INTERNATIONAL DESIGN CONFERENCE – DESIGN 2020
https://doi.org/10.1017/dsd.2020.162

RAPID CREATION OF VEHICLE LINE-UPS BY EIGENSPACE PROJECTIONS FOR STYLE TRANSFER

T. Friedrich ✉, S. Schmitt and S. Menzel

Honda Research Institute Europe GmbH, Germany
✉ timo.friedrich@honda-ri.de

Abstract

In product development, an automated generation of shape variations enables a rapid assessment of potentially appealing design directions. We present a framework for computing a product line-up of automotive body shapes based on spectral methods for mesh processing. We calculate the eigenspace projections of 3D vehicle meshes and identify the relevant style as well as content components based on the eigenvalues. The style of a model is then transferred to arbitrary prototype content car shapes, which allows for a rapid portfolio generation of various car types with minimal user interaction.

Keywords: 3D modelling, conceptual design, product families, style transfer, computational design methods

1. Introduction

The creation of visually appealing designs, which likewise provide an optimal technical performance under given environment conditions, is a core ingredient in any product development process. For some design tasks like automotive development, it is in addition relevant to reflect common features within a set of design instances to increase the recognition factor among a design family or product portfolio. In consequence, these features allow consumers an easy identification of common style and unity, or even branding. However, by working on a novel single design, e.g. a vehicle in a virtual design process, the potential to imagine how style elements of this novel design would transfer to other designs of a family may be limited without actually modelling it in a time-consuming manual process. Hence, in the present paper we propose a framework for computing a product line-up of 3D automotive body shapes, which relies on a transfer of style features on a set of prototypes in a semi-automated fashion. For this, we utilize the spectral eigendecomposition of the Laplace-Beltrami operator discretized for the triangulated 3D surface meshes of car shapes and propose a scheme for 3D style transfer on meshes.

Automated style transfer for 2D images has recently been developed based on convolutional neural networks due to the availability of large image repositories and increased computational (GPU) power. Neural style transfer (Gatys et al., 2016) has made a strong impact on the generation of artistic data as well as the generation of novel shapes and designs. Gatys et al. (2016) proposed a method to automatically divide image data into style and content information and utilized these information to compose novel artistic images comprising the content, e.g. a city skyline, of one image and the style, e.g. the style of famous painters, of a second image. For a more detailed review on available state-of-the-art methods, see Jing et al. (2017).

First attempts to realize neural style transfer for 3D geometries by geometric deep learning concluded in several challenges, e.g. on the choice of a reasonable base representation (voxels or point clouds) or on the layout of the deep neural net architecture. Friedrich et al. (2018) have proposed a framework for 3D neural style transfer based on three-dimensional voxel data for simplified car shapes. An improvement of the framework by a standardized Gram matrix based loss function for style has been introduced by Friedrich and Menzel (2019). Yang et al. (2019) also utilize voxels and point cloud representations for 3D shape transformation between two objects based on geometric deep learning. An alternative hybrid approach combines styles from images and content from 3D meshes resulting in texturized and deformed 3D models (Kato et al., 2018). In parallel to neural style transfer, many researches have created generative neural network setups like autoencoders in order to generate novel three-dimensional shapes by mixing latent features of different models. While Brock et al. (2016) utilized the voxel format, Achlioptas et al. (2018) have achieved similar results with point clouds. Both approaches yield promising results with the downside of geometric data formats, which are typically not used in virtual design. Umetani (2017) on the other hand has focused on autoencoder based shape generation on 3D mesh data by introducing an algorithm for the generation of compatible, homogeneous three-dimensional meshes of cars with a shared topology that enables the direct application of neural network based methods.
In contrast to data-driven neural style transfer approaches like geometric deep learning, the field of geometry processing offers a variety of methods for directly operating on 3D shapes and 3D meshes. Traditional mesh processing was based on mesh operators like Laplacian matrices and spectral methods (Meyer et al., 2003; Sorkine, 2005; Reuter et al., 2006; Zhang et al., 2007). In 2004, Sorkine et al. (2004) proposed to extract high-frequency details from one geometry and transfer them to another shape based on Laplacian processing which provides an excellent starting point for an automated style transfer between 3D vehicle shapes. In this paper, we transfer the general concept of style and content from neural style transfer to classic mesh processing with the help of recent three-dimensional datasets originating from generative neural network setups. We utilize the spectral representation of a 3D mesh object and extract features that can be associated with either style or content. Since we rely on a homogeneous mesh dataset, we are able to mix style and content from different 3D models and create novel shapes. This allows us to generate complete automotive product portfolios in a semi-automatic fashion in a matter of minutes. The generated portfolio shapes are intended to be an inspiration for the designers during the concept development phase for new automotive models. Rather than fully functional 3D meshes, we aim to combine visual features associated with style and content in order to obtain innovative design inspiration prototypes with the proposed design exploration tool. Given compatible datasets, the approach is not limited to automotive development only, but universally applicable on all kinds of object classes given the availability of data sets and mesh constraints.
The paper is structured as follows. In section 2, we introduce the mathematical fundamentals for spectral mesh processing which forms the basis for our proposed framework. In section 3, we detail the connection to style transfer, describe our dataset and formulate our approach, which we apply to a standard benchmark design and representative geometries from the automotive domain. Finally, we conclude our work in section 4 and give ideas for potential improvements.

