide a balance between mesh quality and memory consumption with preservation of sharp features is needed.

(6) Extending 2D meshing methods to surface meshing: Existing 2D meshing methods [3], [54], [69], [79], [85], [181] can be generalized to surface meshing.

(7) Handling defective inputs: There might be special defects, such as self-intersections, noise, or high complexity, in input meshes, where remeshing methods might fail. Therefore, special treatment of these defects or some modules in existing methods, such as [30], [61], [62], would be helpful to work with arbitrary input meshes.

(8) Extending existing methods for high-quality remeshing: Some methods [53], [87], [101], [114], [117] have mostly focused on either simplification or efficiency rather than improving the mesh quality, such as angle improvement and AR. These methods can be further improved for high-quality remeshing.

(9) Practical applications for end-users: A number of methods are available with easy to use tools, such as MeshLab [236], Graphite [237], Triangle [238], and ISO2mesh [62]. These tools are significantly helping the community in their research and developments. However, some methods do not have practical availability to end-users or developers. In particular, for molecular surface remeshing, such tools are in high demand.

(10) Isotropic to anisotropic extension: A number of methods for isotropic meshing [80], [88], [94], [107] can be extended to anisotropic meshing.

(11) Remeshing with highly anisotropic regions: The mesh quality in anisotropic meshing [44] has no significant improvement, especially in highly anisotropic regions, which can be improved by extending the adopted anisotropic metrics.

(12) Abrupt discontinuities in input anisotropic metrics: Handling anisotropy is challenging and is achieved either manually or with automatic algorithms. Although there has been significant work on the implementation of anisotropic metrics [32], [44], handling abrupt discontinuities in the input is still challenging and could be considered in the future.

(13) Theoretical studies on empirical methods: Most of the existing methods [3], [5], [11], [88], [94] have been empirically analyzed but have no theoretical proof. Therefore, theoretical studies regarding these methods are also an interesting direction for future work. For example, researchers in theoretical physics often use results from applied physics as a starting point for their theoretical research.

6.3 Limitations of this survey

There are two main limitations of our survey.

(1) The classification of methods overlaps. This means that an article may simultaneously belong to more than one category. However, we describe each article as being in only one category, which makes for an unbalanced categorization.

(2) Although we have used the SLR methodology, snowballing, and manual search to ensure the inclusion of all relevant articles, due to the huge number of articles in the primary search, it is possible that we may have excluded some relevant articles in the primary selection.

6.4 Our future plan

Remeshing algorithms have been analyzed by comparing their results. However, some algorithms may perform better on some models while failing on others. For a fair comparison, we require a list of models to be remeshed by all methods included in the survey. This is only possible with cooperation of the authors. In the future, we plan to follow the crowd authoring concept [239] and invite different researchers from the meshing domain to participate. We will use specific input models to be remeshed by authors using different methods from their previous researches. Then, we will compare all results based on the mesh quality metrics presented in Section 3. We also plan to apply the general remeshing methods to molecular surface remeshing and conduct a comparative study of the latest methods. Furthermore, we are working on non-obtuse remeshing of molecular surfaces.

6.5 Conclusion

In this article, we presented a comprehensive SLR of surface remeshing methods. We selected 104 articles for detailed analysis and data extraction. The parameters considered for data extraction included: (1) the main remeshing method, (2) the quality measurements used, (3) the pros and cons, and (4) possible future research directions. We classified the articles based on their remeshing techniques and remeshing objectives. We conclude that surface remeshing has made rapid progress in recent years. Furthermore, there still exist a number of issues and challenges for future research.

ACKNOWLEDGMENTS

This work was partially supported by the Japan Society for the Promotion of Science (JSPS) KAKENHI grant number 19K24346. Y. J. Zhang was supported in part by US NSF grants CMMI-1953323 and CBET-1804929. X. Zhang was supported in part by the National Natural Science Foundation of China (NSFC) with grants 61620106003 and 61972459. We are thankful to the anonymous reviewers for their valuable comments.

REFERENCES

[1] D. Khan, D.-M. Yan, S. Gui, B. Lu, and X. Zhang, "Molecular surface remeshing with local region refinement," *International Journal of Molecular Sciences*, vol. 19, no. 5, pp. 1383:1 –1383:20, 2018.

[2] P. Alliez, G. Ucelli, C. Gotsman, and M. Attene, "Recent advances in remeshing of surfaces," in *Shape Analysis and Structuring*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 53–82.

[3] D. Khan, D.-M. Yan, Y. Wang, K. Hu, J. Ye, and X. Zhang, "High-quality 2D mesh generation without obtuse and small angles," *Computers & Mathematics with Applications*, vol. 75, no. 2, pp. 582 – 595, 2018.

[4] X. Liang and Y. Zhang, "Matching interior and exterior all-quadrilateral meshes with guaranteed angle bounds," *Engineering with Computers*, vol. 28, pp. 375–389, 2011.

[5] Y. Wang, D. Yan, X. Liu, C. Tang, J. Guo, X. Zhang, and P. Wonka, "Isotropic surface remeshing without large and small angles," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 25, no. 7, pp. 2430–2442, July 2019.

[6] N. Aghdaii, H. Younesy, and H. Zhang, "5-6-7 Meshes: Remeshing and analysis," *Computer and Graphics (Extended version from GI'12)*, vol. 36, no. 8, pp. 1072–1083, 2012.

[7] L. F. Gutiérrez, I. Aguinaga, M. Harders, and F. Ramos, "Speeding up the simulation of deformable objects through mesh improvement," *Computer Animation and Virtual Worlds*, vol. 23, no. 3-4, pp. 425–433.

[8] X. Liang and Y. Zhang, "An octree-based dual contouring method for triangular and tetrahedral mesh generation with guaranteed angle range," *Engineering with Computers*, vol. 30, pp. 211–222, 2013.

[9] B. Hudson, G. L. Miller, and T. Phillips, "Sparse parallel Delaunay mesh refinement," in *Proceedings of the Nineteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, ser. SPAA '07. New York, NY, USA: ACM, 2007, pp. 339–347.

[10] Z. Chen, M. Qi, and T.-S. Tan, "Computing Delaunay refinement using the GPU," in *Proceedings of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ser. I3D '17. New York, NY, USA: ACM, 2017, pp. 11:1–11:9.

[11] K. Hu, D. M. Yan, D. Bommes, P. Alliez, and B. Benes, "Error-bounded and feature preserving surface remeshing with minimal angle improvement," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 23, no. 12, pp. 2560–2573, 2017.

[12] D. Khan, D.-M. Yan, F. Ding, Y. Zhuang, and X. Zhang, "Surface remeshing with robust user-guided segmentation," *Computational Visual Media*, vol. 4, no. 2, pp. 113–122, Jun 2018.

[13] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering, (EBSE 2007-001). Keele University and Durham University Joint Report." 2007.

[14] F. Payan, C. Roudet, and B. Sauvage, "Semi-regular triangle remeshing: A comprehensive study," *Computer Graphics Forum*, vol. 34, no. 1, pp. 86–102, 2015.

[15] S. J. Owen, "A survey of unstructured mesh generation technology," in *International Meshing Roundtable*, 1998, pp. 239–267.

[16] S. Borah and B. Borah, "A survey on feature remeshing of 3D triangular boundary meshes," in *2016 International Conference on Accessibility to Digital World (ICADW)*, Dec 2016, pp. 57–62.

[17] Y. Zhang, *Challenges and Advances in Image-Based Geometric Modeling and Mesh Generation*. Dordrecht: Springer Netherlands, 2013, pp. 1–10.

