# CAD Reconstruction of Watertight Surface Meshes Using Polycube Parameterization and Subdivision Surfaces

Moustafa Alsayed Ahmad[1(✉)], Majid Hojjat[1], and Kai-Uwe Bletzinger[2]

[1] BMW Group, Petuelring 130, 80809 München, Germany
{Moustafa.Alsayed-Ahmad, Majid.Hojjat}@bmw.de
[2] Technische Universität München, Arcisstraße 21, 80333 München, Germany
kub@tum.de

**Abstract.** Topology optimization can produce optimal geometries in terms of the simulated physics, but they are primarily represented as watertight surface meshes which enclose a volume. As a result, they cannot be incorporated directly into the design process, which is mainly carried out using Computer Aided Design (CAD) data. To enable the integration into the design process, an interpretation step is required to convert these surface meshes into CAD models, known as CAD reconstruction. However, CAD reconstruction proved to be time-consuming and makes it difficult to deploy topology optimization in the industry and in generative design when carried out manually. This contribution presents a method to automatically construct a CAD model starting from a watertight surface mesh. First, the surface mesh is divided into different components based on its topology, each component being called a part. Second, each part is segmented using polycube segmentation methods, where each segment is called a patch. The patches of each part form a polycube, which can be easily meshed using coarse quad elements. The quad meshes of all parts are then assembled to form a quad mesh, which approximates the original watertight surface mesh. The quad mesh serves as a basis for the generation of the subdivision surface. The resulting subdivision surface approximates well the original surface mesh and is flexible for further post-processing or modification in later steps.

**Keywords:** Computer Aided Design (CAD) · CAD reconstruction · Topology Optimization · Polycube parameterization · Subdivision surfaces


J. Rieser et al. (Eds.): *Proceedings of the Munich Symposium on Lightweight Design 2022*, pp. 120–133, 2023. https://doi.org/10.1007/978-3-031-33758-1_10

## 1 Introduction

Converting meshes into CAD models has always been an active area of research. This process may also be known as CAD reconstruction or reverse-engineering. In this process, the raw positional data (position of the vertices of the mesh or the point cloud) gets interpreted and a series of parametric curves and surfaces are constructed to approximate it. A list of scientific developments and publications have been made in this field. In [12] the authors detect surface features on the mesh obtained from topology optimization, then use those features as a basis for the generation of biquartic surface splines; whereas in [17] the authors detect surface types or regions using curvature information and try to compute feature lines between those regions and fit NURBS surfaces over them. Both [12] and [17] rely on computing surface features or types, which is not robust and is strongly sensitive to the mesh quality. In [18], the authors propose a semi-automatic procedure of extracting a curve-skeleton approximating the topology of the underlying mesh and then manually constructing a CAD model (mainly through sweep-operations). This approach has been implemented in an automatic way in [13] and [6]; however, the surface complexity around the curve-skeleton is very low. [21] and [16] rely on similar procedure of approximating the mesh using medial surfaces (include curve- and surface-skeletons), then automatically constructing the CAD model around the skeleton-curves; in [21] the authors use CSG-models (Constructive Solid Geometry models) to approximate the mesh abstracted by the curve-skeleton and in [16] the authors perform offsetting and manipulation of the surface-skeleton to generate a solid geometry. These approaches lack the ability of reconstructing surfaces, which are not tubular or exhibit simple cross section around its skeleton. An approach presented by [11] relies on a voxel structure to re-mesh the given mesh from topology optimization and then using marching cubes to generate a surface mesh and converting every single shell element into a CAD patch, which is very robust but generates a lot of CAD patches which makes the CAD model hard to manipulate. On the other hand, there have been numerous developments from the industry in implementing automatic CAD reconstruction workflows. One approach implemented by CATIA V5 [7] is to some extent similar to [11]; however, as pointed out before, it has the disadvantage that it produces high number of surface patches; the models are also hard to edit and they exhibit no natural parametrization (like feature lines where there is a ridge, or a feature line separating two distinct components in the geometry). Another method, which is followed by many software developers like nTopology, Synera and Autodesk [15] performs re-meshing of the given mesh to create a quad mesh, then the quad elements are merged according to some geometric criteria with adjacent quad elements to form bigger patches, until either there are no more quad elements to merge, or no merge satisfies the specified geometric criteria. This approach is very robust since it can work for all geometries regardless of whether they are watertight or not. However, if one wishes to edit the CAD model produced by this approach, the full underlying quad mesh needs to be edited which is not easy. Furthermore, this approach relies on pure local geometric criteria to assess the merge quality of the quad

elements, and this is why the resulting patches can be small even though the underlying surface is almost flat, or the patches can be large even though the underlying surface is highly curved. In addition, the patches do not represent a natural partitioning of the topology optimization mesh since the feature lines don't necessarily partition two semantic parts or two ridges.

In this paper, we want to develop a new approach, which produces CAD models that contain few CAD patches and are easy to edit and manipulate in a commercial software. In addition, we don't desire the CAD model to exactly match the mesh from topology optimization, since we perform afterwards shape optimization to detect the optimal form. These three criteria form the guidelines in designing our CAD reconstruction workflow as presented in Fig. 1.

