Int j simul model 17 (2018) 2, 308-317
ISSN 1726-4529 Original scientific paper

A TWO-STEP SMOOTHING ALGORITHM FOR AN AUTOMATED PRODUCT DEVELOPMENT PROCESS

Deese, K.; Geilen, M. & Rieg, F.
Chair of Engineering Design and CAD, University of Bayreuth, Germany
E-Mail: kevin.deese@uni-bayreuth.de

Abstract
In this paper, we develop a smoothing algorithm that allows a subsequent production of components directly after topology optimisation. This is achieved by keeping features that are important for production, such as flat surfaces or straight edges.

The algorithm works in two steps. The first step is based on the marching cubes algorithm and is necessary to prepare the optimisation result for the second step. The optimisation result consists of a density distribution and needs to be transformed to a surface representation without further material or density information. The second step makes use of an implicit method for smoothing surfaces, the so-called implicit fairing.

The proposed two-step algorithm is exemplarily shown on two models. The results are compared to those received from a commercial solution to evaluate the quality of the algorithm. We show that the proposed algorithm allows a subsequent production directly after the optimisation and leads to results that are similarly good compared to those obtained by the commercial solution.
(Received in December 2017, accepted in April 2018. This paper was with the authors 2 weeks for 1 revision.)

Key Words: Smoothing, Structural Optimisation, Automated Product Development, Marching Cubes, Implicit Fairing

1. INTRODUCTION

Today, optimisation is widely spread and used in various scientific disciplines. Different optimisation methods are designed for specific optimisation problems, such as multidisciplinary optimisation or constrained mono-objective optimisation. These methods can be used in a variety of use cases, e.g. the layout optimisation of production cells in businesses or the optimisation of inventory routing to reduce emissions. [1, 2] Besides these use cases, optimisation also is used in mechanical engineering, specifically in the form of topology optimisation [3-6].

Lightweight construction is gaining in importance and is increasingly playing an important role in the modern development of new products. Great material savings and associated weight reduction lead to a more efficient use of resources, resulting in better environmental performance and lower operating costs. Topology optimisation is an ideal tool for developing lightweight structures for various applications. The problem with this method is, however, that the topology-optimised structures are very rugged due to the discretisation with finite elements. Therefore, these structures cannot be manufactured and make manual reworking necessary.

In order to avoid this reworking and to realize a direct production from the virtual model, a smoothing of the optimisation results is necessary. This leads to a high degree of automation of the process of optimisation and subsequent production. Smoothing especially is used in computer graphics for creating three-dimensional geometric models from data obtained by scanning, e.g. for computer games or medical applications [7-9].

Currently used algorithms for smoothing are, for example, the explicit Euler method [10, 11] or the λ/μ method [10, 11]. The problem with the explicit Euler method is that in large structures very small time steps have to be chosen in order to obtain an acceptable result. This quickly increases the calculation time [11, 12]. Although the λ/μ method prevents shrinkage

phenomena which frequently occur in various smoothing algorithms [11, 13, 14], the parameters λ and μ must be elaborately determined and defined by the user [11].

In addition, the smoothing changes the surface and geometry and thereby some features proposed by the optimisation (e.g. straight edges) may be lost [3, 10, 12, 15, 16]. As a result, a direct production after optimisation is not possible because under certain circumstances, smoothing could remove production-relevant features.

The proposed two-step smoothing algorithm avoids shrinkage phenomena and is easy to adjust. It still works in an acceptable time with large structures and keeps production-relevant features of the optimised structure.

2. DESCRIPTION OF THE ALGORITHM

The basis for smoothing is an optimised component. It is usually obtained as a result from topology optimisation and is available as a finite element mesh with a density distribution. That is, each element *j* has a value d_j between 0 and 1 (relative density) that indicates whether the element is included in the optimised result or not. An element with a relative density of 1 is included; an element with a relative density of 0 represents a hole and is not included. For elements with a relative density between 0 and 1, it must be decided whether they are included or not. The present work considers hexahedral meshes (see Fig. 1). One of the two steps of the proposed two-step algorithm will be specifically directed to this type of mesh. If this step is slightly modified, the proposed procedure can also be used for tetrahedral elements.

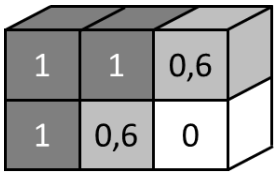

Figure 1: Exemplary density distribution of hexahedral elements.

For the first step of the two-step smoothing algorithm, the density values from the optimisation at the individual nodes are needed. In order to obtain these, for each node *i* the density values of the elements *E*(*i*) containing the node are computed. For this purpose, for each relevant element, the vectors starting from the considered node to the directly adjacent nodes are formed (see Fig. 2).

