Computer-Aided Design, Vol. 33, No 3., pp. 251- 61, 2001

Face Clustering of a Large-scale CAD Model for Surface Mesh Generation

[1]Keisuke Inoue [2]Takayuki Itoh [3]Atsushi Yamada [4]Tomotake Furuhata
[5]Kenji Shimada

[1,2,3]Tokyo Research Laboratory, IBM Research
{inoue, itot}@trl.ibm.co.jp, ayamada@jp.ibm.com

[4]Yamato Software Development Laboratory, IBM Japan Ltd.
furuhata@jp.ibm.com

[5]Mechanical Engineering, Carnegie Mellon University
shimada@cmu.edu

Contact Address: Keisuke Inoue
Tokyo Research Laboratory, IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi,
Kanagawa 228-0852, JAPAN
E-Mail: inoue@trl.ibm.co.jp,
Phone: +81 (46) 215-4888 Fax: +81 (46) 273-7428

Abstract:

A detailed CAD model needs manual clean-up, or simplifying operations, before a finite element mesh can be automatically generated because such a model consists of hundreds or thousands of faces many of which may be smaller than a desired mesh element size. We propose an automated face clustering method used as a pre-process of surface mesh generation. By decomposing a model into face clusters so that each region can be projected onto a simple parametric surface such as a plane, we obtain a final mesh as an aggregation of sub-meshes for respective clusters without time-consuming manual preparation work. The projection onto a surface realises re-parametrization as well as suppression of small details. The main contribution of this work is the integration of: (1) a greedy algorithm for combining faces into clusters, and (2) geometric indices that reflect various geometric aspects of a preferable shape for a cluster. The validity of the approach is demonstrated with results of clustering and mesh generation for a real-scale CAD model.

Keywords: Mesh Generation; Face Clustering; Mesh Simplification; Re-parametrization; Finite Element Analysis

1 Introduction

Although designers prefer to conduct finite element analysis (FEA) on original CAD models rather than to create specific models for analysis alone, a mesh generated automatically from a detailed model is not suitable for FEA. Typically, a CAD model in the final design stage consists of hundreds or even thousands of faces, each of which is represented as a trimmed parametric surface and many of which are smaller than a preferred mesh element size. For such CAD models, the mesh generation problem is not a discretization process that divides a large surface into small elements. On the contrary, one mesh element often has to cover a number of faces that are representing details. Such being the case, assuring conformity of sub-meshes by placing fixed nodes on face boundaries will produce a low-quality mesh.

A straightforward solution to this problem is to generate a mesh over the whole surface ignoring internal face boundaries. This approach, however, works only for simple CAD models. When it is applied to a huge patchwork of faces, the following problems arise:

- It is necessary to traverse parametric spaces for converting values from the physical space to a parametric space, which is frequent and becomes computationally expensive when the number of faces is large.

- Small details of the surface shape degrade the uniformity of the mapping between the physical space and the parametric space, making high-quality meshing difficult.
- A huge patchwork model naturally contains a lot of internal boundaries, where most topological failures and unintended discontinuities reside.

The above problems affect all mesh generation algorithms that use parametric surfaces including physically-based approaches such as our Bubble Mesh algorithm [1][2][3][4]. Therefore, to deal with real-scale CAD models, it is necessary to simplify complex parametric spaces as well as to suppress small features of the surface.

In this paper, we propose the following three-step solution to this problem, also illustrated in Fig. 1:

1. Decompose the entire surface into a set of suitable surface regions by face clustering.

2. For each region, generate a mesh on a simplified geometry (e.g., a projected plane), and then project the mesh back onto the original region.

3. Merge all generated sub-meshes into one mesh.

As the simplified geometry we use the plane since it has the most basic and straightforward parameterisation where the u-, v-derivative vectors are constant as well as orthogonal to each other.

By employing simplified geometry that closely approximates a region, small features are ignored; therefore, the meshing process is drastically accelerated and stabilised without worsening the mesh quality. Although generating nodes on shared boundaries is still necessary to guarantee the conformity of sub-meshes, the restriction is relaxed causing no ill-shaped mesh elements. In addition, by finally projecting every node onto the original region, it lies exactly on the surface and the supporting face is always known.

Among the three steps, the first step—choosing suitable regions—is the key in order to generate a well-shaped mesh, and it is the most labour-intensive step when performed manually. The main contribution of this paper is an automated method for the first step, which clusters all faces in a model into a number of geometrically suitable regions by iteratively merging a pair of adjacent faces to make a larger region. Through the process, each region grows to be a well-shaped, well-sized region by applying merging operations based on the criterion defined as a combination of geometric indices.

The remainder of the paper is organised as follows. After an overview of related work in Section 2, we describe the problem and outline our technical approach in Section 3. Then in Section 4, we discuss how to determine whether two regions should be merged or not and how to prioritise candidate mergers. Finally in Section 5, we show results of clustering and meshing on a realistic-scale CAD model.

2 Related Work

It is widely recognised in industry that a realistic-scale CAD model consisting of multiple surfaces is not suitable for FEA mesh generation due to excessive details and model failures. Approaches to this problem can be categorised into two: pre-meshing and post-meshing approaches. A pre-meshing approach transforms an original model so that it can be handled with some meshing algorithm, while a post-meshing approach takes as input a low-quality mesh, which is an aggregation of meshes for each face, and cleans up the mesh by adaptive simplification.

Mesh simplification is a general technique to convert a mesh into a coarser mesh, which can be used for post-meshing approaches. Its original objective is to simplify a huge mesh that contains millions of triangles, mainly for the purpose of graphical presentation or data processing. Schroeder et al. [5] iteratively selects a vertex in a mesh based on geometric criteria, removes it and re-triangulates the remaining polygonal area. Hoppe et al. [6] introduce an energy function and simplify a mesh by optimisation, using a local topological transformation. Garland and Heckbert [7] adopt both non-edge contraction (i.e., unification of two non-adjacent vertices) and edge contraction based on simple quadric error metrics. His idea of using a 4x4 matrix for error approximation at a vertex is extended to apply to a face normal vector by Willmott et al. [8]. Shephard et al. [9] applied mesh simplification to the problem of multi-surface mesh generation. They start from an aggregation of meshes each coming from a single face, and improve mesh quality by merging sliver elements with adjacent elements. Because ill-shaped elements are resolved locally, quality improvement of the mesh is limited, especially for quadrilateral dominant meshes, in which alignment of elements is of more significance.

Pre-meshing approaches, including ours, simplify a model first, which can include topological correction, re-parametrization of multiple surfaces, and suppression of small features compared with a desired element size. Sheffer et al. [10, 11] propose a 'virtual topology' concept for cleaning up and simplifying only the topology of a model. They introduce clustering of planar facets that are created from each face and identify a simpler topological structure for the whole surface. Their method is suitable for mapping-type algorithms, but not for algorithms that need to work with geometry intensively. Mobley et al. [12] present an object-oriented framework, which aims to defeature CAD models automatically and to resolve topological failures. Marcum [13] introduced global mapping to re-parametrize input faces into a single surface. By solving a Laplacian equation, a single parametric space is obtained.

