Exploiting Local Geometric Features in Vehicle Design Optimization with 3D Point Cloud Autoencoders

Thiago Rios, Bas van Stein, Patricia Wollstadt, Thomas Bäck, Bernhard Sendhoff, Stefan Menzel

2021

Preprint:

This is an accepted article published in IEEE Congress on Evolutionary Computation 2021. The final authenticated version is available online at: https://doi.org/10.1109/CEC45853.2021.9504746

Exploiting Local Geometric Features in Vehicle Design Optimization with 3D Point Cloud Autoencoders

Thiago Rios*, Bas van Stein†, Patricia Wollstadt*, Thomas Bäck†, Bernhard Sendhoff* and Stefan Menzel*
*Honda Research Institute Europe GmbH, Carl-Legien-Str. 30, 63073 Offenbach, Germany
†Leiden Institute of Advanced Computer Science (LIACS), Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
Email: {thiago.rios, patricia.wollstadt, bernhard.sendhoff, stefan.menzel}@honda-ri.de,
{b.van.stein, t.h.w.baeck}@liacs.leidenuniv.nl

***Abstract*—Methods for learning and compressing high-dimensional data allow designers to generate novel and low-dimensional design representations for shape optimization problems. By using compact design spaces, global optimization algorithms require less function evaluations to characterize the problem landscape. Furthermore, data-driven representations are often domain-agnostic and independent of the user expertise, and thus potentially capture more relevant design features than a human designer would suggest. However, more factors than the dimensionality play a role in the efficiency of design representations. In this paper, we perform a comparative analysis of design representations for 3D shape optimization problems obtained with principal component analysis, kernel-principal component analysis and a 3D point cloud autoencoder, which we apply on a benchmark data set of computer aided engineering car models. We evaluate the shape-generative capabilities of these methods and show that we can modify the geometries more locally with the autoencoder than with the remaining methods. In a vehicle aerodynamic optimization framework, we verify that this property of the autoencoder representation improves the optimization performance by enabling potentially complementary degrees of freedom for the optimizer. With our study, we provide insights on the qualitative properties and quantifiable measures on the efficiency of deep neural networks as shape generative models for engineering optimization problems, as well as analyses of geometric representations for engineering optimization with evolutionary algorithms.**

***Index Terms*—design representation, shape optimization, point cloud autoencoder, sensitivity analysis**

I. Introduction

The representation of computer aided engineering (CAE) models is not canonical and often high-dimensional. In evolutionary optimization, where the algorithms estimate the problem landscape through design evaluations, higher-dimensional design representations increase the computational effort to solve the optimization by requiring more design evaluations to characterize the problems' landscape [1]–[3]. Hence, manually selecting a design representation is often challenging and requires experience of the user on the domain of the optimization [4], [5].

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 766186 (ECOLE).

Data-driven dimensionality reduction techniques identify features that optimally describe a data set according to predefined criteria. The low-dimensional set of features obtained with these methods are often not intuitive to a human designer [6], but potentially unlock novel degrees of freedom for design and identification of concepts [7], [8], even though the data has a bias introduced by the designers that created the data set. Nevertheless, most of the methods available in the literature consider input data in Euclidean (structured) domains, whilst engineering applications often require geometric representations in unstructured domains, such as polygonal meshes.

The recent advances in storage technology, 3D-scanning and the development of powerful graphic processing units (GPUs) enabled the introduction of geometric deep learning techniques for processing non-Euclidean (unstructured) data [9]. The methods in this class learn on different types of representation, such as graphs and 3D point clouds [10], and the applications range from classical machine learning to real-time tasks, e.g., scene reconstruction in autonomous driving [11], [12]. Among the existing architectures, state-of-the-art 3D point cloud autoencoders learn compact geometric representations and achieved impressive results in different shape-generative tasks [13]. Furthermore, sampling 3D point clouds from CAE models requires low computational effort, preserve enough geometric details, and suit multiple geometric processing algorithms [14]. Nevertheless, albeit different works approached shape design and optimization problems with point cloud autoencoders [15]–[17], the literature still lacks an optimization-oriented benchmark of data-driven geometric representations that include geometric deep learning techniques.

In the present paper, we analyze features learned by a 3D point cloud autoencoder as design representation with respect to two dimensionality reduction techniques: Principal component analysis (PCA) and kernel principal component analysis (K-PCA). In our analyses, we consider that an existing data set of vehicle CAE models is available, from which we abstract the design features utilizing the selected methods. We compare the characteristics of these representations based on criteria that cover shape-generative capabilities and

performance in design optimization problems. These analyses allowed us to identify and quantify the advantages of the representation learned by the 3D point cloud autoencoder in shape optimization problems, which is the main contribution of our work.

The remainder of the paper is organized as follows: In Section II, we review methods for reducing the dimensionality of data in the context of shape evolutionary optimization. In Section III, we discuss the selected methods and experimental settings and propose the criteria to compare the representations. In Section IV, we discuss the results of the experiments and compare the obtained representations. In Section V, we test our hypotheses on real-world engineering optimization problems, which concern the aerodynamic performance of car shapes with the selected representations. Finally, we conclude our paper with a summary and outlook in Section VI.

II. Related Work

Dimensionality reduction techniques project the data onto a low-dimensional space and enable the visualization of high-dimensional data structures with simpler schemes. However, the vast portfolio of existing techniques and applications hinder the understanding of differences between methods, and thus the selection of a particular technique to address a problem of interest. Hence, different works in the literature propose taxonomies to support the selection of methods [6], [18], [19].

