COMPUTER-AIDED DESIGN & APPLICATIONS, 2017
VOL. 14, NO. 6, 751–759
https://doi.org/10.1080/16864360.2017.1287677

A fast and automatic hole-filling method based on feature line recovery

Chunhong Xia and Hui Zhang

Tsinghua University, China

ABSTRACT

In this paper, an automatic method to fill holes in triangle mesh models is proposed. Different types of holes can be handled by recovering feature lines that pass through them, and then the recovered feature lines are used for splitting complex holes into small and simple ones, which simplifies the hole-filling problem. Two kinds of splits are considered, curve split and corner split. For curve split, Euler spirals are constructed as feature lines. The required end-points and their corresponding tangent vectors can be automatically identified to initialize the curve. For corner split, the corner point contained in the hole is recovered by optimizing a quadric equation, then multiple feature lines can be constructed automatically. An advantage of this method is that it works on the neighborhood of each hole, which makes it possible to handle huge models with high efficiency compared to other state-of-the-art methods.

KEYWORDS

Hole-filling; automatic; feature line

1. Introduction

With the fast development of data acquisition equipment, such as LIDAR, Kinect, it becomes much easier to get the three dimensional representation of a real object. Nevertheless, due to constraints of the equipment and surrounding conditions, the digital data acquired is usually incomplete and contains flaws. As a result, the triangulated mesh model always contains self-intersections, gaps, and holes, and might bring errors to following process or applications. Self-intersections can be solved by adding and deleting mesh faces based on topology, and gaps can be filled by simply connecting them together. The problem is how to deal with holes properly. Holes usually appear at unexpected areas and always have complex boundaries and topology. What's worse, the shape of the missing part is unclear. In this way, hole-filling plays a challenging and indispensable role in the fundamental process of 3D models.

Now that hole-filling is such an important issue to deal with, many effective methods have been employed, which can be grouped into two categories: volume-based methods and mesh-based methods. The basis of volume-based methods is the voxelization of input mesh models. This type of methods process the model globally. Filippov[5] set three requirements to voxels: separability, accuracy, and minimality, in order to ensure a good discrete representation. To be specific, separability means that the discrete surface can avoid ray penetration; accuracy entails requirements on error metrics to ensure that discrete surface satisfies the separability condition; minimality is achieved when the discrete surface contains no voxels that of no use. Nooruddin[16] presented two new methods of voxelization: parity-count and ray-stabbing, which help identifying whether a voxel is interior or exterior to the model by the number of intersections. Admittedly, Nooruddin's method could handle models with holes, double walls, and intersecting parts, however, they worked merely on a global volume space which is very hard to be constructed. In addition, a tiny fault in ray-stabbing may cause a large part on the model of wrong sign. Ju[10], employing an octree grid, converted the input model into a volume, which saved much space and made it possible to deal with huge and complex models. Afterwards, Ju et al., labeled each edge of the octree grid, managed to show the sign of each vertex and reconstruct the surface by contouring. This method is robust and can process models in high efficiency, but cannot recover the missing feature properly. In general, volume-based methods can handle complex holes and get harmonious results, but they are relatively time- and space-consuming.

This research mainly focuses on mesh-based methods, which are easier to implement and consume less time and space. Among these, some manage to fill holes directly by using Advancing Front Methods (AFM), triangulation or Radial Basis Function (RBF). For example, Zhao [6]

CONTACT Hui Zhang huizhang@tsinghua.edu.cn

filled the holes with AFM, and then used topology adjustment to refine newly-inserted faces. In their research, Zhao et al., calculated the desirable normal of each face, rotated the face to the normal's direction, and recovered the missed feature by solving a possion equation. Sharf[2] presented a context-based method. Having found the most similar part to the hole on the model or among a large set of examples, they replaced the filled part with the similar part so that sharp features could be restored. Moreover Sharf's method enabled the requirement of mesh models with textures. Harary[7]'s research, which was based on the similarity descriptor, was an extension of Sharf's method[2] and was able to handle more complex models with better results. Moraru[15] came up with a toolbox to fill the holes on the mesh model. Firstly, they cleaned the hole boundary due to scanner noise, and then used a topological grid built from boundary vertices and an ellipse to fill the hole. Finally, the grid was deformed to satisfy the condition set by surrounding meshes of the hole. Moraru's toolbox worked well on holes with topology like an ellipse. As for other complex topology, future work is needed to adapt the topology and shape of the grid to the original hole. Generally speaking, although the methods mentioned above are effective in many cases, they cannot fill well the holes in large size and with complex topology. Consequently, scholars come up with some methods to split holes into small ones. Jun[11] split holes into sub-holes and applied smoothing and optimizing methods to repair them one by one, however, the process occasionally iterated too many times to converge. Ohtake[1] proposed a hierarchically piecewise function to split a hole, which made it possible to figure out the corner point and feature boundary in it. In order to get the function model suitable for the surface, Ohtake et al., had to conduct many experiments. Apparently the procedure required much time and simplified complicated constraints. What's more, in some cases, Ohtake's function cannot ensure the correctness of the results. Li [13] applied mixed polynomial to restore sharp features in the hole region. They first detected feature points along the hole boundary, and then recovered feature lines with the equation, so that each hole was split into several small ones and was filled by Bézier-Lagrange surface. Ngo [12] made crest-line detection for feature points recognition, and then used Catmull-Rom curve to interpolate sample points of the feature line into the hole area. Then, they projected small holes onto a 2D tangent plane for triangulation. This method is a semi-auto one, which asks for user intervention. Among the above methods, most cannot fill or split the holes automatically, even worse, some are time-consuming. Besides, when it comes to holes of large size and with complex topology, the result is hard to recover the original topology inside the hole.