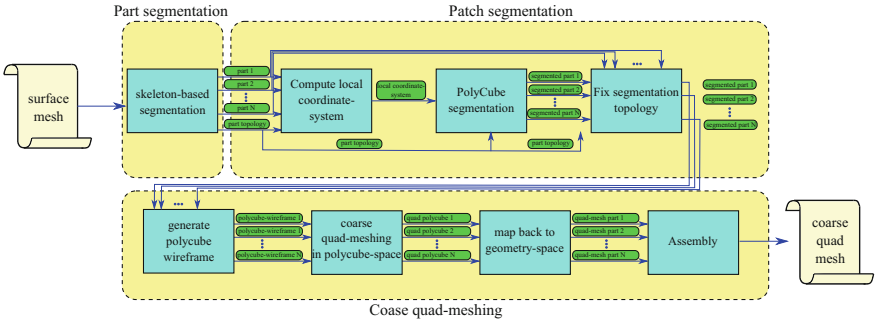


**Fig. 1.** Workflow of reconstructing a watertight surface mesh and creating a coarse quad dominated mesh (base mesh) to be later used as a base to the subdivision surface

## 2 Methods

In this section, the method of constructing a subdivision surface based on a given mesh resulting from topology optimization is presented and discussed. The workflow proposed to construct a coarse quad mesh or also called a base mesh (basis of the subdivision surface) from a given watertight surface mesh is given in Fig. 1.

The workflow starts by partitioning the surface mesh into multiple part meshes; these part meshes are denoted by the set $M_{\mu}^{\text{part}}(V,F), \mu = 1, \cdots, N^{\text{parts}}$ (see Sect. 2.1). Then each part $M_{\mu}^{\text{part}}$ is partitioned further using a polycube parameterization method into patches, which are denoted by the set $M_{\mu,\nu}^{\text{patch}}(V,F), \nu = 1, \cdots, N^{\text{patches}}$ (see Sect. 2.2). Polycube parameterization is first proposed in [20] for texture mapping, and it segments a given surface mesh into a a union of cubes of equal size, see Fig. 2. These cubes are all aligned to an orthogonal coordinate system, and each color in Fig. 2 represents a distinct patch. A collection of patches $M_{\mu,\nu}^{\text{patch}}$ such that $\cup_{\nu} M_{\mu,\nu}^{\text{patch}} = M_{\mu}^{\text{part}}$ is converted

to a polycube structure using a polycube deformation method [22]. We denote the polycube structure $M_{\mu}^{\text{PC}}$, where for each part we get a corresponding polycube structure. The coarse quad mesh is generated based on the polycube structure and then it gets mapped back to the shape of the corresponding part (see Sect. 2.3). The coarse quad mesh is then used as the base mesh for the generation of the subdivision surface (see Sect. 2.4). Lastly, the subdivision surface is post-processed, so that its deviation from the input surface mesh or also called the target mesh is minimal (see Sect. 2.5).

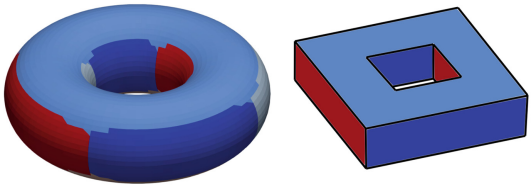

**Fig. 2.** (*left*) A given surface mesh with polycube parameterization (*right*) the corresponding polycube structure

### 2.1 Part Segmentation

Part segmentation is the process of partitioning a surface mesh into multiple-parts $M_{\mu}^{\text{part}}(V, F), \mu = 1, \cdots, N^{\text{parts}}$, where each part is a set of faces $S_{\mu}^{part} = [F_1, F_2, \cdots, F_{N^{\text{part-faces}}}]$ where $N^{\text{part-faces}}$ here is the number of faces within part $\mu$. Each part is represented either as a part mesh $M_{\mu}^{part}(V, F)$ or a part segmentation with the set $S_{\mu}^{\text{part}}(F)$. In [9], they performed part segmentation before patch segmentation using polycube parameterization. This approach has an advantage in that it reduces the number of corner vertices in the polycube structure that is generated later. The reason behind this is that polycube structures are aligned with the axes, so even if a straight geometry, such as a cylinder with small variations in its shape, is segmented using a single polycube structure, it may have more than eight corner points. This is because a perfect cylinder can be ideally segmented with a single polycube structure with eight corner points. Hence, if we seek a polycube structure with few corner-points, we can partition the surface mesh into multiple parts, in such a way each part is aligned to a certain coordinate system. One can either manually partition the surface mesh into multiple parts or use curve-skeletons-based segmentation [2] to achieve that.