In a recent survey [6], the authors quantified the differences in the performance of 44 dimensionality reduction techniques and proposed a simplified taxonomy. The authors evaluated the techniques on 18 benchmark data sets, which comprised different types of data, and used seven different quality metrics, which included similarity of data patterns, preservation of pair-wise distances and data distribution. These experiments indicated that t-distribution stochastic neighborhood embedding (t-SNE) [20], uniform manifold approximation and projection (UMAP) [21] and interactive document map (IDMAP) [22] achieved on average the best quality indicators, and UMAP was the technique that best preserved pair-wise distances between samples. Nevertheless, the current implementations of these methods do not allow the reconstruction of the input representation, which hinders their application in shape optimization problems.

Targeting surrogate-assisted evolutionary optimization, the authors in [23] analyzed four dimensionality reduction techniques: Principal component analysis (PCA), kernel principal component analysis (K-PCA), autoencoder (AE) and variational autoencoder (VAE). In the study, the authors reduced the dimensionality of synthetic data generated from benchmark functions and compared the accuracy of surrogate models and optimization performance achieved with each representation. The authors reported that the surrogate models trained with the representations yielded by the autoencoder achieved the highest accuracy, while the optimizations performed with the PCA representation led to the designs that were the closest to the true optima. Nevertheless, the data used to generate the representations was sampled from a Euclidean space with orthogonal features, which differs from the space where geometric 3D data is typically represented.

In the literature, different authors reported the use of PCA and K-PCA representations in shape optimization problems. In [24], the authors provide a comprehensive analysis of PCA in different applications using data set that comprised results of 32 vehicles on 11 road tests. The authors showed that the representation holds the most relevant information in the first components, and that in regression, ranking and optimization experiments, the authors claim that the representation lacks robustness and could not be adapted to multiple problems. The authors in [25] assessed the performance of the features yielded by a deep autoencoder, PCA and nonlinear variations as the representation of the shape of a Burke-class destroyer. As base parameterization, the authors used a uniform surface mesh with the coordinates of the nodes parameterized by 27 variables. Comparing the reconstruction losses, the authors showed that nonlinear methods outperformed PCA at higher data compression levels, where the deep autoencoder achieved the best performance overall.

The work in [25] already hinted on the potential of autoencoder representations, although the data was defined in a structured domain. Alternatively, 3D point cloud autoencoders learn compact geometric representations from unstructured data, which is the general case for CAE models. According to the survey on algorithms for processing point cloud data in [13], point-based networks performed successfully in classification and shape-generative tasks [26]–[28]. The autoencoders in this class process the input data initially with a point-wise operator, e.g., 1D convolutions, and learn a low-dimensional representation at a bottleneck layer, the so-called latent representation, after a global operator, such as *max pooling* [29]. Hence, the features learned in the latent layer are invariant to the permutation of the points, which increases the applicability of the architecture to shapes with different topology and resampled data sets. In [28], the authors show that the features learned in the latent layer, when forwarded to the decoder, operate as design variables and allow to perform interpolation (Fig. 1) and additive shape operations.

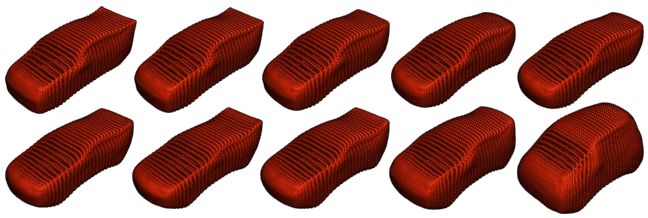

Fig. 1. Example of shape interpolation using the latent variables of the autoencoder proposed in [16] trained on car shapes of the ShapeNetCore.

In [16], the authors assessed the efficiency of the latent variables learned with a similar architecture as shape representation in a target shape matching optimization framework. Their experiments indicated that the shape-generative power of the autoencoder is limited by the geometric features observed

in the training data, since the autoencoder poorly approximated the geometries that were extrapolated from the used data set. The feature visualization method in [29] indicated that the latent features correspond to the occupancy of the input space, and that increasing the dimensionality of the latent space allows the autoencoder to learn more shape details. Although the study in [29] provided an insight on the characteristics of the autoencoder as shape-generative model, a quantitative comparison of the proposed autoencoder to further data-driven methods in shape optimization scenarios is currently missing and, therefore, we address it in the present paper.

III. Learning Low-dimensional Representations

Our objective is to compare the design representation learned by the 3D point cloud autoencoder implemented in [29] with the representations yielded by PCA and K-PCA. The underlying motivation is to search for compact design representations to optimize 3D shapes with evolutionary algorithms, where an existing CAE data set is available prior to the optimization. Hence, we first introduce the data set and sampling techniques that we used to pre-process the data for training the models. Then, we state the settings of the techniques for dimensionality reduction and, last, we define the outline of the experiments to compare the design representations.

A. CAE Data Set

The data set selected for our experiments comprises 3500 shapes from the car class in the ShapeNetCore repository [30]. The data in the repository is represented as polygonal meshes with different topology, centered in the Cartesian space to the coordinates $(0, 0, 0)$ and scaled to the range $[-1, 1]^3$. Since the PCA and K-PCA require input data in Euclidean domains and to simplify the mesh reconstruction on the generated point clouds, we sampled the geometries using a shrink-wrapping algorithm based on [31].

The shrink-wrapping algorithm comprises three steps: Coarse mesh generation, shrinking and mesh smoothing (Fig. 2). In the first step, we generated as initial coarse mesh a parallelepiped with faces tangent to the target shape. In the second step, the algorithm approximates the nodes of the initial mesh to the corresponding nearest points of the target shape. For that, we used Eq. 1, where x_i^t is the position of the *i*-th node of the initial mesh at step t, x_n is the nearest point in the target geometry and α is the step size. In the third step, a smoothing algorithm, which we selected as the Laplacian algorithm proposed in [32], relaxes the generated surface mesh. For completeness, we set the number of shrinking and smoothing steps as 10 and $\alpha = 0.5$ experimentally (results not shown).