In this research, the authors endeavor to recover the topology inside a hole by splitting it into small and simple ones. Two ways to split are proposed: curve split and corner split. The experiment starts with automatically recovering the corner points inside a hole and constructing feature lines by feature detection near the hole area. After that, sample points are interpolated along the feature curve to split the hole. During this process, the feature lines help to retain consistency between the repaired part and its neighborhood.

2. Algorithm overview

The method proposed in this research operates on triangular mesh models, which are supposed to be connected, manifold, and oriented, with no self-intersections. Moreover, no common points exist between any two holes, and no islands are allowed inside each hole. The input model is constructed from clean point clouds, with no noise. The mesh models are represented with half-edge data structure, and the holes can be filled one by one. It is worth mentioning that this novel method manages to split the hole in accordance with features. Two kinds of split are considered as shown in Fig. 1: (a)curve split: the hole contains no corner point, each split line is constructed from two end-points on the hole boundary; (b)corner split: there exists a corner connecting several surfaces inside the hole, the corner point and a feature point on the hole boundary make up the two end-points of the split line.

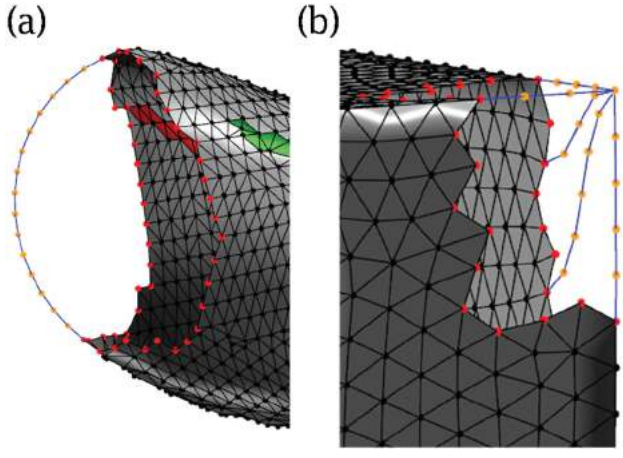

Figure 1. Illustration of two kinds of split and Euler spiral construction. (a)curve split; (b)corner split. Red points are points on the hole boundary, blue lines are recovered feature lines, orange points are sampling points on feature lines.

The whole procedure of this algorithm is: (a)decide whether the hole needs to be split, and how to split, i.e. make curve split or corner split; (b)apply different feature

detecting method according to the split type; (c)construct split lines with feature points and corner point(if any) to divide the hole into sub-holes; (d)repair each sub-hole by AFM. In order to determine the split type, first the normals of 1-ring faces of the hole are collected, then k-means clustering is applied. If the normals distribute evenly in a small range or if only one cluster exists, it implies that these normals change slightly along the hole, so the hole does not need to be split; if they distribute into two ranges, two clusters will be obtained, then the curve split approach is a better choice; and if they distribute into several clusters, the corner split approach will be chosen. In addition, users can choose the split type manually. The flow diagram is shown in Fig. 2.

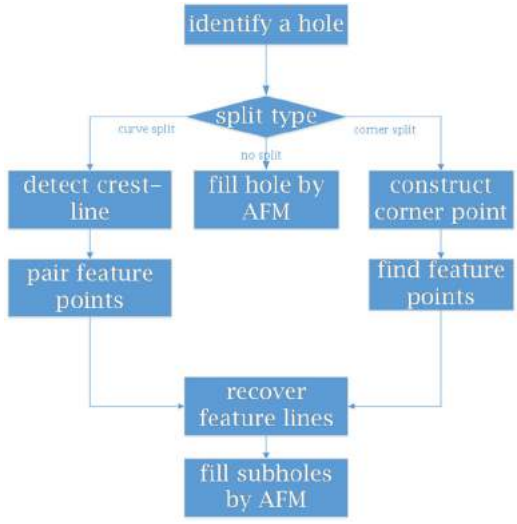

Figure 2. Flow diagram for the hole-filling procedure.

The main contribution of this work can be summarized as follows:

- An automatic method that can split holes containing corner points is presented;
- The number of surfaces intersected at the corner can be obtained by clustering face normal of the 1-ring hole boundary faces;
- The consistency of the recovered mesh can be retained with the neighborhood of the hole.

3. Curve split construction

Split curve construction is the key point in splitting complex holes. With boundary conditions, a common way is to construct Hermite curves, but Euler Spiral [14] seems to be more eye-pleasing: extensible, invariant to similarity transformation, symmetric, smooth and round. Therefore, it is much preferred by the human visual system [9]. To fit an Euler spiral, the two end-points of the curve and their corresponding tangent vectors are needed. In Harry's work [8], researchers chose two end-points and two other points manually to determine the corresponding tangent vectors. By contrast, the point selection procedure, in this research, runs automatically.