Our approach employs similar techniques to those of mesh simplification and Sheffer's approach, but can be characterised by the input model, i.e., a collection of curved surfaces, which has a large variety in:

- Size: some are tiny and others are so large as to cover half of the whole model.
- Complexity: some faces are badly shaped, containing complex boundaries, corner angles close to 0 or 2π, or large variations in normal vectors.

We also impose a stronger requirement for output regions: they have to be well-sized and well-shaped to avoid ill-shaped elements caused by the boundary constraint. Moreover, every region needs to be well approximated by the prescribed types of geometry: planes in our case.

3 Technical Approach

3.1 Problem Description

Our algorithm accepts the following as inputs:

- A set of faces with adjacency links
- A mark for each pair of adjacent faces

Input faces are assumed to be connected without overlapping each other and to make a 2-manifold shell as a whole. As for the marks, the user can constrain any pair of adjacent faces by setting either of two marks, INTERNAL or BOUNDARY. The mark INTERNAL represents that the edges shared by the pair of faces must not be on the boundary of a region, while the mark BOUNDARY represents that they must be on a boundary. Every face pair with no mark is a candidate for iterative merger. Our algorithm clusters all input faces into several groups, each of which makes a connected region suitable for mesh generation.

This clustering problem inherently has a large number of solutions due to a combinatorial explosion. Two trivial solutions are: (1) one region for the whole shape, and (2) every face making its own region. However, by considering the requirements for mesh generation, the proper clustering is found somewhere between the two extremes in most cases.

A well-shaped region for meshing satisfies the following conditions:

- Area size: The area of a region should be large enough in comparison with that of a mesh element. A small region produces a poor quality mesh.
- Boundary smoothness: The boundary of a region should be as smooth as possible. An ill-shaped, jagged boundary results in poor-quality mesh elements in that area.
- Region flatness: Each region should be as flat as possible in order to be well approximated by a plane. A large variation of normal vectors over a region causes a distorted mapping between the surface and the plane, which eventually results in distorted mesh elements.

The definition of a well-shaped region can vary to some extent depending on the method of mesh generation and/or on the downstream applications that will work on the mesh. For example, region flatness is particularly important when we use a plane as the simplified geometry, while the first two conditions are more general requirements. In any case, our clustering method is customisable to various requirements stated in geometric terms.

3.2 Arc Contraction on an Adjacency Graph

By mapping a face to a graph node, and the adjacency of a pair of faces to a graph arc, respectively, we define an adjacency graph from the set of input faces (see Fig. 2). Since the input faces make a 2-manifold shell, the adjacency graph becomes planar. An arc-contraction operation in this graph corresponds to a face-merging operation.

The arc-contraction operation consists of two steps (see Fig. 3). In the first step, a target arc is deleted and its two terminal nodes are merged into one. At the same time, other arcs incident to either or the two original nodes are connected to the merged node. In the second step, if duplicate arcs between the same two nodes are produced in the first step, they are unified into a new arc. In the unifying operation, if at least one of the duplicate arcs is marked with BOUNDARY, the unified arc is marked with BOUNDARY. If at least one of the duplicate arcs is marked with INTERNAL, the unified arc is marked with INTERNAL. There could be a case where one arc of the duplicate arcs is BOUNDARY and the other is INTERNAL. This means that the input marks are contradictory and this merger cannot be allowed.

3.3 Outline of the Algorithm

Our algorithm is basically a sequential arc contraction on an adjacency graph. The process starts from the initial state where every graph node corresponds to one face. Then, pairs of adjacent regions are merged iteratively. If n mergers take place, the total number of regions is reduced by n.

The algorithm consists of the following steps:

1. As a pre-processing step, contract all arcs marked with INTERNAL so that there remain no edges of that type in the adjacency graph.

2. Make a list of all unmarked arcs.

3. For each arc in the list, calculate a score for prioritisation.

4. Scan the list and select the top-scored arc from the arcs that are unmarked and pass the mergeability test (this topic will be discussed in Section 4).

5. During the scan, if any arc marked with BOUNDARY is found, remove the arc from the list. (BOUNDARY arcs can exist in the list since an unmarked arc may be changed into a BOUNDARY arc by arc contraction.)

6. If no arc is selected in step 4, terminate the algorithm.

7. To merge regions, contract the selected arc in the adjacency graph. Remove the arc from the list. If duplicate arcs were created during the contraction, remove them from the list and add the unified arcs to the list.

8. Re-calculate scores for all arcs incident to the merged node.

9. Jump to Step 4.

All arcs marked with INTERNAL are resolved in the pre-processing; therefore, they never appear in the list. On the other hand, arcs marked with BOUNDARY can appear in the list due to the unification of duplicate arcs, even though they are not in the list initially.

The flow of the algorithm is governed by the strategy of scoring and mergeability tests based on the geometry of pairs of regions. That will be discussed in the next section.

4 Conditions for Merging Regions

The output clustering of the method depends totally on the strategy of clustering, that is, which pairs of regions to select for mergers and when to terminate the whole process. In this section, geometric indices for describing the conditions are introduced, and how to define the clustering strategy is presented.

4.1 Scoring and the Mergeability Test

The scoring is used to calculate a scalar value for each arc in order to prioritise arcs for contraction. The arc with the top score is considered to be the best candidate for contraction. The purpose of scoring is to control the shapes of regions, which grow through the process.

The mergeability test it to decide whether or not an adjacent pair of regions can be merged if selected. We need a test of mergeability, which is a Boolean value calculated for each arc, because without proper termination conditions, a sequential contraction based on suitability scores always ends up with the smallest number of regions satisfying the initial constraints (i.e., BOUNDARY arcs). For example, if there is no BOUNDARY arc, the entire surface would always become one large region. The purpose of the mergeability test is to terminate the algorithm in a desirable state.

4.2 Geometric Indices for Scoring and the Mergeability Test

In Section 3.1, we listed three items: (1) Area Size, (2) Boundary Smoothness, and (3) Region Flatness, to measure the quality of a region. In this section, we describe indices to concretely calculate a score and assess mergeability.

4.2.1 Area Size

We employ an area index for the mergeability test of a pair of regions. The area index helps each region to grow larger than the prescribed threshold area size, since an extremely small region always makes a small mesh element.

The area index, A, is defined as:

$$A = \left(\frac{S_0}{S_1}\right)^k + \left(\frac{S_0}{S_2}\right)^k \tag{1}$$

where S_1 and S_2 are the areas of the two regions, S_0 is the threshold size of area, and k is a positive constant. This index favours the merger of a pair when either S_1 or S_2 is smaller than S_0. By making the mergeability test return true when $\lim\limits_{k \to \infty} A$ is smaller than 2, it is guaranteed that there will remain no region smaller than S_0 in the final clustering. S_0 should be at least the size of the desired mesh element, and is preferably larger except for extremely coarse meshes.

