IMEKO International Conference on
Metrology for Archaeology and Cultural Heritage
Lecce, Italy, October 23-25, 2017

Inner Surface Reconstruction of 3D Scanned Vessels

Stefan Spelitz[1, 2]

[1]*Austrian Academy of Sciences in Vienna, Austria, stefan.spelitz@oeaw.ac.at*
[2]*TU Wien (TUW) in Vienna, Austria, spelitz@caa.tuwien.ac.at*

***Abstract* – We present an algorithm for reconstructing the inner geometry of polygonal models with one or more openings (e.g. vessels). Known inner surface parts act as constraints to be fulfilled as part of the resulting solution. The calculation is done in an iterative fashion in order to minimize the difference between expected and achieved object volume. The output is a closed manifold polygonal mesh suitable for calculating properties like its volumetric capacity or for manufacturing (e.g. 3D printing). The algorithm has been applied to 111 antique vessels to allow a comparative study of their filling volumes.**

I. INTRODUCTION

In the field of archaeology extracting measurements of antique vessels is still desired in current research. Cups, jars and mixing bowls were used during antique banquets but it is yet unclear if there were universal measurement units used for the filling capacities of vessels. The manual process of obtaining measurements on real-life objects is labor intense and restricted to non-destructive techniques. In general, calculating the filling volume cannot be done using liquids as they may damage the object. Therefore, approximate solutions with sand or granulated material are often used instead. Those can only be applied to complete vessels without holes or by using an additional coating of the interior. Another possibility is approximating the capacity from a 2D pottery illustration ([1], [2]), although the exact vessel's wall thickness is typically unknown and a rotational symmetry is required.

Nowadays, the usage of 3D scanning technology with e.g. laser or computed tomography (CT) allow precise measurements of objects. Although CT scans result in precise data also about the interior and therefore the capacity of a vessel, obtaining a scan is expensive. In addition, the used radiation might interfere with further examinations like radiocarbon dating. Optical 3D scanning systems (e.g. laser or structured-light) result in scans with missing geometry because of inaccessible surfaces. Especially the interior of objects with narrow openings cannot be captured completely.

In order to measure volumetric capabilities, a fully reconstructed and watertight 3D model is necessary. Creating a watertight model can be done with automatic hole filling algorithms. But this will fail to represent the original object if large parts of the geometry are missing. In case the complete interior of a vessel is missing a simple hole filling algorithm will not succeed. An approximation of an object's interior would be the uniform displacement of the exterior to the inside by a constant amount. Displacing along the normal vectors of the surface (e.g. with displacement maps [3]) will result in possible geometric self-intersections not suitable for manufacturing and with incorrect volumetric statistics. A more suitable approach is to generate the displacement from a distance field ([4], [5]) which avoids overlaps in geometry.

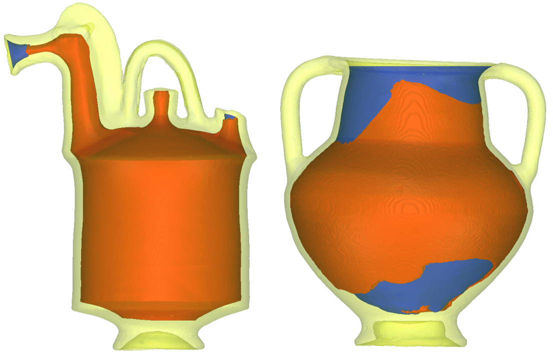

(a) Animal shaped pottery with three openings and sparsely known inner surface
(b) Vessel with one big opening and scanned interior parts at the bottom and around the rim

Fig. 1. Archeological vessels with reconstructed inner surface (red) outer scanned surface (yellow) and scanned inner parts (blue)

One caveat in generating the complete interior geometry from the exterior surface is that existing interior parts are not taken into account and are therefore not part of the resulting solution. To the best of our knowledge we propose the first technique for calculating the interior of an object while using existing inner geometry as constraints to generate a fully reconstructed, closed manifold mesh without self-intersections or overlaps. The resulting mesh's volume is optimized such that it meets a predefined target volume. By doing so we create a reconstructed vessel with an equal material volume to the original object and comparable filling capacities while keeping existing geometrical features

