Title:
A Fast and Automatic Hole-Filling Method Based on Feature Line Recovery

Authors:
Chunhong Xia, xiachunhong7@gmail.com, Tsinghua University
Hui Zhang, huizhang@tsinghua.edu.cn, Tsinghua University

Keywords:
Hole-Filling, Automatic, Feature Line, Corner Point

DOI: 10.14733/cadconfP.2016.262-267

Introduction:

With the fast development of data acquisition equipment, 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 defects. 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, gaps can be filled by simply connecting different parts 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, can handle complex holes and get harmonious results, but are relatively time- and space- consuming. Our research 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 [1] filled the holes with AFM, and then used topology adjustment to refine newly-inserted faces. Sharf [2] presented a context-based method to find a most similar part to the hole on the model, then replaced the filled part with the similar part to restore sharp features. Harary [3]'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 [4] came up with a toolbox to fill the holes on the mesh model and it worked well on holes with topology like an ellipse. 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. There came up some methods to split holes into small ones. Jun[5] 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 [6] proposed a hierarchically piecewise function to split a hole, which made it possible to figure out the corner point and feature boundary in it. However, the procedure required much time and simplified complicated constraints. Li [7] applied mixed polynomial to restore sharp features in the hole region. Ngo [8] made crest-line detection for feature points recognition, and then filled the hole with semi-auto method. Most of the existing methods cannot fill or split a hole automatically, and cannot recover the original topology of the missing part in large size. In this paper, we propose a method that can split the hole automatically, and use simple split lines to retain consistence between the repaired hole and its neighbor regions. And our method operates in local areas to achieve high efficiency.

Main Idea:
Our method operates on triangular mesh models, which are expected to be manifold, oriented, and intersection-free. Moreover, no common points exist between 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. 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: no corner point contained in the hole, each split line is constructed from two end-points on the hole boundary; (b)corner split: a corner exists in the hole, the two end-points of a split line makes up from the corner point and a feature point on the hole boundary. The whole procedure is: (a)decide whether a hole needs to be split, and how to split, i.e. by curve split or corner split; (b)apply different feature detecting methods according to the split type; (c)construct split lines with feature points and the corner point(if have) to divide the hole into sub-holes; (d)repair each sub-hole by AFM.

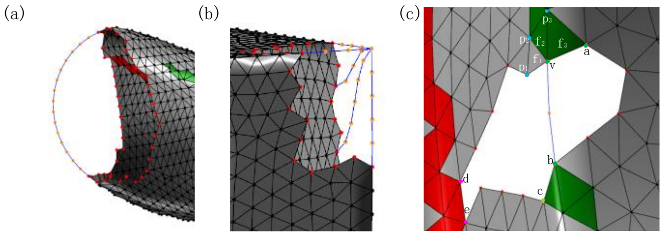

Fig. 1: Illustration of two kinds of split and Euler spiral construction. (a) curve split; (b)corner split; (c)Euler construction. Red points are points on the hole boundary, blue lines are recovered feature lines, orange points are sampling points on feature lines.

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

- An automatic method that can split holes containing corner points is presented;
- The number of surfaces intersected at a corner can be obtained automatically;
- The consistency of the recovered mesh can be retained with the neighborhood of the hole.