4.2.2 Boundary Smoothness

Boundary smoothness reflects a preference for wide, rounded regions over narrow, angular regions. Even if the area of a region is large enough to contain several mesh elements of the desired size, if it has an ill-shaped boundary, the resulting elements will be poor.

We employ two kinds of indices:

- Ratio of the shared boundary, R_{sh}
- Average improvement of the contact angle, Δ

Ratio R_{sh} contributes to the common boundary shared by the face pair and is defined as:

$$R_{sh} = \max\left\{\frac{l}{L_1}, \frac{l}{L_2}\right\} \tag{2}$$

where l stands for the length of the shared boundary, while L_1 and L_2 stand for the perimeters of the two regions, respectively. As shown in Fig. 4, a larger value of R_{sh} implies that two regions share a relatively longer boundary.

The average improvement of the contact angle, Δ, reflects the contribution towards simplifying the boundary. The value is calculated by the following equation:

$$\Delta = \frac{1}{N}\sum^{N}\frac{(\phi_i - \alpha_i) + (\phi_i - \beta_i)}{2} \tag{3}$$

where N is the number of endpoints of discrete segments of the shared boundary (usually equals 2), while α, β, and ϕ are angles at an endpoint between each pair of boundaires as shown in Fig. 5. These angles are between 0 and π, and the ideal value of ϕ is π, which means the merger would produce a straight line at the junction point. A positive value of Δ implies that the merger of the pair contributes to making the boundaries simple.

4.2.3 Region Flatness

For region flatness, we consider four kinds of indices:

- Gauss map angle, ν
- Codirectionality, θ
- Variance of the normal vector, σ^2
- Partial derivative of σ^2, denoted λ

The Gauss map angle, ν, is defined as the half angle of the cone whose axis is aligned with the average normal vector of a region, $\boldsymbol{m}$, and that circumscribes the Gauss map [14] of the region (see Fig. 6). A smaller value of ν contributes to a better score, because the cosine of ν corresponds to the maximum distortion (i.e., shrinkage) caused by the parallel projection from the surface to the working plane whose normal is $\boldsymbol{m}$. The cosine of ν is ideally 1, and for a degenerate case, 0. The value of ν needs to be smaller than $\pi/2$ in order to guarantee a one-to-one mapping by the projection.

The codirectionality of a pair of regions, θ, is an angle between their average normal vectors. A small value of θ indicates that by the merger of the pair the distortion caused by the projection between the surface and a new working plane will not be enlarged significantly. The increase of the maximum distortion also depends on the Gauss map angles of the two regions.

Furthermore, we define the variance of the normal vector, σ^2, as:

$$\sigma^2 = \frac{\int \|\boldsymbol{n} - \boldsymbol{m}\|^2 \, dS}{S} \tag{4}$$

where $\|\cdot\|$ represents a L_2 norm, and the vector $\boldsymbol{n}$ represents a normal vector at a point on the surface as shown in Fig. 6.

Although σ^2 cannot be used to guarantee a one-to-one mapping or to limit the maximum distortion by the projection like the Gauss map angle can, the merit is that σ^2 is not overly affected by the minor features in the region that should be ignored. Such a feature often consists of one or more faces which are small in size but have large normal variations (i.e. ν). Existence of those features in a region drives ν over $\pi/2$, making the index useless. Therefore, the variance σ^2 is useful to evaluate the overall distortion over the region rather than the maximum distortion. As long as the relevant portion is small, the loss of the one-to-one mapping does not cause a serious defect in the final mesh quality.

In practice, values for these three indices are calculated using normal vectors at a limited number of locations on the face boundary by assuming the curvature is roughly constant over the face. For a merged region, the values are calculated from those of the component regions. For example, the value of σ^2 for a merged region can be approximated [15] by:

$$\sigma^2 = \frac{S_1\sigma_1^2 + S_2\sigma_2^2}{S_1 + S_2} + \frac{S_1 S_2}{(S_1 + S_2)^2} \|\boldsymbol{m}_1 - \boldsymbol{m}_2\|^2 \tag{5}$$

where S_i, σ_i^2, and $\boldsymbol{m}_i$ $(i = 1, 2)$ are respectively the area, the variance of normal vectors, and the average normal vector for each region.

From the above equation, we define the last index, the partial derivative of σ^2, λ as follows:

$$\lambda = \max\left\{\left(\frac{\partial \sigma^2}{\partial S_1}\right)_{S_1=0}, \left(\frac{\partial \sigma^2}{\partial S_2}\right)_{S_2=0}\right\} \tag{6}$$

A large value of λ indicates that the variance increases rapidly when the two faces are merged, which is undesirable. This index focuses on the change of the overall distortion caused by the merger, regardless of the size of each region.

4.3 Combining Indices

The geometric indices described in the previous section reflect the respective characteristics of a region or improvements caused by a merger. Using geometric indices we define a score function so that it realises the desired strategy of clustering. The most primitive way is to use only one of the indices as a score function. To allow a more tailored algorithm, a weighted sum of indices is introduced, while the most general form of the score function can be written as:

$$F(r_1, r_2) = F(g_0(r_1, r_2), g_1(r_1, r_2), \ldots) \tag{7}$$

where r_1 and r_2 are two regions and g_i is a geometric index. The mergeability criterion can be, in the simplest form, an inequality involving a geometric index or a score function with a prescribed threshold. More generally, it is a combination of inequalities about geometric indices.

Since different indices reflect different aspects for a pair of regions, so improving one index often leads to worsening another. Therefore, the score function and mergeability criterion should be tuned by experimenting on many sample CAD models from the intended domain.

5 Results and Discussions

The proposed method was implemented in C and executed on an IBM PowerStation RS/6000 Model 43P (PowerPC 604e, 375MHz). In the following experiments, we used a CAD model of a mechanical part shown in Fig. 7, which consists of 1742 faces. There are many tiny or thin faces that are smaller than the desired mesh element size, which is 30(mm).

Table 1 shows four sample settings; Type I, II, and III use one of the indices described in Section 4 as a score function, and the last one (Type IV) employs a hybrid score function. As for the conditions of a well-shaped region, Type I and II consider region flatness only, while Type III considers boundary smoothness only. The hybrid strategy, Type IV, takes care of both conditions using a weighted sum with tentative coefficients. The area size condition is taken into account in the mergeability test for all settings. For calculation of $\underset{k \to \infty}{A}$, allowable minimum area S_0 is set to 1500 (mm^2) (almost 0.1% of the entire area).

We first demonstrate what clusterings the three primitive strategies (Type I-III) generate. We experimented with three levels of granularity; where the 1742 faces were clustered into approximately 30, 80, and 150 groups for each strategy.