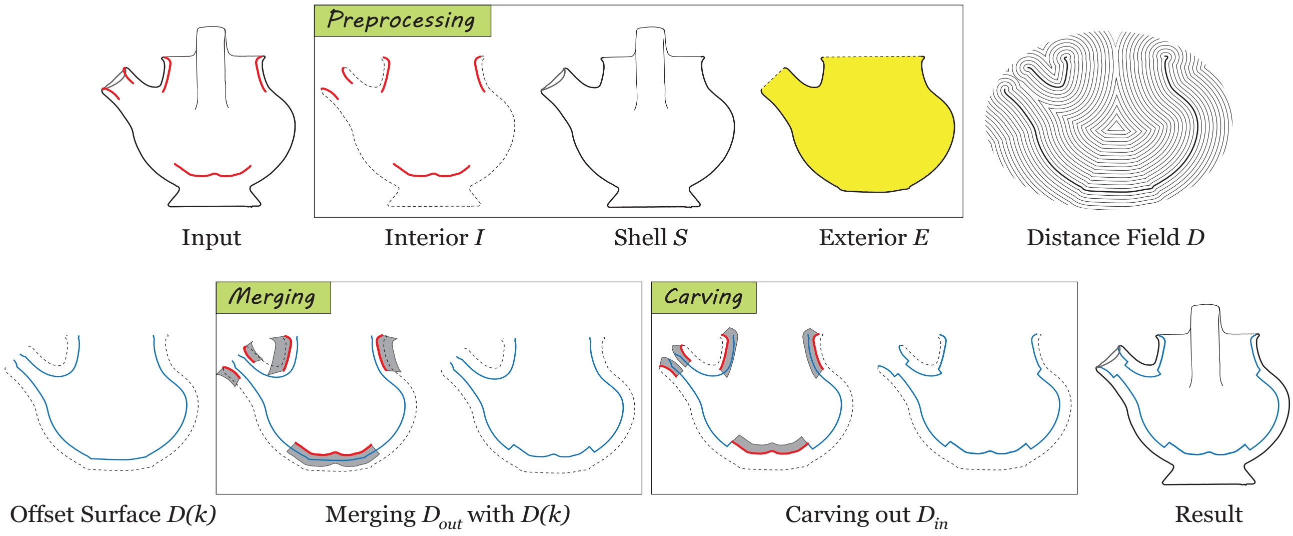


Fig. 2. Algorithmic pipeline (top-left to bottom-right). The input mesh is manually preprocessed and split up in three parts. The inner geometry I, the outer shell S and the exterior used for displacement without solid geometries (e.g. handles, pedestal). The distance field D is calculated from E. Using offset k and Equation 2 results in $D(k)$. The interior is extruded along its negative normals to create I_{out}, converted to a distance field D_{out} and merged with $D(k)$. A second extrusion along the positive normals results in D_{in} which is used to carve out geometry from the offset surface. The modified offset surface is voxelized and connected to the outer shell S which yields the final result.

intact. In other words, we complete a 3D scan by reconstructing the missing interior parts and optimize according to given constraints.

II. ALGORITHM

The algorithm is designed in an iterative manner. In every iteration the interior geometry is created using an offset k and connected to the existing outer geometry. If the resulting object's volume is close enough (in terms of relative error) to the expected target volume it stops. Otherwise, the offset is adapted and the process is repeated. The following sections describe the steps for each iteration.

A. *Input*

The geometric input to the algorithm is

- The complete outer shell S of the object
- The known inner geometry I
- The outer geometry E used for the displacement

Further parameters are

- The target volume V_t
- The maximum error threshold ϵ
- The voxel grid size r

An illustration of the different geometric meshes (S, I, E) is shown in Figure 2. These meshes are created manually in a preprocessing step by splitting up the scanned input object into inner I and outer geometry S. Removing solid parts from S results in geometry E. During each iteration E will be displaced to the inside and defines the general shape of the interior. Mesh I defines the constraints to be fulfilled on the inside and mesh S is used only after reconstructing the interior to connect both meshes and create one combined object from it.