Curve Split Construction
Split curve construction is the key point in splitting holes. With boundary conditions, a common way is to construct Hermite curves, but Euler Spiral [9] seems to be more eye-pleasing: extensible, invariant to similarity transformation, symmetric, smooth and round. To fit a Euler spiral, the two end-points of the curve and their corresponding tangent vectors are needed. In Harary's work [10], they chose two end-points and two other points manually to determine the corresponding tangent vectors. By contrast, the point selection procedure, in our research, runs automatically.
First, crest-line detection [11] is used to locate ridge and ravine areas on the model, which are referred to as feature faces. The ridge and ravine face strips are labelled in green and red respectively in Fig. 2(b). Next the boundary points that belong to feature faces are chosen as candidates of feature points. For each candidate point 'v', if there exist feature faces both in its 1- and 2-ring neighborhood, and both are ridge or ravine faces with a shared point, point 'v' is picked as a feature point, labeled in RIDGE or RAVINE according to the feature faces. As shown in Fig. 1(c), points 'v', 'a', 'b', 'c' in green are RIDGE, 'd', 'e' in purple are RAVINE, they are feature points on the hole boundary. For each feature point, we define its feature normal as the average of two boundary faces that share the point, e.g. in Fig. 1(c), the feature normal of 'v' is equal to the average of f_1's normal and f_3's normal. Then feature points are paired as end-points of a split curve. In order to ensure the correctness of pairing, some constraints are set: (a) the paired points are both RIDGE or RAVINE; (b)they are not neighbors; (c)the distance between them along the hole boundary should be larger than a pre-set threshold; (d)the feature normal variance between the paired points should be the smallest among all pairs. In (c), the distance is measured by the number of points between 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. The reason to satisfy the latter requirement is that if the pair is too near, only a small part of the hole will be split, which contributes

little to the final result. Thus, the second rule helps to filtering out some wrong pairs. On each specific feature curve, the normal of each point is similar, so the constraints on feature normal variance help to find the best pair. For example, in Fig. 1(c), the best pair is {v, b}, treated as the two end-points.

Then we calculate an end-point's tangent vector. 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', we search among its 1-ring neighbor points. 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, we obtain the initial conditions to construct a Euler spiral. Fig. 2 shows the result of constructed Euler spiral.

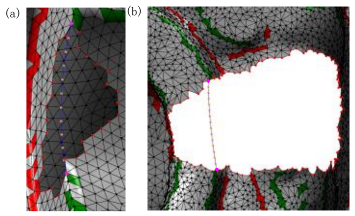

Fig. 2: Illustration of a constructed Euler spiral. Purple points are the best pair. (a)inserted feature line consistent to feature lines around the hole; (b) Euler spiral splits the complex hole.

Corner Split Construction

First, we use Cao's corner recovery method [12] to locate the corner point. Then feature points on the hole boundary need to be found out. As is well-known, corner points usually appear on CAD models. And normal variance between adjacent faces is smaller within a surface; as for the faces separated by feature lines, the difference between them tend to be big. In this way, we measure differently for feature point detection. The first step is to get 1-ring neighbor faces of the hole boundary and their normal, after that, k-means clustering is used to classify the normal set. The problem lies in how to determine the value of k, which is equal to the surface number that a corner connects. Two concepts are defined: (1) face normal variance: the normal variance between two adjacent 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 as follows: (a) iterate k from the lower bound 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, which is denoted as k_s; (c)select k_s points on the boundary in decreasing order of point-related face normal variance. As a corner connects at least three surfaces, lower bound is set to three; the upper bound is set as 1/4 of the number of boundary vertex. If the upper bound is too large, a cluster contains 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. In our method, each center is decided by the largest distance between each other so every cluster can distribute evenly. In Fig. 3, purple points illustrate the feature points. We observe that the number of feature points is larger than or equal to the exact surface number, but feature points on feature lines are always chosen.

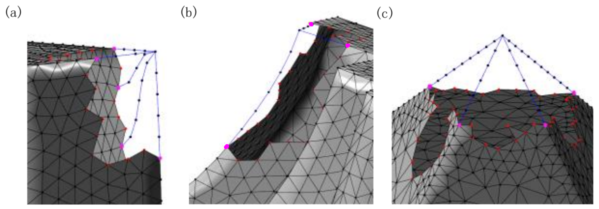

Fig. 3: Illustration of feature points, purple one.

