INTERNATIONAL DESIGN CONFERENCE – DESIGN 2022
https://doi.org/10.1017/pds.2022.177

Exploiting 3D Variational Autoencoders for Interactive Vehicle Design

S. Saha [1,✉], L. L. Minku [2], X. Yao [2,3], B. Sendhoff [1] and S. Menzel [1]

[1] Honda Research Institute Europe GmbH, Germany, [2] University of Birmingham, United Kingdom, [3] Southern University of Science and Technology, China

✉ sneha.saha@honda-ri.de

Abstract

In automotive digital development, 3D prototype creation is a team effort of designers and engineers, each contributing with ideas and technical evaluations through means of computer simulations. To support the team in the 3D design ideation and exploration task, we propose an interactive design system for assisted design explorations and faster performance estimations. We utilize the advantage of deep learning-based autoencoders to create a low-dimensional latent manifold of 3D designs, which is utilized within an interactive user interface to guide and strengthen the decision-making process.

Keywords: data-driven design, collaborative design, 3D modelling

1. Introduction

In automotive digital development designers rely e.g. on sketching for a first ideation as well as computer-aided design (CAD) and engineering (CAE) tools to accelerate various tasks, particularly through virtual prototyping. These tools enrich the design process by providing a means for altering shapes by the designers to create novel variations of the 3D design based on given user requirements or by allowing the engineers to estimate technical performances of the 3D design. However, on one hand, accurate performance simulations still require excessive computational effort, while on the other hand existing additional information would be beneficial to the designer when exploring the design space to find novel prototypes or improve existing prototypes, which are more sustainable according to technical qualities. Thus, to address the challenge of faster and informed design exploration based on technical design quality indications and to support the designer in the creative thinking process, we envision an interactive cooperative design system (CDS), which the designer can utilize as an additional tool for ideation and easy design exploration in the initial automotive digital development phase.

Recent advances in artificial intelligence (AI) aim to collect and utilize existing digital design and simulation data from past development cycles for the application of machine learning techniques and data analytics. In the context of design, two systems have been proposed on a 2D design level, namely SketchRNN (Ha and Eck, 2018) (https://magenta.tensorflow.org/sketch_rnn_demo) and ShadowDraw (Lee *et al.*, 2011), which provide real-time guidance to human users while drawing online. Both models use a deep generative network and are trained on a database of drawing sequences by human users. During interaction for sketching, both networks propose potential design directions for the next drawing steps. In 3D, Umetani (Umetani, 2017) proposed a system using a deep learning model for faster modification of 3D digital vehicles, however, it lacks on explicit assistance for user guidance. Thus, the present paper aims at transferring the idea of guiding the 3D vehicle design process by combining the findings from data-driven AI methods and the designer's intuition in an interactive CDS.

AI systems using deep learning and machine learning techniques learn from existing data and extract significant information hidden in the data. Geometric deep learning methods offer a promising approach for learning and extracting useful information from existing CAE data to form the foundation of our proposed CDS. Deep generative models for design data, like autoencoders (AEs) (Achlioptas *et al.*, 2018; Rios *et al.*, 2019) and variational autoencoders (VAEs) (Saha *et al.*, 2020), learn and compress high-dimensional 3D point cloud data into a low dimensional latent representation in an unsupervised fashion. The extracted latent representation acts as a compact shape manifold, representing 3D designs with a small number of parameters. This learned latent manifold is suitable for downstream processing tasks, like e.g., the generation of novel designs, optimization tasks, and surrogate modeling (Rios *et al.*, 2019; Saha *et al.*, 2020). Thus, these (V)AEs trained on design representations of CAE models can potentially learn enough information in the latent variables, so that these latent variables are applicable as control parameters for our proposed interactive CDS.

