FILLING GAPS BETWEEN MESHES OF TREE CROWN AND TREE TRUNK BASED ON BOUNDARY CONSTRAINTS AND COORDINATE PROJECTION FOR 3D TREE MODELLING

W. X. Wang[1], L. P. Hong[1], H. S. Huang[1], X. M. Li[1], S. J. Tang[1], R. Z. Guo[1], L. F. Xie[1,] *

[1]Research Institute for Smart Cities, School of Architecture and Urban Planning, Shenzhen University, 518060 Shenzhen, China - (wangwx, lixming, shengjuntang, guorz, linfuxie)@szu.edu.cn, (honglinping2022, huanghongsheng2021)@email.szu.edu.cn

Commission II, WG II/1

KEY WORDS: Tree Meshes, Boundary, Gap, Projection, Triangulation.

ABSTRACT:

In order to reconstruct high-quality 3D tree models, trunks and crowns could be reconstructed using appropriate methods separately and merged together. During this process, gaps will appear after tree models are spliced, which will affect the models' topological connectivity and visual effects. In this paper, a gap-repair algorithm for tree mesh models based on boundary restriction and coordinates projection is proposed. The algorithm first extracts all the holes in the tree trunk meshes and crown meshes and matched them according to their relative positions. Then, the gaps in the tree meshes are identified. After that, based on the projection of the hole vertices from 3D space to 2D space, the relative positions of the hole vertices were determined, and vertex connection and surface addition were carried out to generate a 2-manifold patch. Finally, the patch was refined to keep its density of vertices similar to the surrounding meshes. Real 3D Tree meshes of different species were used to conduct experiments, and the quality of the repaired tree models are evaluated qualitatively and quantitatively. Experimental results revealed that the proposed algorithm could fill gaps in tree meshes quickly and effectively, so as to achieve the topology connection and smooth transition between the trunk and crown meshes.

1. INTRODUCTION

The Three-dimensional (3D) tree models are an important part of digital city and digital forestry. They are widely used in the construction of 3D real scenes, simulation of geographic environments, and calculation of forest carbon sinks (Cao et al., 2021). How to reconstruct 3D tree models automatically, efficiently, and realistically has become a spotlight in recent years (Yang et al., 2022). Among them, tree detection technologies represented by light detection and ranging (LiDAR) technology are gradually becoming important methods for tree surface reconstruction (Kelbe et al., 2012). LiDAR point clouds provide data assurance for tree reconstruction and 3D simulation in urban scenes and digital forestry (Li et al., 2001). Point cloud data-driven tree modeling methods are often based on the extraction of tree skeletons or tree crowns (Liu et al., 2021). For example, Liu extracted tree skeletons based on generalized Voronoi diagrams (Liu et al., 2012). As the same to rebuild tree skeletons, Du utilized global optimizations through the computation of minimum spanning graphs, called the AdTree (Du et al., 2019). However, these types of methods do not guarantee local geometric accuracy (Cao et al., 2021). In order to improve the geometric accuracy of the tree models, maintain the realism of the tree models, and improve the efficiency of large-scene tree modeling, Wang merged the trunk and crown triangular mesh models obtained by two different modeling methods to form a refined tree model (Wang et al., 2023).

The fusion of different tree meshes can improve the local geometric accuracy of the tree models. But the mesh models built based on different methods may have inconsistencies in stem thickness, mesh alignment, geometric details, etc. After deleting the overlapping parts of the mesh model and merging the meshes, the result tree meshes will inevitably have gaps between two components of the meshes. The presence of gaps disconnects the trunk mesh and crown mesh components. And the boundaries can cause problems with some continuity-dependent applications of the models (Feng et al., 2020). At the same time, gaps in the meshes lead to faults in the texture mapping, and big gaps can be identified by the naked eye, affecting the visualization of the tree models. When merging two tree meshes, it is important to fill the gaps between the trunk and crown meshes quickly and efficiently. There are several studies that are related to detecting and filling the gaps in triangular meshes. These methods can be roughly divided into voxel-based methods and surface-based methods (Botsch and Kobbelt, 2010).

The voxel-based method first converts the triangular mesh into a voxel model and then converts it back to the surface mesh after completing the repair operation. Ju used an Octree grid for contouring to reconstruct a water-tight surface (Ju, 2004). Li and Huang closed gaps in meshes by applying morphological closing operators to voxels models (Li and Huang, 2004). Bischoff and Kobbelt created an adaptive Octree for the input model and utilized voxel topological simplification to reconstruct the model (Bischoff and Kobbelt, 2005). Voxel-based methods can repair meshes quickly, but one disadvantage of voxel-based methods is that local geometric details on the models may be lost.