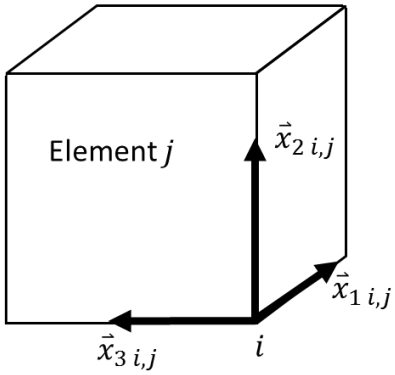

Figure 2: Vectors from the considered node to the neighbouring nodes.

Based on these vectors, the solid angle $\Omega_{i,j}$, which the element *j* occupies around the node *i*, is calculated according to Eq. (1) [17]:

$$\tan\left(\frac{1}{2}\Omega_{i,j}\right) = \frac{\langle \vec{x}_{1\,i,j}, \vec{x}_{2\,i,j}, \vec{x}_{3\,i,j} \rangle}{x_1 x_2 x_3 + (\vec{x}_{1\,i,j} \cdot \vec{x}_{2\,i,j}) x_3 + (\vec{x}_{1\,i,j} \cdot \vec{x}_{3\,i,j}) x_2 + (\vec{x}_{2\,i,j} \cdot \vec{x}_{3\,i,j}) x_1} \tag{1}$$

with $x_n = |\vec{x}_{n\,i,j}|$.

The solid angle is multiplied by the relative density d_j of the considered element. This product is normalised with the sum of all solid angles of the relevant elements formed in the same way. To get the relative node density, this process is repeated and summed for each relevant element (see Eq. (2)).

$$\rho_i = \frac{1}{\sum_{k \in E(i)} \Omega_{i,k}} \sum_{j \in E(i)} d_j \Omega_{i,j} \tag{2}$$

After the preparation of the model, the actual first step of smoothing is performed. This consists of a slight modification of the so-called marching cubes algorithm [18]. The node densities are used to determine which node should be part of the optimised structure and which should be omitted. For this a predetermined limit is used. Any node with a density below the boundary is removed by the marching cubes algorithm by placing triangles in the appropriate place through the hexahedron (see Fig. 3).

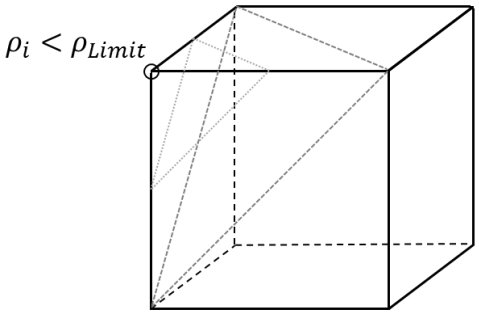

Figure 3: Classic marching cubes (grey dotted) and modified marching cubes (grey dashed).

The first step of the smoothing algorithm is to generate a component with a clearly defined surface from the density distribution within the design volume. The output of the first step is no longer an FE mesh of hexahedrons, but the surface of the part as STL.

The second step is the so-called "implicit fairing" according to Desbrun [10]. This approach is based on Laplace smoothing and is solved with implicit integration, which makes the time steps for the solution much larger compared to the explicit Euler method. The underlying equation is as follows [10, 19]:

$$(I - \lambda dt L) X^{n+1} = X^n \tag{3}$$

Here, X is the mesh (surface) that contains the nodes x_i. λdt is the time step given by the user, n is the current iteration, and L is the umbrella operator, which is calculated as follows [12, 20]:

$$L(x_i) = \frac{1}{|N_1(i)|} \sum_{j \in N_1(i)} x_j - x_i \tag{4}$$

$N_1(i)$ is the set of neighbours of node x_i, i.e. the nodes directly connected to x_i in the finite element mesh.

If the mesh is too coarse, smoothing removes any relevant features. To counter this, the surface is refined after a defined number of steps. Fig. 4 shows the procedure exemplary in the two-dimensional case.

After successful smoothing, the part is exported as STL.

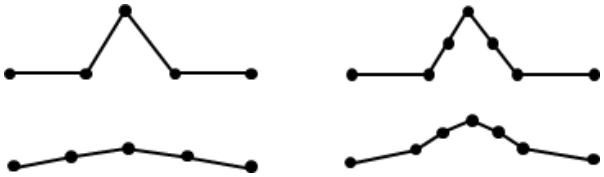

Figure 4: Refinement to preserve geometric features in the 2D case.

3. MODELS

The two-step smoothing algorithm is demonstrated on two models in order to assess its functionality. Input for the smoothing algorithm is the mesh with the corresponding density distribution of the optimisation results. For this study a proprietary format converted from the results is used. Fig. 5 shows the first model.