In summary, in this paper, we propose an interactive framework for 3D design exploration on the latent manifold of a trained (V)AE, which assists in traversing the latent manifold to generate novel 3D designs. In the context of automotive design generation, we consider two scenarios from the perspective of the designers and formulate different forms of assistance using our CDS. First, in the exploratory design concept generation phase, new concepts may often be conceived through rough sketches or rudimentary virtual models to identify and explore novel design ideas. Here, our framework aims to guide the designer for *target-oriented design explorations* in the latent manifold. Second, in automotive digital development, designers explore concepts and simulate the performance of 3D shapes separately. To accelerate the process for faster 3D prototype exploration with its performance measures, we provide a means to *simultaneously generate 3D designs from the latent manifold with accurate performance predictions* using a surrogate model trained on the latent manifold. Our main contributions are as follows: First, we suggest a variational autoencoder (VAE) to construct a low-dimensional latent manifold of 3D shapes. Second, we build on the VAE an interactive framework that offers a slider-based user interface to explore the latent manifold. Third, the interface provides an indication on exploration directions in the latent manifold for changing the topology of 3D designs and at the same time provides performance feedback based on the location in the latent manifold, thus supporting the designer for streamlined design ideation and informed decision-making processes.

The remainder of the paper is organized as follows: In Section 2, we review representations in 3D vehicle design related to our application as well as deep generative models. In Section 3, we lay out the pre-processing steps required for our framework and provide the details of our point cloud variational autoencoder architecture (PC-VAE) as well as of the multi-layer perceptron (MLP) as a surrogate model for design performance prediction tasks. In addition, we give an overview of the interactive workflow embedded in a graphical user interface. In Section 4, we show the application of the interactive framework for the above-mentioned two scenarios: First (Section 4.1), target class-oriented topology changes of 3D designs, and second (Section 4.2), performance feedback for informed 3D design directions. Finally, in Section 5, we present a summary and conclusion of our paper.

2. Prior Art

Over the recent years, the advances made in computing power have strengthened the design process. Product designers may rely on sketching (tools) for initial design development, while the virtual prototyping depends on a variety of (commercial) CAD/E tools for modeling 3D digital designs. These digital designs contain enough information to be used as a base representation for shape parameterization. The general idea is that parameterizing a design with a series of design variables facilitates faster design modification and generation. One popular parameterization technique are parameterized splines with control points as modifiers to define the shape. As an alternative, especially for digital models represented as surface meshes, deformation methods like free-form deformation (FFD) (Sederberg and Parry, 1986) offer effective shape parameterizations. Here, a shape is embedded into a tri-variate polynomial volume defined by a lattice of control points that are capable of handling large-scale design deformations. FFD allows a significant boost in computer-driven design optimizations and

https://doi.org/10.1017/pds.2022.177 Published online by Cambridge University Press

shape modifications since the control point updates directly affect the nodes of the shape surface mesh and, thus, avoids a time-consuming re-meshing phase. However, for modifying complex geometries, this technique may require a large number of variables and if a CAD model of the resulting geometry is required, reverse engineering techniques have to be applied for recreating the CAD model.

Recent advancements in AI focus on data-driven deep learning-based methods to abstract information from prior data, which has been collected over several past product development cycles, for shape parameterization and to build surrogate models as design performance predictors. Among existing deep learning methods, the so-called generative models, which are used for automatically abstracting information of high dimensional domain without human user bias, generate an efficient low-dimensional latent manifold for shape modification. Two families of deep generative models were recently applied for shape parameterization: Generative adversarial networks (GANs) and (variational) autoencoder ((V)AE) models. Existing work on (V)AEs shows that the latent manifold learned by these models can not only be used for operations like data compression and classification tasks, but also for engineering optimizations in comparison with representations based on an FFD lattice (Rios *et al.*, 2019).

CAD software mainly represents geometries in a polygonal mesh form, from which several base representations can be derived, given their intrinsic amount of information. 3D point clouds are a simple and flexible representation, which is extracted from a mesh and consists of a list of nodes contained in the mesh and have been explored by researchers due to their suitability for spatial feature processing. Popular point cloud-based architectures are PointNet (Qi et al., 2017) and point cloud (variational) autoencoders (PC-(V)AEs), which are used to learn and transform high-dimensional 3D point clouds into compressed low-dimensional representations. Many researchers adopted the AE architecture because of its capability to address the permutation invariance of points in point clouds with point-wise operators followed by a global operator, e.g. *max-pooling*. AE architectures typically form a bottleneck layer that learns a compact representation of the input data, the so-called latent space or latent manifold. However, a lack of regularization in the latent space limits the shape generative capabilities of AEs.