The surface-based method directly applies the repair method to the surface meshes. Bøhn and Michael detected the non-closed boundaries through algorithms of graphic theory and used

* Corresponding author

localinformation to repair the gaps (Bøhn and Michael, 1992). Barequet and Sharir used local curve-matching technology and 3D polygon optimal triangulation technology to fill gaps in the mesh output from a CAD system (Barequet and Sharir, 1995). Patel eliminated gaps and overlaps in meshes by iteratively compressing and stretching operations on boundary vertex pairs (Patel et al., 2005). Marchandise discretely parametrized the triangular mesh by using the radial basis function (RBF) and then re-triangulated the surface to reconstruct the triangular surface (Marchandise et al., 2012). Although the surface-based method is inefficient, it can maintain the surface details of the models.

The methods proposed in the above papers only repaired the small gaps between different connecting components of common watertight meshes and did not study the integration of tree meshes. This paper studies the splicing method of tree meshes obtained by two different modeling methods. Our problem is how to fill the gap between the two meshes of a tree to connect the trunk and crown meshes and generate a complete tree model. And make the filling patch as compatible as possible with the surrounding mesh in the tree meshes. In order to obtain a tree model as fine as possible, we choose the surface-based method to fill the gap in the tree mesh.

In this paper, an algorithm that adapts to the gap boundary filling of tree meshes after considering the geometric characteristics of tree gap boundaries is proposed. The algorithm first extracts the boundary vertices and edges and then refines the boundary edges to ensure that the boundaries of the gap in the mesh have a similar density of vertices. The coordinates of the boundary vertices in 3D space are projected onto a fitted two-dimensional(2D) plane to determine the connection sequential order of the gap boundary vertices and then add new faces. Finally, a 2-manifold patch is generated and it is refined according to Liepa's method (Liepa, 2003).

2. METHODS

The method proposed in this paper consists of three steps. The input includes trunk mesh and crown mesh that are constructed by two different methods from one real tree. Before executing the algorithm, the overlapping faces of the two meshes are interactively removed to obtain a tree mesh consisting of two components, trunk mesh, and crown mesh, which are not connected to each other. Between whom exist a gap. Firstly, extract all the border edges and collect them into a set of closed boundaries. A pair of boundaries in the gap between trunk meshes and crown meshes are matched and used as input for the next step. Use the pair of boundaries as a constraint and connect the vertices to generate patch faces in the gap. Finally, refine the patch faces to maintain the Delaunay triangulation rule on the patch. Figure 1 briefly shows the overall flow of this algorithm.

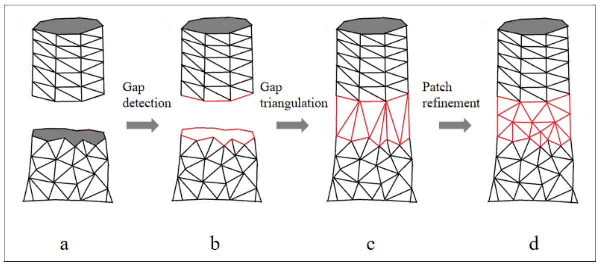

Figure 1. (a)Mesh of tree with a gap. (b)Detection of the gap in the mesh. (c)Result of triangulating the gap. (d)Result of refining the patch.

2.1 Preprocessing

Two different meshes of one tree are built from a moving laser-scanned point cloud of the same tree. The data that is input to our algorithm consists of two parts, one is the trunk mesh generated based on Poisson surface reconstruction (Kazhdan and Hoppe, 2013), and the other is the crown mesh generated based on algorithms called AdTree (Du et al., 2019). Before merging the meshes, make sure that the coordinates of vertices in both meshes are in the same coordinate system.

These two meshes are put together and merged into one mesh. Then the overlap faces in the tree mesh are cut out, leaving a small gap in the tree mesh. In general, the size of the gaps remaining in the tree meshes does not affect the results of the restoration. But it is important to keep the two sizes of boundary in the gap as consistent as possible. At the same time, the boundary of the gap should also be kept as smooth as possible. Jagged boundaries can affect the outcome of the repair. The resulting trunk meshes and crown meshes are input into the algorithm for the next step.

2.2 Gap Extraction

The tree meshes produced in the previous step are non-watertight meshes, with holes at the ends of the trunk and branches of the tree meshes. These holes are similar in appearance to circular curves, and their center can be used to represent their position and the gap can be found by the size of the distance of two holes. Usually, the gap is located at the top of the trunk mesh, and at the bottom of the crown mesh. It is needed to extract all the holes first on the tree meshes, and then find two holes on the trunk mesh and the crown mesh respectively which are closest to each other as the boundary of the gap.

2.2.1 Holes Extraction: A triangular mesh consists of a series of vertices, edges, and faces in 3D space. All interior edges in the mesh are adjacent to two faces. If an edge is adjacent to only one face, it is a boundary edge. The boundary edges can be easily identified using the half-edge data structure. Given a seed border half-edge, by iterative searching for the next border half-edge, a closed loop of border edges can be collected, which can be called a hole in the triangular mesh. All the holes of the tree mesh in our method are divided into trunk mesh holes and crown mesh holes. And they are collected in two list Ht $\{Ht_1, ... Ht_i, Ht_{i+1}, ... Ht_m\}$ and Hc $\{Hc_1, ... Hc_i, Hc_{i+1}, ... Hc_n\}$. Where m is the number of trunk mesh holes and n is the number of crown mesh holes.