### 2.2 Patch Segmentation

After all part meshes $M_{\mu}^{\text{part}}(V, F)$ are extracted, each one of them gets further segmented using a polycube segmentation method. The method of choice in this paper is the Harmonic Boundary-Enhanced Centroidal Voronoi Tesslelation or

short HBECVT [10]. For a visual discription of the patch segmentation procedure see Fig. 3. The procedure of patch segmentation starts by extracting local coordinate systems of each extracted part mesh and the interface (described by the vertices shared by two part meshes $M_{i,j}^{\text{interface}}(V) = M_i^{\text{part}} \cap M_j^{\text{part}}$, where parts $i$ and $j$ are adjacent). For either the part meshes or the interfaces, their corresponding local coordinate system is computed using the Principal Component Analysis (PCA). The PCA-method computes the axes of the local coordinate system by determining the eigenvectors of the following covariance-matrix:

$$\mathbf{C} = \frac{1}{1 - N^{\text{part-vertices}}}(\mathbf{V} - \bar{\mathbf{V}}) \otimes (\mathbf{V} - \bar{\mathbf{V}}), \quad \mathbf{V} \in M_{\mu}^{\text{part}}(V, F) \quad \text{or} \mathbf{V} \in M_{i,j}^{\text{interface}}(V) \tag{1}$$

where $\bar{\mathbf{V}} = \frac{1}{N^{\text{part-vertices}}} \sum_{i=1}^{N^{\text{part-vertices}}} \mathbf{V}_i$, $\mathbf{V}_i$ is the ith row of the matrix $\mathbf{V}$ of vertex-coordinates within the part mesh or the interface. The eigenvectors of the covariance-matrix $\mathbf{C}$ sorted from the smallest to the largest eigenvalue determines the local x,y and z axes of the local coordinate system as shown in Fig. 3b. Now the faces of each part mesh are segmented by minimizing the following energy-functional with respect to the labels $\mathbf{c}$:

$$E(\mathbf{n}_F, \mathbf{c}) = \sum_{i=1}^{N^{\text{part-faces}}} \left[ \frac{L}{\sum_{l=1}^{L} d(\mathbf{n}_{F_i}, \mathbf{c}_l)^{-1}} \right], \tag{2}$$

where $\mathbf{n}_F$ is the matrix of face-normals, $\mathbf{c}$ is matrix of six labels of axis vectors aligned to the previously computed local coordinate system ($+x$,$-x$,$+y$,$-y$,$+z$,$-z$), $L$ is six (number of labels), $\mathbf{n}_{F_i}$ is the ith face's normal vector and $\mathbf{c}_l$ is the lth label vector. Faces, which are adjacent to each other and are assigned to the same label vector are called a patch. The function $d(\mathbf{n}_{F_i}, \mathbf{c}_l)$ mathematically described in Eq. 3 computes in term 1 the deviation of the face's normal vector to the label's vector and penalizes in term 2 the adjacent face from having a different label compared to the label the current face has, where $\tilde{n}_{i,l}$ is the number of faces in the neighborhood of face $F_i$, which have a label different that the assigned label to face $F_i$ and $\lambda$ is a multiplication-parameter and has the value 0.5. In addition, two more terms were added to the original formulation in [10]. Term 3 prevents the current face $F_i$ from being assigned to a label when at least one of its neighboring faces has an opposite label (for example $+x$ and $-x$), where $\tilde{o}_i$ equals the number of faces in the neighborhood of face $F_i$, which have opposite label compared to face $F_i$. Term 4 prevents the face sitting next to an interface from having a label which has a label vector parallel to the x-axis of the interface, where $\tilde{a}_i$ equals the number of those faces. $p$ is the penalty-factor and equals $10^5$.

$$d(\mathbf{n}_{F_i}, \mathbf{c}_l) = \sqrt{\underbrace{\|\mathbf{n}_{F_i} - \mathbf{c}_l\|^2}_{term\ 1} + \lambda(\underbrace{\tilde{a}_{i,l}}_{term\ 2}) + p(\underbrace{\tilde{o}_i}_{term\ 3} + \underbrace{\tilde{a}_i}_{term\ 4})} \tag{3}$$

So far if we perform the patch segmentation as discussed, we would get a segmentation as shown in Fig. 3c; however, one can notice that at the interface,

the patch segmentation of parts 1 and 2 are not matching. This is due to the fact that the local coordinate system of parts 1 and 2 are not aligned. To solve this problem, we propose in this paper blending the local coordinate systems of parts 1 and 2 to the interface's local coordinate system. At Fig. 3d the blending function is plotted at the rightmost side, and this blending distance is a user parameter and is chosen in this paper equal 4 times the average element size.

### 2.3 Coarse Quad Meshing

After extracting the patch segmentation using HBECVT-method (polycube-based segmentation), we need now to convert each part into a polycube structure and perform coarse quad meshing upon it, then the vertices of the coarse quad mesh can be easily mapped from the polycube structure to the part mesh.