Different from AEs, VAEs form a regularized continuous latent space to facilitate shape-generative capabilities (Saha *et al.*, 2020). In our present work, we use a point cloud VAE (PC-VAE) as proposed in (Saha *et al.*, 2020), which has been evaluated specifically for the application to engineering problems, such as design topology changes (Saha *et al.*, 2020), optimization tasks (Saha et al., 2021) and surrogate modelling (Saha *et al.*, 2021a). Our PC-VAE follows the same topology that of AEs but enforces the regularization of the latent space by adding the Kullback-Leibler (KL) divergence as a term in the training loss function of the network. Hence, by improving the distribution of the latent representation, the VAE extends its extrapolation and shape generative capability compared to AEs. Thus, in our proposed interactive framework, we intend to learn from existing shape data using a PC-VAE to produce a latent manifold for user interaction and data-driven guidance during two tasks: First, to provide guidance to explore the latent manifold to generate designs satisfying the designer's target in mind. Second, to train a multi-layer perceptron (MLP) as a surrogate model on the latent manifold for faster performance predictions of 3D designs during latent manifold explorations.

3. Methodology

To create an interactive system for assisting the decision-making process of the designer in the automotive design exploration phase, we first describe necessary preliminary steps for building such a framework as shown in (Figure 1.a). Based on (Figure 1.a) (Step 1), first, we describe the pre-processing step for sampling point cloud data from the CAE models (Section 3.1). Second (Step 2 and 3 in (Figure 1.a), we introduce the architectures of the data-driven deep learning models: A point cloud variational autoencoder (PC-VAE) for learning the point cloud representations of CAE data and the multi-layer perceptron (MLP) as a surrogate model for mapping the CAE data to their performance measures (Section 3.2). Finally, we integrate the trained PC-VAE into a graphical user interface (GUI) and introduce the workflow of our interactive framework (Section 3.3).

3.1. Pre-processing 3D Point Clouds

For 3D car models, we consider the polygonal meshes of the car class of ShapeNetCore (Chang et al., 2015), which comprises a large dataset of 3D shapes. We use the shrink-wrapping algorithm utilized in

https://doi.org/10.1017/pds.2022.177 Published online by Cambridge University Press

(Saha et al., 2021) to sample points uniformly from the external surfaces of the 3D car shapes and sample 3500 shapes from the car class, where each shape consists of 24578 points.
For the future task of training the MLP as a surrogate model with the performance measures of 3D designs, we consider two performance metrics of a 3D car shape: Shape volume (V) and aerodynamic drag coefficients (F_x). We selected the volume due to its low computational cost since it can be directly computed from the geometry, while the drag requires a higher computational effort and is highly non-linear with respect to the latent representation. The drag is calculated by performing computational fluid dynamics (CFD) simulations on the shrink-wrapped meshes using the OpenFOAM toolbox (https://www.openfoam.com). From 3500 shapes in the original data set, we considered a sub-set of the data with converged CFD calculations, which consisted of 600 shapes and performance measures.

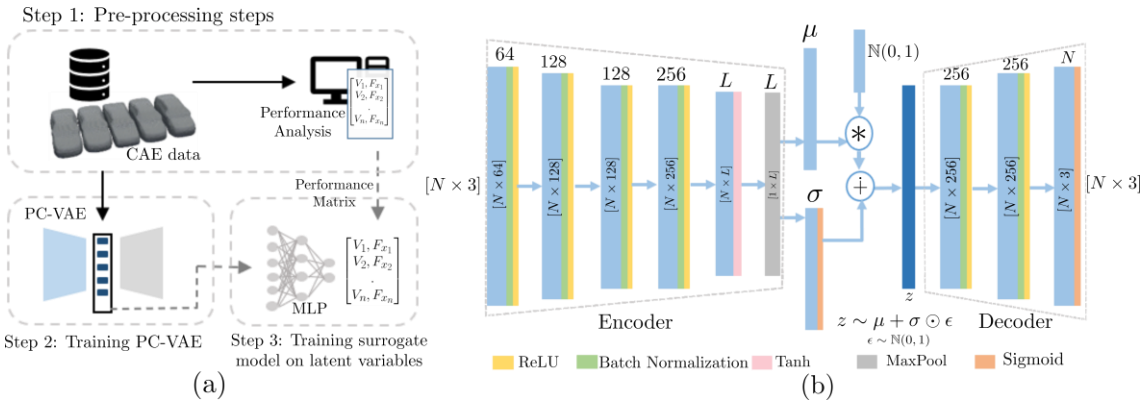

Figure 1. a) Pre-processing steps and workflow for training the PC-VAE and MLP b) PC-VAE architecture.