[18] S. Gui, D. Khan, Q. Wang, D.-M. Yan, and B.-Z. Lu, "Frontiers in biomolecular mesh generation and molecular visualization systems," *Visual Computing for Industry, Biomedicine, and Art*, vol. 1, no. 1, pp. 7:1–7:13, Sep 2018.

[19] B. R. de Araújo, D. S. Lopes, P. Jepp, J. A. Jorge, and B. Wyvill, "A survey on implicit surface polygonization," *ACM Comput. Surv.*, vol. 47, no. 4, pp. 60:1–60:39, May 2015.

[20] P. Heckbert and M. Garland, "Survey of polygonal surface simplification algorithms," in *SIGGRAPH 97 Course Notes: Multiresolution Surface Modeling*, 1997.

[21] A. Shamir, "A survey on mesh segmentation techniques," *Computer Graphics Forum*, vol. 27, no. 6, pp. 1539–1556, 2008.

[22] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, ser. EASE '14. New York, NY, USA: ACM, 2014, pp. 38:1–38:10.

[23] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2Mesh: Generating 3D mesh models from single RGB images," in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 55–71.

[24] Y. J. Zhang, *Geometric Modeling and Mesh Generation from Scanned Images*, 1st ed., ser. Chapman & Hall/CRC Mathematical and Computational Imaging Sciences Series. CRC Press, Taylor & Francis Group, 2010.

[25] Y. Zhang, C. Bajaj, and B.-S. Sohn, "Adaptive and quality 3D meshing from imaging data," in *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications*, ser. SM '03. New York, NY, USA: ACM, 2003, pp. 286–291.

[26] Y. Zhang, C. B. Bajaj, and B.-S. Sohn, "3D finite element meshing from imaging data." *Computer Methods in Applied Mechanics and Engineering*, vol. 194 48-49, pp. 5083–5106, 2005.

[27] Z. C. Marton, R. B. Rusu, and M. Beetz, "On fast surface reconstruction methods for large and noisy point clouds," in *2009 IEEE International Conference on Robotics and Automation*, May 2009, pp. 3218–3223.

[28] M. W. Beall, J. Walsh, and M. S. Shephard, "Accessing CAD geometry for mesh generation," in *in: 12th International Meshing Roundtable, Sandia National Laboratories*, 2003, pp. 2003–3030.

[29] Y. Zhang, G. Xu, and C. L. Bajaj, "Quality meshing of implicit solvation models of biomolecular structures," *Computer aided geometric design*, vol. 23 6, pp. 510–530, 2006.

[30] D.-M. Yan and P. Wonka, "Non-obtuse remeshing with centroidal Voronoi tessellation," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 22, no. 9, pp. 2136–2144, 2016.

[31] Y. Liu, W. Wang, B. Lévy, F. Sun, D.-M. Yan, L. Lu, and C. Yang, "On centroidal Voronoi tessellation energy smoothness and fast computation," *ACM Trans. Graph.*, vol. 28, no. 4, pp. 101:1–101:17, Sep. 2009.

[32] Z. Zhong, W. Wang, B. Lévy, J. Hua, and X. Guo, "Computing a high-dimensional Euclidean embedding from an arbitrary smooth Riemannian metric," *ACM Trans. Graph.*, vol. 37, no. 4, Jul. 2018.

[33] S. Valette, J. M. Chassery, and R. Prost, "Generic remeshing of 3D triangular meshes with metric-dependent discrete Voronoi diagrams," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 14, no. 2, pp. 369–381, March 2008.

[34] H. Nguyen, J. Burkardt, M. Gunzburger, L. Ju, and Y. Saka, "Constrained CVT meshes and a comparison of triangular mesh generators," *Comput. Geom. Theory Appl.*, vol. 42, no. 1, pp. 1–19, Jan. 2009.

[35] P. Frey and H. Borouchaki, "Surface mesh evaluation," in *6th International Meshing Roundtable*, 1997, pp. 363–374.

[36] E. Medeiros and M. Siqueira, "Good random multi-triangulation of surfaces," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 24, no. 6, pp. 1983–1996, June 2018.

[37] X. Huang and D. Xu, "Aspect-ratio based triangular mesh smoothing," in *ACM SIGGRAPH 2017 Posters*, 2017, pp. 68:1–68:2.

[38] C. A. Dietrich, C. E. Scheidegger, J. Schreiner, J. L. D. Comba, L. P. Nedel, and C. T. Silva, "Edge transformations for improving mesh quality of marching cubes," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 15, no. 1, pp. 150–159, Jan 2009.

[39] S. Liu, X. Yin, X. Jin, and J. Feng, "High quality triangulation of implicit surfaces," in 9^{th} *Int. Conference on Computer Aided Design and Computer Graphics (CAD-CG'05)*, 2005, pp. 133–138.

[40] S. Cheng, T. Dey, E. Ramos, and T. Ray, "Sampling and meshing a surface with guaranteed topology and geometry," *SIAM Journal on Computing*, vol. 37, no. 4, pp. 1199–1227, 2007.

[41] F. Labelle and J. R. Shewchuk, "Anisotropic Voronoi diagrams and guaranteed-quality anisotropic mesh generation," in *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*, ser. SCG 03. New York, NY, USA: Association for Computing Machinery, 2003, pp. 191–200.

[42] H. Erten and A. Üngör, "Computing triangulations without small and large angles," in *Sixth International Symposium on Voronoi Diagrams, ISVD 2009*, 2009, pp. 192–201.

[43] F. Sun, Y.-K. Choi, W. Wang, D.-M. Yan, Y. Liu, and B. Lévy, "Obtuse triangle suppression in anisotropic meshes," *Computer Aided Geometric Design*, vol. 28, no. 9, pp. 537 – 548, 2011.

[44] Q. Xu, D.-M. Yan, W. Li, and Y. Yang, "Anisotropic surface remeshing without obtuse angles," *Computer Graphics Forum*, vol. 38, no. 7, pp. 755–763, 11 2019.

[45] C.-H. Chiang, B.-S. Jong, and T.-W. Lin, "A robust feature-preserving semi-regular remeshing method for triangular meshes," *The Visual Computer*, vol. 27, no. 9, pp. 811–825, Sep 2011.

[46] M. Bartoň, I. Hanniel, G. Elber, and M.-S. Kim, "Precise Hausdorff distance computation between polygonal meshes," *Comput. Aided Geom. Des.*, vol. 27, no. 8, pp. 580–591, Nov. 2010.

[47] M. Guthe, P. Borodin, and R. Klein, "Fast and accurate Hausdorff distance calculation between meshes," *Journal of WSCG*, vol. 13, pp. 41–48, 2005.

[48] X. Ma, J. Zheng, Y. Shui, H. Zhou, and L. Shen, "A novel method of mesh simplification using Hausdorff distance," in *2012 Third World Congress on Software Engineering*, Nov 2012, pp. 136–139.

[49] P. Cignoni, C. Rocchini, and R. Scopigno, "Metro: measuring error on simplified surfaces," *Computer Graphics Forum*, vol. 17, no. 2, pp. 167–174, 1998.

[50] N. Aspert, D. Santa-Cruz, and T. Ebrahimi, "Mesh: measuring errors between surfaces using the Hausdorff distance," in *Proceedings. IEEE International Conference on Multimedia and Expo*, vol. 1, Aug 2002, pp. 705–708 vol.1.

[51] Y. Li, E. Zhang, Y. Kobayashi, and P. Wonka, "Editing operations for irregular vertices in triangle meshes," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 29, no. 6, pp. 153:1–153:11, 2010.

[52] G. Peyré and L. D. Cohen, "Surface segmentation using geodesic centroidal tesselation," *Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004.*, pp. 995–1002, 2004.