2. Mesh processing

In the following, we review the triangular mesh definition and its mesh Laplacian operators, which we will use to perform the mesh processing. Our focus lies on mesh compression and mesh smoothing based on spectral mesh methods. For a comprehensive overview of all applications based on mesh Laplacians, we refer to Zhang et al. (2007). Furthermore, Meyer et al. (2003), Reuter et al. (2006) and Sorkine et al. (2004, 2005) give further inside on the theoretical principles of mesh Laplacians and their applications.

2.1. Mesh notation

We use the notation $\mathcal{M} = (\mathcal{E}, V)$ for a geometric, triangular, three-dimensional mesh, where V describes the vertices and $\mathcal{E}$ the edges of the mesh. The edge list $\mathcal{E}$ includes the topological

https://doi.org/10.1017/dsd.2020.162 Published online by Cambridge University Press

information in form of a list of index pairs (i, j) which indicates an edge between vertex i and vertex j. The vertices are organized in a matrix $V \in \mathbb{R}^{n\times 3}$ where n denotes the total number of vertices and each row encodes the three dimensional coordinate vector of the vertex, $v_i = [x_i, y_i, z_i]$ with $i \in [1, n]$.

2.2. Mesh Laplacians

Mesh Laplacians are linear operators which can be applied to discrete meshes and which are derived from the Laplace-Beltrami operator defined on locally Riemannian Manifolds (Sorkine, 2005).

The Laplace-Beltrami operator applied to a function f on a continuous 2D Riemann manifold is defined as

$$\Delta(f) := div\big(grad(f)\big) = \nabla \cdot \nabla f \tag{1}$$

Equation (1) can be discretized for the application to discrete triangulated mesh $\mathcal{M}$ in three dimensions and is generally written as

$$Lf = b_i^{-1} \sum_{j\in N(i)} \omega_{ij}\big(f_i - f_j\big) \tag{2}$$

The operator acts on each vertex v_i taking only the local neighbourhood $N(i)$ of the vertex into account. Usually, only first order neighbours are considered. The edge weight ω_{ij} is a positive scalar representing the interconnection of vertices i and j, and the factor b_i^{-1} represents a suitable normalization which might be different for each vertex i. In the general form, f is an abritrary function defined on each vertex.

If we take the actual mesh's vertex coordinates v_i as function value f, we obtain a vector δ_i for each vertex. This so-called differential coordinate δ_i (Sorkine, 2005) is normal to the local mesh surface and its norm encodes the curvature of the surface. The differential coordinate is a local mesh descriptor used for mesh processing and manipulation (Sorkine, 2005). Most relevant for our considerations are the specifications of the general Mesh Laplacian in the combinatorial or in the geometric form.

2.3. Combinatorial Laplacian

The combinatorial, or also known as graph Laplacian, is solely derived from the graph information encoded in the edge list $\mathcal{E}$. The matrix representation of the graph Laplacian is given in its normalized form by

$$L_{comb} = 1 - \mathrm{D}^{-1}A, \tag{3}$$

where the adjacency matrix A and the diagonal matrix D are defined as follows.

$$A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in \mathcal{E}, \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$$D_{ij} = \begin{cases} d_i = |N(i)| & \text{if } i = j, \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

If we combine Equations (2) and (3), we obtain the formula

$$\delta_i^{graph} = \frac{1}{d_i} \sum_{j\in N(i)} \big(\mathrm{v}_i - \mathrm{v}_j\big) \tag{6}$$

which gives us the differential coordinate of each vertex with the absolute norm approximating the curvature of the mesh at the vertex v_i. For low-resolution meshes, the curvature approximation as given by $|\delta_i^{graph}|$ is not very accurate (Figure 1, left). Some vertices, which should have rather high curvature values, since they are pointy edged, are assigned a rather low value, whereas neighbouring vertices with expected low curvature are assigned a rather high curvature value.

https://doi.org/10.1017/dsd.2020.162 Published online by Cambridge University Press

Figure 1. Visualization of the curvature approximation at each vertex given by the norm of the differential coordinates calculated with the graph Laplacian (left) and geometric Laplacian (right). Purple colours imply low curvature, while red and yellow imply larger curvature. Curvatures are calculated for the vertices and interpolated on the mesh triangles.

2.4. Geometric Laplacian

In order to achieve better local approximation capabilities, Pinkall and Polthier (1993) and Meyer (2003) proposed to use geometric relations of the edges and vertices involved in Equation (1). Effectively, ω_{ij} and b_i^{-1} become a function of $\mathcal{E}$ and V in contrast to the combinatorial Laplacian. The geometric information to be included is provided by the angles α_{ij} and β_{ij} between edges opposing the edge connecting v_i and v_j in the mesh triangles adjacent to v_i and v_j. By using these angles to define so-called cotangent weights, the geometric Laplacian is given as follows:

$$\left(L_{geom}f\right)_i = \frac{1}{|\Omega_i|}\sum_{j\in N(i)}\frac{1}{2}\left(\cot\alpha_{ij} + \cot\beta_{ij}\right)\left(f_i - f_j\right). \tag{7}$$

Hereby, Ω_i denotes the size of the Voronoi cell of the vertex v_i. We are now able to calculate the differential coordinates δ_i^{geom} and the curvature in the same way as done in section 2.3. Apparently, the approximation of the curvature is more in line with the expected curvature (Figure 1, right).