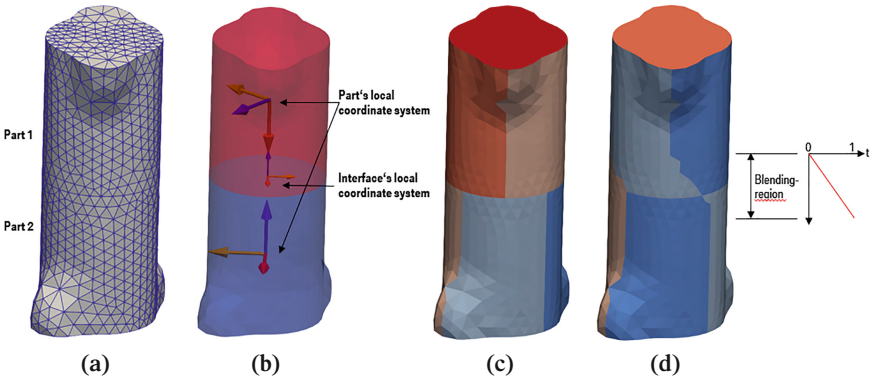


**Fig. 3.** Patch segmentation procedure; (**a**) given a surface mesh (**b**) a part segmentation is computed and local coordinate systems for each part and the interface between the parts is determined (**c**) patch segmentation using HVECVT-method without using the blending procedure (**d**) patch segmentation using the HBECVT-method using the blending procedure

To generate a polycube structure, we use the method proposed in [22], which takes a part mesh with its patch segmentation and deforms the part mesh in such a way that the faces' normal vectors are aligned to the normal vectors dictated by the label vectors assigned to each face by minimizing the energy in Eq. 2. This deformation is formulated as a successive solution of a linear system of equations; for full details of the deformation-procedure see [22]. The number of iterations needed to re-solve this linear system in our paper is 20. After deforming the mesh, we get a mesh similar to the one shown in Fig. 4a.

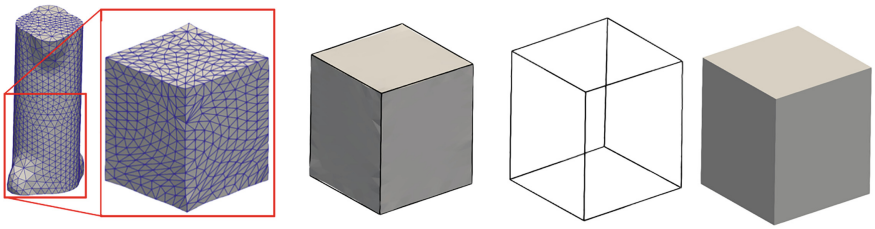


**Fig. 4.** Procedure of generating the polycube structure; (**a**) the surface mesh is deformed according to [22] (**b**) the wireframe connecting corner-vertices is constructed (**c**) the wireframe is deformed to form the polycube structure (**d**) quad elements are created based on the polycube structure

Mesh deformation according to [22] does not produce perfect polycube structures (see not fully aligned faces in Fig. 4b). A solution would be to extract the corresponding wireframe structure, which consists of vertices located at the mesh's corner-vertices (vertices surrounded by three or more distinct labels) and edges connecting them. Each edge of the wireframe structure lies between two perpendicular label vectors. For a perfect polycube structure, those wireframe edges should be perpendicular to the two labels-vectors. After the mesh-deformation and the extraction of the wireframe structure, the edges are not in general perpendicular to the two label vectors. To make them so, we deform the wireframe until the edge-vectors are perpendicular to the two label vectors; see Fig. 4c.

After having a perfectly aligned wireframe structure, quad elements are created on each patch and then their vertices are mapped back the shape of the surface mesh. These quad elements are computed by firstly using triangulation algorithms for 2-dimensional polygons, then triangles are merged together according to some mesh quality criteria to form quad elements; see Fig. 4d.

After each part mesh is meshed with quad elements, they get assembled to form the coarse quad mesh or the base mesh needed for the generation of the subdivision surface.

### 2.4 Generation of Subdivision Surfaces

Now given the coarse quad mesh generated by the methods presented in Sect. 2.3, which also will be called the base mesh, the subdivision surface is created using the Catmull-Clark method [4]. The Catmull-Clark method is chosen since the base mesh is quad dominated; however, for triangular meshes one can use the Loop subdivision method [14]. The Catmull-Clark method is based on the idea of successively adding new vertices at the middle of quad faces and at the middle of each edge and then determining the position of those new vertices and modifying the position of the original ones based on some sort of averaging scheme as shown in Fig. 5a–d; the reader is adviced to read [4] to see full details of the averaging

schemes used. Those refined meshes shown in Fig. 5b–d are called subdivision meshes and the subdivision surface is the limit of those subdivision meshes as the number of subdivision iterations reaches infinity.