2.2.2 Detecting the Gap: To search the gap between trunk mesh and crown mesh, the distance between the holes was calculated. Our goal is to find a hole in Ht and a hole in Hc that is geometrically closest to each other. Firstly, the centroid point coordinates of the holes are calculated from all the boundary vertices in the holes. A hole H consisting of a set of vertices $\{v_1, \dots v_i, v_{i+1}, \dots v_n\}$, its centroid $v_{hc} = \frac{1}{n}\sum v_i$. Then m holes in Ht and n holes in Hc are combined into m*n pairs of holes. For each pair of holes, calculate the distance of their centroid coordinates and assign this value as a label value to this pair of holes. The pair of holes that have a minimum label value is the boundary of the gap in the tree mesh.

2.3 Triangulating the Gap

A hole in the mesh consists of a set of directed vertices. The boundary of the gap consists of two directed vertex chains.

These vertices form two geometrically similar curves and look like two parallel circles. The vertex density of the two curves is not the same. In order to obtain a better-quality triangular patch, vertex encryption needs to be performed on holes with sparser vertices before doing gap triangulation. After encrypting the hole boundary, the vertices of the boundaries of the gap are projected onto a 2D plane. Then determine the order of connections between the vertices and add new faces to the mesh.

2.3.1 Boundaries Encryption: The mesh models generated by different reconstruction algorithms often have different face sizes and side lengths. The vertex density of the gap boundary will affect the result of the patch. If the boundaries of the trunk and crown are directly connected, the patches in the resulting triangulated patch are of poor quality. Therefore, it is needed to encrypt the boundary where the vertex is too sparse, that is, insert a new vertex between two adjacent vertices, thereby dividing the adjacent face into two new faces. Firstly, calculate the number of vertices of the two boundary vertices chains. Suppose that two vertex chains are called chain1 and chain2, and the number of vertices in chain1 is m, and the number of vertices in chain2 is n, where m > n. Define the ratio value r, where r is the greatest integer less or equal to m / n. If the integer r > 1, the hole where chain1 is located is encrypted. The encryption method is for the set of vertices $\{v_1, \dots v_i, v_{i+1}, \dots v_m\}$ in chain1, insert r - 1 new vertices every two points v_i and v_{i+1} on the edge (v_iv_{i+1}), and generate r - 1 new face in the triangular mesh. The process is shown in Figure 2, where the value of r is 2.

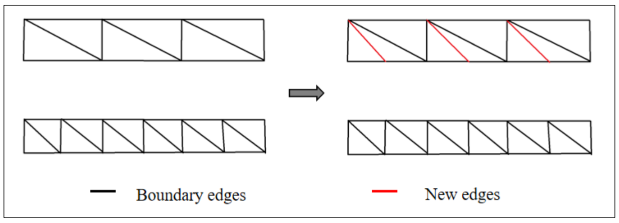

Figure 2. Encryption of the boundary

2.3.2 Vertex Projection: With the method of the previous steps, two new vertex chains, chain1$\{v_{11}, \dots v_{1i}, v_{1i+1}, \dots v_{1m}\}$ and chain2$\{v_{21}, \dots v_{2j}, v_{2j+1}, \dots v_{2n}\}$ can be obtained. where m and n are the numbers of vertices in chain1 and chain2 respectively. To repair the gap between the two parts of the meshes, it is necessary to stitch the vertices in two vertex chains together and form a 2-manifold patch. The connection order between vertices is determined under a polar coordinate system on a 2D plane by projecting all the vertices of the boundary onto the plane and inserting new triangle faces. The projection program of the algorithm is organized as follows:

(1) Collect all the vertices of one chain, and assume that chain1 is chosen. By calculating the average coordinate of all the vertices in chain1, a virtual centroid vertex $v_c(x_c,y_c,z_c)$ of vertices in chain1 can be obtained. The values in parentheses are the coordinates of the vertex.

(2) Collect the coordinates of all the vertices in chain1 $\{v_{11}, \dots v_{1i}, v_{1i+1}, \dots v_{1m}\}$. Every two adjacent vertex v_{1i} and v_{1j}, together with the centroid point v_c determine a plane α_i. Calculate the normal n_i (x_i, y_i, z_i) of α_i, and form a vector set $\{n_1, \dots n_i, n_{i+1}, \dots n_m\}$. Specifically, n_m is calculated by v_{1m}, v_{11}, and v_c. Equation 1 presents the calculation method of normal n_i.

$$n_i = \overrightarrow{p_c p_i} \times \overrightarrow{p_c p_{i+1}} \tag{1}$$

Calculate the average value of the elements in $\{n_1 \dots n_i, n_{i+1}, n_m\}$ to obtain the average normal n_c (a, b, c).