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \alpha(\mathbf{x}_n - \mathbf{x}_i^t) \tag{1}$$

Since the generated meshes were isomorphic, we sampled the point clouds from vertices of the meshes according to the same ordering, which allowed us to handle the coordinates as variables in a Euclidean space. For reducing the dimensionality of the point clouds using PCA and K-PCA, we organized the data set by concatenating the point clouds in a $(S, 3N)$ matrix, where S is the number of shapes and N is the size of the point clouds. Hence, each line of the matrix comprised the coordinates ordered according to the principal axes, such as $(x_1, \ldots, x_N, y_1, \ldots, y_N, z_1, \ldots, z_N)$. Since the autoencoder processes the input data with 1D convolutions, we reshaped the matrix into a $(S, N, 3)$ tensor.

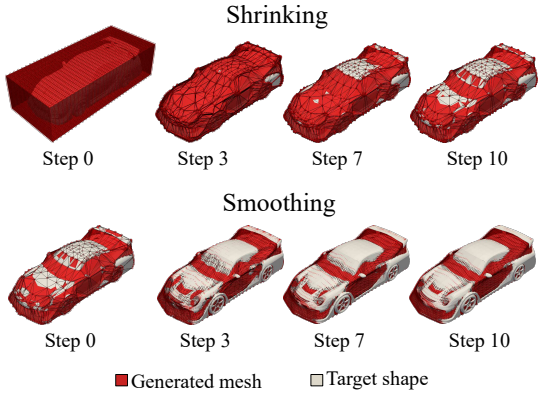

Fig. 2. Shrink-wrapping algorithm applied to a car mesh sampled from the ShapeNetCore data set.

B. Shape Representations

We reduced the dimensionality of the point cloud data set with PCA and K-PCA using the implementations available in [33]. We adopted the standard hyperparameters of the algorithms, varying only the number of components and the type of kernel for the K-PCA algorithm between linear, polynomial, radial basis, sigmoid and cosine functions. For the 3D point cloud autoencoder, we utilized the same architecture as in [29] (Table I). The encoder comprises five 1D convolutions, followed by a *max pooling* layer, which yields the latent variables, and a decoder, which comprises three fully-connected layers. The last convolutional layer is activated with a hyperbolic tangent function, which limits the values of the latent variables to the interval $[-1, 1]$, and thus eases the definition of design constraints and sampling in the latent space.

TABLE I
Architecture of our 3D point cloud autoencoder.

Layer	Type	Activation	Features	Output dimensions
1	1D-C	ReLU	64	[N×64]
2	1D-C	ReLU	128	[N×128]
3	1D-C	ReLU	128	[N×128]
4	1D-C	ReLU	256	[N×256]
5	1D-C	tanh	L	[N×L]
6	maxPool	-	L	[1×L]
7	FC	ReLU	256×3	[256×3]
8	FC	ReLU	256×3	[256×3]
9	FC	sigmoid	N×3	[N×3]

1D-C: 1D-convolution
N: Size of the point cloud
L: Number of latent variables
FC: *Fully-connected*

To train the models, we randomly selected 90% of the sampled point clouds and normalized the Cartesian coordinates to the range $[0.1, 0.9]^3$, preserving the aspect ratio of the shapes. Specifically for the autoencoder, we trained the network using the Adam optimizer [34] with learning rate $\eta = 5\text{E}{-}04$, $\beta = 0.99$ and $\beta_1 = 0.9$, and organized the training data into batches of 50 shapes. We defined the loss function as the mean squared distance (MSD) between corresponding input and predicted points as defined in Eq. 2, where $\mathbf{x}_i$ and $\tilde{\mathbf{x}}_i$ are the i-th points in the input and predicted point clouds, respectively.

$$\text{MSD} = \frac{1}{N}\sum_{i=1}^{N} \|(\mathbf{x}_i - \tilde{\mathbf{x}}_i)\|^2 \tag{2}$$

C. Experimental Set-up

Considering the discussed methods and data set, we implemented the algorithms for generating the models using Python and a machine with two CPUs Intel® Xeon® Silver, clocked at 2.10 GHz, and four GPUs Nvidia® GeForce® RTX 2080 Ti. In order to consider the effects of the dimensionality L of the representations, we generated representations with $L = (5, 10, 20)$.

For comparing the representations, we proposed the following criteria:

1) Reconstruction losses: The reconstruction accuracy is often a baseline criterion for evaluating shape-generative models. We evaluated the reconstruction losses on the training and test data by calculating the Chamfer distance (CD) [35] between the reconstructed and original shapes.

2) Shape sensitivity analysis: For this criterion, we analyzed the sensitivity of the displacement of the points with respect to each variable in the low-dimensional representations. We calculated the total sensitivity for 30 shapes sampled from the data set using the Sobol's method with Satelli's sampling scheme [36]. For each shape, we generated $30(L{+}2)$ variations in a range of $\pm 30\%$ of each variable of the corresponding low-dimensional representations, where L is the dimensionality of the low-dimensional representation. For comparing the techniques, we analyzed the maximum sensitivities obtained with each variable, which hints on their relevance for shape modification. We also projected the sensitivity values onto the 3D point cloud representations as color maps, similar to the study in [29], which revealed the mapping between the representation features and geometric characteristics.