The vertex position of the base mesh is represented by the $(N \times 3)$ matrix $\mathbf{V}^{(0)}$, where 0 indicates that this is the original vertex position or the 0th subdivision iteration. The successive vertex positions using the Catmull-Clark method can be computed using the following linear operation:

$$\mathbf{V}^{(i+1)} = \mathbf{S}^{(i)} \cdot \mathbf{V}^{(i)}, \tag{4}$$

where $\mathbf{S}^{(i)}$ is called the subdivision matrix used in the ith iteration, and it is a rectangular matrix, since the number of vertices at the subdivision iteration $i+1$ is larger than the number of vertices at the subdivision iteration $i$. The subdivision-matrix which gets multiplied by the original vertex positions $\mathbf{V}^{(0)}$ to get $\mathbf{V}^{(i+1)}$ can be computed as follows:

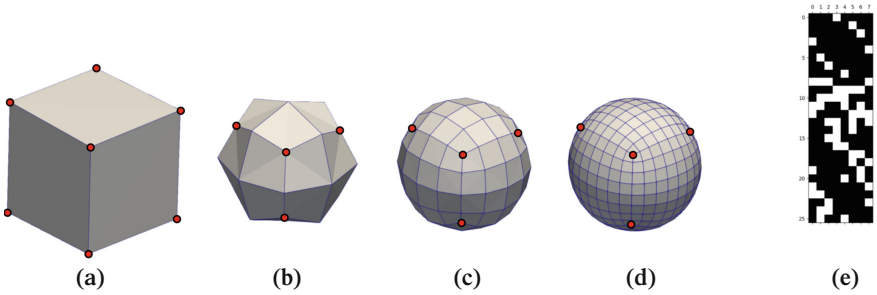


**Fig. 5.** Generation of subdivision meshes (**a**) starting from the base mesh (**b**) 1st subdivision iteration (**c**) 2nd subdivision iteration (**d**) 3rd subdivision iteration, where the red vertices are the original vertices getting averaged according to the Catmull-Clark method and (**e**) represents the non-zero structure of the subdivision matrix $\mathbf{S}^{(0)}$, where the non-zero entries are represented as black squares

$$\mathbf{S}_i = \prod_{j=0}^{i} \mathbf{S}^{(j)}. \tag{5}$$

This subdivision matrix will be used in the following section to map variation in quantities back and forth between the base mesh and the subdivision mesh; for the non-zero structure of the subdivision matrix $\mathbf{S}^{(0)}$ (after one subdivision iteration of the base mesh shown in Fig. 5a) see Fig. 5e.

### 2.5 Post-processing of Subdivision Surfaces

After the subdivision surface is computed, we get a surface close to the surface mesh which originated from topology optimization or as we call it the target mesh. However, this subdivision surface due to the subdivision averaging does

shrink and it deviates from the target mesh; see Fig. 6. To mitigate this deviation, a post-processing step is proposed to modify the base mesh in such a way that the subidivion surface becomes as close to the target mesh as possible. To do that we use node-based (or vertex-based) shape optimization in order to modify the vertex position of the base mesh. Node-based shape optimization uses the vertex position as the design variable. In order to modify the base mesh such that the corresponding subdivision surface is as close as possible to the target-mesh, we propose to maximize the volume of the subdivision surface $f_{\text{volume}}(\mathbf{V}^{(i)})$ and use a packaging-constraint $f_{\text{packaging}}(\mathbf{V}^{(i)})$ (using the penalty-method with penalty-factor $p$ equals $10^5$) to limit the volume-growth in such a way that the subdivision surface stays contained within the target mesh. An additional constraint on the thickness $f_{\text{thickness}}(\mathbf{V}_j^{(0)})$ of the base mesh is imposed to prevent self-penetration during the optimization. The optimization-statement is presented in Eq. 6.

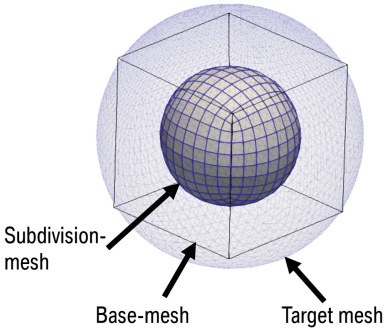


**Fig. 6.** Deviation of the subdivision mesh from the target mesh (to be approximated)

$$\begin{aligned} &\text{minimize} \quad -f_{\text{volume}}(\mathbf{V}^{(i)}) + p \cdot f_{\text{packaging}}(\mathbf{V}^{(i)}), \\ &\text{design variables} \quad \mathbf{V}^{(0)} \\ &\text{s.t} \quad \mathbf{V}^{(i)} = \mathbf{S}_i \cdot \mathbf{V}^{(0)}, \\ &\text{s.t} \quad f_{\text{thickness}}(\mathbf{V}_j^{(0)}) \geq t_{min} \quad \forall \mathbf{V}_j^{(0)} \in \mathbf{V}^{(0)} \end{aligned} \tag{6}$$

The detailed formulation for the functions $f_{\text{volume}}$, $f_{\text{packaging}}$ and $f_{\text{thickness}}$ along with their shape derivative is presented in Eqs. 7 [5], 8 and 9 respectively.