Fig. 8(a)-(c) show the results of codirectionality-based clustering (Type I) at the three levels with the threshold θ_0 to 30.0, 50.0, and 80.0(deg). The total numbers of clusters (i.e., regions) are 148, 81, and 33, respectively. The larger threshold produces a smaller number of clusters as we expected. The obtained clusters seem to be flat, but some clusters have poorly shaped boundaries. Fig. 8 (d)-(f) show the results of clustering based on normal vector variance (Type II). The threshold value, σ_0, was set to 1.5, 11.5, and 20.0(deg), generating 147, 80, and 32 clusters, respectively. The larger σ_0^2 naturally produces a smaller number of clusters. The character of Type II clusterings is similar to that of Type I. Fig. 8 (g)-(i) show the opposite approach to clustering, in which boundary smoothness is considered. By setting the threshold R_0 to 0.60, 0.46, and 0.40, we obtained 146, 81, and 34 clusters, respectively. Clusters at each granularity have well-shaped boundaries in comparison with the clusters generated with Type I and Type II settings.

Since we prefer a clustering that satisfy all conditions for a well-shaped region, we tested the hybrid strategy (Type IV). To evaluate quantitatively the hybrid strategy and the other ones, we introduced two metrics that indicate the well-shapedness of a region:

- R_l : Ratio of the perimeter of the region to the circumference of a circle having the same area
- σ : Standard deviation of normal vector (i.e., square root of variance of normal vector)

We evaluated the above metrics for regions in each clustering, and calculated average and maximum values. Fig. 9 shows some results for Type I to IV and for two reference clusterings, which are:

- A clustering with only one region that covers the whole
- A clustering where each face makes its own cluster (total of 1742 regions)

The statistics show regions in Type I and II clusterings have more poorly shaped boundaries (i.e., larger values of R_l) compared with Type III; however, they are flatter than regions obtained by Type III. By those metrics it is clear that final clusterings can be controlled by geometric indices used in a clustering strategy. As to the hybrid strategy, metrics show the resultant clustering achieves well the multiple goals set by the two geometric indices. The key is to heuristically find the best configuration of the geometric indices which define the clustering strategy.

We also evaluated the quality of the final meshes generated using the regions in the clusterings. The following metrics were adopted:

- Ratio of warped elements (a pair of triangles in a quadrilateral element forms a dihedral angle greater than 30 degrees)
- Ratio of elements that have short (under 0.5 times the given element size) or long (over 1.5 times the given element size) edges
- Ratio of elements that have small (under 30 degrees) or wide (over 135 degrees) angles

Fig. 10 shows the values of the above metrics for each clustering. It is reasonable that a large number of clusters (i.e., fine decomposition) results in many poorly shaped elements because regions are inevitably small and the boundary constraints become intractable. Meshes generated by using hybrid clustering show the best or second best ratios for ill-sized or ill-angled elements and medium ratios for warped elements. This shows that the well-shaped regions generated by the hybrid clustering contribute much to the final mesh quality.

Finally, a sample clustering with 15 clusters using the hybrid strategy is shown in Fig. 11. A final merged mesh is also shown in Fig. 12.

6 Conclusion

We have presented a method for clustering a large number of faces for the purpose of surface mesh generation. To deal with the large CAD models actually used in industry, decomposition of the whole model into well-sized, well-shaped regions contributes significantly to the improvement of the quality of the generated mesh. Our central idea is a sequential merger of adjacent regions. The most promising pair for making a well-shaped region is selected based on a score that is a combination of geometric indices. The results of experiments show that a hybrid strategy of combining different geometric indices actually realises clusterings that are reasonably good in multiple respects. Evaluation of quality metrics on obtained meshes also proved that those geometrically fair clusterings contribute much to the quality of meshes. By configuring the combination of geometric indices and tuning parameters, the method is customisable to various applications that have different requirements on the well-shapedness of a region.

One limitation of our approach is that a face with a large variation of normal vector (e.g., a half cylinder) or with a long complicated boundary can cause poorly shaped regions. When such faces are small compared to the desired element size, they are properly ignored. But otherwise, essential to a good solution is either to break such faces before clustering or to extend the surface types as mentioned below. It is also a potential limitation that our algorithm always selects the best available pair at a particular moment in the clustering process. This does not always lead to an optimal solution. To obtain a better result, 'tactical' merging might be necessary, such as accepting an ill-shaped region at first and then merging it into a well-shaped region.

As future work, the type of simplified geometry for approximating the original surface should be extended to other type of surfaces. The cylindrical surface is a promising one since mesh generation over 'developable surfaces' [14] such as a cylindrical surface can be done in the same way as that over a plane. We are also thinking of introducing a post-processing function that will produce better shaped regions by locally exchanging faces between two regions. Another topic would be a systematic way of tailoring a clustering strategy, which consists of the combination of geometric indices and of determining threshold values for mergeability tests.

7 REFERENCES

[1] Shimada, K. Physically-Based Mesh Generation: Automated Triangulation of Surfaces and Volumes. PhD Thesis. Massachusetts Institute of Technology, USA, 1993.

[2] Shimada K, Itoh T. Automated Conversion of 2D Triangular Mesh into Quadrilateral Mesh. Proceedings of International Conference on Computational Engineering Science '95, 1995. p. 350-355.

[3] Shimada K, Yamada A, Itoh T. Anisotropic triangulation of parametric surfaces via close packing of ellipsoids. Proceedings of 6th International Meshing Roundtable, 1997. p. 375-390.

[4] Shimada K, Liao JH, Itoh T. Quadrilateral Meshing with Directionality Control through the Packing of Square Cells. Proceedings of 7th International Meshing Roundtable, 1998. p. 61-75.

[5] Schroeder WJ, Zarge JA, Lorensen WE. Decimation of Triangle Meshes. SIGGRAPH '92 Conference Proceedings, 1992. p. 65-70.

[6] Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W. Mesh Optimization. SIGGRAPH '93 Conference Proceedings, 1993. p. 19-26.

[7] Garland M, Heckbert PS. Surface Simplification Using Quadric Error Metrics. SIGGRAPH '97 Conference Proceedings, 1997. p. 209-216.

[8] Willmott AJ, Heckbert PS, Garland M. Face Cluster Radiosity. Eurographics Workshop on Rendering, 1999. p. 293-304.

[9] Shephard MS, Beall MW, O'Bara RM. Revisiting the Elimination of the Adverse Effects of Small Model Features in Automatically Generated Meshes. Proceedings 7th International Meshing Roundtable, 1998. p. 119-131.

[10] Sheffer A, Blacker T, Clements J, Bercovier M. Virtual Topology Operations for Meshing. Proceedings 6th International Meshing Roundtable, 1997. p. 49-66.

[11] Sheffer A, Blacker T. Bercoivier M. Clustering: Automated Detail Suppression Using Virtual Topology. Trends in Unstructured Mesh Generation, 1997. p. 57-64.

[12] Mobley AV, Carroll MP, Canann SA. An Object Oriented Approach to Geometry Defeaturing for Finite Element Meshing. Proceedings of 7th International Meshing Roundtable, 1998. p. 547-564.