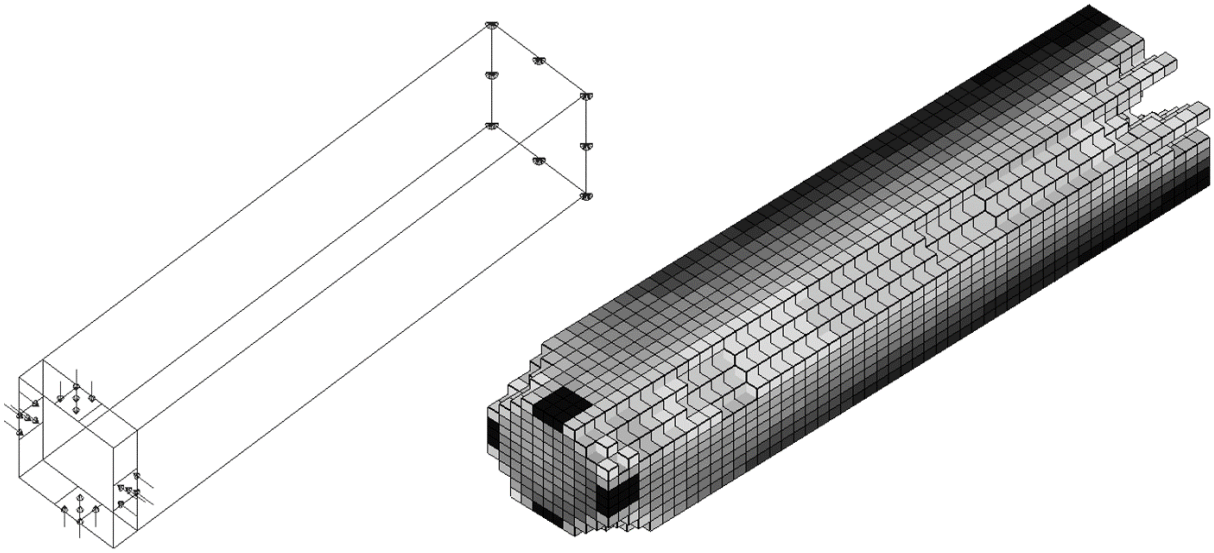

Figure 5: Model 1 – Beam model with pressure load (left) and the optimisation result (right).

The model is a beam of dimension 100×20×20 mm, which is fixed in one end and loaded with pressure at the other end in the middle of each side surface on an area of size 6×6 mm (Fig. 5, left). It consists of 6171 hexahedron elements. Optimisation was performed with Tosca [21] using the SIMP method [5, 6]. The result is shown on the right. Elements with a relative density below 0.25 are not displayed, which are 1823 of the original elements.

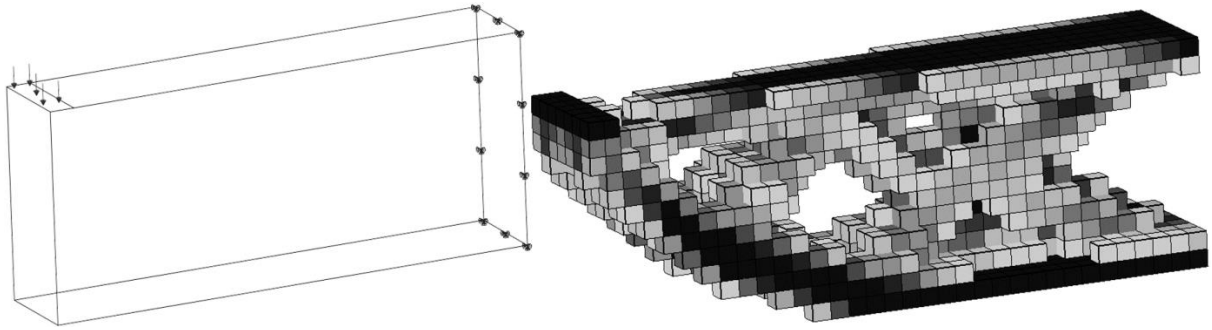

Figure 6: Model 2 – Loaded beam model (left) and the optimisation result (right).

The second model (see Fig. 6, left) is a beam of dimension 100×20×40 mm. The fixture was applied to one end face as in the first model, while a force was applied to the opposite upper edge on a width of 6. This model consists of 3094 hexahedron elements. The optimisation again was carried out with Tosca using the SIMP method (Fig. 6, right). Not displayed are elements with a relative density below 0.15, which are 1919 of the original elements.