2.5. Mesh compression

Differential coordinates and the according Laplacians are used in a variety of applications for mesh and graph processing. For a comprehensive overview, we refer to the overviews given by Sorkine (2005) and Zhang et al. (2007). The application described in this work relies on the mesh processing and compression based on the eigenvalue decomposition of the Laplacians. The eigenvectors of the Laplacian matrix L are organized into columns of an orthogonal matrix

$$E = \{\mathrm{e}_1, \cdots, e_n\} \in \mathbb{R}^{n\times n} \text{ where } E^T E = E\,E^T = 1 \tag{8}$$

with the corresponding sorted eigenvalues

$$0 = \lambda_1 < \lambda_2 \leq \lambda_3 \leq \cdots \leq \lambda_n\,. \tag{9}$$

In order to project the vertices v_i into the mesh's spectral domain and obtain the spectral coefficients c_i, the matrix of eigenvectors has to be multiplied with the vertices,

$$C = E^T V \text{ with } V \in \mathbb{R}^{n\times 3}\,. \tag{10}$$

The original vertices are consequently reconstructed with the inverse operation,

$$\mathrm{V} = E\mathrm{C}. \tag{11}$$

It turns out that the coefficients ordered according to (9) relate to low and high frequency mesh properties in a similar way as Fourier coefficients (Sorkine, 2005; Zhang et al., 2007). Thus, we can use these coefficients to perform mesh compression in form of low-pass filtering. In Figure 2, we show the compressed Stanford bunny in a 242 vertices low polygon version based on the mesh operators L_{comb} and L_{geom}. We took only the smallest 24 eigenvalues and corresponding spectral coefficients of Equation (10) and reconstructed the shape according to Equation (11) with the truncated versions of E and C. The reconstructions are far from being perfect with both approaches, but in general, the geometric approach

https://doi.org/10.1017/dsd.2020.162 Published online by Cambridge University Press

produces smoother results with less artefacts and reproduces the overall shape and features most similar to the original. Hence, we base all following transformations on the geometric Laplacian L_{geom}.

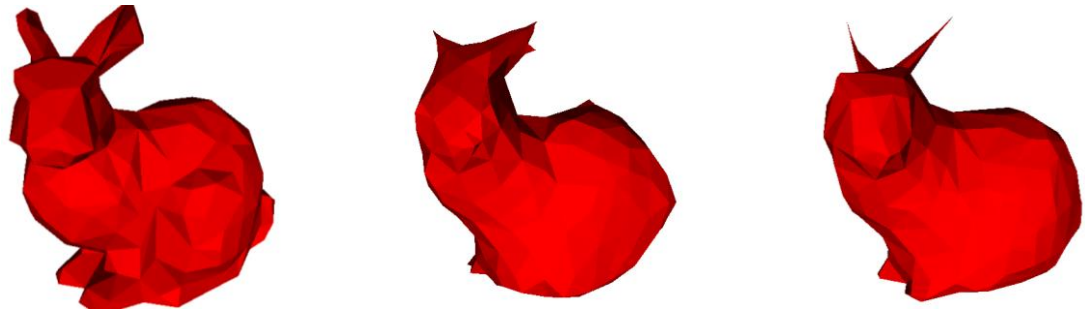

Figure 2. 24 Eigenvalue based, compressed Stanford Bunny with 242 vertices (left: original, middle: Reconstruction based on combinatorial Laplacian, right: Reconstruction based on geometric Laplacian)

It is common knowledge that the lowest non-zero eigenvalue λ_2 encodes the centre of mass of the mesh. It takes a minimum of three non-zero spectral coefficients to generate three-dimensional structures with the inverse transformation according to Equation (11). Figure 3 illustrates various reconstructions of shapes with truncated sets of spectral coefficients from the geometric Laplacian. Taking the coefficients related to the first four eigenvalues already produces a three-dimensional shape (top left). Increasing the number of included coefficients leads to a steady improvement of the reconstructed shape, where higher order eigenvectors introduce more high-frequency details. Including only 100 coefficients, which is less than half the total number of eigenvectors, produces a reasonable approximation of the original mesh. The overall structure of the mesh is quickly lost, once the low order coefficients are omitted (Figure 3, bottom right).