Secondly, split lines are constructed with the end-points set. In order to get better-looking results, Euler spiral is chosen too. Therefore, the tangent vector of each end-point needs to be calculated. For each feature point, such as ‘p’ in Fig. 4, face pair $\{f_1, f_2\}$ has the largest normal variance among its 1-ring neighbor faces $\{f_1, f_2\}$, so ‘a’ is chosen as a chain point. Likewise, the following chain points ‘b’, ‘c’, ‘d’, and ‘e’ are found, and the line which these chain points lay on is regarded as feature line. When searching for a new chain point, the faces shared with former chain point must be excluded. 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 a new chain point. In other words, only $\{f_3, f_4, f_5, f_6\}$ should be considered. Moreover, some of the chain points found should not be used in the following process. In order to do the filter, we calculate the tangent vector of each chain point starting from boundary point ‘p’, if a vector has a large variance to the former ones, we stops there. Eqn. (1) is used for the vector calculation:

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

Where T represents tangent vector, Arc represents arc length, D represents Euclidean distance. As is shown in Fig. 4, tangent vector at ‘d’ has a large variance from ‘p’, ‘a’, ‘b’ and ‘c’, the procedure has to stop at ‘c’. In this way, we get different tangent vectors of the corner point with different feature points.

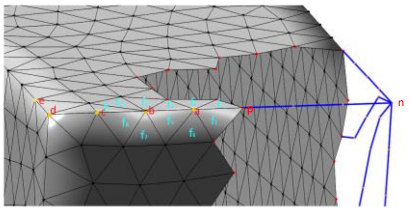

Fig. 4: Illustration of chain point calculation.

After the curve or corner split, large and complex holes are decomposed into simple and small sub-holes, which can be filled easily by AFM. Fig. 5 and Fig. 6 show some results.

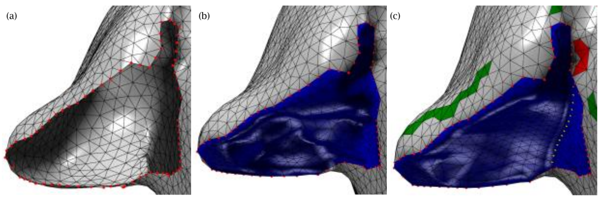

Fig. 5: Comparison on a repaired pig ear. (a)original hole; (b)repaired by AFM; (c)repaired by curve split.

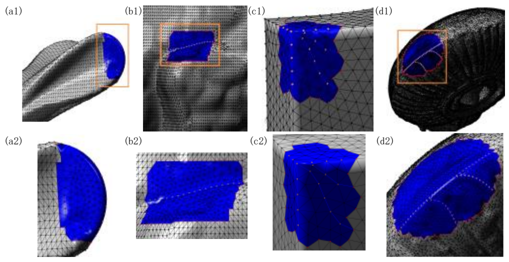

Fig. 6: Hole-filling results. (a1) a bunny ear; (b1) a bunny leg; (c1) a hole with corner; (d1) a lamp model; (a2), (b2), (c2), (d2) are zoom in version.

The comparison in Fig. 5 shows that with feature line recovery, we can split large complex holes into small ones, and the filled region can recover the original shape of the missing part.

Conclusion:

In this paper, we propose a fast and automatic method to fill holes in mesh models, it can handle both CAD and free-form ones. We can automatically pair two feature points together and construct a Euler spiral to split a hole into sub-holes. If a corner point exists, a quad programming equation will be 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, our method presents a new way to automatically find chain points along a 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 it. Undoubtedly, there exist some limitations. It is well-known that the input models must contain no noise or self-intersections. Although, the pre-set thresholds are suitable for most cases, when it comes to some holes with very complex topology, they need to be adjusted manually in order to get accurate results. Additionally, granted that the hole-filling procedure 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 applied in the future to obtain more suitable thresholds automatically.

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).

References:

[1] 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

[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] 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

[4] 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

[5] 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

[6] 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

[7] 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

[8] Lee, W.-S.; Ngo, T.-M.: Feature-First Hole Filling Strategy for 3D Meshes, Computer Vision, Imaging and Computer Graphics. Theory and Applications, Springer Berlin Heidelberg, 2013, 53-68. http://dx.doi.org/10.1007/978-3-642-32350-8_4

[9] 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

[10] 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

[11] 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

[12] 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