4. METHOD

In order to evaluate the two-step algorithm, the smoothing results are compared to the results of the smoothing step available in Tosca.

The actual target is to achieve a result that keeps relevant features and therefore resembles the topology optimisation result. Assuming that elements of the optimised result with a relative density *d* should also be included in the smoothed result for this proportion, a statement can be made as to how well the smoothed result maps the optimised component. For this purpose, the smoothing result is superimposed on the optimised component and divided into four areas:

- volume to be included after optimisation and included after smoothing (inside / inside),
- volume to be included but is not included (inside / outside),
- volume that should not be included and is not included (outside / outside),
- volume that should not be included but is included (outside / inside).

Ideally, the "inside / outside" and "outside / inside" parts tend towards zero. Smoothing would not remove any needed volume nor add unnecessary volume.

This method gives a good first understanding of how well the algorithm works. Nevertheless, if two different smoothed results are compared this way, they can still both show similar good numbers. The reason for the differences in the smoothed results, although the volume mapping is virtually identical, is that the comparison of the optimised and the smoothed structure is based solely on the relative density of the element and the volume of the smoothed component in the region of the element. In which way the volume is arranged is not considered in the comparison. Fig. 7 illustrates this situation.

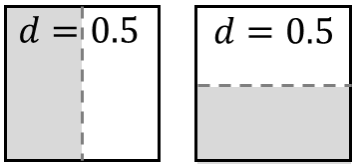

Figure 7: Two possible volume representations of an element with a given relative density.

Here, both elements have a density of 0.5, i.e. they should only be filled halfway in the smoothed result. The grey part resembles the actual way of how the volume may be distributed.

For a second way of evaluation, the smoothing results are simply compared optically to the optimisation results and the smoothing results of Tosca. The goal of this comparison is to see if the proposed algorithm can preserve optimisation features, such as flat surfaces or straight edges.

Besides these methods for comparing the result of the proposed two-step smoothing algorithm, additional comparisons are made regarding the computational effort, i.e. the total time the smoothing takes with the two-step algorithm and Tosca. For this, the smoothing is conducted ten times, of which the average time effort is calculated.

5. RESULTS

Fig. 8 shows the respective proportions both after smoothing with Tosca and with the two-step smoothing algorithm. Both smoothing algorithms lead to similar good results and almost eliminate the “inside / outside” and “outside / inside” parts.

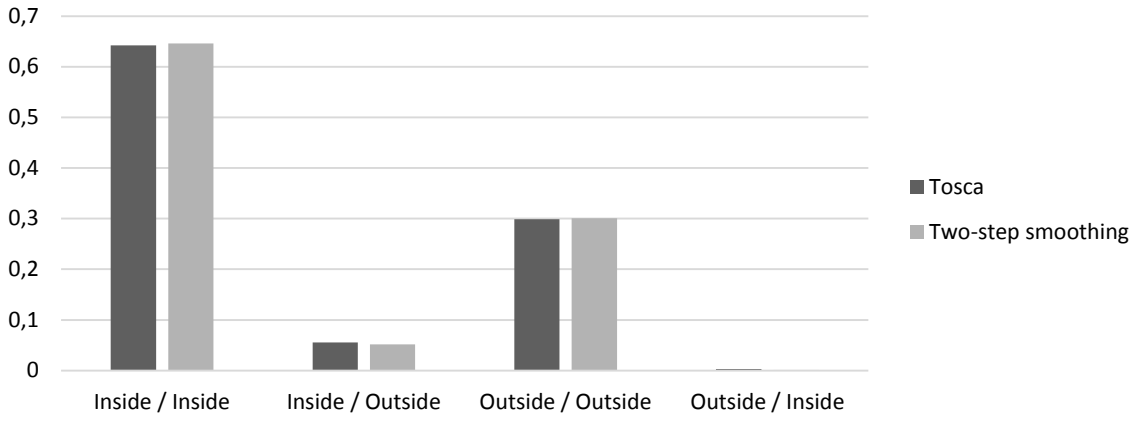

Figure 8: Volume image of the optimised result by smoothing on model 1.

The optical results of smoothing model 1 with the two-step algorithm are shown in Fig. 9. The intermediate result after the first step (modified marching cubes) can be seen on the left side. The final result is shown on the right. For comparison, the smoothing result from Tosca is shown in Fig. 10. The results clearly differ and resemble the topology optimisation result in different ways. The two-step algorithm tends to create straight edges and flat surfaces and strictly keep optimised features.