$$f_{\text{volume}}(\mathbf{V}^{(i)}) = \frac{1}{6} \sum_{\mu,\nu,\gamma \in F^{(i)}} \mathbf{V}_\mu^{(i)} \cdot (\mathbf{V}_\nu^{(i)} \times \mathbf{V}_\gamma^{(i)}), \quad \frac{\partial f_{\text{volume}}(\mathbf{V}^{(i)})}{\partial \mathbf{V}_j^{(i)}} = A(\mathbf{V}_j^{(i)}) \mathbf{n}(\mathbf{V}_j^{(i)}), \tag{7}$$

where $A(\mathbf{V}_j^{(i)})$ is the vertex area of the vertex $\mathbf{V}_j^{(i)}$, and $\mathbf{n}(\mathbf{V}_j^{(i)})$ is the normal vector of the vertex $\mathbf{V}_j^{(i)}$

$$\begin{aligned} f_{\text{packaging}}(\mathbf{V}^{(i)}) &= \textstyle\sum_{\mu \in V^{(i)}} \max\big((\mathbf{V}_\mu^{(i)} - \tilde{\mathbf{V}}_\mu^{(i)}) \cdot \mathbf{n}^{\text{target mesh}}(\tilde{\mathbf{V}}_\mu^{(i)}), 0\big), \\ \frac{\partial f_{\text{packaging}}(\mathbf{V}^{(i)})}{\partial \mathbf{V}_j^{(i)}} &= \mathbf{n}^{\text{target mesh}}(\tilde{\mathbf{V}}_\mu^{(i)}), \end{aligned} \tag{8}$$

where $\tilde{\mathbf{V}}_\mu^{(i)}$ is the closest point projection (CPP) of the vertex $\mathbf{V}_\mu^{(i)}$ on the target mesh, and $\mathbf{n}^{\text{target mesh}}(\tilde{\mathbf{V}}_\mu^{(i)})$ is the normal vector at the CPP-point $\tilde{\mathbf{V}}_\mu^{(i)}$.

$$f_{\text{thickness}}(\mathbf{V}_j^{(i)}) = (\mathbf{V}_j^{(i)} - \mathbf{V}_j^{(i),*}) \cdot -\mathbf{n}(\mathbf{V}_j^{(i)}), \quad \frac{f_{\text{thickness}}(\mathbf{V}_j^{(i)})}{\mathbf{V}_j^{(i)}} = -\mathbf{n}(\mathbf{V}_j^{(i)}), \tag{9}$$

where $\mathbf{n}(\mathbf{V}_j^{(i)})$ is the normal vector at the vertex $\mathbf{V}_j^{(i)}$ and $\mathbf{V}_j^{(i),*}$ is the intersection of the ray originating at $\mathbf{V}_j^{(i)}$ in the direction of $-\mathbf{n}(\mathbf{V}_j^{(i)})$.

The shape gradient of the functions is needed since the node-based shape optimization will be carried out using a gradient-based algorithm called the Relaxed Gradient Projection (RGP) [1]. The RGP method computes the search direction in the design space of the position of the base mesh's vertices by considering the gradient of both the objective function and the active constraints. Since the functions 7, 8 are represented as a function of the subdivision mesh's vertices, a mapping of the gradients between the subdivision mesh and base mesh is needed. This mapping is realized using the subdivision matrix $\mathbf{S}_i$ in Eq. 5. To map search direction (in base mesh space) to shape change of vertices of the subdivision mesh, the following relation is used:

$$\delta \mathbf{V}^{(i)} = \mathbf{S}_i \cdot \delta \mathbf{V}^{(0)}, \tag{10}$$

and for mapping shape gradient from the space of vertices of the subdivision mesh to the space of vertices of the base mesh, the following relation is used:

$$\frac{\partial f}{\partial \mathbf{V}^{(i)}} = \mathbf{S}_i^T \frac{\partial f}{\partial \mathbf{V}^{(0)}}. \tag{11}$$

The relation represented in Eqs. 10 and 11 are similar to the forward- and backward-mapping operations used in the Vertex Morphing method to map between the design space and the geometry space [3,8].

## 3 Results and Discussion

Two industrial test-cases are used to demonstrate the workflow discussed in Sect. 2. They both are bracket structures developed for different purposes. Both part- and patch-segmentation are shown in Figs. 7a and 8a.

The different colors shown in Figs. 7a and 8a represent different patches, where each patch gets meshed using quad- and triangular elements to result in the coarse meshes (represented as a wireframe) shown in Fig. 7b and 8b. The

subdivision mesh is obtained after two subdivision iterations of the base mesh represented by the black wireframe. The influence of the postprocessing step detailed in Sect. 2.5 is shown in Fig. 7b and 8b, where it can clearly be seen that the subdivision meshes are getting closer to the target mesh (represented as a transparent grey-surface). However a low deviation of the subdivision mesh from the target-surface is hard to achieve due to the fact that the base mesh has low number of vertices (or degrees of freedom) to allow for richer deformation of the subdivision mesh. Nevertheless, the RGP-method finds a way to exploit the existing degrees of freedom to get as close to the target mesh as possible; see the sharp feature at the left side of the right leg in Fig. 8b.

Looking at Fig. 7c and 8c one sees clearly the benefit of using subdivision surfaces for representing the CAD-models as it gives very smooth transition between the patches, which is clearly observed by the light-reflections. In CATIA V5, we only need to import the base mesh, and the software generates instantly a corresponding subdivision surface. The subdivision surface can be modified in terms of topology and shape very easily using the readily available tools in CATIA V5.