[13] Marcum DL, Gaither JA. Unstructured Surface Grid Generation Using Global Mapping and Physical Space Approximation. Proceedings of 8th International Meshing Roundtable, 1999. p. 397-406.

[14] Hsiung C. A First Course in Differential Geometry. John Wiley & Sons, 1981.

[15] Lindgren BW, Mcelrath GW. Introduction to Probability and Statistics. The Macmillan Company, 1969.

Table 1. Settings of scoring and mergeability

Type	Score function	Mergeability test
I: codirectionality	$F = -\theta$	$\theta < \theta_0 \quad \vee \quad \underset{k\to\infty}{A} < 2$
II: variance of normal vector	$F = -\lambda$	$\sigma^2 > \sigma_0^2 \quad \vee \quad \underset{k\to\infty}{A} < 2$
III: ratio of shared boundary	$F = R_{sh}$	$R_{sh} > R_0 \quad \vee \quad \underset{k\to\infty}{A} < 2$
IV: combination of I & III	$F = R_{sh} - \dfrac{\theta}{5\pi}$	$\theta < \theta_0 \quad \vee \quad \underset{k\to\infty}{A} < 2$

Figure 1 Three-step solution.

For top-left: input CAD model
For top-right: Step 1: clustering
For bottom-left: Step 2: meshes of regions
For bottom-right: Step 3: merged mesh

Figure 2 Conversion from faces to a graph

For left: (a) a set of faces
For right: (b) an adjacency graph

Figure 3 Arc contraction.

Figure 4 Ratio of shared boundary.

Fop left to right: $R_{sh} = 0.2$ $R_{sh} = 0.3$ $R_{sh} = 0.5$ $R_{sh} = 0.8$ $R_{sh} = 1.0$

Figure 5 Angles at a contact point.

Figure 6 Gauss map angle and average normal.

Figure 7 Input CAD model with 1742 faces.

Figure 8 Clusterings with different types and thresholds.

Top row
Fop left: (a) Type I, θ_0=30.0(deg) (148 clusters)
For middle: (b) Type I, θ_0=50.0(deg) (81 clusters)
For right: (c) Type I, θ_0=80.0(deg) (33 clusters)
Middle row
Fop left: (d) Type II, σ_0 = 1.5(deg) (147 clusters)
For middle: (e) Type II, σ_0 = 11.5(deg) (81 clusters)
For right: (f) Type II, σ_0 = 20.0(deg) (33 clusters)
Bottom row
Fop left: (g) Type III, R_0 = 0.60 (146 clusters)
For middle: (h) Type III, R_0 = 0.46 (81 clusters)
For right: (i) Type III, R_0 = 0.40 (34 clusters)

Figure 9 Comparison of four types of clusterings.

For top: (a) Average of R_l
For bottom left: (b) Average of σ (deg)
For bottom right: (c) Maximum of σ (deg)

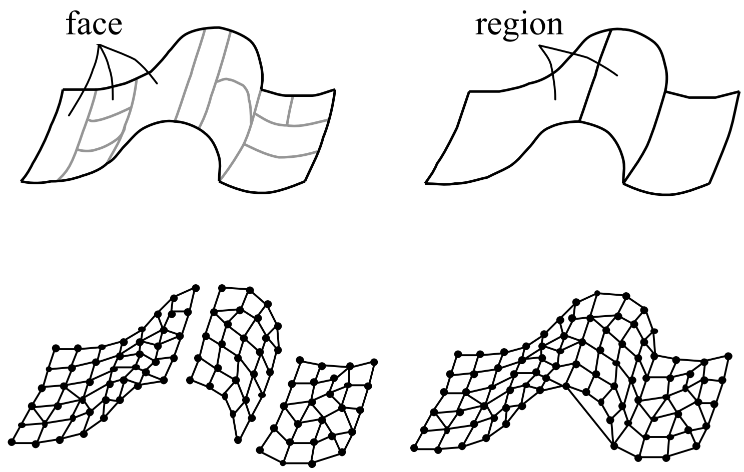
Figure 10 Comparison of mesh qualities.

For top: (a) Ratio of warped elements
For bottom left: (b) Ratio of ill-sized elements

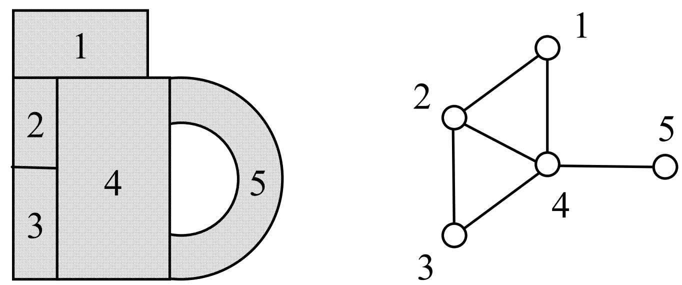
For bottom right: (c) Ratio of ill-angled elements

Figure 11 Clustering by Type IV setting (θ_0 =73(deg), S_0=20000(mm^2), 15 clusters).

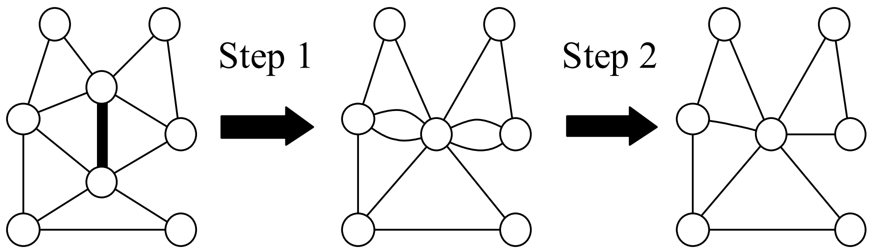
Figure 12 Merged mesh.



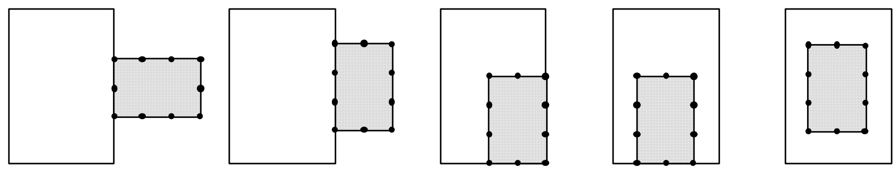
The Journal of Computer-Aided Design, Keisuke Inoue, Figure 1



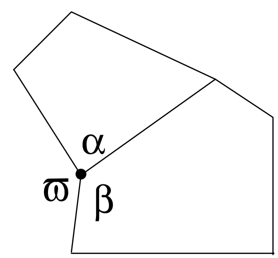
The Journal of Computer-Aided Design, Keisuke Inoue, Figure 2