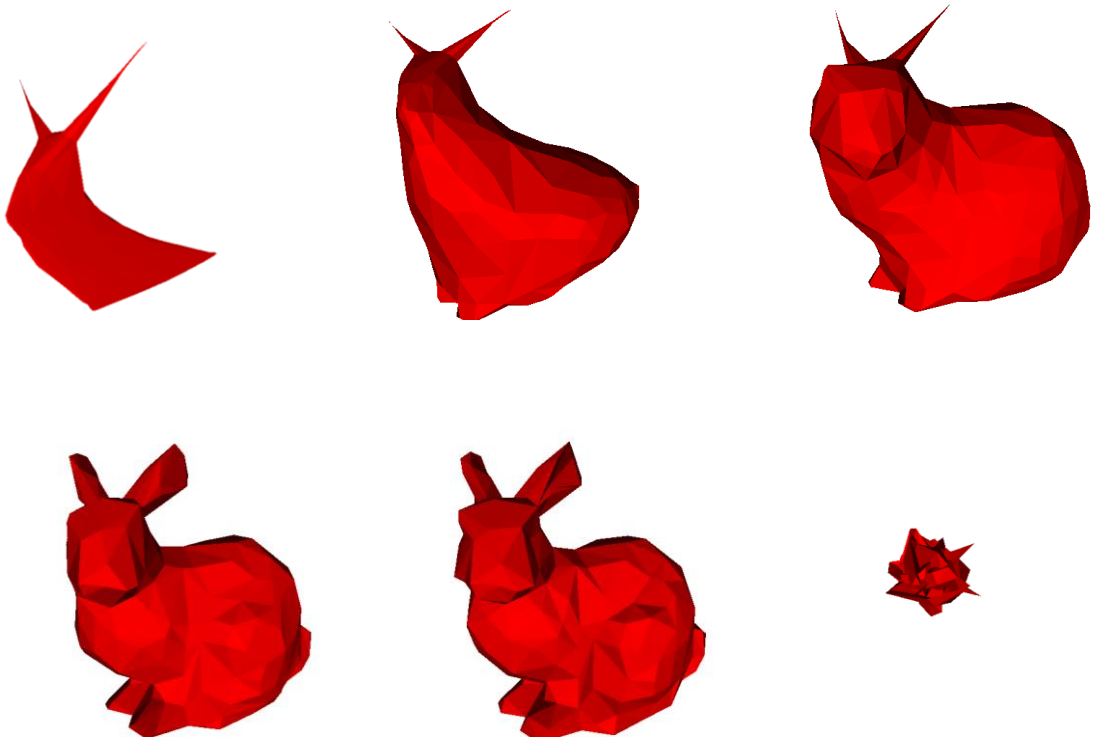

Figure 3. Mesh reconstructions based on different numbers of eigenvectors according to Equation (11); top row: $\lambda_1 \cdots \lambda_4$, $\lambda_1 \cdots \lambda_6$, $\lambda_1 \cdots \lambda_{25}$; bottom row: $\lambda_1 \cdots \lambda_{100}$, , $\lambda_1 \cdots \lambda_{n=242}$, , $\lambda_9 \cdots \lambda_{=n=242}$ (strong zoom)

https://doi.org/10.1017/dsd.2020.162 Published online by Cambridge University Press

In addition, we compute and viusualize the low and high frequency relation of the eigenvectors by overlaying the components $e_{ik};\ k \in [1, n]$ of the individual eigenvectors onto the mesh geometry. The lower order eigenvectors represent smooth functions on the complete mesh, whereas increasing oscillations or strong localizations are visible for higher order eigenvectors (Figure 4). For example, the lowest eigenvalues influence only very localized areas of the mesh.