[53] K. Ng and Z. Low, "Simplification of 3D triangular mesh for level of detail computation," in *2014 11th International Conference on Computer Graphics, Imaging and Visualization*, Aug 2014, pp. 11–16.

[54] A. Üngör, "Off-centers: A new type of steiner points for computing size-optimal quality-guaranteed Delaunay triangulations," *Computational Geometry*, vol. 42, no. 2, pp. 109 – 118, 2009.

[55] J. R. Shewchuk, "Delaunay refinement mesh generation," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, May 1997, CMU CS Tech Report CMU-CS-97-137.

[56] A. Abdelkader, A. H. Mahmoud, A. A. Rushdi, S. A. Mitchell, J. D. Owens, and M. S. Ebeida, "A constrained resampling strategy for mesh improvement," *Computer Graphics Forum*, pp. 189–201, 2017.

[57] M. Hussain, "Fast decimation of polygonal models," in *Advances in Visual Computing*, G. Bebis, R. Boyle, B. Parvin, D. Koracin, P. Remagnino, F. Porikli, J. Peters, J. Klosowski, L. Arns, Y. K. Chun, T.-M. Rhyne, and L. Monroe, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 119–128.

[58] M. Hussain, Y. Okada, and K. Niijima, "A fast and memory-efficient method for LOD modeling of polygonal models," in *2003 International Conference on Geometric Modeling and Graphics, 2003. Proceedings*, July 2003, pp. 137–142.

[59] M. Dunyach, D. Vanderhaeghe, L. Barthe, and M. Botsch, "Adaptive remeshing for real-time mesh deformation," in *Eurographics Short Papers Proceedings*, 2013, pp. 29–32.

[60] X. Du, X. Liu, D.-M. Yan, C. Jiang, J. Ye, and H. Zhang, "Field-aligned isotropic surface remeshing," *Computer Graphics Forum*, vol. 37, no. 6, pp. 343–357, 2018.

[61] G. Rong, Y. Liu, W. Wang, X. Yin, X. D. Gu, and X. Guo, "GPU-assisted computation of centroidal Voronoi tessellation," *IEEE Trans. on Vis. and Comp. Graphics*, pp. 345–356, March 2011.

[62] Q. Fang and D. A. Boas, "Tetrahedral mesh generation from volumetric binary and grayscale images," in *2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, June 2009, pp. 1142–1145.

[63] T. Liu, M. Chen, Y. Song, H. Li, and B. Lu, "Quality improvement of surface triangular mesh using a modified laplacian smoothing approach avoiding intersection," *PLOS ONE*, vol. 12, no. 9, pp. 1–16, 09 2017.

[64] H. Si, "TetGen, a Delaunay-based quality tetrahedral mesh generator," *ACM Trans. Math. Softw.*, vol. 41, no. 2, pp. 11:1–11:36, Feb. 2015.

[65] S. Valette and J.-M. Chassery, "Approximated centroidal Voronoi diagrams for uniform polygonal mesh coarsening," *Computer Graphics Forum*, pp. 381–389, 2004.

[66] X. Wang, X. Ying, Y.-J. Liu, S.-Q. Xin, W. Wang, X. Gu, W. Mueller-Wittig, and Y. He, "Intrinsic computation of centroidal Voronoi tessellation (CVT) on meshes," *Computer-Aided Design*, vol. 58, pp. 51 – 61, 2015, solid and Physical Modeling 2014.

[67] J. Edwards, W. Wang, and C. L. Bajaj, "Surface segmentation for improved remeshing," in *Proceedings of the 21st International Meshing Roundtable, IMR 2012, October 7-10, 2012, San Jose, CA, USA*, 2012, pp. 403–418.

[68] M. Ma, X. Yu, N. Lei, H. Si, and X. Gu, "Guaranteed quality isotropic surface remeshing based on uniformization," *Procedia Engineering*, vol. 203, pp. 297 – 309, 2017, 26th International Meshing Roundtable, IMR26, 18-21 September 2017, Barcelona, Spain.

[69] Y. You, X. Kou, and S. Tan, "Adaptive meshing for finite element analysis of heterogeneous materials," *Computer-Aided Design*, vol. 62, no. C, pp. 176–189, May 2015.

[70] J. Guo, F. Ding, X. Jia, and D.-M. Yan, "Automatic and high-quality surface mesh generation for CAD models," *Computer-Aided Design*, vol. 109, pp. 49 – 59, 2019.

[71] J. Guo, D.-M. Yan, X. Jia, and X. Zhang, "Efficient maximal Poisson-disk sampling and remeshing on surfaces," *Comput. Graph.*, vol. 46, no. C, pp. 72–79, Feb. 2015.

[72] D.-M. Yan and P. Wonka, "Gap processing for adaptive maximal Poisson-disk sampling," *ACM Trans. Graph.*, vol. 32, no. 5, pp. 148:1–148:15, 2013.

[73] E. Ruiz-Gironés, J. Sarrate, and X. Roca, "Generation of curved high-order meshes with optimal quality and geometric accuracy," *Procedia Engineering*, vol. 163, pp. 315 – 327, 2016, 25th International Meshing Roundtable.

[74] A. Abdelkader, C. L. Bajaj, M. S. Ebeida, A. H. Mahmoud, S. A. Mitchell, J. D. Owens, and A. A. Rushdi, "Vorocrust: Voronoi meshing without clipping," *ArXiv*, vol. abs/1902.08767, 2019.

[75] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, "Laplacian mesh optimization," in *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia*, ser. GRAPHITE '06. New York, NY, USA: ACM, 2006, pp. 381–389.

[76] Z. Zhong, L. Shuai, M. Jin, and X. Guo, "Anisotropic surface meshing with conformal embedding," *Graphical Models*, vol. 76, no. 5, pp. 468 – 483, 2014.

[77] D.-M. Yan, J. Guo, X. Jia, X. Zhang, and P. Wonka, "Blue-noise remeshing with farthest point optimization," *Comput. Graph. Forum (Proc. SGP)*, vol. 33, no. 5, pp. 167–176, 2014.

[78] D.-M. Yan, J. Wallner, and P. Wonka, "Unbiased sampling and meshing of isosurfaces," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 20, no. 11, pp. 1579–1589, 2014.

[79] S. Lo, "Dynamic grid for mesh generation by the advancing front method," *Computers & Structures*, vol. 123, pp. 15 – 27, 2013.

[80] D.-M. Yan, B. Lévy, Y. Liu, F. Sun, and W. Wang, "Isotropic remeshing with fast and exact computation of restricted Voronoi diagram," *Comput. Graph. Forum (Proc. SGP)*, vol. 28, no. 5, pp. 1445–1454, 2009.

[81] Y.-J. Liu, C.-X. Xu, R. Yi, D. Fan, and Y. He, "Manifold differential evolution (mde): A global optimization method for geodesic centroidal Voronoi tessellations on meshes," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 35, no. 6, pp. 243:1–11, 2016.

[82] X.-M. Fu, Y. Liu, J. Snyder, and B. Guo, "Anisotropic simplicial meshing using local convex functions," *ACM Trans. Graph.*, vol. 33, no. 6, Nov. 2014.

[83] Z. Zhong, X. Guo, W. Wang, B. Lévy, F. Sun, Y. Liu, and W. Mao, "Particle-based anisotropic surface meshing," *ACM Trans. Graph.*, vol. 32, no. 4, Jul. 2013.

[84] W. Yue, Q. Guo, J. Zhang, and G. Wang, "3D triangular mesh optimization in geometry processing for CAD," in *Proceedings of the 2007 ACM Symposium on Solid and Physical Modeling*, ser. SPM '07. New York, NY, USA: ACM, 2007, pp. 23–33.