3) Shape-generative performance: We assessed the shape-generative performance with two experiments. First, we linearly interpolated the 3D point clouds and corresponding low-dimensional representations of 50 different pairs of shapes, which were randomly sampled from the data set, and calculated the MSD between reconstructed and interpolated shapes at intermediate steps. Second, we targeted generating *crossover* shapes by transferring sets of variables between representations of different shapes. Hence, we evaluated the representations by inspecting the reconstruction of the shapes and calculating the distance between the points of the reconstructed shape with respect to the initial point cloud and shape with the targeted features.

IV. Analysis of the Representations

In the sections IV-A to IV-C, we analyze the different representations according to the criteria defined in the previous section.

A. Shape Reconstruction Losses

We analyzed the reconstruction losses based on the distribution of the Chamfer Distance over the data samples, where we divided the cases by the dimensionality of the representations (Fig. 3). We observed that the quality of the reconstructions slightly increased with the increasing dimensionality of the representations, which is intuitive. Comparing the dimensionality reduction techniques, the 3D point cloud autoencoder (AE) and PCA achieved the best performance, followed by the K-PCA with polynomial kernel (K-PCA-P). Hence, for simplicity, we only considered in the following analyses the results of these three representations.

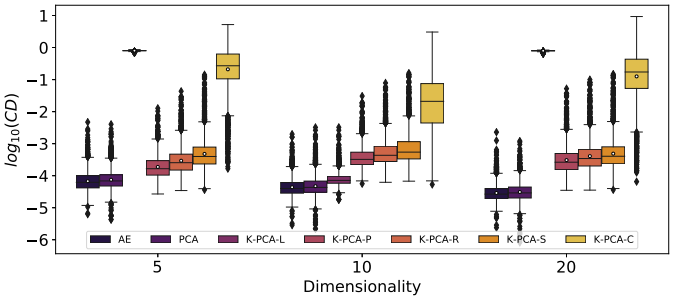

Fig. 3. Distribution of samples over the values of the Chamfer Distance on the data set. The subindices *L, P, R, S, C* identify the *linear, polynomial, radial-basis function, sigmoid and cosine* K-PCA kernels, respectively.

Comparing shape reconstructions obtained with the AE, PCA and K-PCA-P (Fig. 4), we observed that the K-PCA-P yielded coarse reconstructions of the input shapes. Differently, the AE and PCA gradually increased the level of shape details with the dimensionality of the representations.

B. Shape Sensitivity to Design Variables

We first analyzed the sensitivity of the shapes using a box plot of the maximum sensitivity values associated to each variable in the low-dimensional representations (Fig. 5). Regardless of the dimensionality, the maximum sensitivity of the PCA and K-PCA-P variables decreased with the dimensionality of the latent space, while the latent variables of the AE presented more uniform values. We interpreted the results as in line with the background of the methods: The PCA ranks the components based on data variance, and thus the relevance of the components to shape modifications decrease with the dimensionality of the representation. Differently, the AE learns variables that describe the occupancy of the input space [29]. Hence, the AE representation describes geometric features that are more local than obtained with the PCA, yet more sensitive to changes in the latent variables.

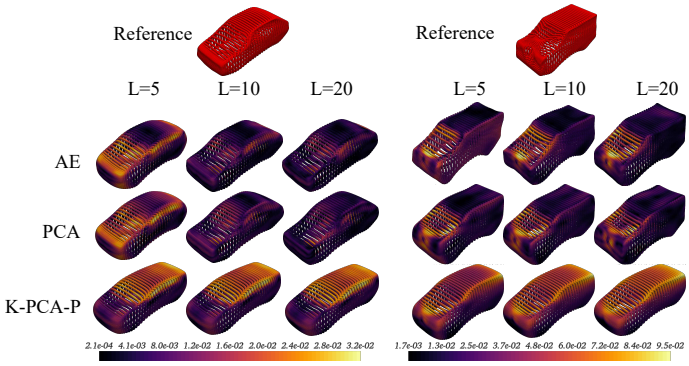

Fig. 4. Reconstruction of shapes sampled from the data set with different methods and dimensionalities. The colors indicate the distance to the corresponding points in the reference point clouds, where the brighter colors indicate higher distances.

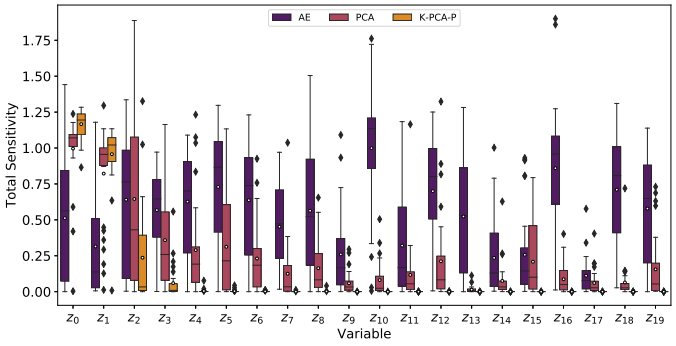

Fig. 5. Box plot of the maximum sensitivity values achieved on 30 shapes from the data set with the 20D representations.

To verify our interpretation, we projected the sensitivities obtained with the 20D representations onto the 3D point clouds selected for the analyses (Fig. 6). We observed that, in general, the variables of the AE representation were associated to changes in smaller regions in the point clouds than obtained with the other representations. Furthermore, the sensitivity of the first PCA components were associated to regions that potentially describe the dimensions of the shape along the main axes (length, height, and width), which also coincided with the Cartesian axes of the space where the shapes were embedded.