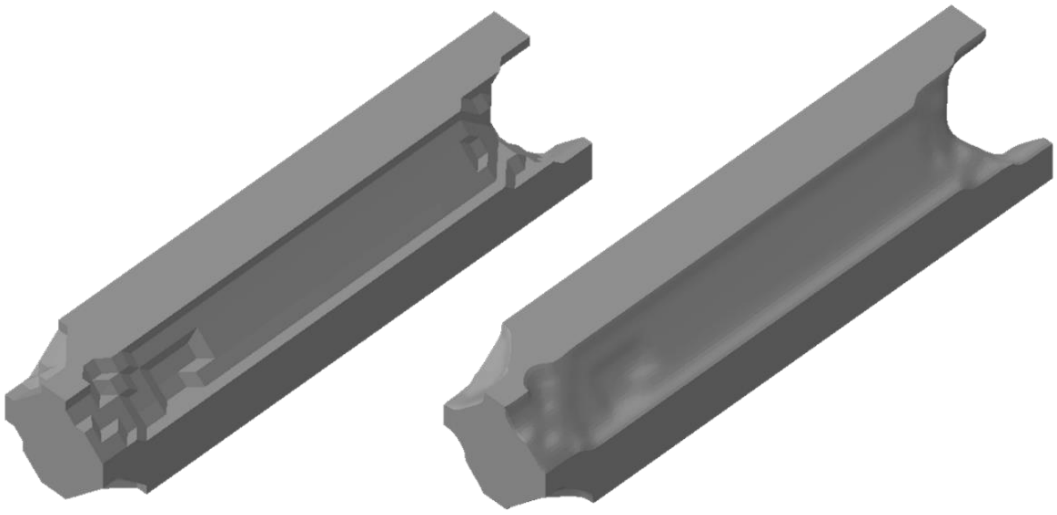

Figure 9: Model 1 smoothed with two-step algorithm, first step (left) and second step (right).

One specific detail besides the flat surfaces is the small pocket on the front. Fig. 11 shows this pocket after smoothing with the two compared algorithms. With the two-step algorithm, the pocket is clearly visible and consists of flat surfaces. The result from Tosca does not show this pocket.

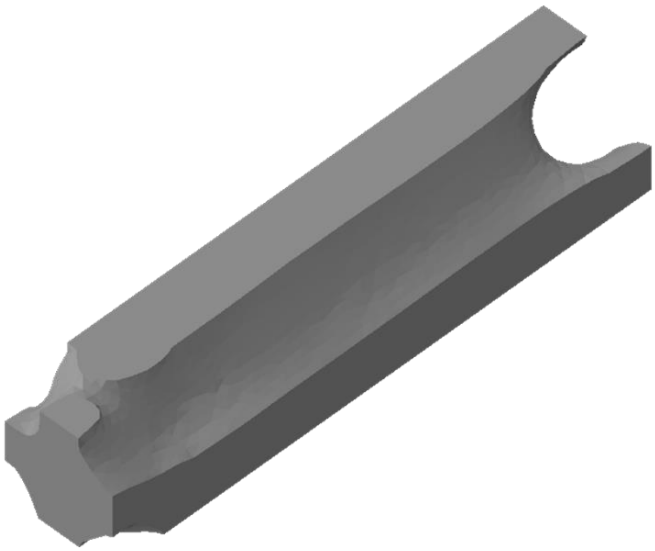

Figure 10: Model 1 smoothed with Tosca.

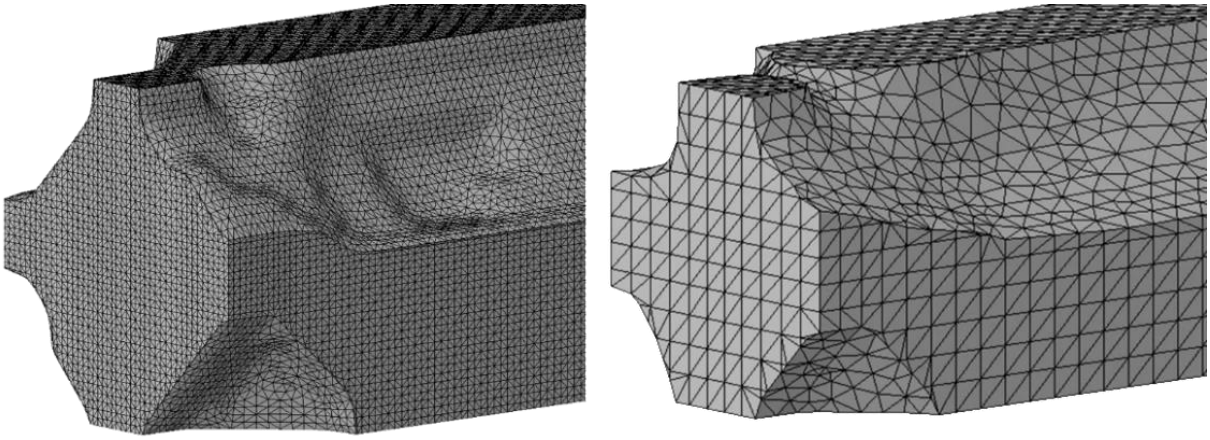

Figure 11: Front pocket in model 1 after smoothing with two-step algorithm (left) and Tosca (right).