[85] S. Har-Peled and A. Üngör, "A time-optimal Delaunay refinement algorithm in two dimensions," in *Proceedings of the Twenty-first Annual Symposium on Computational Geometry*, ser. SCG '05. New York, NY, USA: ACM, 2005, pp. 228–236.

[86] Y. Fu and B. Zhou, "Direct sampling on surfaces for high quality remeshing," in *ACM symposium on Solid and physical modeling*, 2008, pp. 115–124.

[87] R. Dyer, H. Zhang, and T. Möller, "Delaunay mesh construction," in *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, ser. SGP '07. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2007, pp. 273–282.

[88] Z. Chen, J. Cao, and W. Wang:, "Isotropic surface remeshing using constrained centroidal Delaunay mesh," *Computer Graphics Forum*, vol. 31, no. 7-1, pp. 2077–2085, 2012.

[89] V. Surazhsky and C. Gotsman, "Explicit surface remeshing," in *Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, ser. SGP '03. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2003, pp. 20–30.

[90] Z. Liu, H. Leung, L. Zhou, and H. P. H. Shum, "High quality compatible triangulations for 2D shape morphing," in *Proceedings of the 21st ACM Symposium on Virtual Reality Software and Technology*, ser. VRST '15. New York, NY, USA: ACM, 2015, pp. 85–94.

[91] Z. Liu, L. Zhou, H. Leung, and H. P. Shum, "High-quality compatible triangulations and their application in interactive animation," *Computers & Graphics*, vol. 76, pp. 60 – 72, 2018.

[92] Z. Gao, Z. Yu, and M. Holst, "Feature-preserving surface mesh smoothing via suboptimal Delaunay triangulation," *Graphical Models*, vol. 75, no. 1, pp. 23 – 38, 2013.

[93] V. Surazhsky and C. Gotsman, "High quality compatible triangulations," *Engineering with Computers*, vol. 20, no. 2, pp. 147–156, Jul 2004.

[94] X.-X. Cheng, X.-M. Fu, C. Zhang, and S. Chai, "Practical error-bounded remeshing by adaptive refinement," *Computers & Graphics*, vol. 82, pp. 163 – 173, 2019.

[95] S. Mansouri and H. Ebrahimnezhad, "Segmentation-based semi-regular remeshing of 3D models using curvature-adapted subdivision surface fitting," *Journal of Visualization*, vol. 19, no. 1, pp. 141–155, Feb 2016.

[96] B. Lévy and Y. Liu, "L_p centroidal Voronoi tesselation and its applications," *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 29, no. 4, pp. 119:1–119:11, 2010.

[97] W. Jakob, M. Tarini, D. Panozzo, and O. Sorkine-Hornung, "Instant field-aligned meshes," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 34, no. 6, pp. 189:1–189:15, 2015.

[98] L. Chen, Y. Zheng, J. Chen, and Y. Liang, "An improved Laplacian smoothing approach for surface meshes," in *Computational Science – ICCS 2007*, Y. Shi, G. D. van Albada, J. Dongarra, and P. M. A. Sloot, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 318–325.

[99] E. Marchandise, C. C. de Wiart, W. G. Vos, C. Geuzaine, and J.-F. Remacle, "High-quality surface remeshing using harmonic maps—part II: Surfaces with high genus and of large aspect ratio," *International Journal for Numerical Methods in Engineering*, vol. 86, no. 11, pp. 1303–1321, 2011.

[100] J. Schreiner, C. E. Scheidegger, S. Fleishman, and C. T. Silva, "Direct (re)meshing for efficient surface processing," *Comput. Graph. Forum (Proc. EUROGRAPHICS)*, vol. 25, no. 3, pp. 527–536, 2006.

[101] F. Dassi, A. Mola, and H. Si, "Curvature-adapted remeshing of CAD surfaces," *Procedia Engineering*, vol. 82, pp. 253 – 265, 2014, 23rd International Meshing Roundtable (IMR23).

[102] F. Dassi, P. Farrell, and H. Si, "An anisoptropic surface remeshing strategy combining higher dimensional embedding with radial basis functions," *Procedia Engineering*, vol. 163, pp. 72 – 83, 2016, 25th International Meshing Roundtable.

[103] J. Schreiner, C. Scheidegger, and C. Silva, "High-quality extraction of isosurfaces from regular and irregular grids," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 12, no. 5, pp. 1205–1212, Sep. 2006.

[104] R. Yi, Y.-J. Liu, and Y. He, "Delaunay mesh simplification with differential evolution," *ACM Trans. Graph.*, vol. 37, no. 6, Dec. 2018.

[105] Y.-J. Liu, C. Xu, D. Fan, and Y. He, "Efficient construction and simplification of Delaunay meshes," *ACM Trans. Graph.*, vol. 34, no. 6, pp. 174:1–174:13, 2015.

[106] S. Methirumangalath, A. Dev Parakkat, S. S. Kannan, and R. Muthuganapathy, "Reconstruction using a simple triangle removal approach," in *SIGGRAPH Asia 2017 Technical Briefs*, ser. SA '17. New York, NY, USA: ACM, 2017, pp. 27:1–27:4.

[107] K. Su, N. Lei, W. Chen, L. Cui, H. Si, S. Chen, and X. Gu, "Curvature adaptive surface remeshing by sampling normal cycle," *Computer-Aided Design*, vol. 111, pp. 1–12, 2019.

[108] J. Wei and Y. Lou, "Feature preserving mesh simplification using feature sensitive metric," *Journal of Computer Science and Technology*, vol. 25, no. 3, pp. 595–605, May 2010.

[109] H. Ozaki, F. Kyota, and T. Kanai, "Out-of-core framework for QEM-based mesh simplification," in *Proceedings of the 15th Eurographics Symposium on Parallel Graphics and Visualization*, ser. PGV '15. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2015, pp. 87–96.

[110] J. Vollmer, R. Mencl, and H. Müller, "Improved laplacian smoothing of noisy surface meshes," *Computer Graphics Forum*, vol. 18, pp. 131–138, 1999.

[111] L. Xiao, G. Yang, K. Zhao, and G. Mei, "Efficient parallel algorithms for 3D Laplacian smoothing on the gpu," *Applied Sciences*, vol. 9, no. 24, pp. 5437:1–5437:14, 2019.

[112] L. Liu, C.-L. Tai, Z. Ji, and G. Wang, "Non-iterative approach for global mesh optimization," *Computer-Aided Design*, vol. 39, no. 9, pp. 772 – 782, 2007.

[113] G. Rong, M. Jin, L. Shuai, and X. Guo, "Centroidal Voronoi tessellation in universal covering space of manifold surfaces," *Comput. Aided Geom. Des.*, vol. 28, no. 8, pp. 475–496, Nov. 2011.

[114] R. Zangeneh and C. F. Ollivier-Gooch, "Thread-parallel mesh improvement using face and edge swapping and vertex insertion," *Computational Geometry*, vol. 70-71, pp. 31 – 48, 2018.

[115] N. Singh, T. Ray, C. Parimi, and S. Kuchibhotla, "Input size independent efficient quality meshing of the interior of 2D point cloud data," *Journal of Computational Design and Engineering*, vol. 6, no. 3, pp. 316 – 326, 2019.

[116] J. Vorsatz, C. Rössl, and H.-P. Seidel, "Dynamic remeshing and applications," in *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications*, ser. SM '03. New York, NY, USA: ACM, 2003, pp. 167–175.

[117] M. Hussain, "Efficient simplification methods for generating high quality lods of 3D meshes," *Journal of Computer Science and Technology*, vol. 24, no. 3, pp. 604–604, May 2009.