C. Shape-generative Capabilities

In a first analysis, we interpolated 50 pairs of shapes both in the low-dimensional and Cartesian spaces, and we calculated the MSD between the generated point clouds at 20 intermediate steps of the interpolation (Fig. 7). We observed that the AE and PCA representations performed similarly, with a slight trend of the autoencoder to increase the MSD values at central interpolation steps ($\approx [0.4, 0.8]$). Our interpretation is that in some of the interpolations, the AE representation crossed regions in the latent space that map features not observed in any of the shapes in the selected pair. Therefore, when reconstructing the point clouds, the AE generated local

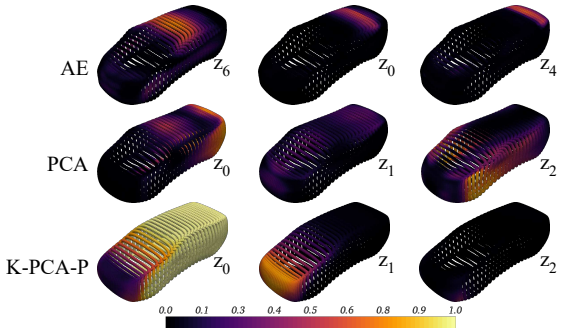

Fig. 6. Visualization of the displacement sensitivity of points in a 3D point cloud of a sedan car to three components of the 20D design representations. The brighter colors indicate higher sensitivity.

features that increased the difference to the point clouds obtained with the interpolation in the Cartesian space.

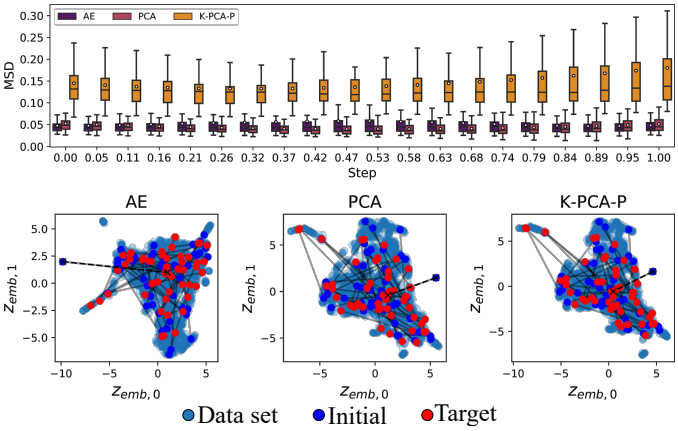

Fig. 7. Box plot of the MSD values calculated on 50 random pairs of shapes sampled from the data set, and 2D embedding of the 10D representations showing the data set and selected samples for the interpolations.

By visually inspecting the reconstructions (Fig. 8), we confirmed that the K-PCA-P reconstructed shapes with less detail than the other representations. Also, the PCA modified the shapes during the interpolation more smoothly than the autoencoder, which modified distinct geometric features at different rates (e.g. the airfoil structure at step 0.3 in Fig. 8).

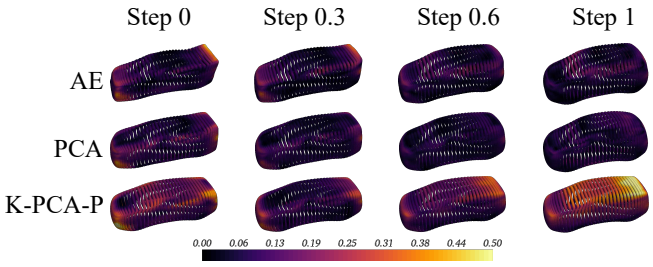

Fig. 8. Comparison between the interpolations in the low-dimensional and Cartesian spaces. The brighter colors indicate higher deviation of the points with respect to the reference.

In a second analysis, we generated *crossover* shapes through transfer of low-dimensional variables. In the experiment, we

transferred the variables associated to the front design of a car shape (target features) to the representations of 50 data set samples (initial shapes). Then, we reconstructed the shapes and measured the differences to the initial shapes and the shape with the target features (Fig. 9). We observed that despite transferring only the variables that mapped the front of the car according to the sensitivity analysis, the PCA also modified other regions of the shape, while the AE modified the targeted region and preserved the remaining geometric characteristics. We confirmed our observations by calculating the MSD between modified and initial shapes, where the AE achieved on average $8.87\text{E}{-02}$ and the PCA $9.52\text{E}{-02}$, and between the modified and target shapes, where AE and PCA achieved on average $2.88\text{E}{-01}$ and $3.06\text{E}{-01}$, respectively.

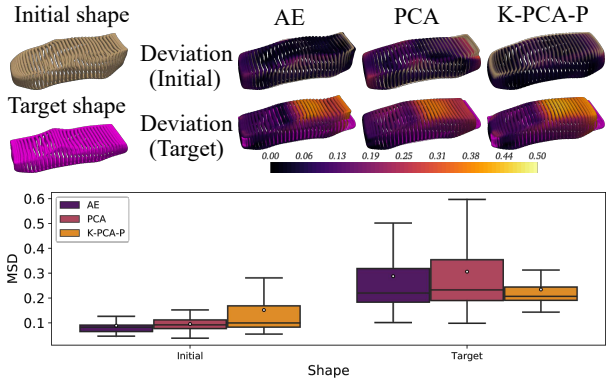

Fig. 9. Reconstruction of shapes after transfer of features and comparison to the original geometry and shape with target feature. The brighter colors indicate larger deviations.

These analyses showed that the AE exploits local geometric features more than the other methods. This property provides a potential advantage in shape optimization problems, especially for problems where the fitness of the designs is more sensitive to changes in shape details. Therefore, in a last set of experiments, we compared the performance of the representation in an industrial-like scenario of vehicle aerodynamic optimization, as discussed in the following section.