The results of model 2 are shown below. As with model 1, both smoothing algorithms almost eliminate the undesired volume parts and show similar results.

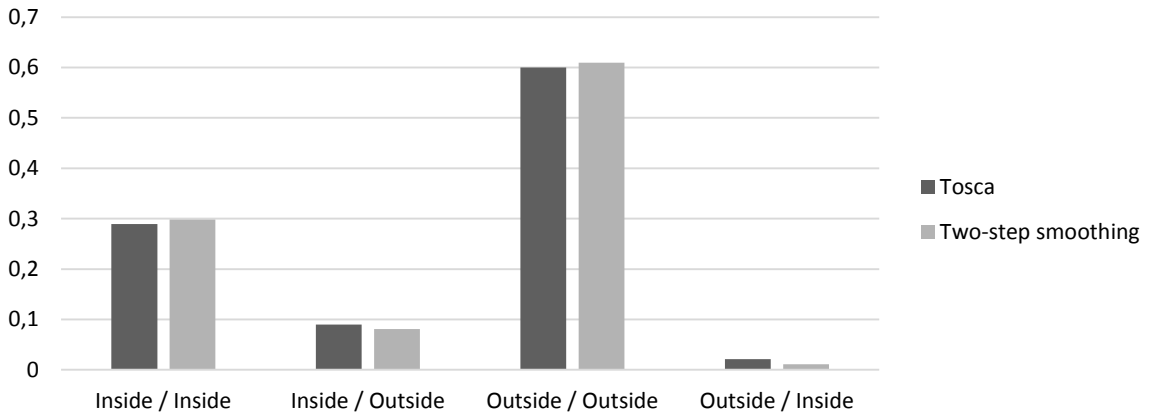

Figure 12: Volume image of the optimised result by smoothing on model 2.

Figs. 13 and 14 show the optical comparison of the smoothing results of the two-step algorithm and Tosca. Both results generally go into the same direction but still look different. As with the previous model, the two-step algorithm leads to flat surfaces and straight edges.

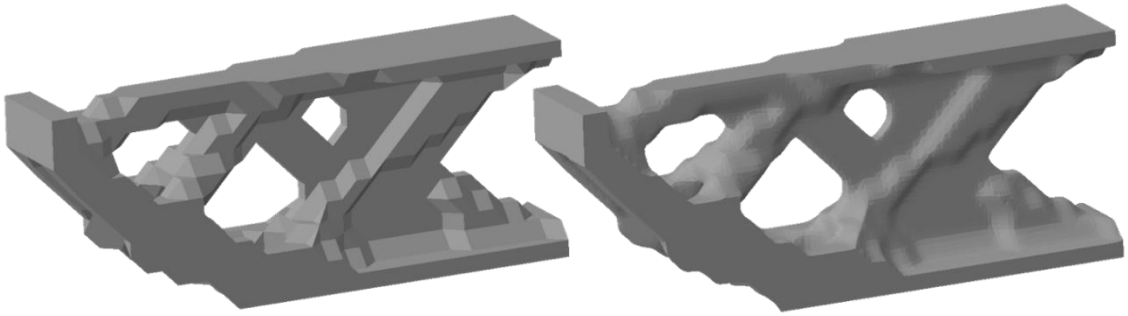

Figure 13: Model 2 smoothed with two-step algorithm, first step (left) and second step (right).

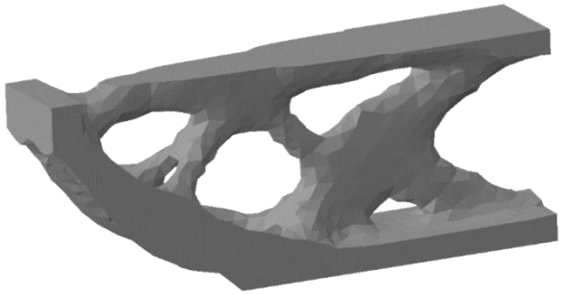

Figure 14: Model 2 smoothed with Tosca.

Fig. 15 shows the rear part of model 2. Smoothing with Tosca leads to larger holes and to holes, where the two-step algorithm did not introduce holes. Additionally, the two-step algorithm leads to flat surfaces and does not create a bloated structure.