[118] A. Lee, H. Moreton, and H. Hoppe, "Displaced subdivision surfaces," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '00. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000, pp. 85–94.

[119] D. Boltcheva, M. Yvinec, and J.-D. Boissonnat, "Feature preserving Delaunay mesh generation from 3D multi-material images," *Computer Graphics Forum*, pp. 1455–1464, Jul. 2009.

[120] P. Alliez, É. C. d. Verdiére, O. Devillers, and M. Isenburg, "Centroidal Voronoi diagrams for isotropic surface remeshing," *Graphical Models*, vol. 67, no. 3, pp. 204–231, 2005.

[121] D. Sieger, P. Alliez, and M. Botsch, "Optimizing Voronoi diagrams for polygonal finite element computations," in *Proceedings of the 19th International Meshing Roundtable, IMR 2010*, 2010, pp. 335–350.

[122] H. Edelsbrunner and D. Guoy, "Sink-insertion for mesh improvement," in *Symposium on Computational Geometry*. ACM, 2001, pp. 115–123.

[123] B. Hudson, G. Miller, and T. Phillips, "Sparse Voronoi refinement," in *Proceedings of the 15th International Meshing Roundtable*, P. P. Pébay, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 339–356.

[124] L. Xing, X. Zhang, C. C. L. Wang, and K. Hui, "Highly parallel algorithms for visual-perception-guided surface remeshing," *IEEE Computer Graphics and Applications*, vol. 34, no. 1, pp. 52–64, Jan 2014.

[125] C. R. S. Nunes, P. C. G. Mayrink, R. C. Mesquita, and D. A. Lowther, "A parallel remeshing method," *IEEE Transactions on Magnetics*, vol. 47, no. 5, pp. 1202–1205, May 2011.

[126] Q. Du and D. Wang, "Anisotropic centroidal Voronoi tessellations and their applications," *SIAM J. Sci. Comput.*, vol. 26, no. 3, pp. 737–761, Mar. 2005.

[127] Y. Lai, M. Jin, X. Xie, Y. He, J. Palacios, E. Zhang, S. Hu, and X. Gu, "Metric-driven rosy field design and remeshing," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 16, no. 1, pp. 95–108, 2010.

[128] J.-D. Boissonnat, K.-L. Shi, J. Tournois, and M. Yvinec, "Anisotropic Delaunay meshes of surfaces," *ACM Trans. Graph.*, vol. 34, no. 2, Mar. 2015.

[129] M. Budninskiy, B. Liu, F. de Goes, Y. Tong, P. Alliez, and M. Desbrun, "Optimal Voronoi tessellations with Hessian-based anisotropy," *ACM Trans. Graph.*, vol. 35, no. 6, Nov. 2016.

[130] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '97. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1997, pp. 209–216.

[131] Y. Cai, X. Guo, Y. Liu, W. Wang, W. Mao, and Z. Zhong, "Surface approximation via asymptotic optimal geometric partition," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 23, no. 12, pp. 2613–2626, Dec 2017.

[132] Q. Du, V. Faber, and M. Gunzburger, "Centroidal Voronoi tessellations: Applications and algorithms," *SIAM Review*, vol. 41, no. 4, pp. 637–676, 1999.

[133] P. J. Frey and H. Borouchaki, "Geometric surface mesh optimization," *Computing and Visualization in Science*, vol. 1, no. 3, pp. 113–121, Nov 1998.

[134] L. Kobbelt, S. Campagna, and H.-P. Seidel, "A general framework for mesh decimation," in *Proceedings of the Graphics Interface 1998 Conference, June 18-20, 1998, Vancouver, BC, Canada*, June 1998, pp. 43–50.

[135] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Mesh optimization," in *Proc. ACM SIGGRAPH*, 1993, pp. 19–26.

[136] M. Rabinovich, R. Poranne, D. Panozzo, and O. Sorkine-Hornung, "Scalable locally injective mappings," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 16:1–16:16, Apr. 2017.

[137] B. Lévy and N. Bonneel, "Variational anisotropic surface meshing with Voronoi parallel linear enumeration," in *Proceedings of the 21st International Meshing Roundtable*, X. Jiao and J.-C. Weill, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 349–366.

[138] J. Leng, G. Xu, Y. Zhang, and J. Qian, *Quality Improvement of Segmented Hexahedral Meshes Using Geometric Flows*. Dordrecht: Springer Netherlands, 2013, pp. 195–221.

[139] K. Hu, Y. J. Zhang, and G. Xu, "CVT-based 3D image segmentation and quality improvement of tetrahedral/hexahedral meshes using anisotropic

Giaquinta-Hildebrandt operator," *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, vol. 6, no. 3, pp. 331–342, 2018.

[140] K. Hu and Y. J. Zhang, "Centroidal Voronoi tessellation based polycube construction for adaptive all-hexahedral mesh generation," *Computer Methods in Applied Mechanics and Engineering*, vol. 305, pp. 405 – 421, 2016.

[141] D. Cohen-Steiner, P. Alliez, and M. Desbrun, "Variational shape approximation," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 905–914, 2004.

[142] K. Zhou, J. Snyder, B. Guo, and H.-Y. Shum, "Iso-charts: Stretch-driven mesh parameterization using spectral analysis," in *ACM Symposium on Geometry Processing*, 2004, pp. 45–54.

[143] F. de Goes, S. Goldenstein, M. Desbrun, and L. Velho, "Exoskeleton: Curve network abstraction for 3D shapes," *Computers & Graphics*, vol. 35, no. 1, pp. 112–121, 2011.

[144] Y. Zhuang, M. Zou, N. Carr, and T. Ju, "Anisotropic geodesics for live-wire mesh segmentation," *Computer Graphics Forum*, vol. 33, no. 7, pp. 111–120, Oct. 2014.

[145] M. S. Floater, "Parametrization and smooth approximation of surface triangulations," *Computer Aided Geometric Design*, vol. 14, no. 3, pp. 231 – 250, 1997.

[146] M. S. Smit, "Efficient remeshing and analysis views for integration of design and analysis," PhD Thesis, Delft University of Technology, 2011.

[147] B. Delaunay, "Sur la sphère vide. A la mémoire de Georges Voronoï," *Bulletin de l'Académie des Sciences de l'URSS*, no. 6, pp. 793–800, 1934.

[148] S.-W. Cheng, T. K. Dey, and J. R. Shewchuk, *Delaunay Mesh Generation*. CRC Press, 2012.

[149] L. Chen and M. Holst, "Efficient mesh optimization schemes based on optimal Delaunay triangulations," *Computer Methods in Applied Mechanics and Engineering*, vol. 200, no. 9, pp. 967 – 984, 2011.

[150] F. d. Goes, P. Memari, P. Mullen, and M. Desbrun, "Weighted triangulations for geometry processing," *ACM Trans. Graph.*, vol. 33, no. 3, pp. 28:1–28:13, Jun. 2014.

[151] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.

[152] J.-D. Boissonnat and F. Cazals, "Natural neighbor coordinates of points on a surface," *Computational Geometry*, vol. 19, no. 2, pp. 155 – 173, 2001, combinatorial Curves and Surfaces.

[153] N. Amenta and M. Bern, "Surface reconstruction by Voronoi filtering," *Discrete Comput. Geom.*, vol. 22, no. 4, pp. 481–504, 1999.

[154] T. K. Dey and W. Zhao, "Approximating the medial axis from the Voronoi diagram with a convergenceguarantee," *Algorithmica*, vol. 38, no. 1, pp. 179–200, Jan 2004.