V. Vehicle Aerodynamic Optimization

The vehicle aerodynamic performance is associated to the magnitude of the drag and lift forces, which have influence on the fuel consumption [37] and grip on the tires, respectively. In this section, we compare the optimization performance obtained with the AE, PCA and K-PCA-P representations in two sets of experiments, where we minimize the aerodynamic drag and lift forces of car shapes. Since the aerodynamic performance was neglected when generating the representations, their relation to fitness of the designs is a priori unknown. Thus, the differences in optimization performance depend only on the shape-generative capability of the models.

A. Settings

We calculated the aerodynamic forces using computational fluid dynamics (CFD) simulations implemented with OpenFOAM®. For the simulations, the vehicle was embedded in a large fluid domain, with an inlet velocity $U_x = 110$ km/h and a lower wall tangent to the car shape with no-slip condition moving at the same velocity (Fig. 10). The simulations were set on a cluster environment comprising machines with 2 Westmere 4 Core Xeon E5620 clocked at 2.4 GHz and each simulation solved in parallel with 16 processors.

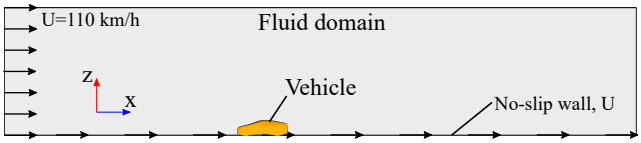

Fig. 10. Control volume and boundary conditions defined for the CFD simulations.

To optimize the shapes, we selected the covariance matrix adaptation evolutionary strategy (CMA-ES) algorithm due to the low number of hyperparameters, fast convergence and suitability for small population sizes [38]. Regarding the settings of the algorithm, we utilized the strategy (μ,λ) with $\mu = 3$ and $\lambda = 10$, initial step size of $5\text{E}{-02}$, and the maximum number of generations of 15.

We defined the objective function (Eq. 3) as the sum of the aerodynamic performance $G(S_x)$ (drag or lift force) and a penalty term, which is the squared MSD between the shape S_x and the data set sample $S_{x,N}$ with the nearest representation in the low-dimensional space. The penalty was weighted by a penalty factor of $\rho = 750$, defined experimentally. Finally, we randomly selected five designs from the data set, which we optimized using the 20D representations that were generated for the previous experiments.

$$\min_x F(x) = G(x) + \rho(\text{MSD}(S_x, S_{x,N}))^2 \tag{3}$$

B. Analysis of the Results

The analysis of the convergence (Fig. 11) showed that the AE and PCA led to similar performance in the aerodynamic drag optimization. However, the designs obtained in the lift optimization with the AE had on average 30% better fitness than with the PCA. Since the lift force is more sensitive to disturbances in the air flow, our interpretation of the results is that the locality of the shape modifications performed by the AE allowed the optimizer to find geometric features that decreased the lift without severely disturbing the air flow. This interpretation also justifies the poor performance of the K-PCA-P, since this method mapped only large-scale geometric features.

We confirmed our interpretation by visually inspecting the optimized designs and calculating the differences with respect to the initial shapes (Fig. 12). The K-PCA-P yielded shapes that were similar to the initial design, as implied by the numerical results. The shapes obtained with AE and PCA presented similar geometric features, which resemble aerodynamic devices for generating downforce, e.g., spoilers. However, the AE enabled more localized shape modifications and thus, a more diverse set of degrees of freedom for the optimization, which is in line with our previous interpretation.

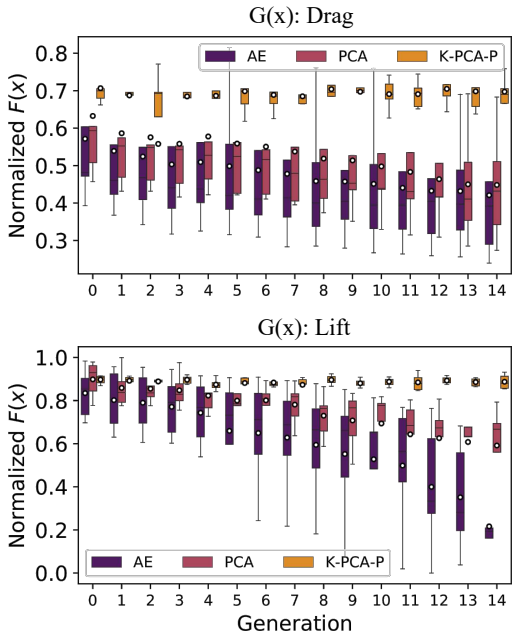

Fig. 11. Normalized fitness of the best individuals per generation for drag and lift optimizations.

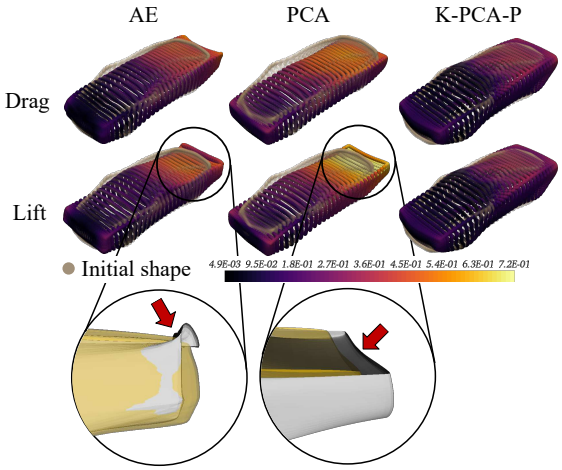

Fig. 12. Reconstruction of optimized samples with distance between corresponding points in the initial shape, and detail of spoiler-like geometric features obtained with the autoencoder and PCA representations.