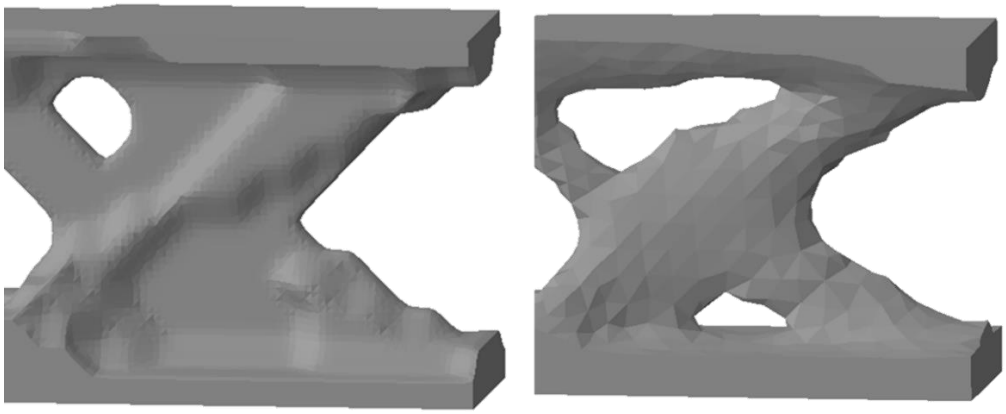

Figure 15: Rear part of model 2 after smoothing with two-step algorithm (left) and Tosca (right).

Regarding the time consumption of the two-step smoothing algorithm and Tosca, Fig. 16 shows that the two-step algorithm is slower.

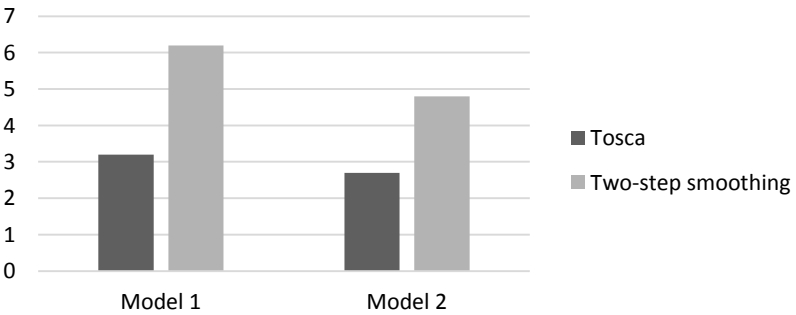

Figure 16: Average time consumption for smoothing.

For the first model, Tosca takes an average of 3.2 seconds while the two-step algorithm takes an average of 6.2 seconds. For the second model, due to the reduced number of elements, both algorithms are faster. Tosca takes an average of 2.7 seconds while the two-step algorithm takes an average of 4.8 seconds. It has to be mentioned that for Tosca the default settings of five iterations was used. For the two-step algorithm a setting of ten iterations for the second step was used.

6. CONCLUSION

The illustrated two-step smoothing algorithm is an implicit method. This means that a linear system of equations must be solved for each iteration. What initially appears to be a disadvantage has a positive effect on large systems compared to an explicit method, since a large time step can be chosen. On the other hand, if the method is explicit, the time step must be small, which increases the calculation time [10].

The presented algorithm is easy to handle since only two parameters are needed for the configuration: the size of the time step λdt (see Eq. (3)) and the number of iterations to be performed. In addition, the choice of parameters is not very complicated because the algorithm is robust to changes in the parameters.

As can be seen in the results, features of the optimised component are well preserved. In particular, straight edges and flat surfaces are well represented and thus require little or no rework. This is especially good compared to the results of smoothing with Tosca. If the optimisation has been carried out in compliance with production restrictions, the resulting features are retained, whereby the smoothed result is also suitable for production. The duration of the smoothing with the proposed two-step algorithm is higher than with Tosca but still short. In comparison to the overall time needed for topology optimisation, the smoothing plays only a little role.

The illustrated two-step smoothing algorithm thus represents a possibility to make optimised results easier to interpret for the human eye and, moreover, to pass them directly to the production. This closes the gap between optimisation and production and enables a fully automated product development process.

ACKNOWLEDGEMENTS

The European Union supported this research via the European Regional Development Fund (ERDF).