(3) The normal n_c (a, b, c) and the centroid vertex $v_c(x_c,y_c,z_c)$ can form a plane б (shown in Figure 3). The plane б is represented as Equation 2:

$$a(x - x_c) + b(y - y_c) + c(z - z_c) = 0 \tag{2}$$

Project the vertices in chain1 $\{v_{11}, \dots v_{1i}, v_{1i+1}, \dots v_{1m}\}$ and chain2 $\{v_{21}, \dots v_{2j}, v_{2j+1}, \dots v_{2n}\}$ onto the plane to obtain two sets of mapped vertices $\{v_{11}', \dots v_{1i}', v_{1i+1}', \dots v_{1m}'\}$ and $\{v_{21}', \dots v_{2j}', v_{2j+1}', \dots v_{2n}'\}$. The coordinates of the mapping points are given by Equation 3 and Equation 4.

$$\begin{cases} x_i' = x_i + at \\ y_i' = y_i + bt \\ z_i' = z_i + ct \end{cases} \tag{3}$$

$$t = \frac{a(x_i - x_c) + b(y_i - y_c) + c(z_i - z_c)}{a^2 + b^2 + c^2} \tag{4}$$

Where (x_i, y_i, z_i) is the coordinate of vertex v_i in the boundary of the gap, and (x_i', y_i', z_i') is the coordinate of vertex v_i', which is the mapping point of v_i on the plane б.

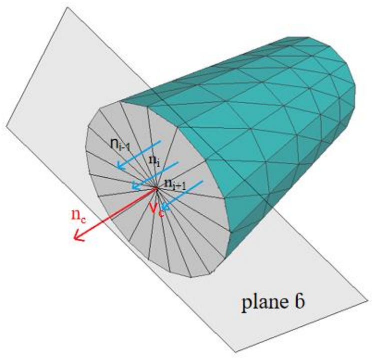

Figure 3. Getting the projection plane

2.3.3 Gap Triangulation: The vertices are projected onto the plane б, and the topological position relationship between the projected points determines the connection order of vertices on the original mesh. For the mapping points obtained in 2.3.2, a vertex v_k' is randomly selected from one mapped vertices chain and the vertex v_l' which is the closest vertex to v_k' is selected from the other mapped vertices chain. These two vertices become the seed vertices of the triangulation program., and the coordinates of v_0' are the coordinates of the seed vertex v_k'. Set two pointers (u and v) to each of these two mapped vertices chains. Suppose now u points to v_{1i}' and v points to v_{2j}'. If the following conditions apply:

$$\angle v_0' v_c v_{1i}' < \angle v_0' v_c v_{2j}' \tag{5}$$

Then add a new face (v_{1i}, v_{2j}, v_{1i+1}) to the tree mesh, and the pointer u moves forward. Otherwise, add a new face (v_{1i}, v_{2j}, v_{2j+1}) and the pointer v moves forward. When both pointers complete the closed loop and return to the seed vertices at the same time, the iteration ends. Figure 4 shows the triangulation result.

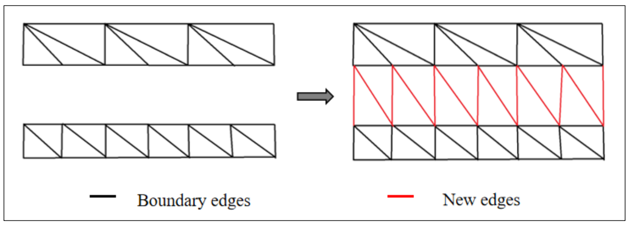

Figure 4. Gap triangulation

2.4 Refining the Patches

In order to keep the density of the faces in the patch to be similar to the surrounding mesh, the patches generated in 2.3 need to be refined. The method presented by Liepa (Liepa, 2003) is used in this work. The main idea is to calculate the edge length of the patch boundary and adjust the patch face density by interpolating vertices and relaxing edges, and maintain the Delaunay rule of the patch. Relaxing an edge refers to the process of exchanging an edge if, for two adjacent triangles, a sphere is constructed with their common edge as the diameter, and both of the other two vertices are located inside the sphere (shown in Figure 5). the density control factor α is recommended by Liepa, where $\alpha = \sqrt{2}$. The steps of refining the patch are listed as follows:

(1) For each boundary vertex v_i in the patch, a label value $\varphi(v_i)$ is calculated, which is the average length of the adjacent edges of v_i.