First, crest-line detection [3] is used to locate ridge and ravine areas on the model, which are referred as feature faces. The ridge and ravine face strips are labeled in green and red respectively in Fig. 3 and Fig. 4. The boundary points that belong to feature faces are chosen as candidates of feature points. In order to determine which candidate is a feature point, the 1-ring neighborhood of the point is usually used. In this research, due to the fact that crest-line detection is not so robust and precise when a model is incomplete, e.g. with holes, both 1-ring and 2-ring faces of a point are used, so as to ensure the continuousness of feature and the accuracy of the result. To be specific, for each candidate point 'v', if feature face exists in both its 1-ring neighborhood and its 2-ring neighborhood, and the two faces are both ridge or ravine faces that share a common point or edge, then point 'v' is picked as a feature point, labeled in RIDGE or RAVINE.

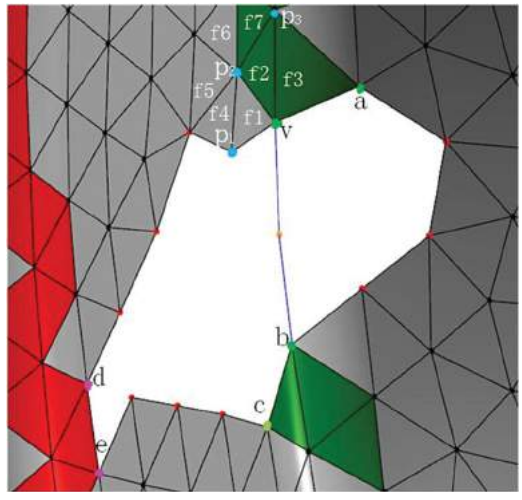

Figure 3. Feature points identification.

As shown in Fig. 3, faces $\{f_1, f_2, f_3\}$ are 1-ring neighbor faces of point 'v', and f_2, f_3 are feature faces. Likewise, the authors iterate the procedure from point p_2 to p_3 to find the 2-ring neighbor of 'v', and get faces $\{f_1, f_2, f_4, f_5, f_6, f_7\}$. Face f_1 and f_2 are 1-ring faces of 'v' and f_4 is on the boundary, so only faces f_5, f_6, f_7 are taken into consideration. Since f_7 is a feature face of 2-ring, f_2 is a feature face of 1-ring faces of 'v', and both f_2 and f_7 are ridge faces who share the same point 'p_2', then point 'v' is a RIDGE point. In Fig. 3, the points 'v', 'a', 'b', 'c' in green

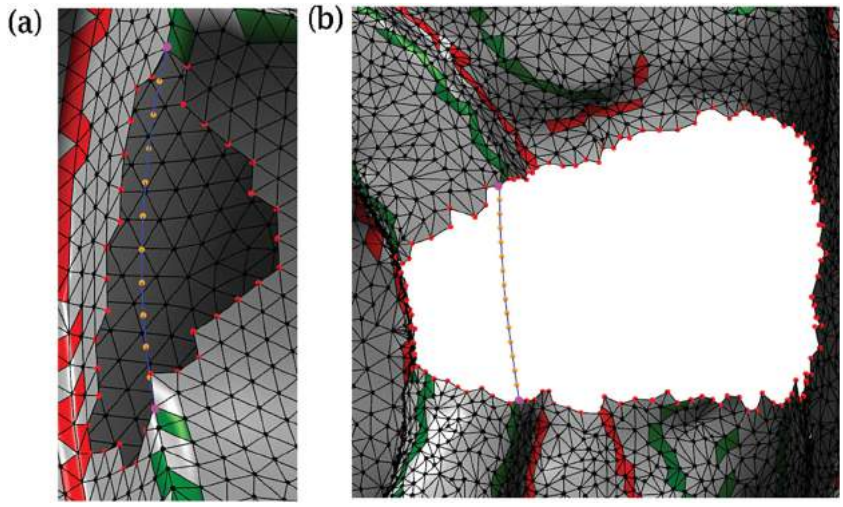

Figure 4. Illustration of a constructed Euler spiral. Purple points are the best pair. (a) The inserted feature line consistent to the feature line which goes through the hole; (b)Euler spiral splits a complex hole into two parts.

are RIDGE, points 'd', 'e' in purple are RAVINE. These ridge and ravine points are referred as feature points on the hole boundary.

The feature normal of a feature point is defined as the average normal of two boundary faces that share the point. What's more, for each feature point, its 1-ring faces lie on different sides of the feature line, so the two faces chosen to calculate the feature normal must belong to different sides. In this way, the feature normal conduce to maximize the difference of different feature lines and the similarity of the same feature line, so that the paired feature points will be more accurate. In Fig. 3, the feature normal of point 'v' is equal to the average of f_1's normal and f_3's normal. Then the feature points are paired. In order to ensure the correctness of pairing, some constraints are set in advance: (a)the two points paired together are both RIDGE or RAVINE; (b)the paired points are not neighbors; (c)the distance between the paired points along the hole boundary should be larger than a pre-set threshold; (d)the feature normal variance between the paired points should be the minimum among all the pairs. In (c), the distance is measured by the number of points between the paired points along the hole boundary, and the threshold is set to be 1/5 of the number of boundary points. The above constraints help ensuring that the two end-points are of the same feature, while not to be very close to each other along the hole boundary. The reason to satisfy the latter requirement is that if two points near to each other are paired, only a small part of the hole will be split, which has little contribution to the final result. Thus the second rule contributes to filtering out some wrong pairs. On each specific feature curve, the normal of each point is always similar, so the constraints on feature normal variance also help to find the best pair. For example, in Fig. 3, the best pair is {v, b}. In this research, the best pair is treated as the two end-points of the Euler split curve.