REFERENCES

[1] Zupan, H.; Herakovic, N.; Zerovnik, J.; Berlec, T. (2017). Layout optimization of a production cell, *International Journal of Simulation Modelling*, Vol. 16, No. 4, 603-616, doi:10.2507/IJSIMM16(4)4.396

[2] Balamurugan, T.; Karunamoorthy, L.; Arunkumar, N.; Santhosh, D. (2018). Optimization of inventory routing problem to minimize carbon dioxide emission, *International Journal of Simulation Modelling*, Vol. 17, No. 1, 42-54, doi:10.2507/IJSIMM17(1)410

[3] Fiebig, S.; Sellschopp, J.; Manz, H.; Vietor, T.; Axmann, J. K.; Schumacher, A. (2015). Future challenges for topology optimization for the usage in automotive lightweight design technologies, *Proceedings of the 11th World Congress on Structural and Multidisciplinary Optimisation*, Sydney, 8 pages

[4] Ramadani, R.; Belsak, A.; Kegl, M.; Predan, J.; Pehan, S. (2018). Topology optimization based design of lightweight and low vibration gear bodies, *International Journal of Simulation Modelling*, Vol. 17, No. 1, 92-104, doi:10.2507/IJSIMM17(1)419
[5] Bendsøe, M. P. (1989). Optimal shape design as a material distribution problem, *Structural Optimization*, Vol. 1, No. 4, 193-202, doi:10.1007/BF01650949
[6] Bendsøe, M. P.; Sigmund, O. (1999). Material interpolation schemes in topology optimization, *Archive of Applied Mechanics*, Vol. 69, No. 9-10, 635-654, doi:10.1007/s004190050248
[7] Qin, X. J.; Duan, Z. J.; Zheng, H. B.; Tang, Y. (2017). Efficient smoothness-preserving fusion modelling method for mesh models, *International Journal of Simulation Modelling*, Vol. 16, No. 3, 527-540, doi:10.2507/IJSIMM16(3)CO14
[8] Spradley, J. P.; Pampush, J. D.; Morse, P. E.; Kay, R. F. (2017). Smooth operator: The effects of different 3D mesh retriangulation protocols on the computation of Dirichlet normal energy, *American Journal of Physical Anthropology*, Vol. 163, No. 1, 94-109, doi:10.1002/ajpa.23188
[9] Miura, K. T.; Suzuki, S.; Gobithaasan, R. U.; Salvi, P.; Usuki, S. (2017). Log-aesthetic flow governed by heat conduction equations, *Computer-Aided Design and Applications*, Vol. 14, No. 2, 227-233, doi:10.1080/16864360.2016.1223436
[10] Desbrun, M.; Meyer, M.; Schröder, P.; Barr, A. H. (1999). Implicit fairing of irregular meshes using diffusion and curvature flow, *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, 317-324, doi:10.1145/311535.311576
[11] Taubin, G. (1995). A signal processing approach to fair surface design, *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, 351-358, doi:10.1145/218380.218473
[12] Kobbelt, L.; Campagna, S.; Vorsatz, J.; Seidel, H.-P. (1998). Interactive multi-resolution modeling on arbitrary meshes, *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, 105-114, doi:10.1145/280814.280831
[13] Oliensis, J. (1993). Local reproducible smoothing without shrinkage, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 3, 307-312, doi:10.1109/34.204914
[14] Wang, J.; Yu, Z. (2009). A novel method for surface mesh smoothing: applications in biomedical modeling, *Proceedings of the 18th International Meshing Roundtable*, 195-210, doi:10.1007/978-3-642-04319-2_12
[15] Zhao, H.; Xu, G. (2006). Triangular surface mesh fairing via Gaussian curvature flow, *Journal of Computational and Applied Mathematics*, Vol. 195, No. 1-2, 300-311, doi:10.1016/j.cam.2005.03.094
[16] Jones, T. R.; Durand, F.; Desbrun, M. (2003). Non-iterative, feature-preserving mesh smoothing, *ACM Transactions on Graphics*, Vol. 22, No. 3, 943-949, doi:10.1145/882262.882367
[17] Van Oosterom, A.; Strackee, J. (1983). The solid angle of a plane triangle, *IEEE Transactions on Biomedical Engineering*, Vol. BME-30, No. 2, 125-126, doi:10.1109/TBME.1983.325207
[18] Lorensen, W. E.; Cline, H. E. (1987). Marching cubes: A high resolution 3D surface construction algorithm, *ACM SIGGRAPH Computer Graphics*, Vol. 21, No. 4, 163-169, doi:10.1145/37401.37422
[19] Taubin, G. (1995). Curve and surface smoothing without shrinkage, *Proceedings of IEEE – 5th International Conference on Computer Vision*, 852-857, doi:10.1109/ICCV.1995.466848
[20] Kobbelt, L. (1994). *Iterative Erzeugung glatter Interpolanten*, Dissertation, Universität Karlsruhe
[21] Dassault Systèmes. SIMULIA – TOSCA, from *https://www.3ds.com/de/produkte-und-services/simulia/produkte/tosca/*, accessed on 20-12-2017