(2) For each triangle f (v_i, v_j, v_k) in the patch, calculate the centroid vertex v_c and the label value $\varphi(v_c)$, where $\varphi(v_c)$ equals $(\varphi(v_i) + \varphi(v_j) + \varphi(v_k)) / 3$. For m = i, j, k, if $\alpha \|v_c - v_m\| > \varphi(v_c)$, and $\alpha\|v_c - v_m\| > \varphi(v_m)$, Then insert a new vertex v_c to the patch, delete the face f (v_i, v_j, v_k), and add the new face (v_i, v_j, v_c), (v_j, v_k, v_c) and (v_k, v_i, v_c) to the mesh.

(3) Relax the new edges (v_i, v_c), (v_j, v_c), and (v_k, v_c).

(4) If there are no new faces added to the mesh, relax all the edges inside the patch.

(5) If there are no edges swapped, return to step 2. Else, return to step 4. If no new faces were added and no edges were relaxed, the procedure was ended.

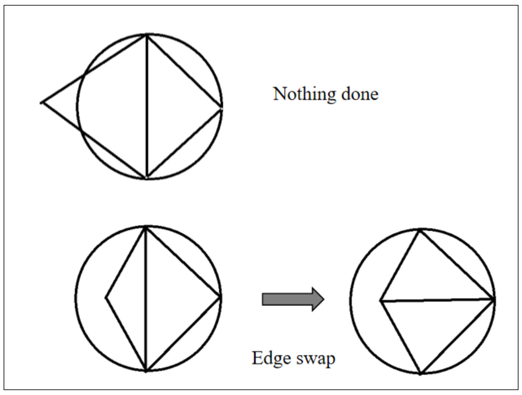

Figure 5. Edge relaxation

3. RESULT AND DISCUSSION

3.1 Experimental Results and Analysis

We used a handheld mobile laser scanner to acquire point clouds of trees in Lianhua Mountain, Shenzhen, Guangdong Province, China. Based on the tree point clouds, the Poisson modeling method was used to obtain the trunk meshes, and the AdTree modeling method was used to obtain the crown meshes. And the overlapping part of the two meshes is clipped by manual operation in the CloudCompare software (EDF R&D and Telecom ParisTech 2023). The trunk mesh faces density is usually larger than that of crown mesh. To verify the effectiveness of our algorithm, we obtained six tree meshes from different tree species (shown in Figure 6). We use the CGAL algorithm library (Utrecht University, Faculty of Mathematics and Computer Science, 2022) to integrate our algorithm and run the procedure on a 2.5GHz computer with 64G RAM and a Windows 10 system.

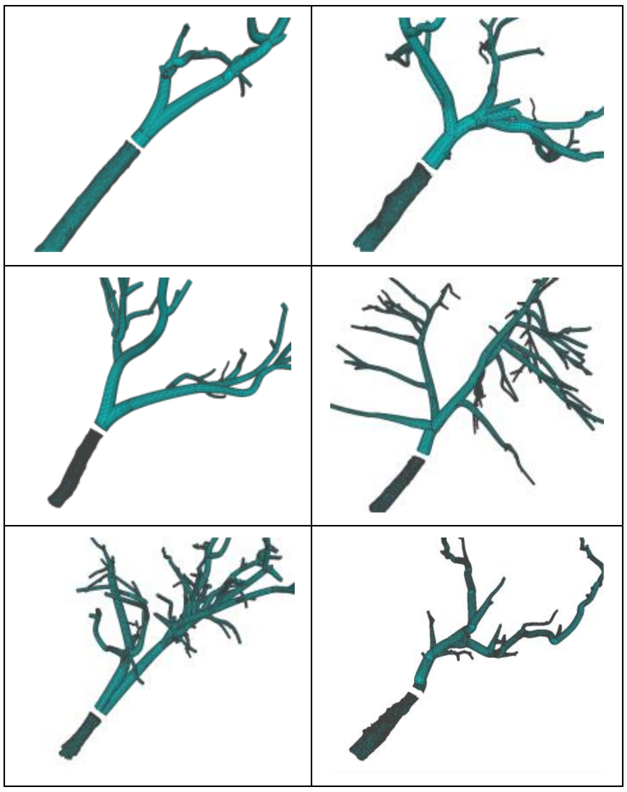

Figure 6. Six of Tree meshes with a gap between the trunk and crown.

Experimental results show that our method can quickly identify the gap between the trunk and crown mesh and perform triangulation on it. Figure 7 shows the gap-filling results of the six tree meshes. Meanwhile, the attribute information of the original meshes, the information of the patches, and the program running time are given in Table 1. The patch faces density is basically consistent with the gap boundary face density. When the border edge lengths of the gaps are similar to each other, the patch faces are uniform in size. In the experiments, we obtained the patches that were articulated with the original model of the trees, which were 2-manifold, and the faces on the patches did not self-intersect.