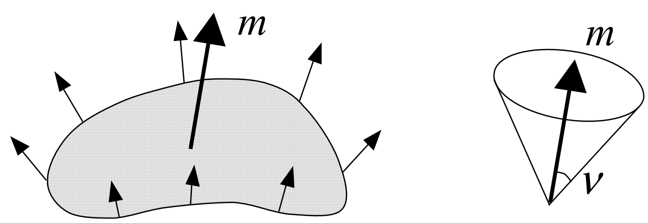
The Journal of Computer-Aided Design, Keisuke Inoue, Figure 3



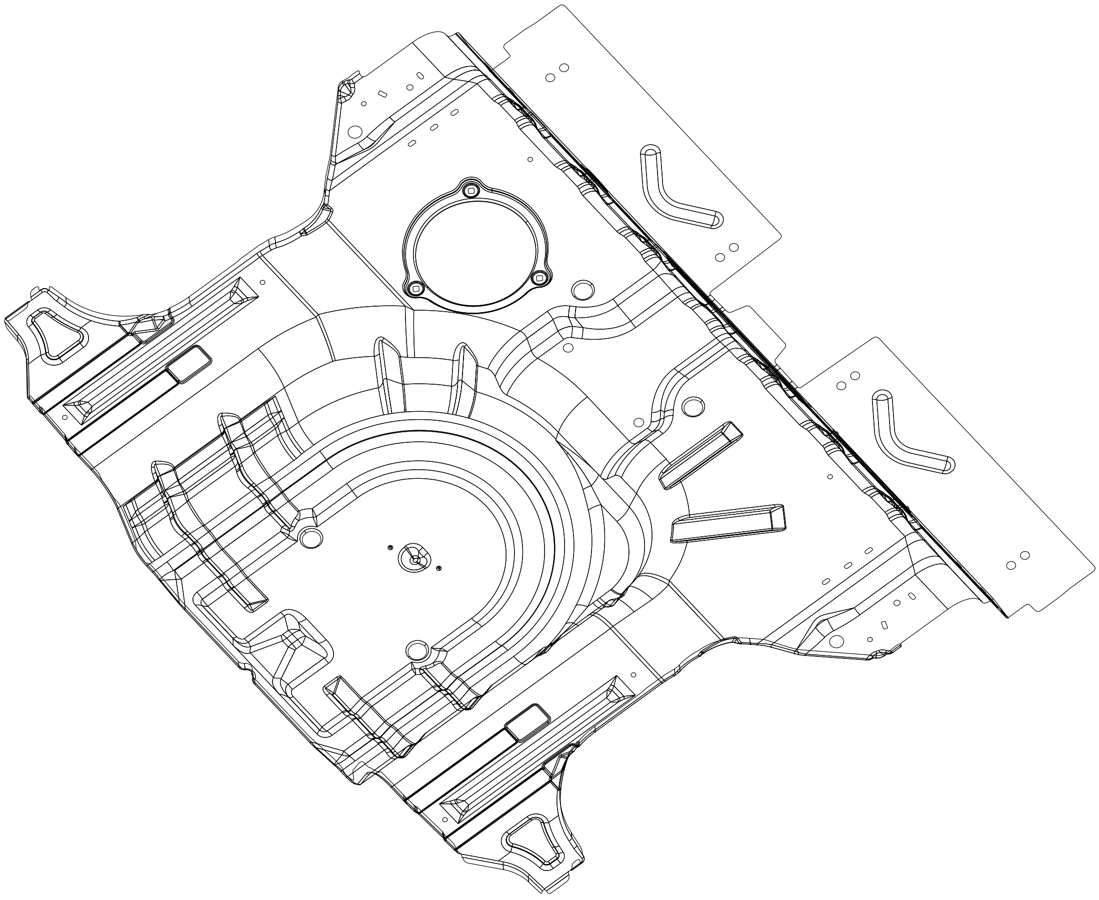
The Journal of Computer-Aided Design, Keisuke Inoue, Figure 4



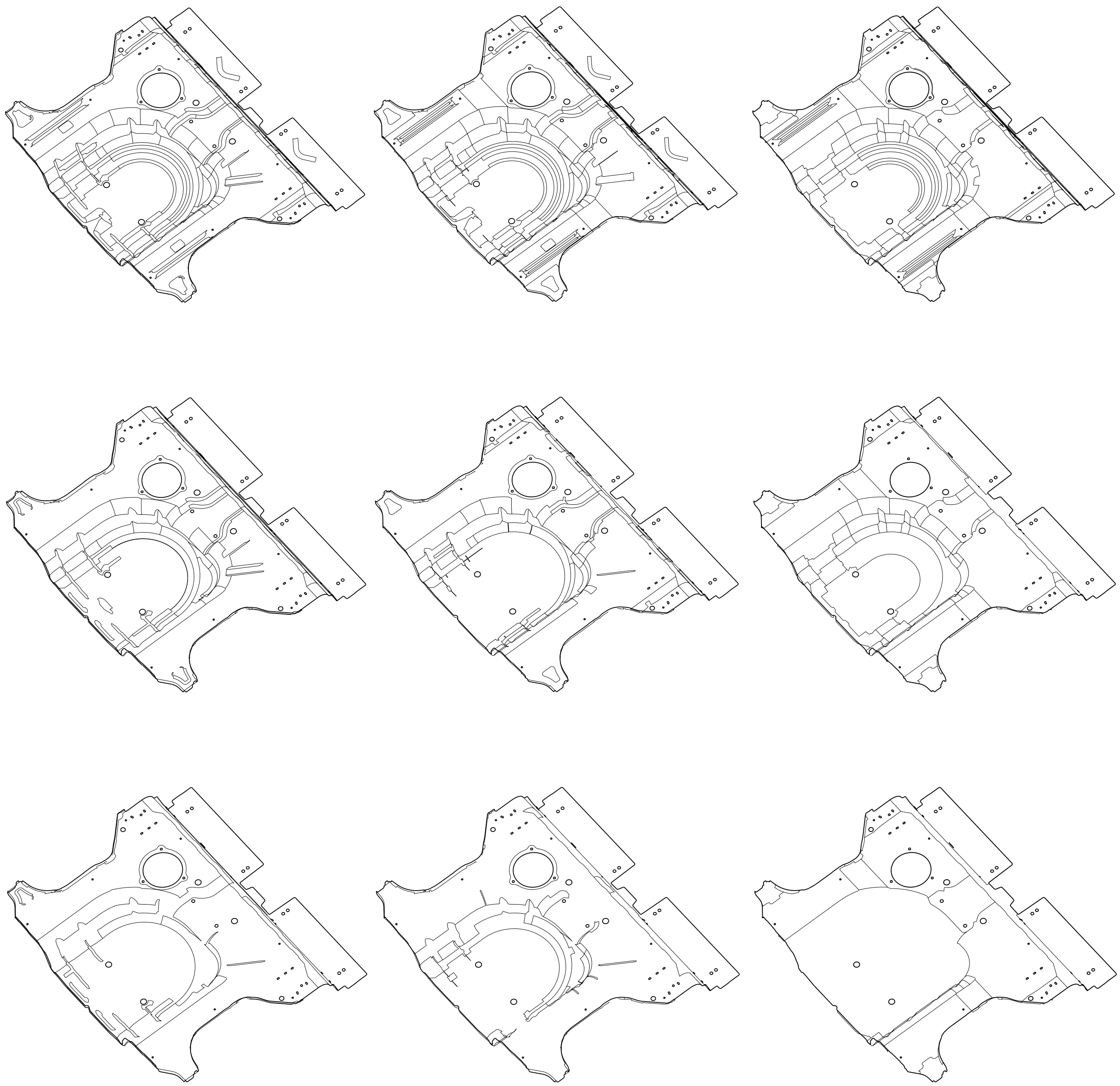
The Journal of Computer-Aided Design, Keisuke Inoue, Figure 5