Afterwards, the tangent vector of each end-point is calculated. For instance, let D be the vector between two end-points, which indicates the trend that the curve moves on. For each end-point 'v', researchers search among its 1-ring points to find the vector. Although, each neighbor point can form a vector with point 'v', only the vector that has the minimum angle with D is referred as the tangent vector of point 'v'. In this way, researchers obtain the initial conditions to construct an Euler spiral. Fig. 4 shows the result of a constructed Euler spiral.

4. Corner split construction

First, Cao's corner recovery method [4] is employed to locate the corner point. It used boundary edges and boundary points along the hole to formulate two kinds of tetrahedrons. By optimizing a quad programming equation, the position of the corner point is acquired. Then the feature points on the hole boundary are to be found out. As is well-known, corner points usually appear on CAD models. In addition, the variance between changes to normals of adjacent faces is smaller within each surface; as for the faces separated by feature lines, the difference between them tend to be big. In this way, this research takes a different measurement for feature point detection. The first step is to get 1-ring neighbor faces of the hole boundary and their corresponding normal, after that, k-means clustering will be used to classify the face normal set. The problem lies in how to determine the value of k, which is equal to the number of surfaces that a corner point connects. Two concepts need

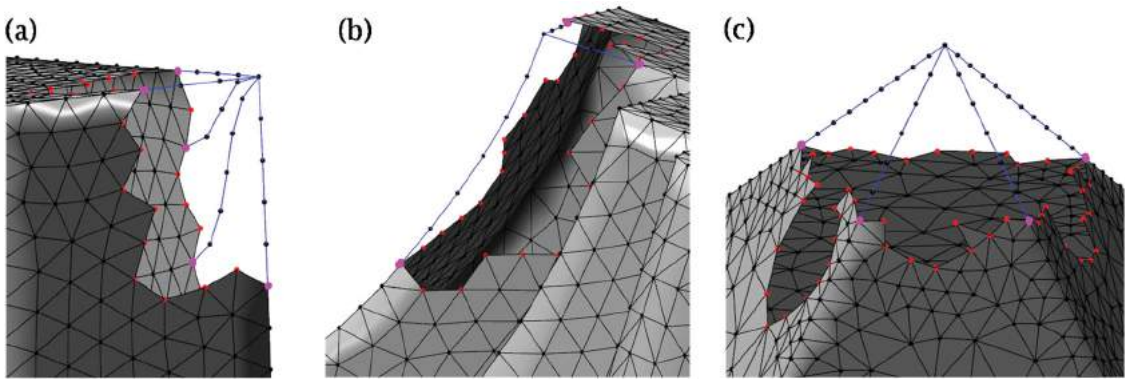

Figure 5. Illustration of feature points, purple points. (a) Five feature points are figured out on the hole boundary, three surfaces intersect together; (b)three feature points are figured out, three surfaces intersect together; (c)four points are figured out, four surfaces intersect together.

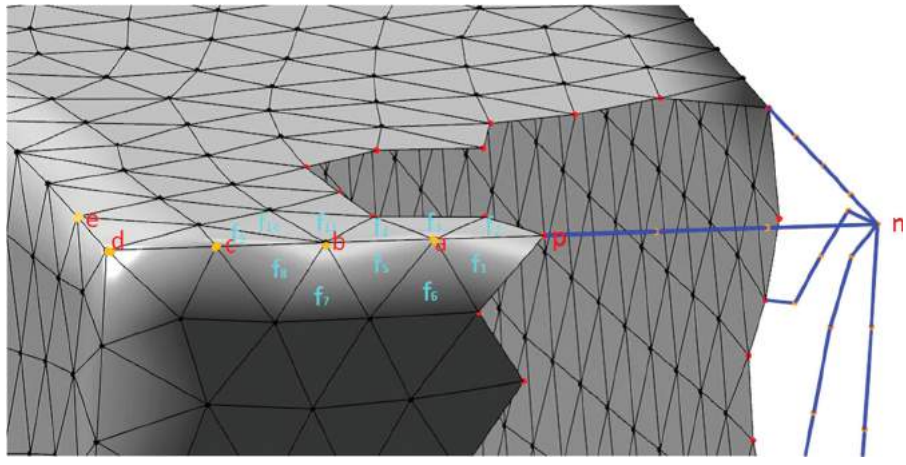

Figure 6. Illustration of chain point calculation.

to be defined: (1) face normal variance: the normal variance between two neighbor faces; (2)point-related face normal variance: the largest face normal variance among the 1-ring neighbor faces of a point. The k feature points are obtained in the following steps: (a)considering that a corner connects at least three surfaces, researchers iterate k from the lower bound(which is equal to three) to a pre-set upper bound, and denote the k with minimum variance as k_{min}; (b)iterate back from k_{min} to the lower bound, and set the k where each cluster has a reasonable number of elements as the number of surfaces a corner connects, which is denoted as k_s; (c)select k_s points on the hole boundary in decreasing order of point-related face normal variance. Here the upper bound is set to be 1/4 of the number of boundary vertex. If the upper bound is too large, then a cluster will contain only few elements, it is of none sense; if it is too small, the clustering will be at the risk of losing generality.