Note that E can be identical to S which results in an interior being just an offset version of S. This is not always desirable since solid parts like handles would be hollowed out (see Figure 5). Therefore E is chosen in such a way that it only contains the outer surface of non-solid geometry. Examples for E can be seen in Figures 2 and 9.

The algorithm's goal is to optimize the reconstructed vessel's volume towards the target volume V_t within the given error margin ϵ. V_t can be calculated from the vessel's known weight w and an estimated guess of the material density ρ as

$$V_t = \frac{w}{\rho} \tag{1}$$

The voxel grid size r controls the resolution of the resulting triangle mesh and also affects the computational costs.

B. *Distance field*

We voxelize geometry E to first create a binary volume from it such that every voxel representing a point on the surface is logical *true* and every other *false*. From this a discretized distance field D_E is calculated. Every value

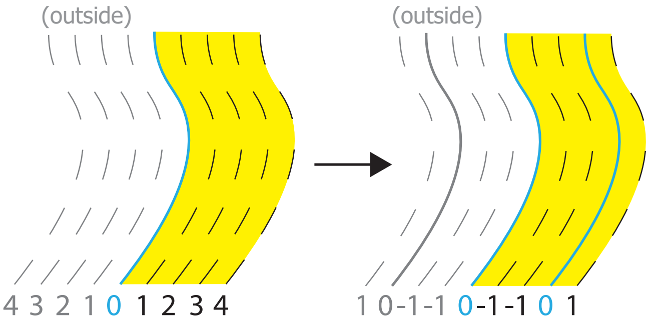
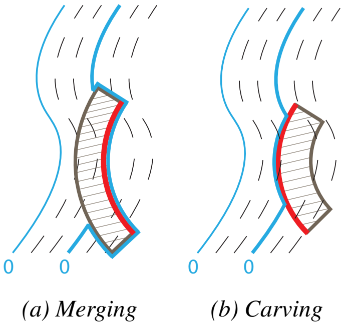


Fig. 3. Example for applying Equation 2 with $k = 3$ to an unsigned distance field (left). Zero distance classifies a point on the surface. The result (right) is a signed distance field with negative values inside the geometry (i.e. between walls). Interior mask B_{inner} depicted in yellow.



(a) Merging *(b) Carving*

Fig. 4. Manipulating the distance field of the offset surface with the known interior surface (red). Merging D_{out} with the offset surface (left) vs. Carving out D_{in} (right). The operations result in new isosurfaces at isolevel zero (blue) complying to the inner surface constraints.

in D_E is the distance to the closest point of the geometry. A distance equal to zero is a point exactly on the surface. Since E is an open polygon mesh with one or more boundaries, the distance field will be initially unsigned since there is no concept of inside and outside.

To classify which voxels are located outside of the vessel we use a simple mesh processing algorithm [6] to close all existing holes of E and create a new mesh from it. This result is again voxelized and acts as a binary mask B_{inner} to differentiate between inside and outside of the vessel for later use. An example of B_{inner} can be seen in Figure 2 depicted in yellow for exterior E.

C. *Offset surface*

The offset value k controls the distance the inner surface will be displaced from the outer surface. It is adapted in each iteration and implicitly controls the object's volume. Increasing k leads to thicker walls and therefore an increased material volume. The outer geometry's unsigned distance field D_E is offset by:

$$D(k) = |D_E - \frac{k}{2}| - \frac{k}{2} \tag{2}$$

This results in two potential isosurfaces at isovalue zero around the original surface E with one surface inside and one outside of the object (see Figure 3). The signed distance field $D(k)$ represents a solid object with one wall of thickness k to the inside and another wall with thickness k to the outside of the vessel. We will only concentrate on the inner wall and disregard the outer displacement by using the binary mask B_{inner} to exclude all voxels on the outside.