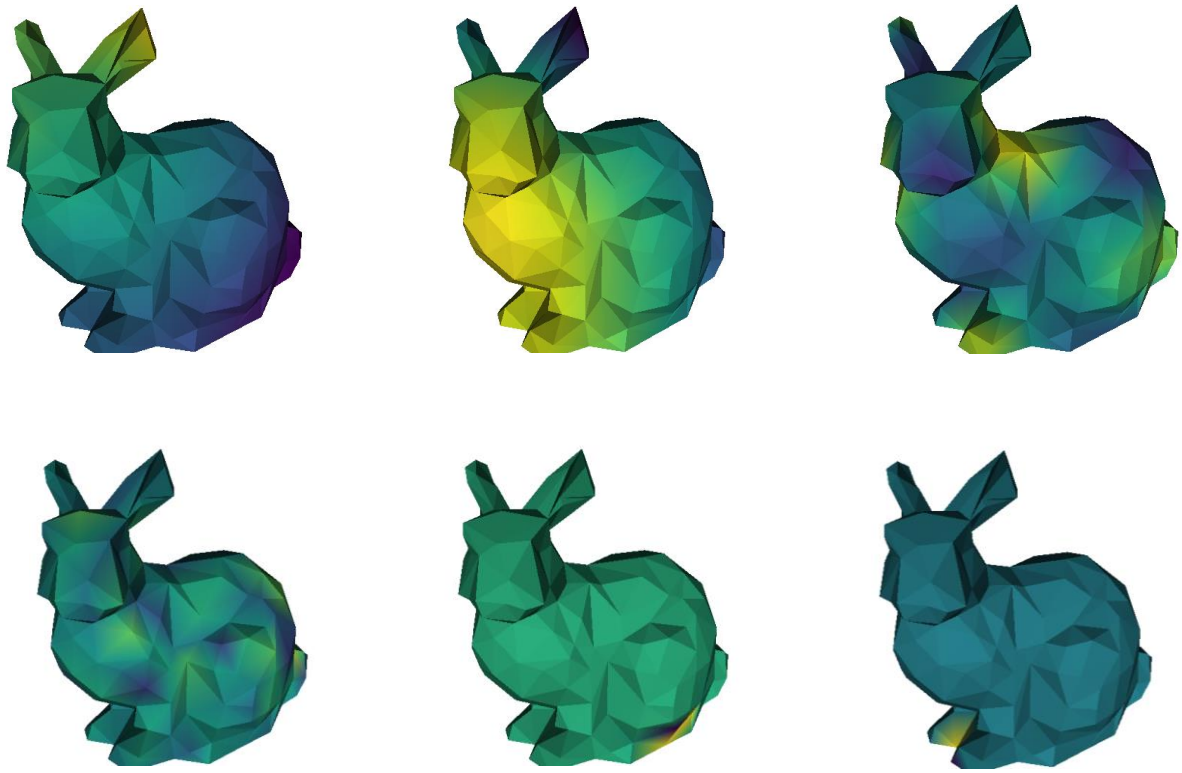

Figure 4. Overlay of eigenvectors e_i onto the mesh geometry; top row; $i = 1$, $i = 4$, $i = 20$; bottom row: $i = 100$, $i = 241$, $i = 242$

3. Eigendecomposition based mesh mixing

At this point, we draw an analogy to the idea of neural style transfer (Gatys et al., 2016; Friedrich et al., 2018; Friedrich and Menzel, 2019) and interpret the low frequency components of a mesh as content of the model and the high frequency components as style. Low frequency components let the viewer recognize the overall object shape but omit certain details like surface texture, which is in direct correlation to the content representation in neural style transfer. Similarly, we argue that the high frequency transformation, as depicted in Figure 3 bottom right, contains model information usually interpreted as style information like small structures and texture.

Therefore, we propose to perform eigendecomposition based style transfer by mixing the high and low frequency parts of two separate triangular meshes. However, since we directly transfer spectral coefficients with their corresponding eigenvectors, we are limited to compatible meshes that share the same graph structure $\mathcal{E}$ and only the vertex positions V are allowed to differ. Additionally, similar semantic regions have to be encoded by similar sets of vertices. In case of e.g. cars, we have to assume that the set of vertices modelling the front lights of the first triangular mesh has to be at least in a similar geometric region on the second triangular mesh.

Consequently, in this first simple approach we do not achieve a style representation with invariance to e.g. translation or rotation, which is in contrast to neural style transfer. Hence, we cannot perform a general style transfer of arbitrary objects but instead focus on a model family like different types of cars with the same mesh topology.

https://doi.org/10.1017/dsd.2020.162 Published online by Cambridge University Press

3.1. Vehicle dataset for style transfer

In the following, we utilize the automotive datasets provided by Umetani (2017) to illustrate our proposed framework. The automotive body shapes have homogenous meshes, which results in identical mesh graph structures given by $\mathcal{E}$. Furthermore, due to the way Umetani's meshes are generated, the vertices have a rough semantic similarity from one mesh to another, meaning vertices on the front bumper of one car are also located on the front bumper of all other cars, etc. This is true for most car types like sedans and SUVs, but might differ slightly for more distinct types like pickups or buses. The individual meshes consist of $n = 6146$ vertices. We transform the original quad meshes by Umetani to triangular meshes.
In addition to the original mesh geometries, we derive a set of automotive prototypes representing different categories like compact, SUV and sport. For doing this, we take the original meshes of a representative of each category and manually remove fine details in order to obtain car shapes, which solely embody the coarse shape of the category. The editing preserves the topology and the rough vertex positions. The resulting five category prototypes are depicted in Figure 5.

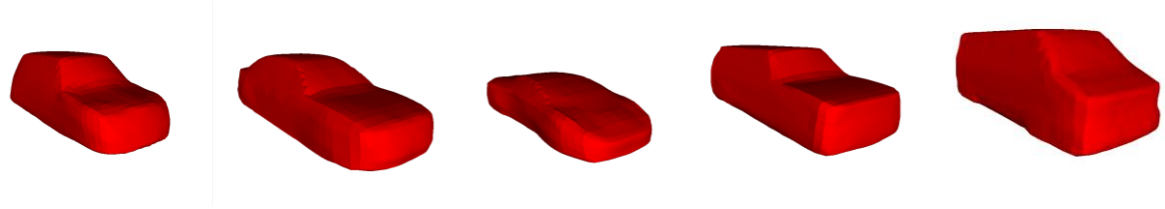

Figure 5. Automotive prototypes derived from Umetani's (2017) dataset

However, in general, our method does not rely on shape prototypes necessarily. We achieve very similar prototype models when applying spectral low pass filtering but as we include more high frequency components, the prototypes change in an unfavourable manner as more detailed structures appear on the prototype instead of refining only the global appearance (Figure 6). Thus, we base our final portfolio stylization on the shape prototypes derived from Umetani's dataset.