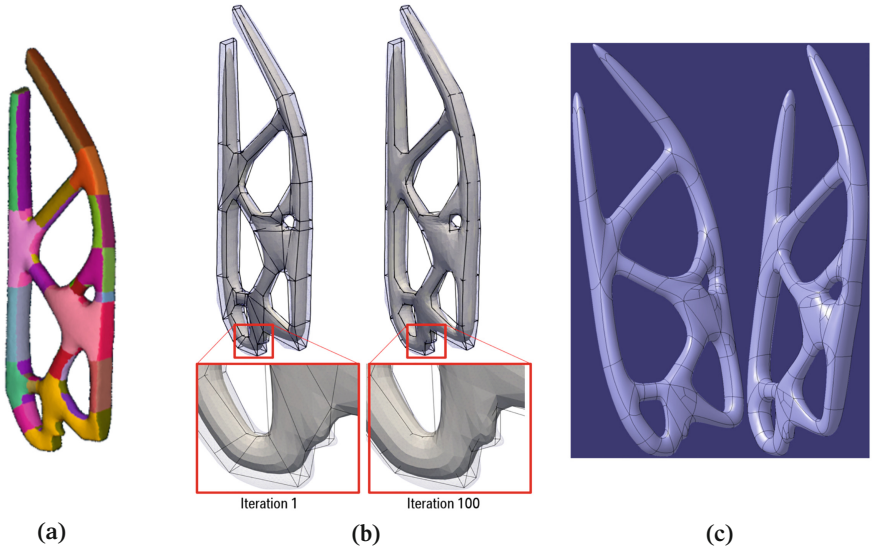


**Fig. 7.** Bracket-structure (**a**) after part- and patch-segmentation, (**b**) after undergoing shape-optimization with 100 iterations to bring its shape close to the target mesh (transparent grey) and (**c**) represented as subdivision surface in CATIA V5 [19]

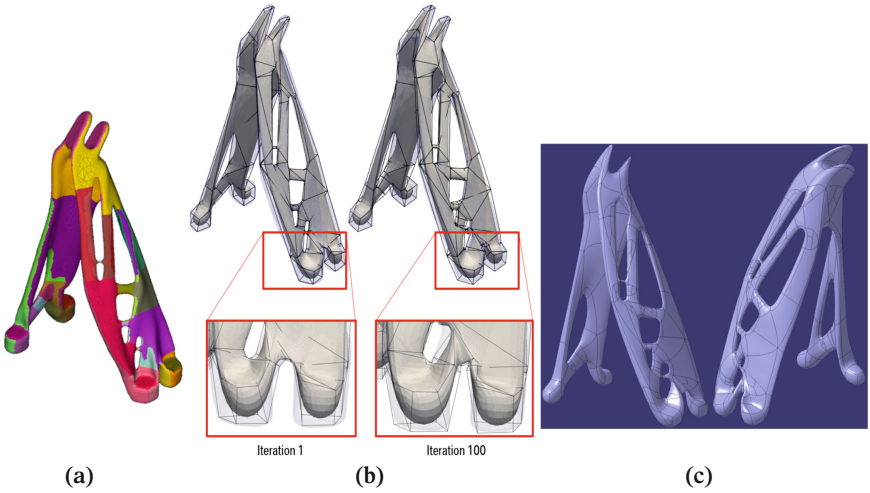


**Fig. 8.** Holder-structure (**a**) after part- and patch-segmentation, (**b**) after undergoing shape-optimization with 100 iterations to bring its shape close to the target mesh (transparent grey) and (**c**) represented as subdivision surface in CATIA V5 [19]

## 4 Conclusion

This paper presents a workflow to perform CAD reconstruction of watertight surface meshes obtained from topology optimization using subdivision surfaces. The workflow is designed such that it generates subdivision surfaces that have low number of vertices in its base mesh and are easy to modify and adjust in a commercial software. The core idea of the proposed workflow is the hierarchical segmentation of the mesh using part- and patch segmentation. To be able to model complex geometries, the polycube parameterization method is chosen to compute the patches in patch segmentation, which form the basis of the quad- and triangular elements of the base mesh. The workflow's post-processing stage, which eliminates the need for the CAD reconstruction to have a complex parameterization (with a large number of degrees of freedom in the base mesh), is an additional advantage. Instead, it uses shape optimization during post-processing to make up for the base mesh's low number of degrees of freedom. With this method, the target mesh and computed subdivision surface are intended to be tightly aligned. The workflow is proved to work on two industrial topology optimization meshes, where the subdivision surfaces are close to the target meshes and can be very easily modified and adjusted using the CATIA V5 environment.