[155] J. Ruppert, "A new and simple algorithm for quality 2-dimensional mesh generation," Berkeley, CA, USA, Tech. Rep., 1992.

[156] T. Boubekeur and C. Schlick, "A flexible kernel for adaptive mesh refinement on GPU," *Computer Graphics Forum*, vol. 27, no. 1, pp. 102–113.

[157] M. S. Ebeida, S. A. Mitchell, A. Patney, A. A. Davidson, and J. D. Owens, "A simple algorithm for maximal Poisson-disk sampling in high dimensions," *Comput. Graph. Forum (Proc. EUROGRAPHICS)*, vol. 31, no. 2, pp. 785–794, 2012.

[158] P. J. Frey and P.-L. George, *Mesh Generation: Application to Finite Elements, 2nd Edition*. Wiley-ISTE, 2013, p. 848.

[159] J. A. George, "Computer implementation of the finite element method," PhD Thesis, Dept. of Computer Science, Stanford Univ., 1971.

[160] J. R. Tristano, S. J. Owen, and S. A. Canann, "Advancing front surface mesh generation in parametric space using a Riemannian surface definition," 1998, pp. 429–445.

[161] M. Attene, B. Falcidieno, M. Spagnuolo, and G. Wyvill, "A mapping-independent primitive for the triangulation of parametric surfaces," *Graphical Models*, vol. 65, no. 5, pp. 260 – 273, 2003, special Issue on SMI 2002.

[162] R. Löhner, "Progress in grid generation via the advancing front technique," *Engineering with Computers*, vol. 12, no. 3, pp. 186–210, Sep 1996.

[163] E. Hartmann, "A marching method for the triangulation of surfaces," *The Visual Computer*, vol. 14, no. 3, pp. 95–108, Jul 1998.

[164] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *SIGGRAPH Comput. Graph.*, vol. 21, no. 4, pp. 163–169, Aug. 1987.

[165] L. P. Kobbelt, M. Botsch, U. Schwanecke, and H.-P. Seidel, "Feature sensitive surface extraction from volume data," in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '01. New York, NY, USA: ACM, 2001, pp. 57–66.

[166] F. Labelle and J. R. Shewchuk, "Isosurface stuffing: Fast tetrahedral meshes with good dihedral angles," *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 26, no. 3, pp. 57.1–57.10, Jul. 2007.

[167] D. A. Field, "Laplacian smoothing and Delaunay triangulations," *Communications in Applied Numerical Methods*, pp. 709–712, 1988.

[168] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel, "Laplacian surface editing," in *Proceedings of the EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing*. ACM Press, 2004, pp. 179–188.

[169] O. Sorkine, "Differential representations for mesh processing," *Computer Graphics Forum*, vol. 25, no. 4, pp. 789–807, 2006.

[170] T. Liu, M. Chen, and B. Lu, "Efficient and qualified mesh generation for Gaussian molecular surface using adaptive partition and piecewise polynomial approximation," *SIAM Journal on Scientific Computing*, vol. 40, no. 2, pp. B507–B527, 2018.

[171] M. Botsch and L. Kobbelt, "A remeshing approach to multiresolution modeling," in *Proc. of Symp. of Geometry Processing*, 2004, pp. 189–196.

[172] Y. Wang, D.-M. Yan, C. Tang, and X. Liu, "Obtuse triangle elimination for isotropic remeshing," in *ACM SIGGRAPH 2017 Posters*, ser. SIGGRAPH '17. New York, NY, USA: ACM, 2017, pp. 81:1–81:2.

[173] P. Alliez, M. Meyer, and M. Desbrun, "Interactive geometry remeshing," *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 21(3), pp. 347–354, 2002.

[174] V. Surazhsky, P. Alliez, and C. Gotsman, "Isotropic remeshing of surfaces: a local parameterization approach," in *12th International Meshing Roundtable*, 2003, pp. 204–231.

[175] E. Marchandise, J.-F. Remacle, and C. Geuzaine, "Optimal parametrizations for surface remeshing," *Engineering with Computers*, vol. 30, no. 3, pp. 383–402, 2014.

[176] A. Sheffer, E. Praun, and K. Rose, "Mesh parameterization methods and their applications," *Foundations and Trends® in Computer Graphics and Vision*, vol. 2, no. 2, pp. 105–171, 2007.

[177] X. Fang, H. Bao, Y. Tong, M. Desbrun, and J. Huang, "Quadrangulation through morse-parameterization hybridization," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 92:1–92:15, Jul. 2018.

[178] D.-M. Yan, G. Bao, X. Zhang, and P. Wonka, "Low-resolution remeshing using the localized restricted Voronoi diagram," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 20, no. 10, pp. 418–1427, 2014.

[179] A. Ahmed, J. Guo, D.-M. Yan, J.-Y. Franceschi, X. Zhang, and O. Deussen, "A simple push-pull algorithm for blue-noise sampling," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 23, no. 12, pp. 2496–2508, 2017.

[180] G. Taubin, "A signal processing approach to fair surface design," in *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '95. New York, NY, USA: ACM, 1995, pp. 351–358.

[181] L. Chen, "Mesh smoothing schemes based on optimal Delaunay triangulations," in *13th International Meshing Roundtable, Williamsburg*, 2004, pp. 109–120.

[182] L. Chen and J. Xu, "Optimal Delaunay triangulations," *Journal of Computational Mathematics*, vol. 22(2), pp. 299–308, 2004.

[183] T. Schlömer, D. Heck, and O. Deussen, "Farthest-point optimized point sets with maximized minimum distance," in *High Performance Graphics Proceedings*, 2011, pp. 135–142.

[184] T. Liao, Y. Zhang, P. M. Kekenes-Huskey, Y. Cheng, A. P. Michailova, A. D. McCulloch, M. J. Holst, and J. A. McCammon, "Multi-core CPU or GPU-accelerated multiscale modeling for biomolecular complexes." *Molecular Based Mathematical Biology*, vol. 1, 2013.

[185] M. Nieser, J. Palacios, K. Polthier, and E. Zhang, "Hexagonal global parameterization of arbitrary surfaces," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 18, no. 6, pp. 865–878, 2011.

[186] X. Gao, W. Jakob, M. Tarini, and D. Panozzo, "Robust hex-dominant mesh generation using field-guided polyhedral agglomeration," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 114:1–114:13, Jul. 2017.

[187] S. Ni, Z. Zhong, J. Huang, W. Wang, and X. Guo, "Field-aligned and lattice-guided tetrahedral meshing," *Computer Graphics Forum*, vol. 37, no. 5, pp. 161–172, 2018.

[188] A. Tropsha and B. Krishnamoorthy, "Development of a four-body statistical pseudo-potential to discriminate native from non-native protein conformations," *Bioinformatics*, vol. 19, no. 12, pp. 1540–1548, 08 2003.

[189] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, March 1982.

[190] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Math. Program.*, vol. 45, no. 3, pp. 503–528, 1989.

[191] Q. Du, M. D. Gunzburger, and L. Ju, "Constrained centroidal Voronoi tessellations for surfaces," *SIAM Journal on Scientific Computing*, vol. 24, no. 5, pp. 1488–1506, 2003.

[192] I. H. Sloan and R. S. Womersley, "Constructive polynomial approximation on the sphere," *Journal of Approximation Theory*, vol. 103, no. 1, pp. 91 – 118, 2000.

[193] Z. Chen, T. Zhang, J. Cao, Y. J. Zhang, and C. Wang, "Point cloud resampling using centroidal Voronoi tessellation methods," *Computer-Aided Design*, vol. 102, pp. 12 – 21, 2018, special Issue on SPM 2018.