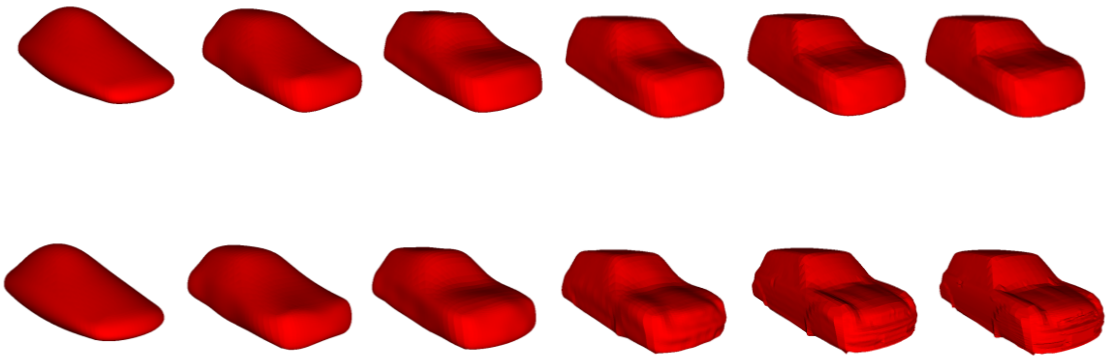

Figure 6. Low pass filtered automotive model. Top row: Compact car prototype derived from Umetani's dataset; Bottom row: Original model of compact car prototype

3.2. Style transfer method based on eigenspace projections

As stated above, the meshes of all different geometries in our dataset have the same topology. Hence, we create a new set of vertices V_{mix} by a modification of the mesh reconstruction of Equation (11) and generate a linear combination of the spectral coefficients and eigenvectors of two different geometries, i.e., the content and style meshes as follows:

$$V_{mix} = E_{content} W_{content} C_{content} + E_{style} W_{style} C_{style} \tag{12}$$

Here, W are diagonal $n \times n$ matrices with $0 \leq w_{ii} \leq 1$. Setting $w_{ii} = 0$ omits eigenfrequency information corresponding to the eigenvalue λ_i. For our models generated in this work, we usually

https://doi.org/10.1017/dsd.2020.162 Published online by Cambridge University Press

consider the first 100 eigenvalues of the content source mesh and eigenvalues from no. 100 to 3000 as style source. The exact setting depends on the used models for the content and style source as well as on the subjective preference of the user. The weighting W is not necessarily binary. We can easily model smooth transitions or even over amplify e.g. high frequency features.

3.3. Evaluation of style transfer for vehicles models

In order to estimate the success of the generated automotive models by eigendecomposition mixing, we compare the results to a simple baseline method, which just takes the average of the vertex positions between content and style mesh for each vertex. The mean models smoothly blend between content and style shape and do not consider the fact that the overall shape should be close to the content source (Figure 7). For example, the mean models have a length exactly in between the content and style mesh instead of keeping the length of the content source mesh. We also observe in the area of the trunk on both examples that the rough shape of the content source is noticeably neglected while our eigendecomposition approach performs as desired.

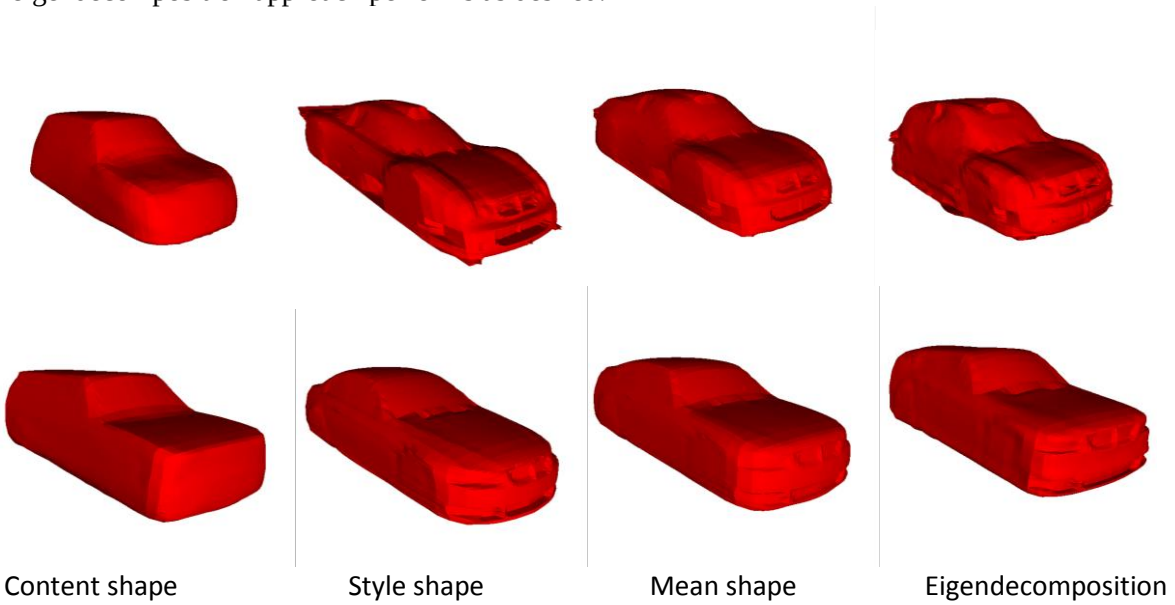

Figure 7. Comparison of automotive body shapes produced by a mean operator and eigendecomposition mixing. 1. Column: Prototype/content source, 2. Column: Style source, 3. Column: Mean result, 4. Column: Eigendecomposition result

Additionally, we note that shape-mixing approaches based on latent vector representations from autoencoders, as presented by Umetani (2017), in principal suffer from the same issues as a simple mean mesh. In adverse situations, the generated latent vector does not represent a geometry even close to one of the source meshes, which is especially true for non-variational autoencoders. Furthermore, our own experience has shown that autoencoders tend to smooth details, in which we are especially interested. In contrast, the proposed approach has the ability to transfer sharp features from the style shape onto the content shape.