3.2. Learning the Point Could Variational Autoencoder

In step 2 of (Figure 1.a) we trained the PC-VAE proposed in (Saha et al., 2020). The detailed PC-VAE architecture is shown in (Figure 1.b) where the encoder comprises five 1D convolutional layers, followed by a max-pooling layer that is divided into two parts: A mean vector μ and a standard deviation vector σ with a sigmoid activation function (Figure 1.b). The latent representation z (Eq. 1) is sampled from the encoder's output distribution with mean μ and standard deviation σ, from which the decoder generates a 3D point cloud. The network is trained by minimizing two loss functions (Eq. 2), where $S_i, S_o \subset R^3$ are the input and output point clouds with points x_i, x_o respectively. The first term in the loss function is the reconstruction loss between the input and reconstructed point cloud calculated using the mean-square distance (MSD) as loss function (Eq. 3), and the second term is the KL-divergence loss between the learned latent distribution and an assumed prior normal distribution.

$$\mathrm{z} = [\mathrm{z_i}],\ z_\mathrm{i} \sim \mu_\mathrm{i} + \sigma_\mathrm{i} \odot \epsilon,\ \epsilon \sim N(0,1) \text{ and } \mathrm{i} = 1,2,..,5 \tag{1}$$

$$\mathcal{L}_{\mathrm{VAE}}(x_i, x_o) = \alpha\, L_{recon} + \beta\, \mathcal{D}_{KL}, \alpha = 1000, \beta = 0.001 \tag{2}$$

$$L_{recon}(MSD) = \frac{1}{N}\sum_{j=1}^{N} |x_{i,j} - x_{o,j}|_2^2 \tag{3}$$

We trained the PC-VAE architecture in (Figure 1.b) with a 5D latent vector using the Adam optimizer with a learning rate of $5E^{-03}$. The network was trained on a single GPU Nvidia®GeForce ®RTX8000 with 48 GB. Further, for training a surrogate model from the learned 600 latent representations z to their desired properties, here, volume and drag coefficients (Step 3 in Figure 1.a), we trained a 2-output multi-layer perceptron (MLP) to map the latent representations z to the two output variables: Normalized volume (V) and normalized drag coefficients (F_x). To train the MLP, we divided the 600 shapes with their performance metrics into 85% training and 15% test set. We trained the MLP on the training set and achieved an R-square (R2) score of 0.86 on the training set and 0.75 on the test set. Since training the MLP with 600 shapes show comparable R2 score for training and test set, we considered the trained MLP for perfromance prediction of 3D car shapes as adequate. However for improving the accuracy of

https://doi.org/10.1017/pds.2022.177 Published online by Cambridge University Press

the MLP, we augmented the performance data to improve prediction accuracy. For further details on the PC-VAE and the MLP, please refer to (Saha *et al.*, 2021a).