Finally, we calculated the mutual information [39] between the low-dimensional features and seven performance metrics calculated on 500 shapes of the data set (Fig. 13). Since the mutual information quantifies the non-linear dependency between two independent variables, the analysis of the mutual information potentially indicates differences in the degrees of freedom enabled by the representations. Comparing the distribution of the values, we observed similar patterns to the shape sensitivity analysis: The PCA-based representations concentrated the highest values in the first three components, while the AE have a more uniform distribution over the variables. Hence, we concluded that the AE spreads the information in the low-dimensional space more evenly and potentially in a synergetic fashion, which enables more degrees of freedom for shape-generative tasks and led to better performance in the optimizations.

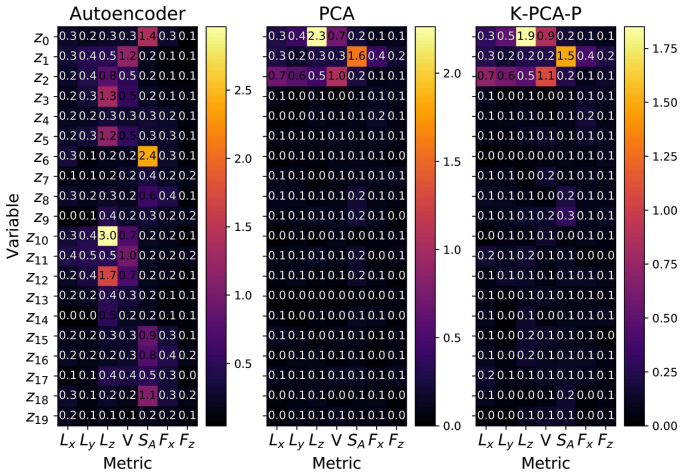

Fig. 13. Mutual information calculated between the variables obtained with the discussed techniques and metrics calculated from the designs: Length (L_x), width (L_y), height (L_z), volume (V), surface area (S_A), aerodynamic drag (F_x) and lift (F_z).

VI. Conclusions

In this paper, we analyzed the representations obtained for shape optimization problems with three data-driven techniques: PCA, K-PCA and a 3D point cloud autoencoder (AE). Our objective was to propose a set of criteria to compare different data-driven design representations and identify potential advantages of the AE representation in shape optimization problems. We performed our analyses on a benchmark data set of 3D shapes of vehicles and extended the results to a real-world aerodynamic optimization scenario.

We evaluated the representations according to four criteria: Data reconstruction losses, shape sensitivity, shape-generative and optimization performance. The AE and PCA yielded the best shape reconstructions, however, the AE exploited better local geometric features than the other representations. This characteristic of the AE was particularly evident when we generated *crossover* shapes by transferring low-dimensional features, where the AE modified only the regions targeted by the transferred variables. The sensitivity analyses also seconds this interpretation and extended the results of the feature visualization method in [29].

In an industrial-like experiment, we optimized the aerodynamic drag and lift of five car shapes. Since minimizing the drag depends mostly on large-scale features (frontal area of the shape), the optimizations with AE and PCA representations performed similarly. However, since the lift force is more sensitive to disturbances in the flow, the fitness of the shapes obtained with the AE was significantly better than with the other methods. Hence, the AE enables a more diverse set of degrees of freedom, which we also verified by analyzing the mutual information between the low-dimensional features and different design performance metrics.

REFERENCES

[1] Y. Qiu, J. Bai, N. Liu, and C. Wang, "Global Aerodynamic Design Optimization Based on Data Dimensionality Reduction," *Chinese Journal of Aeronautics*, vol. 31, no. 4, pp. 643–659, 2018.

[2] D. D'Agostino, A. Serani, and M. Diez, "On the Combined Effect of Design-space Dimensionality Reduction and Optimization Methods on Shape Optimization Efficiency," *2018 Multidisciplinary Analysis and Optimization Conference*, no. June, 2018.

[3] Y. Zhang, Z.-H. Han, L. Shi, and W.-P. Song, "Multi-round Surrogate-based Optimization for Benchmark Aerodynamic Design Problems," in *54th AIAA Aerospace Sciences Meeting*.

[4] S. Menzel and T. Friedrich, "Addressing Complex Engineering Systems: The Value of Evolvability," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 731–739.

[5] S. N. Skinner and H. Zare-Behtash, "State-of-the-art in Aerodynamic Shape Optimisation Methods," *Applied Soft Computing Journal*, vol. 62, pp. 933–962, 2018.

[6] M. Espadoto, R. M. Martins, A. Kerren, N. S. T. Hirata, and A. C. Telea, "Towards a Quantitative Survey of Dimension Reduction Techniques," *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–1, 2019.

[7] J. Matejka, M. Glueck, E. Bradner, A. Hashemi, T. Grossman, and G. Fitzmaurice, "Dream Lens: Exploration and Visualization of Large-Scale Generative Design Datasets," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, ser. CHI '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1–12.

[8] N. Dommaraju, M. Bujny, S. Menzel, M. Olhofer, and F. Duddeck, "Identifying Topological Prototypes using Deep Point Cloud Autoencoder Networks," in *2019 International Conference on Data Mining Workshops (ICDMW)*, 2019, pp. 761–768.

[9] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric Deep Learning: Going Beyond Euclidean Data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[10] E. Ahmed, A. Saint, A. E. R. Shabayek, K. Cherenkova, R. Das, G. Gusev, D. Aouada, and B. Ottersten, "Deep Learning Advances on Different 3D Data Representations: A Survey," *arXiv preprint arXiv:1808.01462v1 [cs.CV]*, 2018.