3.4. Portfolio generation by eigenspace projections

The framework proposed above enables us to generate an automotive design portfolio in very short time in a semi-automated manner. At first, we identify style features, i.e. high-frequencies of the eigendecomposition, and transfer them on the set of car shape prototypes. Since we only have to calculate the eigenvectors of a potentially unseen new style source, while we can fall back to our precomputed prototypes for each category, the computational effort is minimal. We experienced that a default configuration for W yields reasonable results in most cases. Nevertheless, it is possible to perform a manual fine-tuning of the eigendecomposition mixing in real time afterwards for each prototype. The results of the style transfer for two example vehicle portfolios sets are visualized in Figure 8.

https://doi.org/10.1017/dsd.2020.162 Published online by Cambridge University Press

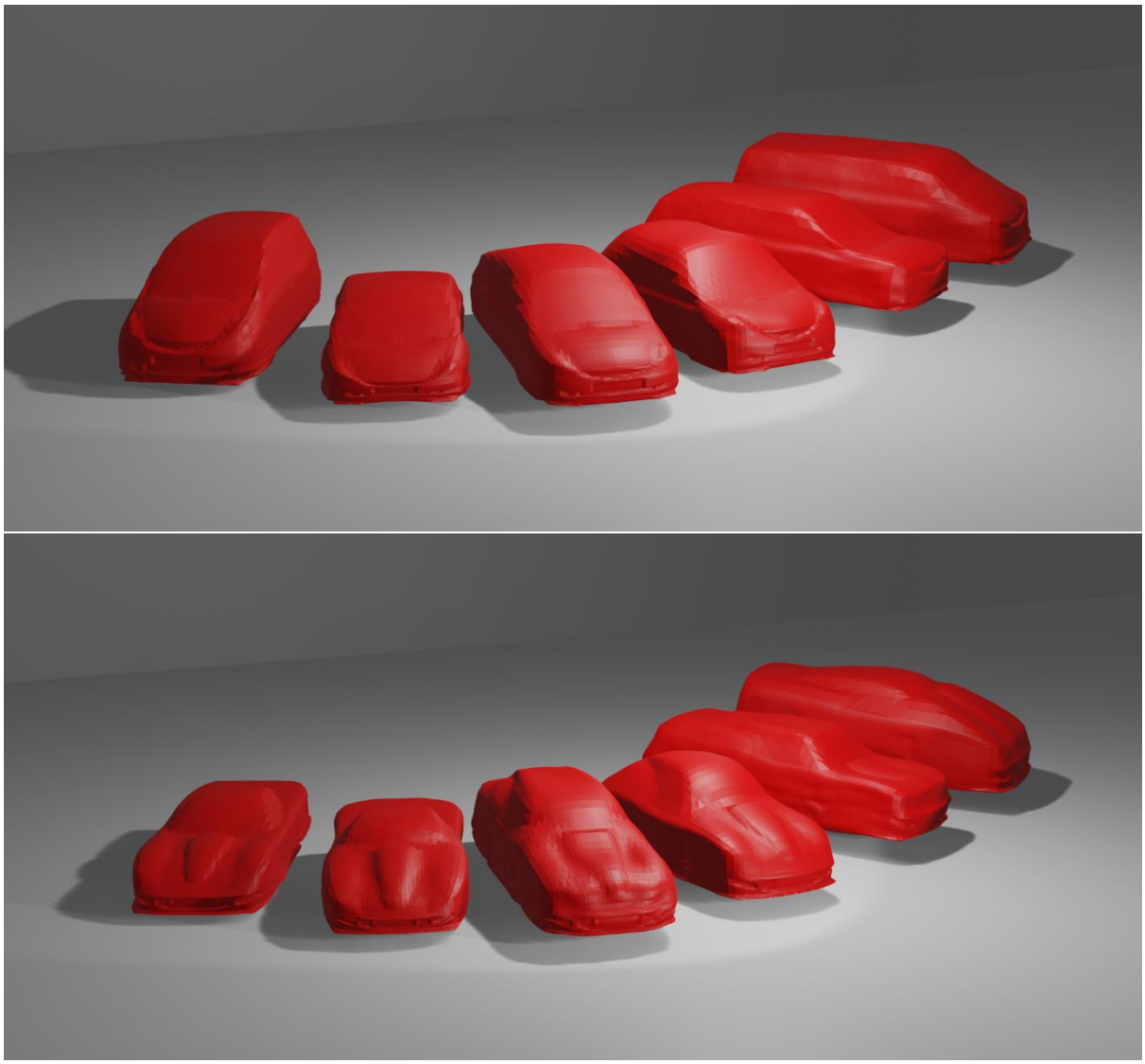

Figure 8. Two exemplary automotive portfolios, the style features of the left car are transferred to the set of five prototypes (Figure 5)