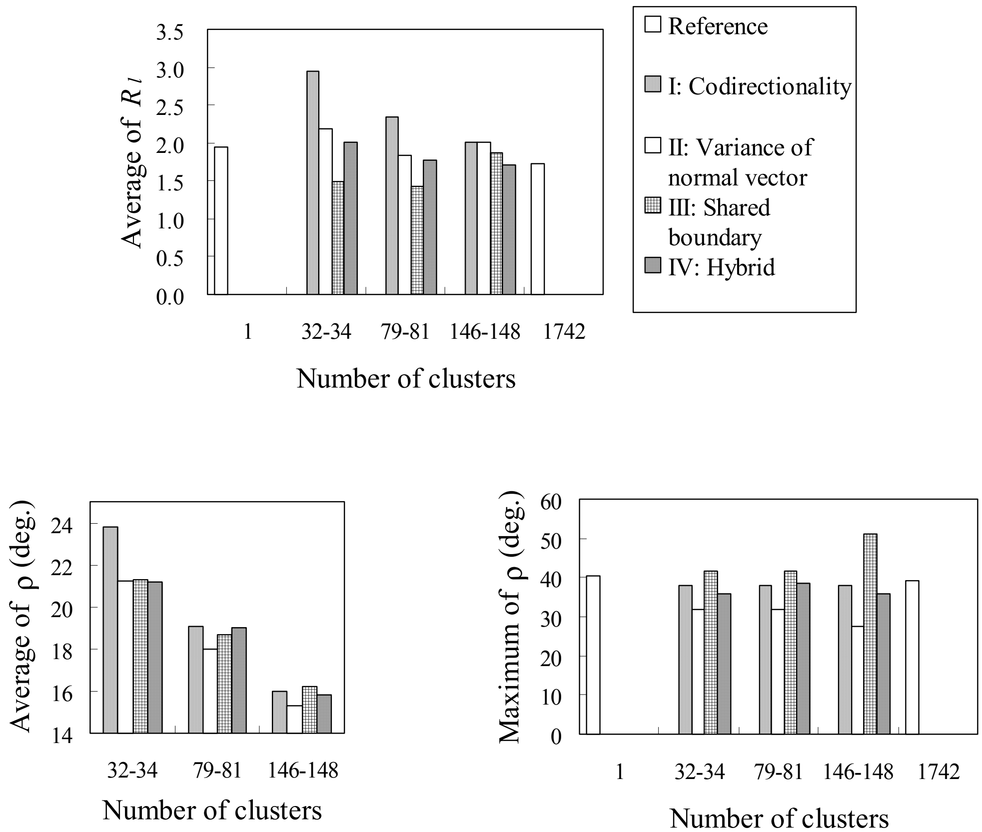
The Journal of Computer-Aided Design, Keisuke Inoue, Figure 6



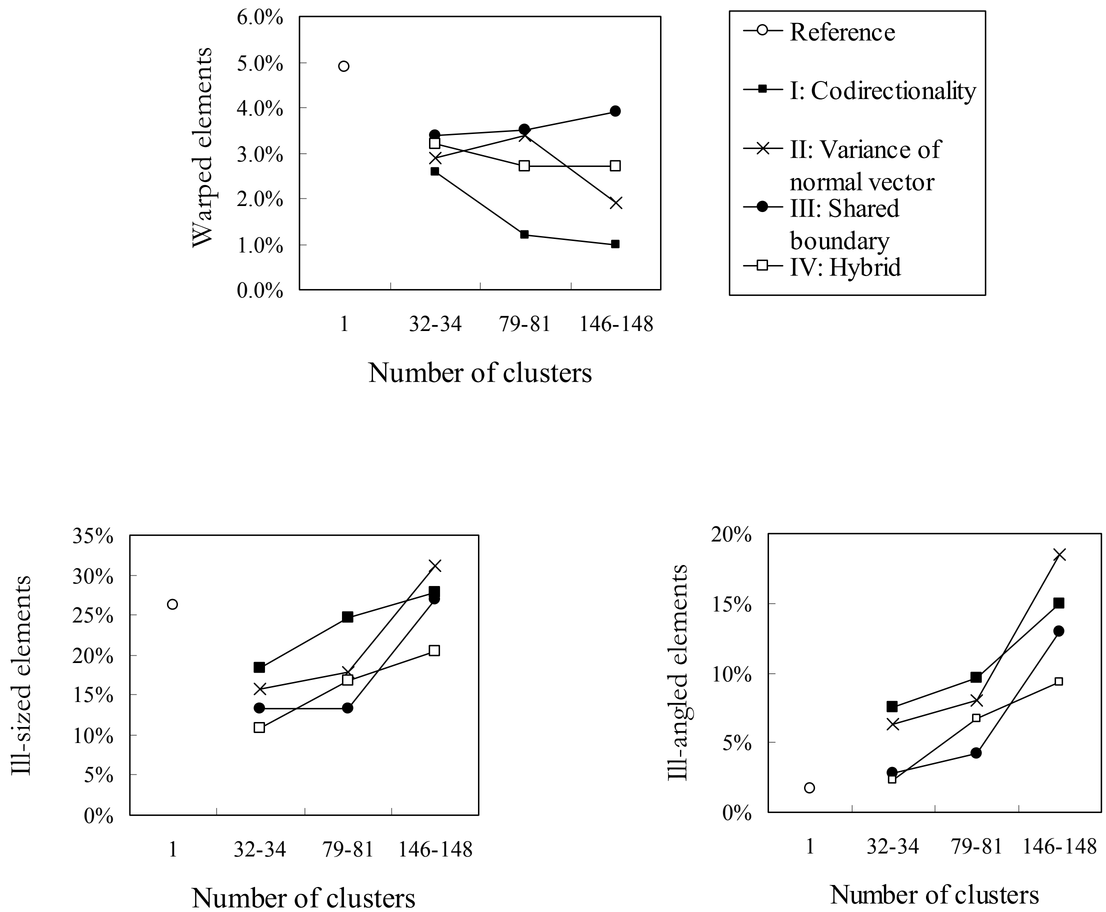
The Journal of Computer-Aided Design, Keisuke Inoue, Figure 7