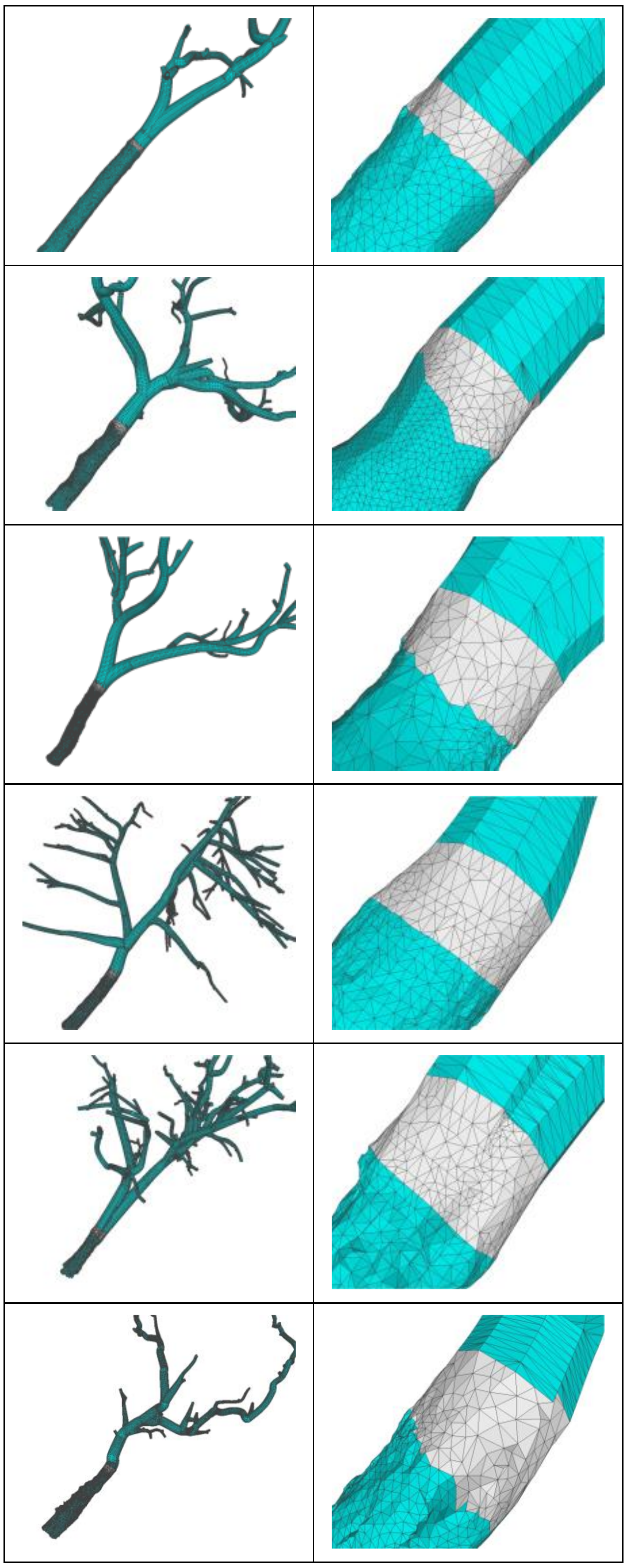

Figure 7. Single gap triangulation. (a) Gap-filling result of the tree meshes. (b) Detail of the patch in the gap.

Model	Original mesh		Gap filling patch		Filling time(ms)
	N_f	N_v	N_f	N_v	
Tree1	28,729	15,072	342	117	3
Tree2	43,010	22,358	491	175	4
Tree3	46,842	24,608	573	219	4
Tree4	99,903	62,414	603	246	5
Tree5	59,808	30,840	653	261	5
Tree6	71,692	43,956	701	292	5

Table 1. Attributes of the meshes (N_f is the number of faces, N_v is the number of vertices)

Although no mesh fairing method is applied in our gap repairing algorithm, the resulting patches are still quite smooth at the original mesh junction, that is no curvature mutation at the boundary of the gap (shown in Figure 8). This is because the curvature of the boundary of the trunk mesh and the crown mesh are similar, so neither the initial triangulation nor the refinement causes curvature mutations.

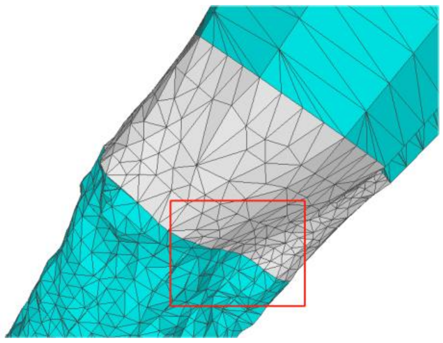

Figure 8. Smooth boundaries

When the boundary curves of the gap are smooth, the repair result is good. A not smooth boundary curve will cause the production of an elongated triangle in the patch (shown in Figure 9). This reminds us that we should try to flatten the gap boundary when doing data preprocessing work or improve our method to avoid generating elongated triangles or even degenerate triangles.