4. Conclusion

In this paper, we introduced a 3D style transfer based on the spectral eigendecomposition of mesh Laplacians. Starting from prototype content shapes, we transfer style elements from a different car mesh to all prototype shapes. This framework enables a rapid generation of automotive portfolios homogeneously stylized as defined by the style car shape. As advantage, the proposed method allows a user working on a novel (car) design to visualize the current style features on a portfolio of designs in an online fashion, which supports decision making processes. The generated portfolio acts as a creative inspiration and starting point for follow up design tasks and studies. The method is not limited to automotive models as used in this work but generally applicable on any compatible dataset under the given mesh constraints. We have discussed the differences and advantages of our approach in the style transfer context compared to other mesh mixing methods. We are also able to selectively combine global and local features enabling us to respect given constraints like overall car length or the height of specific parts such as the trunk. This is essential for a proper portfolio generation process. Since the computational expensive parts of the algorithm are executed beforehand, the proposed approach and design exploration tool provide an excellent usability with real-time editing capabilities. Additionally, the proposed method has only very few and very intuitive free parameters. By adjusting those mixing coefficients, the style transfer can be fine-tuned in order to increase or avoid desired or unwanted features.

https://doi.org/10.1017/dsd.2020.162 Published online by Cambridge University Press

On the contrary, the current approach only supports style transfer between meshes with the exact same topology. One future direction is to extend the approach to handle meshes with different topologies. Besides, different topology meshes style transfer would allow for functional models closer to production instead of meshes for pure visual evaluation, which would further facilitate development and design. Additionally, a homogenous mesh generator with semantic awareness would strongly increase the output quality. This generator could place vertices on identical components across highly differing automotive categories that would prevent undesired artefacts like e.g. windshield wipers on trunks.

References

Achlioptas, P. et al. (2018), "Learning representations and generative models for 3d point clouds", in *35th International Conference on Machine Learning, ICML 2018*. pp. 67-85. Available at: http://arxiv.org/abs/1707.02392 (accessed 14 November 2019)

Brock, A. et al. (2016), "Generative and Discriminative Voxel Modeling with Convolutional Neural Networks". Available at: http://arxiv.org/abs/1608.04236 (accessed 14 November 2019).

Friedrich, T., Aulig, N. and Menzel, S. (2018), "On the Potential and Challenges of Neural Style Transfer for Three-dimensional Shape Data", *EngOpt 2018: Proceedings of the 6th International Conference on Engineering Optimization*, Springer International Publishing. https://doi.org/10.1007/978-3-319-97773-7

Friedrich, T. and Menzel, S. (2019), "Standardization Of Gram Matrix For Improved 3D Neural Style Transfer", *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, Xiamen, China. (accepted).

Gatys, L., Ecker, A. and Bethge, M. (2016), "A Neural Algorithm of Artistic Style", *Journal of Vision*, Vol. 16 No. 12. https://doi.org/10.1167/16.12.326

Jing, Y. et al. (2017), "Neural Style Transfer: A Review", pp. 1-25. Available at: http://arxiv.org/abs/1705.04058.

Kato, H., Ushiku, Y. and Harada, T. (2018), "Neural 3D Mesh Renderer", *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3907-3916. https://doi.org/10.1109/CVPR.2018.00411

Meyer, M. et al. (2003), "Discrete Differential-Geometry Operators for Triangulated 2-Manifolds", *Visualization and Mathematics III*, pp. 35-57. https://doi.org/10.1007/978-3-662-05105-4_2

Pinkall, U. and Polthier, K. (1993), "Computing discrete minimal surfaces and their conjugates", *Experimental Mathematics*, Vol. 2 No. 1, pp. 15-36. https://doi.org/10.1080/10586458.1993.10504266

Reuter, M., Wolter, F.E. and Peinecke, N. (2006), "Laplace-Beltrami spectra as 'Shape-DNA' of surfaces and solids", *CAD Computer Aided Design*, Vol. 38 No. 4, pp. 342-366. https://doi.org/10.1016/j.cad.2005.10.011

Sorkine, O. et al. (2004), *Laplacian Surface Editing*, Eurographics Symposium on Geometry Processing.

Sorkine, O. (2005), "Laplacian Mesh Processing", *Eurographics, (Section 4)*, pp. 53-70. https://doi.org/10.1128/JVI.00468-10.

Umetani, N. (2017), "Exploring Generative 3D Shapes Using Autoen-coder Networks", SIGGRAPH Asia Technical Brief. https://doi.org/10.1145/3145749.3145758.

Yang, Z., Jiang, H. and Zou, L. (2019), "3D Conceptual Design Using Deep Learning", *Science and Information Conference*. Springer, pp. 16-26. https://doi.org/10.1007/978-3-030-17795-9_2

Zhang, H., Van Kaick, O. and Dyer, R. (2007), "Spectral Methods for Mesh Processing and Analysis", *{STAR} Proceedings of Eurographics*, Vol. 92 No. RC-20404, pp. 1-22. https://doi.org/10.1.1.132.8135.

https://doi.org/10.1017/dsd.2020.162 Published online by Cambridge University Press