[194] L. Shuai, X. Guo, and M. Jin, "GPU-based computation of discrete periodic centroidal Voronoi tessellation in hyperbolic space," *Computer-Aided Design*, vol. 45, no. 2, pp. 463 – 472, 2013, solid and Physical Modeling 2012.

[195] G. Peyré and L. D. Cohen, "Geodesic remeshing using front propagation," *Int. J. Comput. Vision*, vol. 69, no. 1, pp. 145–156, 2006.

[196] O. Sifri, A. Sheffer, and C. Gotsman, "Geodesic-based surface remeshing," in *Proceedings of the 12th International Meshing Roundtable, IMR 2003*, 2003, pp. 189–199.

[197] Y.-J. Liu, D. Fan, C.-X. Xu, and Y. He, "Constructing intrinsic Delaunay triangulations from the dual of geodesic Voronoi diagrams," *ACM Trans. Graph.*, vol. 36, no. 2, Apr. 2017.

[198] D. Dunbar and G. Humphreys, "A spatial data structure for fast Poisson-disk sample generation," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 503–508, Jul. 2006.

[199] Y. Men, Z. Shen, D. Khan, and D. Yan, "Improving regularity of the centoridal Voronoi tessellation," in *SIGGRAPH Posters*. ACM, 2018, pp. 66:1–66:2.

[200] J.-D. Boissonnat, C. Wormser, and M. Yvinec, "Locally uniform anisotropic meshing," in *Proceedings of the 24th Annual Symposium on Computational Geometry*, ser. SCG 08. New York, NY, USA: Association for Computing Machinery, 2008, pp. 270–277.

[201] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun, "Anisotropic polygonal remeshing," *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 22, no. 3, pp. 485–493, 2003.

[202] R. Simpson, "Anisotropic mesh transformations and optimal error control," *Applied Numerical Mathematics*, vol. 14, no. 1, pp. 183 – 198, 1994.

[203] P. S. Heckbert and M. Garland, "Optimal triangulation and quadric-based surface simplification," *Computational Geometry*, vol. 14, no. 1, pp. 49 – 65, 1999.

[204] F. Bossen and P. S. Heckbert, "A pliant method for anisotropic mesh generation," in *International Meshing Roundtable*, 1996, pp. 63–74.

[205] K. Shimada, A. Yamada, and T. Itoh, "Anisotropic triangular meshing of parametric surfaces via close packing of ellipsoidal bubbles," in *In 6th International Meshing Roundtable*, 1996, pp. 375–390.

[206] J. Nash, "C1 isometric imbeddings," *Annals of Mathematics*, vol. 60, no. 3, pp. 383–396, 1954.

[207] J.-D. Boissonnat, F. Nielsen, and R. Nock, "Bregman voronoi diagrams," *Discrete Comput. Geom.*, vol. 44, no. 2, pp. 281–307, Sep. 2010.

[208] G. Rong, M. Jin, and X. Guo, "Hyperbolic centroidal Voronoi tessellation," in *Proceedings of the 14th ACM Symposium on Solid and Physical Modeling*, ser. SPM 10. New York, NY, USA: Association for Computing Machinery, 2010, pp. 117–126.

[209] G. D. Cañas and S. J. Gortler, "Surface remeshing in arbitrary codimensions," *The Visual Computer*, vol. 22, pp. 885–895, 2006.

[210] I. Babuska and A. Aziz, "On the angle condition in the finite element method," *SIAM Journal on Numerical Analysis*, vol. 13, no. 2, pp. 214–226, 1976.

[211] X. Liang and Y. Zhang, "Hexagon-based all-quadrilateral mesh generation with guaranteed angle bounds," *Computer Methods in Applied Mechanics and Engineering*, vol. 200, no. 23, pp. 2005 – 2020, 2011.

[212] B. S. Baker, E. Grosse, and C. S. Rafferty, "Nonobtuse triangulation of polygons," *Discrete Comput. Geom.*, vol. 3, no. 2, pp. 147–168, Jun 1988.

[213] T. R. Jones, F. Durand, and M. Desbrun, "Non-iterative, feature-preserving mesh smoothing," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 943–949, Jul. 2003.

[214] J. Wang, X. Zhang, and Z. Yu, "A cascaded approach for feature-preserving surface mesh denoising," *Computer-Aided Design*, vol. 44, no. 7, pp. 597–610, Jul. 2012.

[215] Q. Zhou, T. Weinkauf, and O. Sorkine, "Feature-based mesh editing," in *Proc. EUROGRAPHICS, Short Papers*, pp. 214–218.

[216] A. Nealen, O. Sorkine, M. Alexa, and D. Cohen-Or, "A sketch-based interface for detail-preserving mesh editing," *ACM Trans. Graph. (Proceedings of ACM SIGGRAPH)*, vol. 24, no. 3, pp. 1142–1147, 2005.

[217] J. Qian, Y. Zhang, W. Wang, A. C. Lewis, M. A. S. Qidwai, and A. B. Geltmacher, "Quality improvement of non-manifold hexahedral meshes for critical feature determination of microstructure materials," in *International Meshing Roundtable*, 2009, pp. 211–230.

[218] X. Liang, M. S. Ebeida, and Y. Zhang, "Guaranteed-quality all-quadrilateral mesh generation with feature preservation," in *International Meshing Roundtable*, 2009, pp. 45–63.

[219] S. Fuhrmann, J. Ackermann, T. Kalbe, and M. Goesele, "Direct resampling for isotropic surface remeshing," in *Vision, Modeling, and Visualization (2010)*, R. Koch, A. Kolb, and C. Rezk-Salama, Eds. The Eurographics Association, 2010, pp. 9–16.

[220] X. Gu, F. Luo, J. Sun, and T. Wu, "A discrete uniformization theorem for polyhedral surfaces," *Journal of Differential Geometry*, vol. 109, no. 2, pp. 223–256, 6 2018.

[221] K. Hormann and G. Greiner, "Quadrilateral remeshing," in *Proceedings of Vision, Modeling and Vizualization, 2000*, 2000, pp. 153–162.

[222] P. V. Sander, S. J. Gortler, J. Snyder, and H. Hoppe, "Signal-specialized parametrization," in *Proceedings of the 13th Eurographics Workshop on Rendering*, ser. EGRW '02. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2002, pp. 87–98.

[223] L. Kobbelt, J. Vorsatz, U. Labsik, and H.-P. Seidel, "A shrink wrapping approach to remeshing polygonal surfaces," *Computer Graphics Forum*, vol. 18, pp. 119–130, 1999.

[224] O. Deussen, M. Spicker, and Q. Zheng, "Weighted Linde-Buzo-Gray stippling," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 233:1–233:12, 2017.

[225] L. Lu, F. Sun, H. Pan, and W. Wang, "Global optimization of centroidal Voronoi tessellation with Monte Carlo approach," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 18, no. 11, pp. 1880–1890, 2012.

[226] L. Wang, F. Hétroy-Wheeler, and E. Boyer, "A hierarchical approach for regular centroidal Voronoi tessellations," *Computer Graphics Forum*, vol. 35, no. 1, pp. 152–165, 2016.

[227] V. Kraevoy and A. Sheffer, "Cross-parameterization and compatible remeshing of 3D models," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 861–869, Aug. 2004.

[228] J. D'Amato and M. Vénere, "A CPU-GPU framework for optimizing the quality of large meshes," *Journal of Parallel and Distributed Computing*, vol. 73, no. 8, pp. 1127 – 1134, 2013.

[229] C. R. S. Nunes, P. C. G. Mayrink, R. C. Mesquita, and D. A. Lowtherx, "An efficient parallel remeshing method," in *Digests of the 2010 14th Biennial IEEE Conference on Electromagnetic Field Computation*, May 2010, pp. 1–1.