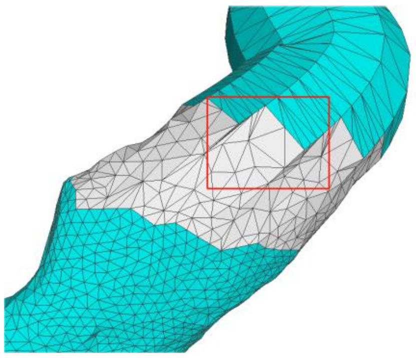

Figure 9. Elongated triangles

3.2 Evaluation of the patches

To assess the quality of the patch triangles, we use the triangle quality value q introduced by Field (Field, 2000). Suppose R_a is the radius of the inscribed circle of the triangle and R_b is the radius of the circumscribed circle of the triangle. Defined the mesh quality value $q = 2R_a / R_b$. If the triangle is regular, $q = 1$.

The closer q is to 1, the better the quality of the triangle. When q is greater than 0.5, the triangle can be considered to be of good quality (Sun et al. 2014).

Figure 10 shows the density distribution plot of the quality value of all patch faces generates from six tree meshes. The results of the evaluation show that our method produced patch faces of high quality, with most of the quality values q greater than 0.5(94.91%) out of a total of 3363 new patch faces.

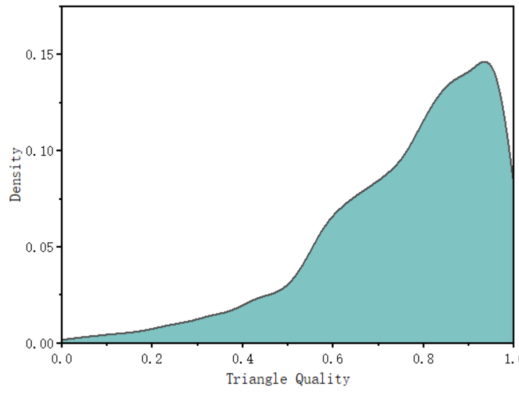

Figure 10 Distribution of face quality values

3.3 Other example

To adapt to the splicing requirements of different tree meshes, and improve the robustness of the algorithm, we did another set of experiments with other tree mesh data. These meshes of trees have more than one gap. These gaps arise when the same trunk merges with multiple branches in the crown. We detected, matched and repaired all the gaps at the same time. Experimental results show that our gap boundary recognition method is also effective in the presence of multiple gaps. The repairing result is shown in Figure 11 and Table 2.

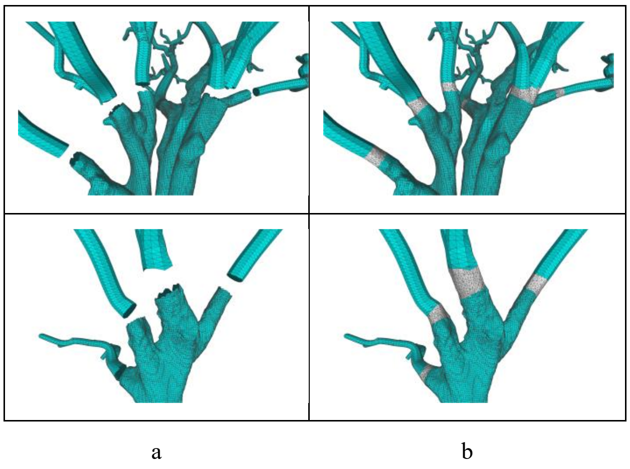

Figure 11. Multiple gap triangulation. (a)Mesh with multiple gaps. (b)The result after filling the gap

Model	Original mesh		Gap filling patch		Filling time(ms)
	N_f	N_v	N_f	N_v	
Tree1	204,467	102,964	2,239	824	15
Tree2	90,768	49,472	1,525	603	11

Table 2. Attributes of meshes with multiple gaps (N_f is the number of faces, N_v is the number of vertices)

4. CONCLUSIONS

This paper proposes a mesh gap repair algorithm to solve the problem of the existing gap caused by the merging of two different meshes of one tree. Firstly, all the holes in the two-part mesh are extracted. Then the boundary of the gap is identified according to the hole centroid vertex coordinates. The connection order of the hole vertices is determined through the spatial coordinate mapping and the gap is triangulated to generate a 2-manifold patch. Finally, refine the patch.

We performed a triangle quality evaluation on the output patches. Experimental results show that our method can quickly and efficiently identify the gap between two tree meshes and connect the meshes by triangulating the gap and forming a complete mesh of the tree. A smooth transition of the tree mesh from trunk to crown can be achieved.

Our method is only applied to the tree mesh fusion with relatively simple geometric shapes, and the robustness of the algorithm may cross over. This method can also be applied to the fusion of other mesh objects, but further research is needed. Our future research will improve the robustness of the algorithm and expand the application field.