D. *Inner surface constraints*

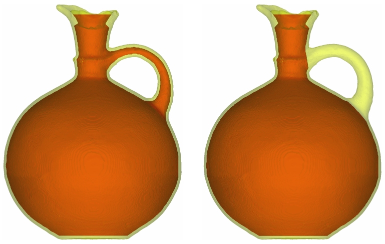
So far the distance field only represents a simple displacement and the inner wall does not necessarily pass through the given known interior parts. To satisfy the constraints given by the interior surface I we will further manipulate $D(k)$.

It is assumed that the given interior geometry I consists of (not necessarily connected) thin surface patches with boundaries. The normal vectors are correctly oriented and point to the inside of the vessel. If parts of I are located within the inner wall defined by $D(k)$ (Figure 4b) we want to remove (carve out) surrounding parts of the wall. In case parts of I are not connected to the wall (Figure 4a) we want to have the wall extend to (merge with) these parts. In both cases the resulting, modified wall should interpolate the geometry of I.

This approach is implemented by creating two solidified meshes I_{in} and I_{out} from the original interior I. These meshes represent solid volumes extruded from the surface patches either along the surface normals (I_{in}) or along the negative normals (I_{out}).

The extrusion depth needs to be chosen carefully. The patches need to be extruded deep enough such that they overlap with the current offset surface but little enough to minimize self-intersections and avoid intersecting other geometry of the vessel. The extrusion depth for I_{in} is chosen to be k. This guarantees to overlap with the offset surface if I is between the original surface and the offset surface. The extrusion depth for I_{out} can be chosen in a more liberal way and was empirically set to $r/10$, assuming the offset surface is not farther away from the interior than this distance which was true for all tested vessels.

The extruded meshes are each voxelized and converted to distance fields (D_{in}, D_{out}), such that they can either be merged with the offset surface of $D(k)$ or carved out from it. We create a new distance field $D(k)_{new}$ where isovalue zero defines an isosurface interpolating the given inner surface and otherwise connecting to the previously generated offset surface. The new distance field including the inner



(a) Using $E = S$ *(b) Using E without solid geometry*

Fig. 5. Example of the undesired effects of hollowing out solid geometry (left) vs. using an exterior mesh E with solid parts explicitly removed (right). Both reconstructions provide identical ceramic volumes and filling capacities.

surface and illustrated in Figure 4 is calculated with:

$$D(k)_{new} = \max(\min(D(k), D_{out}), -D_{in}) \qquad (3)$$

E. *Polygonization*

Converting the distance field to a polygonal mesh is done with Marching Cubes [7] on the zero valued isosurface. By only considering the inner voxels the triangulated result will be an open, polygonal surface patch. Since Marching Cubes can create a polygonal mesh with a high triangle count, a decimation step [8] has been introduced to reduce the number of triangles.

F. *Connecting inside and outside*

The resulting inner surface reconstruction is then connected to the open exterior mesh S by connecting the boundaries of both meshes using the algorithm of Zou et al. [9]. The connected mesh might contain possible self intersections, small holes or flipped normals. These are fixed with a post-processing step [6] resulting in a one component, closed manifold mesh.

G. *Offset refinement and termination*

From the combined mesh created with offset k the mesh's volume $V(k)$ can be calculated [10]. Using the target volume V_t, the absolute value of the volume difference $V_{diff}(k) = V(k) - V_t$ is expected to be zero. Solving for k is equivalent to finding the root of $V_{diff}(k)$ and can be achieved with e.g. *regula falsi* or the secant method. Since an exact solution can not be expected the algorithm terminates as soon as the absolute relative error

$$\delta = |\frac{V(k) - V_t}{V_t}| \leq \epsilon \qquad (4)$$

is below a given threshold ϵ. The algorithm is said to not converge if either the number of iterations exceed a predefined maximum amount or if the change of $V_{diff}(k)$ between the last two iterations is not sufficiently large enough (i.e. no progress).