Another factor that has crucial impact on the result is the initial center of each cluster. Many methods choose the center randomly, but in this research, each center is decided by the largest distance between each other where every cluster can distribute evenly. In Fig. 5, the purple points illustrate the feature points figured out by our method. We observe that the number of feature points is larger than or equal to the exact surface number, but it will not bring negative effects on the final result. The reason is that for all the points on the hole boundary, only the ones on feature lines have larger point-related face normal variance. Therefore once these points are selected, whether other points are selected or not makes no difference.

Secondly, the split lines will be constructed with the points gathered above. In order to get better-looking results, Euler spiral is chosen to fit feature lines, just like in the curve split procedure. Therefore the tangent vector of each end-point needs to be calculated. For a feature point, such as point 'p' in Fig. 6, face pair $\{f_1, f_2\}$ has the largest normal variance among the 1-ring neighbor faces $\{f_1, f_2\}$, so point 'a' is chosen as a chain point. Likewise, it is possible to find the following chain points 'b', 'c', 'd', and 'e', and the line on which these chain points are connected

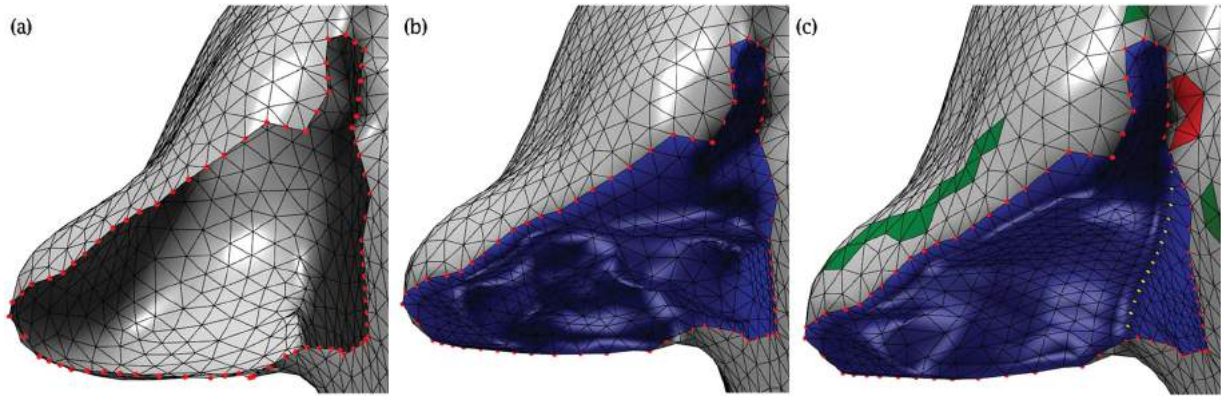

Figure 7. Comparison of hole-filling part on a pig model. (a) The original hole of a pig ear; (b)the hole filled directly by AFM; (c)the hole filled by curve split.

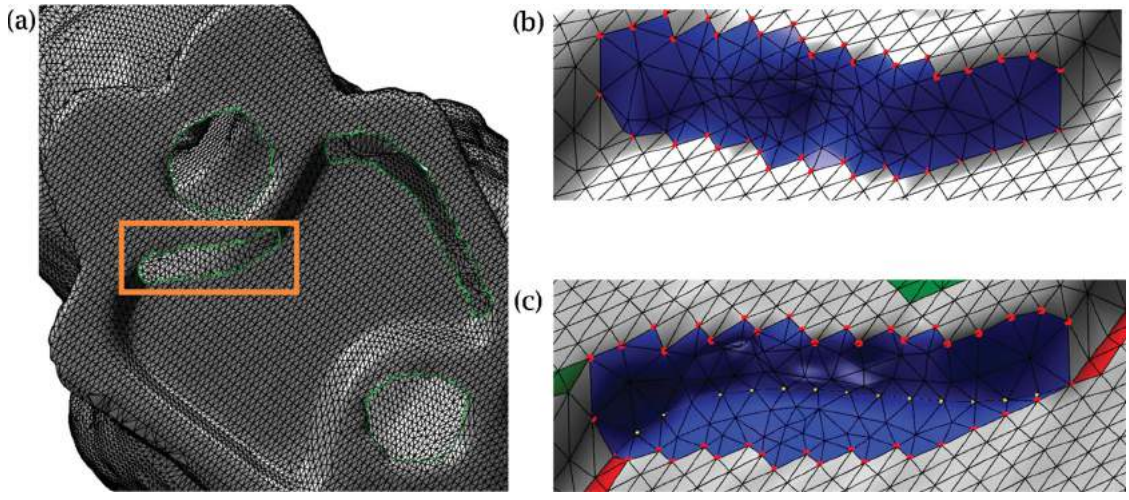

Figure 8. Comparison of the hole-filling result for a hole on Stanford bunny base. (a) The original hole; (b)the hole filled by AFM; (c)the hole filled by curve split.

is regarded as the feature line. When searching for a new chain point, it is important to notice that the faces shared with former chain point must be excluded from its 1-ring neighbor face set. For example, having got the chain point 'a', face f_1 and f_2 should be excluded from its 1-ring neighbor faces in finding new chain point. In other words, only $\{f_3, f_4, f_5, f_6\}$ should be considered. Moreover, although a set of chain points is collected, some of them should not be used in the following process. In order to do the selection, researchers calculate the tangent vector of each chain point starting from boundary point 'p', if a vector has a large variance to the ones ahead of it, the procedure stops at that point. Eqn. (1) is used to calculate the tangent vector of the corner point:

$$\frac{T_n - T_p}{T_p - T_b} = \frac{Arc_{np}}{Arc_{pb}} = \frac{D_{np}}{D_{pb}} \tag{1}$$

Where T represents tangent vector, Arc represents arc length, D represents Euclidean distance. As is shown in Fig. 6, since tangent vector at 'd' has a large variance from point 'p', 'a', 'b' and 'c', the procedure has to stop at point 'c'.