ACKNOWLEDGEMENTS

This work is supported by the National Key Research and Development Program of China (Projects Nos. 2022YFB3903703), the National Natural Science Foundation of China (42001407, 41971341, 41971354), the Shenzhen Science and Technology Program (Grant No. RCBS20221008093124064), the Guangdong Science and Technology Strategic Innovation Fund (the Guangdong－Hong Kong-Macau Joint Laboratory Program) (Project No.: 2020B1212030009).

REFERENCES

Cao, W., Chen, D., Shi, Y. F., Cao, Z., Xia, S. B., 2021. Progress and Prospect of LiDAR Point Clouds to 3D Tree Models. Geomatics and Information Science of Wuhan University, 46(2), 203-220.

Yang, J., Wen, X. R., Wang, Q. L., Ye, J. S., 2022. Shortest Path Extraction Algorithm of Tree Branch Point Cloud Skeleton Based on Geometric Characteristics. Journal of Northwest Forestry University, 37(6), 129-137.

Kelbe, D., Romanczyk, P., Aardt, J. V., Cawse-Nicholson, K., Krause, K., 2012. Automatic extraction of tree stem models from single terrestrial lidar scans in structurally heterogeneous forest environments. The 12th International Conference on LiDAR Applications for Assessing Forest Ecosystems.

Li, Q. Q., Li, B. J., Chen, J., 2001. Research on laser range scanning and its application. Geo-spatial Information Science, 4(1), 37-42.

Liu, Y. C., Guo, J. W., Benes, B., Deussen, O., Zhang, X. P., Huang, H., 2021. TreePartNet: neural decomposition of point clouds for 3D tree reconstruction. ACM Trans. Graph., 40(6), 1-16.

Liu, H. Z., Wu, Z. H., Hsu, D. F., Peterson, B. S., Xu, D. R.,

2012. On the generation and pruning of skeletons using generalized Voronoi diagrams. Pattern Recognition Letters, 33(16), 2113-2119.

Du, S. L., Lindenbergh, R., Ledoux, H., Stoter, J., Nan, L. L., 2019. AdTree: Accurate, Detailed, and Automatic Modelling of Laser-Scanned Trees. Remote Sensing, 11(18).

Wang, W. X., Huang, H. S., Du, S. Q., Li, X. M., Xie, L. F., Tang, S. J., Hong, L. P., Guo, R. Z., 2023. A highly realistic 3D reconstruction method for tree models created for virtual geographic environments. National Remote Sensing Bulletin, 1-12.

Botsch, M., Kobbelt, L., 2010. Polygon Mesh Processing. A. K. Peters, Ltd, Natick, Massachusetts.

Ju, T., 2004. Robust repair of polygonal models. Acm Transactions on Graphics, 23(3), 888-895.

Li, T. Y., Huang, P. Z., 2004. Planning humanoid motions with striding ability in a virtual environment. IEEE International Conference on Robotics and Automation, 4(4), 3195-3200.

Feng, C., Liang, J., Ren, M. D., Qiao, G., Lu, W., Liu, S. F., 2020. A Fast Hole-Filling Method for Triangular Mesh in Additive Repair. Applied Sciences, 10(3), 969.

Bischoff, S., Kobbelt, L., 2005. Structure preserving CAD model repair. Computer Graphics Forum, 24(3), 527-536.

Bøhn, J. H., Michael, W. J., 1992. Automatic CAD-model Repair: Shell-Closure. International Solid Freeform Fabrication Symposium.

Barequet, G., Sharir, M., 1995. Filling gaps in the boundary of a polyhedron. Computer Aided Geometric Design, 12(2), 207-229.

Patel, P. S., 2005. Marcum, D. L. Stitching and Filling: Creating Conformal Faceted Geometry. Proceedings of the 14th International Meshing Roundtable, 239-256.

Marchandise, E., Piret, C., Remacle, J. F., 2012. CAD and mesh repair with Radial Basis Functions. Journal of Computational Physics, 231(5), 2376-2387.

Liepa, P., 2003. Filling Holes in Meshes. Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing, 2200-205.

Kazhdan, M., Hoppe, H., 2013. Screened Poisson Surface Reconstruction. Acm Transactions on Graphics, 32(3), 13.

EDF R&D, Telecom ParisTech, 2023. CloudCompare, Version 2.13, GNU General Public License. www.cloudcompare.org. (6 April 2023).
Utrecht University, Faculty of Mathematics and Computer Science, 2023. The Computational Geometry Algorithms Library (CGAL), Version 5.5, European Union's information technologies programme Esprit. www.cgal.org. (5 May 2022).

Field, D. A., 2000. Qualitative measures for initial meshes. International Journal for Numerical Methods in Engineering, 47(4), 887-906.

Sun, Z. H., Guo, H. Y., Lu, S. L., Wen, W. L., Chen, Y. J., 2014. Filling Holes in Triangular Meshes of Plant Organs. Computer and Computing Technologies in Agriculture VII, 222-231.