[11] A. Ioannidou, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris, "Deep Learning Advances in Computer Vision with 3D Data," *ACM Computing Surveys*, vol. 50, no. 2, pp. 1–38, 2017.

[12] Z. Cai, J. Han, L. Liu, and L. Shao, "RGB-D Datasets Using Microsoft Kinect or Similar Sensors: A Survey," *Multimedia Tools and Applications*, vol. 76, no. 3, pp. 4313–4355, 2017.

[13] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep Learning for 3D Point Clouds: A Survey," *CoRR*, vol. abs/1912.12033, 2019.

[14] M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, J. a. Levine, A. Sharf, C. T. Silva, A. Tagliasacchi, L. M. Seversky, C. T. Silva, J. a. Levine, and A. Sharf, "State of the Art in Surface Reconstruction from Point Clouds," *Proceedings of the Eurographics 2014, Eurographics STARs*, vol. 1, no. 1, pp. 161–185, 2014.

[15] N. Umetani, "Exploring Generative 3D Shapes Using Autoencoder Networks," in *SIGGRAPH Asia 2017 Technical Briefs*. New York, NY, USA: ACM, 2017, pp. 24:1–24:4.

[16] T. Rios, B. Sendhoff, S. Menzel, T. Bäck, and B. van Stein, "On the Efficiency of a Point Cloud Autoencoder as a Geometric Representation for Shape Optimization," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 791–798.

[17] S. Saha, S. Menzel, L. L. Minku, X. Yao, B. Sendhoff, and P. Wollstadt, "Quantifying the Generative Capabilities of Variational Autoencoders for 3D Car Point Clouds," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2020, pp. 1469–1477.

[18] L. van der Maaten, E. Postma, and H. Herik, "Dimensionality Reduction: A Comparative Review," *Journal of Machine Learning Research - JMLR*, vol. 10, 01 2007.

[19] H. Yin, "Nonlinear Dimensionality Reduction and Data Visualization: A Review," *International Journal of Automation and Computing*, vol. 4, no. 3, pp. 294–303, Jul 2007.

[20] L. van der Maaten and G. Hinton, "Visualizing Data Using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.

[21] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction," *arXiv preprint arXiv:1802.03426 [stat.ML]*, 2018.

[22] R. Minghim, F. V. Paulovich, and A. de Andrade Lopes, "Content-based Text Mapping Using Multi-dimensional Projections for Exploration of Document Collections," in *Visualization and Data Analysis 2006*, vol. 6060, International Society for Optics and Photonics. SPIE, 2006, pp. 259 – 270.

[23] S. Ullah, D. A. Nguyen, H. Wang, S. Menzel, B. Sendhoff, and T. Bäck, "Exploring Dimensionality Reduction Techniques for Efficient Surrogate-Assisted Optimization," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2020, pp. 2965–2974.

[24] R. Reris and J. P. Brooks, "Principal Component Analysis and Optimization: A Tutorial," in *Operations Research and Computing: Algorithms and Software for Analytics*. INFORMS, Jan. 2015, pp. 212–225. [Online]. Available: https://doi.org/10.1287/ics.2015.0016

[25] D. D'Agostino, A. Serani, E. F. Campana, and M. Diez, "Nonlinear Methods for Design-Space Dimensionality Reduction in Shape Optimization," in *Machine Learning, Optimization, and Big Data*, G. Nicosia, P. Pardalos, G. Giuffrida, and R. Umeton, Eds. Cham: Springer International Publishing, 2018, pp. 121–132.

[26] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 5105–5114.

[27] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point Cloud Auto-encoder via Deep Grid Deformation," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 206–215, 2018.

[28] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning Representations and Generative Models for 3D Point Clouds," in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, vol. 80. Stockholmsmässan, Stockholm Sweden: PMLR, 2018, pp. 40–49.

[29] T. Rios, B. van Stein, S. Menzel, T. Back, B. Sendhoff, and P. Wollstadt, "Feature Visualization for 3D Point Cloud Autoencoders," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–9.

[30] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," *arXiv preprint arXiv:1512.03012v1 [cs.GR]*, 2015.

[31] B. K. Koo, Y. K. Choi, C. W. Chu, J. C. Kim, and B. T. Choi, "Shrink-wrapped Boundary Face Algorithm for Mesh Reconstruction from Unorganized Points," *ETRI Journal*, vol. 27, no. 2, pp. 235–238, 2005.

[32] J. Vollmer, R. Mencl, and H. Müller, "Improved Laplacian Smoothing of Noisy Surface Meshes," *Computer Graphics Forum*, vol. 18, no. 3, pp. 131–138, 1999.

[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[34] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[35] H. Fan, H. Su, and L. Guibas, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image," in *30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Januar, 2017, pp. 2463–2471.

[36] A. Saltelli, P. Annoni, I. Azzini, F. Campolongo, M. Ratto, and S. Tarantola, "Variance Based Sensitivity Analysis of Model Output. Design and Estimator for the Total Sensitivity Index," *Computer Physics Communications*, vol. 181, no. 2, pp. 259 – 270, 2010.

[37] L. Dumas, *CFD-based Optimization for Automotive Aerodynamics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 191–215.

[38] T. Bäck, F. Hoffmeister, and H.-P. Schwefel, "A Survey of Evolution Strategies," in *Proceedings of the Fourth International Conference on Genetic Algorithms*. Morgan Kaufmann, 1991, pp. 2–9.

[39] A. Kraskov, H. Stögbauer, and P. Grassberger, "Estimating Mutual Information," *Phys. Rev. E*, vol. 69, p. 066138, Jun 2004.