After the curve or the corner split, large and complex holes are decomposed into simple and small sub-holes, which can be filled easily by AFM. In most cases, it is possible to automatically split and fill the hole.

5. Results

Different split methods can be used according to the types of holes. When it comes to holes which contain a corner point connecting several surfaces, corner split is employed; as for holes through which feature lines pass, curve split is chosen. Fig. 7–10 are the demonstration of some results. In Fig. 7 and Fig. 8, we compare our hole-filling results with those directly filled by AFM, it shows that our results can recover the feature in the hole. In Fig. 9, we make comparison of our method, hole-filling with curvature preserving, and the original part,

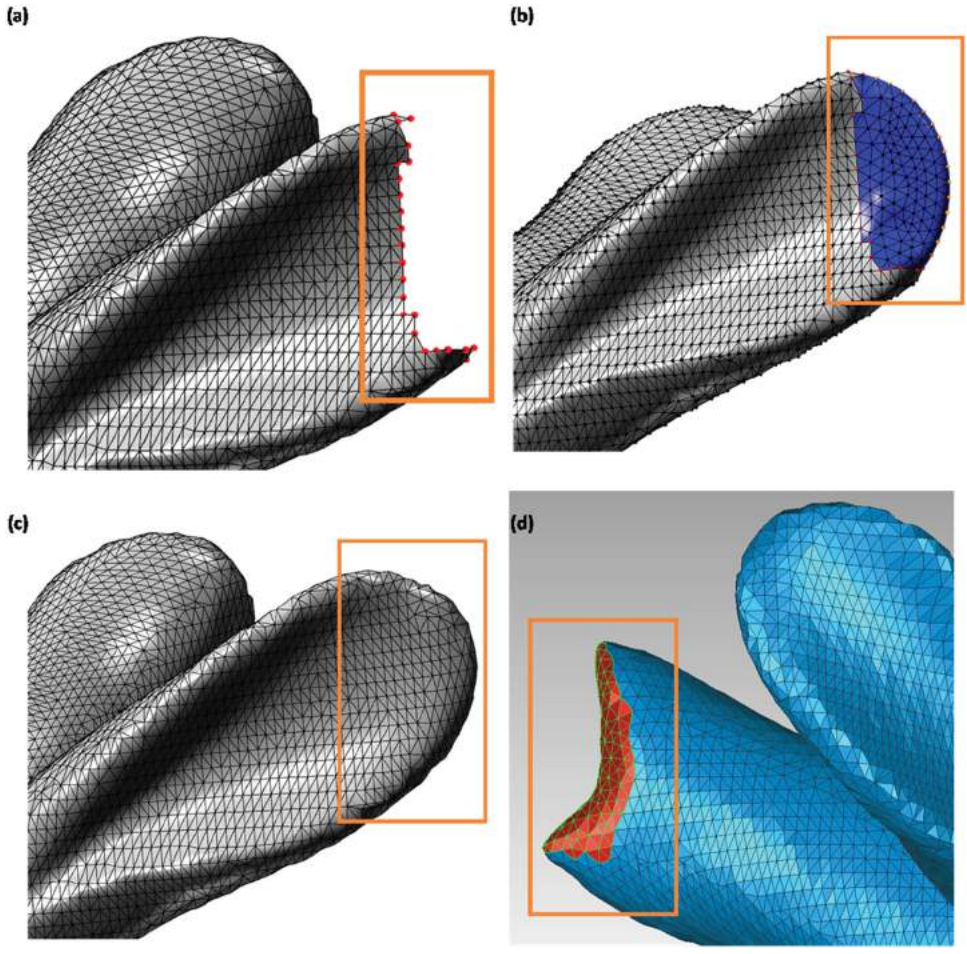

Figure 9. Comparison of the filled part, original part and other algorithm of a man-made hole on Stanford bunny ear. (a) The hole on the ear; (b)the filled result by our method; (c)the original part on the model; (d) the hole filled by curvature preserving.