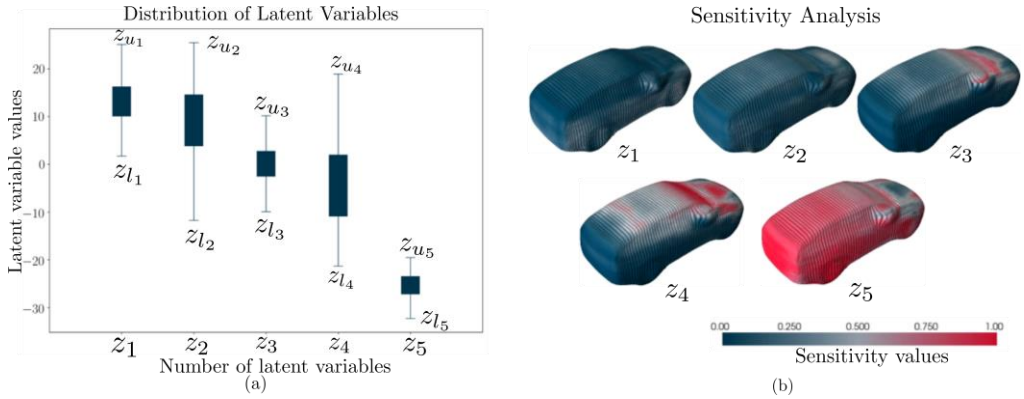

Figure 2. a) Distribution of latent variables of the training set shapes. b) Visualization of the displacement sensitivity of points in a 3D point cloud of the initial design (S_{in}) to the 5 latent variables.

In our application examples, we used the learned 5D latent variables as interactive elements for modifying 3D designs. To analyze the range of the distributions of the latent variables, we plotted a boxplot of the latent variables ($z_i, i = 1,2,..,5$) from the training data set in Figure 2.a). Ideally, we considered the lower ($z_{l_i}, i = 1,2,..5$) and upper limit ($z_{u_i}, i = 1,2,..5$) of each latent variable to determine the lower and upper limits (min and max values) of the sliders, which we use in the GUI (Section 3.3). We observed that the latent variables have values of different magnitudes, which is due to the latent values generated by sampling from the latent distribution of PC-VAE. Further, to analyze the impact of each latent variable modification on the generated shape, we performed a sensitivity analysis to determine the regions of the shapes mapped by each latent variable z_i. We calculated the total sensitivity of the displacement of each point of 30 point clouds sampled from the training dataset and estimated the association with each latent variable z_i. For that, we used the Sobol method with Satelli's sampling (Saltelli *et al.*, 2010), where samples are generated as 30(L+2) variations in a range of $\pm 30\%$ of each latent variable, considering L=5. Next, we calculated the sensitivity values of each point of the point cloud representation associated with each latent variable (z_i). In (Figure 2.b) we projected the sensitivities obtained from the 5 latent variables onto a 3D point cloud selected for the analyses. We observed that each latent variable z_i is associated with changes in larger and different regions in the output point cloud. However, the impact of change on the design topology of the output point cloud is different for each latent variable z_i, e.g, the sensitivity of displacement concerning z_5 is higher compared to other latent variables. The sensitivity analysis is presented to the user when the user starts the GUI interface.

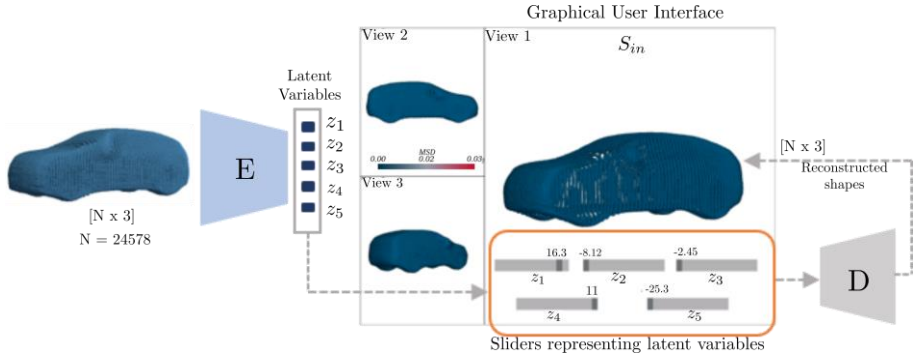

Figure 3. Schematic overview of the interactive framework.

In the GUI, interacting with the sliders updates the new values of the latent variables, and the PC-VAE decoder automatically calculates and displays the updated 3D car point cloud based on the modified 5 latent parameters. To support the human user in the decision-making process, the framework aims to

https://doi.org/10.1017/pds.2022.177 Published online by Cambridge University Press

provide support in two different ways for two different scenarios: First, the system aims to guide the designer for target-oriented design exploration in the latent manifold of the PC-VAE (Section 4.1). For this application, our framework suggests a personalized range of the latent variable sliders to generate a target design. Second, the system aims to provide performance predictions of the generated 3D design simultaneously, to fasten the separate process of design generation and expensive performance simulations (Section 4.2). Thus, for the second scenario, the framework provides feedback on the estimated performance measure of the generated 3D point cloud design by projecting the performance value onto the 3D point cloud representation as color maps (Figure 6). Note that depending on the type of application requested by the designer, the framework uses only the trained PC-VAE (Section 4.1) or both, the trained PC-VAE and the MLP as a surrogate model (Section 4.2).