## References

1. Antonau, I., Hojjat, M., Bletzinger, K.U.: Relaxed gradient projection algorithm for constrained node-based shape optimization. Struct. Multidiscip. Optim. **63**(04), 1633–1651 (2021)

2. Au, O., Tai, C.-L., Chu, H.-K., Cohen-Or, D., Lee, T.-Y.: Skeleton extraction by mesh contraction. ACM Trans. Graph. **27**(08) (2008)
3. Bletzinger, K.U.: A consistent frame for sensitivity filtering and the vertex assigned morphing of optimal shape. Struct. Multidiscip. Optim. **49**(01), 873–895 (2014)
4. Catmull, E., Clark, J.: Recursively generated b-spline surfaces on arbitrary topological meshes. Comput. Aided Des. **10**(6), 350–355 (1978)
5. Crane, K., de Goes, F., Desbrun, M., Schröder, P.: Digital Geometry Processing with Discrete Exterior Calculus. CM SIGGRAPH 2013 courses. SIGGRAPH '13. ACM, New York. https://www.cs.cmu.edu/~kmcrane/Projects/DDG/ (2013)
6. Denk, M., Rother, K., Paetzold, K.: Fully automated subdivision surface parametrization for topology optimized structures and frame structures using euclidean distance transformation and homotopic thinning. In: Pfingstl, S., Horoschenkoff, A., Höfer, P., Zimmermann, M. (eds.) Proceedings of the Munich Symposium on Lightweight Design 2020, pp. 18–27. Springer, Berlin (2021)
7. Doutre, P.-T., Morretton, E., Vo, T.H., Marin, P., Pourroy, F., Prudhomme, G., Vignat, F.: Comparison of Some Approaches to Define a CAD Model From Topological Optimization in Design for Additive Manufacturing, pp. 233–240. Springer International Publishing, Cham (2017)
8. Hojjat, M., Stavropoulou, E., Bletzinger, K.-U.: The vertex morphing method for node-based shape optimization. Comput. Methods Appl. Mech. Eng. **268**(01), 494–513 (2014)
9. Hu, K.: Centroidal Voronoi Tessellation with Applications in Image and Mesh Processing 7. Carnegie Mellon University, Pittsburgh (2016)
10. Hu, K., Zhang, Y.J., Liao, T.: Surface segmentation for polycube construction based on generalized centroidal voronoi tessellation. Comput. Methods Appl. Mech. Eng. **316**, 280–296 (2017). Special Issue on Isogeometric Analysis: Progress and Challenges
11. Joshi, S., Medina, J.C., Menhorn, F., Reiz, S., Rueth, B., Wannerberg, E., Yurova, A.: CAD-Integrated Topology Optimization. Technical University of Munich, Munich (2016)
12. Koguchi, A., Kikuchi, N.: A surface reconstruction algorithm for topology optimization. Eng. Comput. **22**, 1–10 (2006)
13. Kresslein, J., Haghighi, P., Park, J., Ramnath, S., Sutradhar, A., Shah, J.J.: Automated cross-sectional shape recovery of 3d branching structures from point cloud. J. Comput. Des. Eng. **5**(3), 368–378 (2018)
14. Loop, C.: Smooth Subdivision Surfaces Based on Triangles. University of Utah, Salt Lake City (1987)
15. Marinov, M., Amagliani, M., Barback, T., Flower, J., Barley, S., Furuta, S., Charrot, P., Henley, I., Santhanam, N., Finnigan, G., Meshkat, S., Hallet, J., Sapun, M., Wolski, P.: Generative design conversion to editable and watertight boundary representation. Comput. Aided Des. **115**, 194–205 (2019)
16. Mayer, J., Wartzack, S.: A concept towards automated reconstruction of topology optimized structures using medial axis skeletons. In: Pfingstl, S., Horoschenkoff, A., Höfer, P., Zimmermann, M. (eds.) *Proceedings of the Munich Symposium on Lightweight Design 2020*, pp. 28–35. Springer, Berlin (2021)
17. Park, J.M., Lee, B.C., Chae, S.W., Kwon, K.Y.: Surface reconstruction from fe mesh model. J. Comput. Des. Eng. **6**(2), 197–208 (2019)

18. Weber, C., Husung, S., Cantamessa, M., Cascini, G., Marjanovic, D., Graziosi, S. (eds.): International Conference on Engineering Design (ICED 15) 6, 235–244. ISBN: 978-1-904670-69-8. ISSN: 2220-4334. https://www.designsociety.org/publication/37838/FEATURE+BASED+INTERPRETATION+AND+RECONSTRUCTION+OF+STRUCTURAL+TOPOLOGY+OPTIMIZATION+RESULTS (2015)
19. Dassault Systèmes. Catia
20. Tarini, M., Hormann, K., Cignoni, P., Montani, C.: Polycube-Maps. *ACM SIGGRAPH 2004 Papers* (2004)
21. Yin, G., Xiao, X., Cirak, F.: Topologically robust cad model generation for structural optimisation. Comput. Methods App. Mech. Eng. **369**, 113102 (2020)
22. Zhao, H., Lei, N., Li, X., Peng, Z., Ke, X., Xianfeng, G.: Robust edge-preserving surface mesh polycube deformation. Comput. Visual Media **4**, 01 (2018)