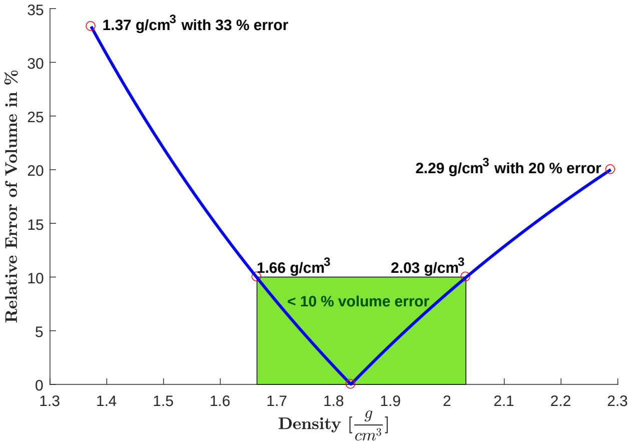


Fig. 6. Illustrating the non-linear error in volume in relation to different material density values. The density range is $[1.37, 2.29]$, i.e. $\pm 25\%$ of the optimal density of $1.83\frac{g}{cm^3}$. Densities between $[1.66, 2.03]$ result in less than 10 % error in volume.

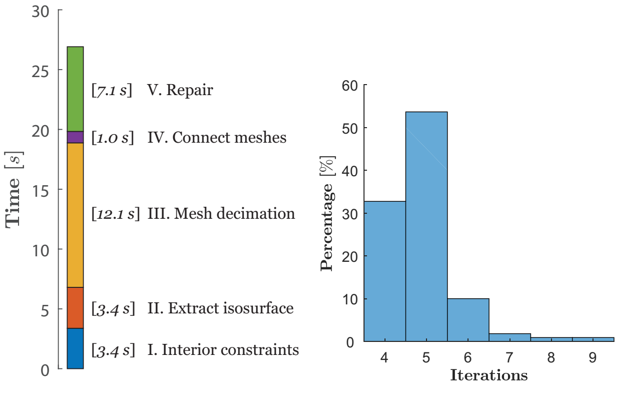
III. RESULTS

In order to study the filling capacities of antique vessels (especially attic and cypriotic pottery) a collection of 3D scans (with laser and structured light) was obtained. Some items (e.g. bowls) were available as complete scans and could therefore be used to calculate a representative average material density. These densities were then used as input to the algorithm for reconstructing the remaining, partly-scanned vessels.

The algorithm's implementation was done in Matlab. So far, 111 vessels were used as input to the algorithm. These vessels were successfully reconstructed with a maximum relative error (Equation 4) of $\epsilon = 0.0001$ (i.e. 0.01%). The voxel grid size for calculating the distance fields was chosen to be 300^3, i.e. $r = 300$ voxels in each dimension. This voxel count is high enough for fine adjustments to the offset k and low enough for fast execution.

The run-time statistics are shown in Figure 7. Execution was done on a laptop PC with an Intel Core i7-6600U CPU and 12 GB RAM. The average time for each iteration is split up into several stages. Applying the interior constraints was described in Section D, extracting the isosurface in Section E, connecting and repairing in Section F. With one iteration taking about 27 seconds on average and requiring a median of 5 iterations a full reconstruction can be done in under 3 minutes with no user interaction necessary.

Execution was repeated for different values of r (see Figure 8). While connecting the meshes (step IV) and re-



(a) Average time per iteration *(b) Iterations necessary*

Fig. 7. Runtime statistics for 111 *vessels with* $r = 300$. *The average time for one iteration is split up into several stages (left). The iteration count (right) shows that* 96% *of all runs need* 6 *or less iterations to converge and* 86% *need* 5 *or less.*

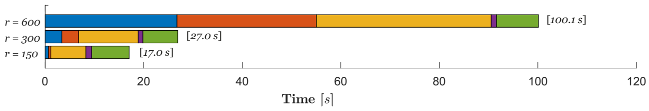


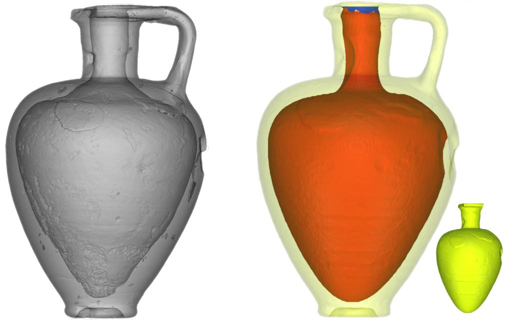
Fig. 8. Average time per iteration for different voxel grid sizes r.

pairing (step V) take up a constant amount of time regardless of the choice for r, every other step is dependent on the voxel grid size. Voxelization of the inner surface and polygonization of the reconstruction are naturally related to r and since a higher voxel count will create a finer triangulation the decimation step has to deal with a larger amount of data.