4. Evaluation of the Interactive Framework

We performed two different experiments to illustrate the evaluation of our interface for the above-mentioned two scenarios. The initial setup for the experiments is the same, where at the beginning the GUI loads the initial design (S_{in}) in the form of a 3D point cloud representation with the current position of the sliders representing the latent variables of S_{in} (Figure 3).

4.1. Target-oriented Design Exploration

This experiment demonstrates the applicability of the framework for providing assistance in the form of personalized ranges of the latent variable sliders to guide the user for target-oriented design generation on the learned latent manifold of the PC-VAE (Section 3.3, first scenario). Often the designer in the concept generation phase has some idea of a potential design target. Here, we considered a design target to be defined by the user as one of the car classes from the following car classes: "SUV", "Sedan", "Convertibles" and "Sportscar" as provided with labels by the ShapeNetCore data set. For example, if the user's target is to explore sports car designs, the user chooses "Sportscar" as the target car class from the above-mentioned car classes. Based on the selection of a target class (Sportscar) we need to estimate the region (cluster) in the latent manifold of the PC-VAE that represents the target sportscar designs to provide a personalized range of the latent sliders to the users. The user modifies the personalized latent sliders for changing the initial design (S_{in}) to a 3D shape representing a sportscar design.

However, many shapes from ShapeNetCore car classes are not labeled correctly. Thus, we used the shape generative capability of the PC-VAE to generate a cluster of similar yet different sportscar shapes from the 10 correctly labeled sportscar templates selected from the ShapeNetCore car dataset. We considered 10 distinct sportscar templates from the dataset with the "Sportscar" label after visual verification since various categories of sportscar exist in the dataset. From the 10 chosen sportscar template shapes, we estimated the latent distribution of these 10 shapes ($\mu_j, \sigma_j, j = 1,2,..10$) using the encoder of PC-VAE and sampled 10 latent representation z (Eq. 1) from each of these 10 distributions (μ_j, σ_j). Thus, exploiting the generative capability of the PC-VAE assists in generating a cluster of 100 latent representations z for the sportscar class. We analyze the latent variables of these 100 samples representing different types of sportscar designs and estimate the lower and upper limits (min and max values) of these 100 latent variables ($z_i, i = 1,2,..,5$) to approximate the personalized ranges of the sliders in our framework. We use a similar template-based sampling approach to generate a cluster of 100 latent variables for other target classes (SUV, sedan, and convertibles).

In (Figure 4.b), we show the latent distributions of the 100 samples from the target sportscar cluster and the initial design (S_{in}) to determine the lower ($z_{l_i}, i = 1,2,..5$) and upper value (z_{u_i}) of each of the 5 sliders. Next to customize the slider ranges before starting the GUI interface, we consider the latent variables of S_{in} as one of the lower (z_{l_i}) or upper limit (z_{u_i}) of the slider and the other limit is determine from the latent variables of the target sportscar class (similar to the range of z_2, z_3, z_4 *and* z_5 in Figure 4.b). However, if a latent variable of S_{in} is within the range of the target class latent range (similar to the range of z_1 in Figure 4.b), we consider both the upper (z_{u_1}) and lower limits (z_{l_1}) of the target class as the personalized range of the slider z_1. Thus, based on the selection of a target car class by the user, the templates of the selected target class are used to determine the personalized ranges of the latent variables (Figure 5.a)

https://doi.org/10.1017/pds.2022.177 Published online by Cambridge University Press