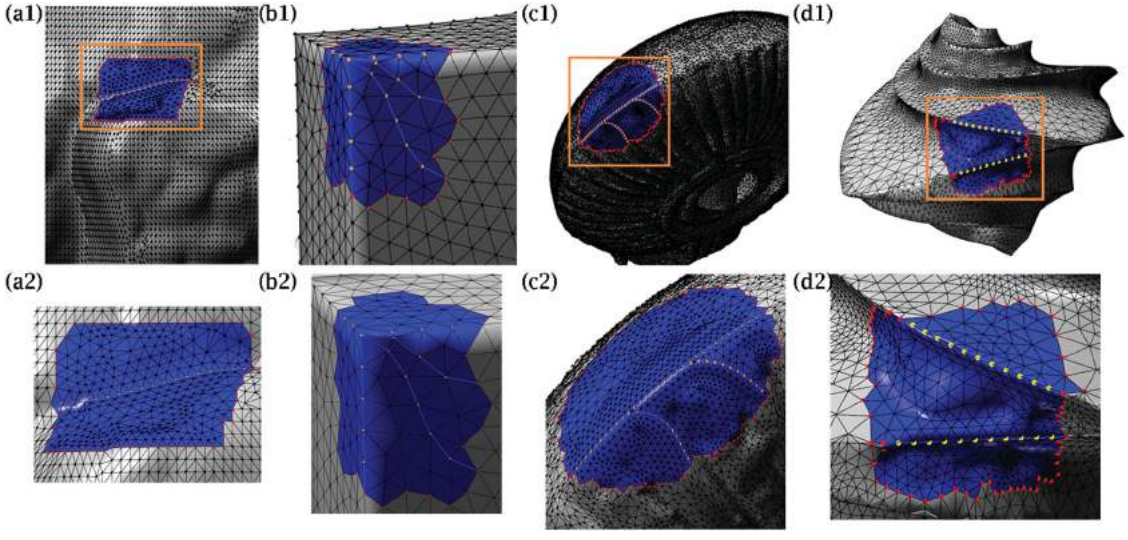

Figure 10. Hole-filling results. (a1) Filled part of Stanford bunny leg; (b1)filled part of a fandisk hole with a corner point missed; (c1)filled part of the lamp model; (d1)filled part of octa flower. (a2), (b2), (c2), (d2) are zoom in version.

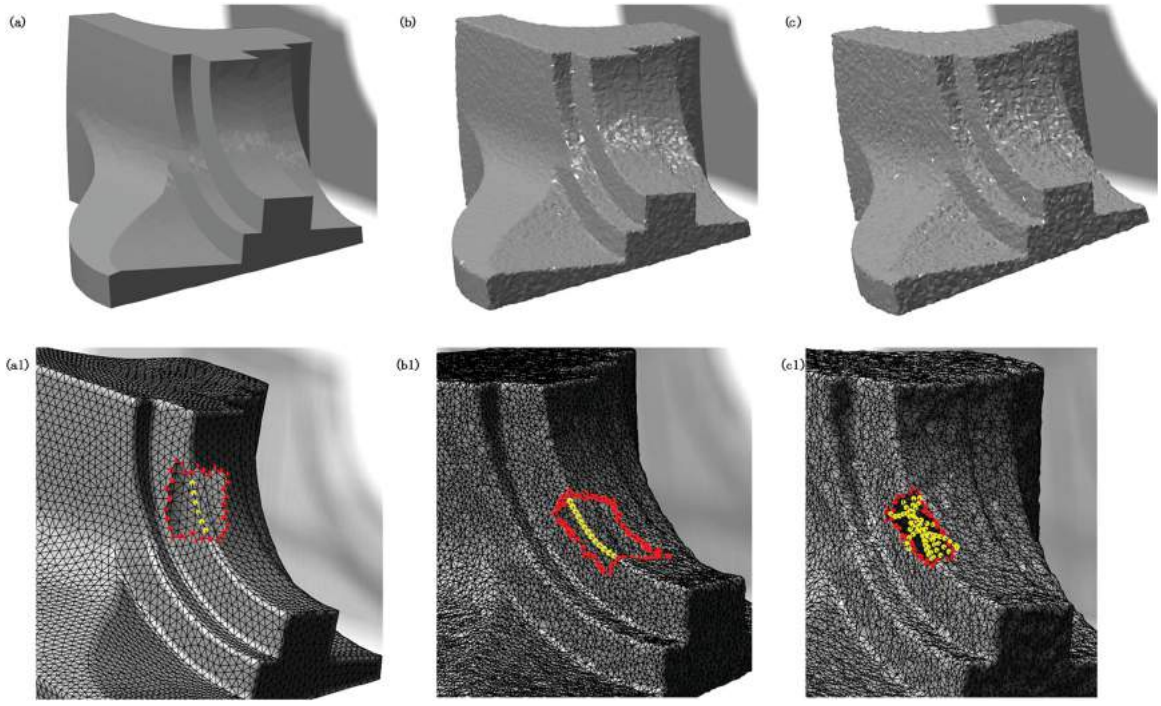

Figure 11. Hole-filling results on noise. (a) Original model with no noise; (b)model with little noise; (c)model with much noise; (a1), (b1) and (c1) are corresponded automatic split results.

Table 1. Result on input size and time consuming.

Model	Number of Input Faces	Number of Input Points	Number of Hole Points	Number of New Faces	Number of New Points	Time of Crest-line Detection(s)	Time of Filling Hole(s)
Pig(Fig. 7)	30969	16181	76	638	282	0.517	0.313
Bunny(Fig. 8)	68354	34339	42	158	59	0.987	0.132
Fandisk(Fig. 10(b))	13801	6995	32	144	57	NA	0.015
Lamp(Fig. 10(c))	60399	30257	117	1747	816	0.741	2.206

it shows that our result is very similar to the original part.

The comparison in Fig. 7–10 shows that with feature line recovery, large complex holes could be split into small ones, and the filled region is much more similar to the original shape of the input model compared to results directly filled by AFM, which is the basis of most mesh-based hole-filling methods. Fig. 11 shows the feature split result of a hole on a clean model and on models with noise. We can observe that if the noise is not too much, we can get a satisfying result.

Tab. 1 shows the sizes of some input models and the time spent in filling the hole. The research program runs on a 2 core i5 CPU with 16G memory. In the process of Fandisk, corner split is used, so there is no need for crest-line detection, and thus no data is shown in this cell.