[230] Y. Wang, L. Ju, D. Wang, and X. Wang, "Generalized edge-weighted centroidal Voronoi tessellations for geometry processing," *Computers & Mathematics with Applications*, vol. 64, no. 8, pp. 2663 – 2681, 2012.

[231] L. Feng, P. Alliez, L. Busé, H. Delingette, and M. Desbrun, "Curved optimal Delaunay triangulation," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 61:1–61:16, 2018.

[232] J. Huang, M. Zhang, W. Pei, W. Hua, and H. Bao, "Controllable highly regular triangulation," *Science China Information Sciences*, vol. 54, no. 6, pp. 1172–1183, Jun 2011.

[233] T. Igarashi and J. F. Hughes, "Smooth meshes for sketch-based freeform modeling," in *Proceedings of the 2003 Symposium on Interactive 3D Graphics*, ser. I3D '03. New York, NY, USA: ACM, 2003, pp. 139–142.

[234] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, "Fibermesh: Designing freeform surfaces with 3D curves," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 41:1–41:9, Jul. 2007.

[235] A. Gargallo-Peiró, M. Avila, H. Owen, L. Prieto-Godino, and A. Folch, "Mesh generation, sizing and convergence for onshore and offshore wind farm atmospheric boundary layer flow simulation with actuator discs," *Journal of Computational Physics*, vol. 375, pp. 209 – 227, 2018.

[236] 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*, V. Scarano, R. D. Chiara, and U. Erra, Eds., 2008, pp. 129–136.

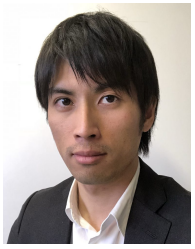
[237] "Graphite," http://alice.loria.fr/index.php/software/3-platform/22-graphite.html, [Online; accessed Feb. 13, 2018].

[238] J. R. Shewchuk, "Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator," in *Applied Computational Geometry Towards Geometric Engineering*, M. C. Lin and D. Manocha, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 203–222.

[239] A. E. Al Lily et al., "Academic domains as political battlegrounds: A global enquiry by 99 academics in the fields of education and technology," *Information Development*, vol. 33, no. 3, pp. 270–288, 2017.

Dawar Khan was an Assistant Professor at the Interactive Media Design Lab, Nara Institute of Science and Technology (NAIST), Japan from 2018 to 2020. Since June 2020, he has been working as a Postdoctoral Researcher with the Social Computing Lab, NAIST. He received his PhD in Computer Science from the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, University of Chinese Academy of Sciences (UCAS), Beijing, China in 2018. During his PhD study, he was supported by the Chinese Government Scholarship and the National Natural Science Foundation of China. He was awarded with the 2018 Excellent International Graduates Award of UCAS. Prior to that, he received his BS and MS degrees in Computer Science from University of Malakand, Pakistan in 2011 and 2014, respectively. His research interests include computer graphics, computational geometry, mesh processing, and augmented reality.

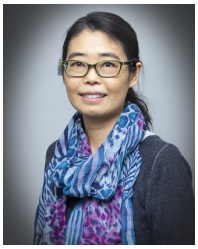
Alexander Plopski is a Postdoctoral Fellow at the HCI group, University of Otago, New Zealand. Before this, he worked as an Assistant Professor at the Interactive Media Design Lab, Nara Institute of Science and Technology (NAIST), Japan. He received his BSc and MSc degrees in Informatics from the Technical University of Munich, Germany in 2010 and 2012, respectively. In 2016, he completed his PhD in Information Science and Technology at Osaka University, Japan and joined NAIST in April 2016. His research interests include computer vision, eye tracking, and augmented, mixed, and virtual reality.



Yuichiro Fujimoto is an Assistant Professor at the Interactive Media Design Lab, Nara Institute of Science and Technology (NAIST), Japan. He received his BS degree in Engineering from Osaka University, Osaka, Japan in 2010 and MS degree and PhD in Engineering from Nara Institute of Science and Technology, Nara, Japan in 2012 and 2015, respectively. From 2015 to 2018, he was an Assistant Professor with the Engineering Department, Tokyo University of Agriculture and Technology, Japan. His research interests include spatial augmented reality and analysis of internal states of humans from 2D/3D information.

Masayuki Kanbara received his PhD degree in Engineering from Nara Institute of Science and Technology (NAIST) in 2002. He was an assistant professor at the Information Science Department, NAIST from 2002-2010. He was a visiting researcher of the University of California, Santa Barbara in 2008-2009. He has been an associate professor at NAIST since 2010. His research is focused on augmented reality, computer vision, and human-robot interaction.

Gul Jabeen recieved her PhD from Tsinghua University, Beijing, China in 2019. She received her BS degree in Computer Sciences from Karakuram International University, Gilgit, Pakistan in 2006 and MS degree from International Islamic University, Islamabad in 2012. Currently, she is a Lecturer at the Department of Computer Science, Karakuram International University, Gilgit, Pakistan. Her research interests include software and information security, software reliability, and prediction modelling.



Yongjie Jessica Zhang is the George Tallman Ladd and Florence Barrett Ladd Professor of Mechanical Engineering at Carnegie Mellon University (CMU) with a courtesy appointment to the Department of Biomedical Engineering. She received her BEng degree in Automotive Engineering and MEng degree in Engineering Mechanics from Tsinghua University, China in 1996 and 1999, respectively; she also received her MEng degree in Aerospace Engineering and Engineering Mechanics and PhD in Computational Engineering and Sciences from the Institute for Computational Engineering and Sciences (now Oden Institute), The University of Texas, Austin in 2002 and 2005, respectively. She joined CMU in 2007 as an assistant professor. She was promoted to associate professor in 2012 and to full professor in 2020. She has received numerous awards, including the US Presidential Early Career Award for Scientists and Engineers, NSF Career Award, and ONR Young Investigator Award. She is a fellow of AIMBE, ASME, USACM, and ELATE at Drexel. Her research interests include computational geometry, mesh generation, computer graphics, visualization, isogeometric analysis, and their application in computational biomedicine, materials science, and engineering.

Xiaopeng Zhang is a research professor in National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academic of Sciences, and a teaching professor in University of Chinese Academy of Sciences (UCAS). He received his BS and MS degrees in Mathematics from Northwest University, China in 1984 and 1987, respectively, and he received his PhD in computer science from the Institute of Software, Chinese Academy of Sciences in 1999. He was invited as a specialist in computer graphics to INRIA Nancy, France from October 2002 to August 2004. He received the second prize of the State Scientific and Technological Progress Award in 2004, won the Chinese Award of Excellent Patents in both 2008 and 2012, and received second prize of the Scientific and Technological Progress Award by Ministry of Education of China in 2015. His main research interests include computer graphics, computer vision, and image processing.

Hirokazu Kato is a professor at the Nara Institute of Science and Technology (NAIST), Japan, since 2007. He received his BEng, MEng, and PhD degrees from Osaka University, Japan in 1986, 1988, and 1996, respectively. He was with the Faculty of Engineering Science at Osaka University from 1989 to 1999. In 1999, he joined the Faculty of Information Sciences at Hiroshima City University, Japan. In 2003, he joined Osaka University. Since 1998, when he was a visiting scholar with the Human Interface Technology Laboratory (HIT Lab) at the University of Washington, he has been working on augmented reality and related topics. He received the Virtual Reality Technical Achievement Award from IEEE VGTC in 2009 and Lasting Impact Award at the 11^{th} IEEE International Symposium on Mixed and Augmented Reality in 2012.