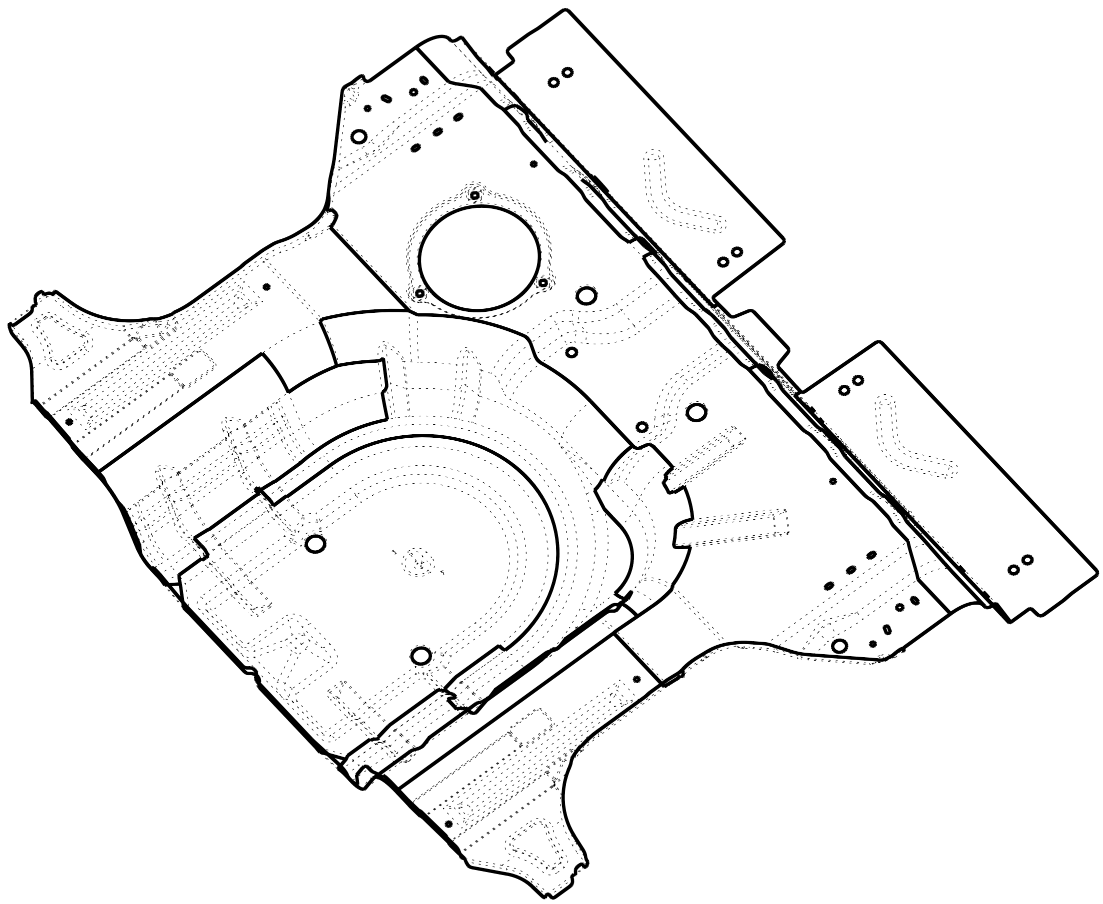
The Journal of Computer-Aided Design, Keisuke Inoue, Figure 8



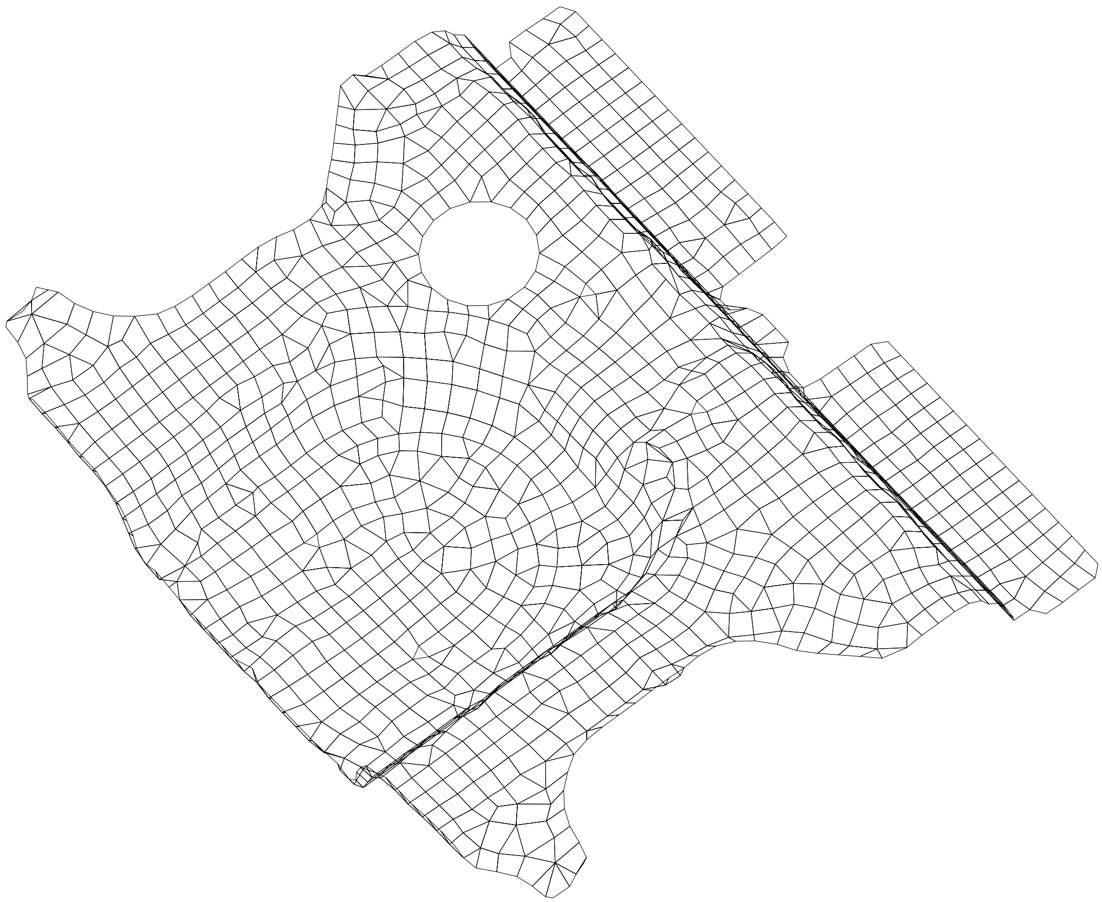
The Journal of Computer-Aided Design, Keisuke Inoue, Figure 9



The Journal of Computer-Aided Design, Keisuke Inoue, Figure 10



The Journal of Computer-Aided Design, Keisuke Inoue, Figure 11



The Journal of Computer-Aided Design, Keisuke Inoue, Figure 12