6. Conclusion

In this paper, we propose a fast and automatic method to fill holes in mesh models. This novel approach can handle both CAD models and free-form ones. It also enables researchers to automatically pair two feature points together and construct an Euler spiral to separate a hole into sub-holes. In cases that a hole contains a corner point, a quad programming equation is applied to optimize its position, and a local method is proposed according to the face normal variance so as to figure out the feature points on the hole boundary. What's more, this research presents a new way to automatically find chain points of a specific feature point along its feature line, which helps to calculate the tangent vector of the corner point. For each sub-hole generated from splitting the original hole, AFM is employed to fill the hole. Undoubtedly, there are some limitations of the method proposed in this paper. It is well-known that the input models must contain no noise or self-intersections. Although, the pre-set thresholds in this program are suitable for most cases, when it comes to some holes with very complex topology, the thresholds still need to be adjusted manually in order to get accurate results. Additionally, granted that the hole-filling procedure of this research is fast, the crest-line detection for feature points costs much more time compared to hole-filling. Furthermore, machine

learning algorithms are considered to be employed in future work so as to obtain more suitable thresholds automatically, and other feature detection methods will be used to improve the stability as well as the speed of this method.

Acknowledgements

This work was supported by the National Key Technologies R&D Program of China (2015BAF23B03), the National Nature Science Foundation of China (61373070), and Tsinghua University Initiative Scientific Research Program (2012Z02170).

ORCID

Chunhong Xia http://orcid.org/0000-0001-6935-6142
Hui Zhang http://orcid.org/0000-0001-6563-9890

References

[1] Alexa, M.; Belyaev, A.; Ohtake, Y.: Multi-level partition of unity implicits, ACM Transactions on Graphics, 22(3), 2003, 463–470. http://dx.doi.org/10.1145/882262.882293

[2] Alexa, M.; Cohen-Or, D.; Sharf, A.: Context-based surface completion, ACM Transactions on Graphics, 23(3), 2004, 878–887. http://dx.doi.org/10.1145/1015706.1015814

[3] Belyaev, A.; Seidel, H.-P.; Yoshizawa, S.: Fast and robust detection of crest lines on meshes, Proc of ACM Symposium on Solid & Physical Modeling, 2005, 227–232. http://dx.doi.org/10.1145/1060244.1060270

[4] Cao, J.; Li, B.; Liu, X.; Lu, L.; Shi, X.; Wang, X.; Yin, B.: Automatic hole-filling of CAD models with feature-preserving, Computers & Graphics, 36(2), 2012, 101–110. http://dx.doi.org/10.1016/j.cag.2011.12.007

[5] Filippov, V.; Huang, J.; Yagel, R.: An accurate method for voxelizing polygon meshes, Volume Visualization, IEEE Symposium on IEEE, 1998, 119–126. http://dx.doi.org/10.1145/288126.288181

[6] Gao, S.; Lin, H.; Zhao, W.: A robust hole-filling algorithm for triangular mesh, The Visual Computer, 23(12), 2007, 987–997. http://dx.doi.org/10.1007/s00371-007-0167-y

[7] Grinspun, E.; Harary, G.; Tal, A.: Context-based coherent surface completion, ACM Transactions on Graphics, 33(1), 2014, 57–76. http://dx.doi.org/10.1145/2532548

[8] Grinspun, E.; Harary, G.; Tal, A.: Feature-preserving surface completion using four points, Computer Graphics Forum, 33(5), 2014, 45–54. http://dx.doi.org/10.1111/cgf.12430

[9] Harary, G.; Tal, A.: 3D Euler spirals for 3D curve completion, Computational Geometry, 45(3), 2012, 115–126. http://dx.doi.org/10.1016/j.comgeo.2011.10.001

[10] Ju, T.: Robust repair of polygonal models, ACM Transactions on Graphics, 23(3), 2004, 888–895. http://dx.doi.org/10.1145/1015706.1015815

[11] Jun, Y.: A piecewise hole filling algorithm in reverse engineering, Computer Aided Design, 2(2), 2005, 263–270. http://dx.doi.org/10.1016/j.cad.2004.06.012

[12] Lee, W.-S.; Ngo, T.-M.: Feature-first hole filling strategy for 3D meshes, Computer Vision, Imaging and Computer Graphics, Theory and Applications, 274, 2013, 53–68. http://dx.doi.org/10.1007/978-3-642-32350-8_4

[13] Li, Z.; Meek, D.-S.; Walton, D.-J.: Polynomial blending in a mesh hole-filling application, Computer-Aided Design, 42(4), 2010, 340–349. http://dx.doi.org/10.1016/j.cad.2009.12.006

[14] Meek, D.-S.; Walton, D.-J.: G1 interpolation with a single Cornu spiral segment, Journal of Computational & Applied Mathematics, 223(1), 2009, 86–96. http://dx.doi.org/10.1016/j.cam.2007.12.022

[15] Moraru, G.; Pernot, J.-P.; Véron, P.: Repairing triangle meshes built from scanned point cloud, Journal of Engineering Design, 18(5), 2007, 459–473. http://dx.doi.org/10.1080/09544820701403797

[16] Nooruddin, F.-S.; Turk, G.: Simplification and repair of polygonal models using volumetric techniques, IEEE Transactions on Visualization and Computer Graphics, 9(2), 2003, 191–205. http://dx.doi.org/10.1109/TVCG.2003.1196006