A collection of some representative results is given in Figure 10. In Figure 9 the reconstruction of a vessel is compared to its CT scan.

IV. DISCUSSION AND FUTURE WORK

Material density So far, the vessel's target volume is calculated from the vessel's weight and material density (Equation 1). In order to get realistic reconstruction results a correct average material density ρ is necessary. For ancient vessels ρ is typically unknown but can be inferred from other objects with known material manufactured during the same time period. An average value can be determined with non-destructive complete 3D scans (e.g. CT, laser) or destructive material tests. Choosing an incorrect material density leads to a non-linear error in volume as depicted in Figure 6. But even using a correct average material density might still only give approximate results. As one can see in Figure 9 the reconstruction result has a surplus capacity of about 5% although both objects have the same material volumes. This is likely due to air bubbles in the material which don't contribute to the original object's capacity but still are part of the reconstruction's filling volume. By collecting statistics about frequency and size of enclosed air in pottery through CT scans one could account for this discrepancy by reducing the filling capacity of the reconstruction accordingly.



(a) Direct Volume Rendering (DVR) of a CT scan *(b) Reconstruction result by using exterior* E *(bottom right)*

Fig. 9. Comparison between the 'ground truth' of a vessel's interior captured through computed tomography[1] *(CT) and the reconstructed inner geometry. Both vessels have identical ceramic volumes and similar capacities of 44.4 ml (left) vs. 46.6 ml (right).*

Preprocessing The manual preprocessing step of splitting up the mesh into different parts (S, I, E) could be partly automated. Dey et al. [12] proposed a possibility to identify topological features like handles. These parts of the mesh could then be removed and the holes automatically closed to create an exterior mesh E. However, it still remains a challenge to automatically remove solid parts which have no distinct topological features (like pedestals or ornaments) since identifying these parts needs expert domain knowledge about pottery.

Runtime The runtime of the current implementation is suboptimal. Especially with a high voxel count the calculation time becomes unfeasible. The mesh decimation step used for reducing the triangle count after Marching Cubes can be eliminated by using a different polygonization algorithm instead (e.g. [13]) or a decimation algorithm specifically designed for Marching Cubes (e.g. [14]). Another possible improvement is not extruding and voxelizing the interior I in every iteration but doing it once in a preprocessing step independent of the offset k.

[1] Compare to Trinkl, 2013 [11]. Vessel from Universalmuseum Joanneum (UMJ), inventory number 4214

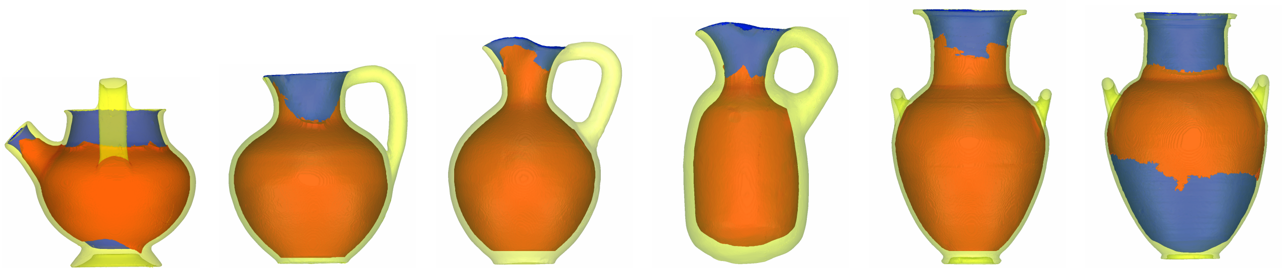


Fig. 10. Reconstruction result examples. Reconstructed inner surface (red), outer scanned surface (yellow) and scanned inner parts (blue).

Artifacts Marching cubes might introduce block-like artifacts and non-smooth surfaces. For volume calculations this does not impose a problem but it does not provide realistic reconstruction results. These artifacts can be reduced by increasing the voxel resolution or by using a different isosurface extraction technique (e.g. dual contouring [15] or marching squares [16]).

REFERENCES

[1] L. M. Senior and D. P. Birnie III, "Accurately estimating vessel volume from profile illustrations," *American Antiquity*, vol. 60, no. 2, pp. 319–334, 1995.

[2] L. Engels, L. Bavay, and A. Tsingarida, "Calculating vessel capacities: A new web-based solution," in *Shapes and Uses of Greek Vases (7th-4th centuries BC). Proceedings of the Symposium held at the Université libre de Bruxelles 27-29 April 2006*, pp. 129–133, CReA-Patrimoine, 2009.

[3] R. L. Cook, "Shade trees," *ACM Siggraph Computer Graphics*, vol. 18, no. 3, pp. 223–231, 1984.

[4] D. Pavić and L. Kobbelt, "High-resolution volumetric computation of offset surfaces with feature preservation," in *Computer Graphics Forum*, vol. 27, pp. 165–174, Wiley Online Library, 2008.

[5] S. Liu and C. C. Wang, "Fast intersection-free offset surface generation from freeform models with triangular meshes," *IEEE Transactions on Automation Science and Engineering*, vol. 8, no. 2, pp. 347–360, 2011.

[6] M. Attene, "A lightweight approach to repairing digitized polygon meshes," *The visual computer*, vol. 26, no. 11, pp. 1393–1406, 2010.

[7] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," in *ACM siggraph computer graphics*, vol. 21, pp. 163–169, ACM, 1987.

[8] S. Forstmann, "Fast-quadric-mesh-simplification," `https://github.com/sp4cerat/Fast-Quadric-Mesh-Simplification`, 2014, Accessed 2017-09-25.

[9] M. Zou, T. Ju, and N. Carr, "An algorithm for triangulating multiple 3d polygons," in *Computer Graphics Forum*, vol. 32, pp. 157–166, Wiley Online Library, 2013.

[10] C. Zhang and T. Chen, "Efficient feature extraction for 2d/3d objects in mesh representation," in *Image Processing, 2001. Proceedings. 2001 International Conference on*, vol. 3, pp. 935–938, IEEE, 2001.

[11] E. Trinkl, *Corpus Vasorum Antiquorum, Oesterreich, Beiheft 1: Interdisziplinaere Dokumentations- und Visualisierungsmethoden*. Austrian Academy of Sciences Press, 2013.

[12] T. K. Dey, K. Li, J. Sun, and D. Cohen-Steiner, "Computing geometry-aware handle and tunnel loops in 3d models," in *ACM Transactions on Graphics (TOG)*, vol. 27, p. 45, ACM, 2008.

[13] R. Shu, C. Zhou, and M. S. Kankanhalli, "Adaptive marching cubes," *The Visual Computer*, vol. 11, no. 4, pp. 202–217, 1995.

[14] R. Shekhar, E. Fayyad, R. Yagel, and J. F. Cornhill, "Octree-based decimation of marching cubes surfaces," in *Proceedings of the 7th conference on Visualization'96*, pp. 335–ff, IEEE Computer Society Press, 1996.

[15] S. Schaefer, T. Ju, and J. Warren, "Manifold dual contouring," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 3, pp. 610–619, 2007.

[16] C. Ho, F.-C. Wu, B.-Y. Chen, Y.-Y. Chuang, M. Ouhyoung, *et al.*, "Cubical marching squares: Adaptive feature preserving surface extraction from volume data," in *Computer graphics forum*, vol. 24, pp. 537–545, Wiley Online Library, 2005.