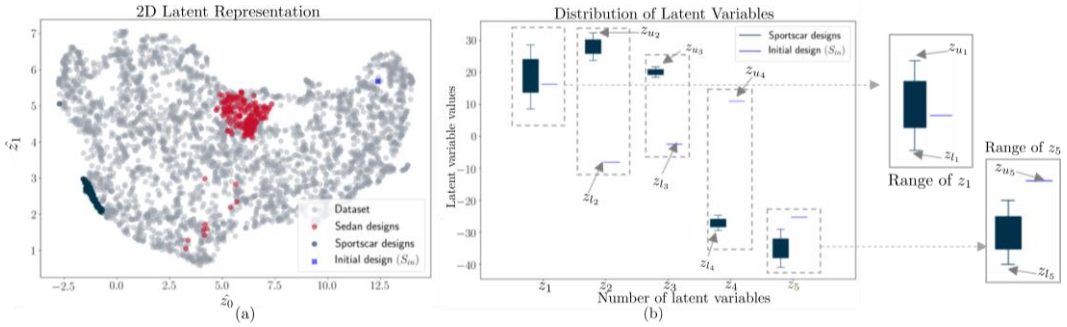

Figure 4. a) 2D representation of the 5D latent variables of the training dataset and the target car class shapes. b) Distribution of the latent variables of the initial design (S_{in}) and the selected target class templates.

In addition, to visualize the distinct locations of these target car class designs in the latent manifold, we embedded the 5D latent space of the training dataset and 100 shapes from each cluster of the two-car classes (sedan and sports car) into a 2D representation using UMAP (Figure 4.a). We observed that the designs from the two target classes are located in a distinct region of the latent space, showing that the reliability of the latent representation since the sedan designs are very different from sports car designs.

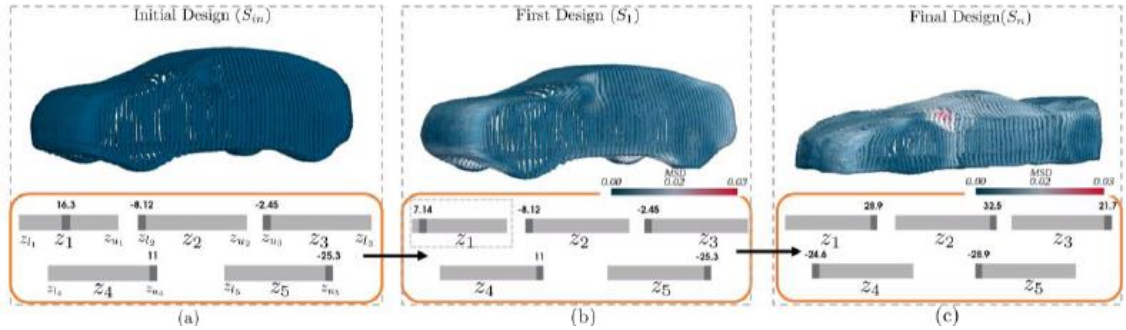

Figure 5. Visualization of the interactive framework with personalized slider ranges to change the design topology from the initial design to a sportscar design. a) Initial design with customized slider ranges derived from (Figure 4.b). b) First point cloud design (S_1) generated with z_1 latent variable modification. c) Final design (S_n) generated by moving the sliders.

The framework loads the initial design (S_{in}) with personalized slider ranges provided at the bottom of the GUI (Figure 5.a), where the current position of each slider reflects the value of the latent variables of S_{in}. The user is able to move each slider to generate new designs, where each design is automatically calculated by the PC-VAE decoder. The first design (S_1) is generated by changing only the latent variable z_1 from the initial design (S_{in}) (Figure 5.b). Modifying each latent variable modifies a region of the point cloud, and thus we measured the mean square distance (MSD) between the current and the previous design. The MSD value between each point of the current shape S_1 and the previous design S_{in} is projected on each point onto the current 3D point cloud representation S_1 as color maps (Figure 5.b) to make it comprehensible to the user to display the region of modification in the generated point cloud concerning the current latent variable modification. Likewise, after a n number of latent variable modifications the final design of the sports car class is generated (Figure 5.c). The color map in the final design (S_n) shows the MSD difference between the point in the shape S_n and previous shape S_{n-1} (not shown in the Figure). Other than visually verifying that the final design S_n is representing a sports car, we estimated the similarity of the final design with regards to the user target (selected 5 sports cars templates). We measured the pairwise similarity in the latent manifold between the final design and the selected templates from the sports car class and observed that these Euclidean distances are always lower than the maximum pairwise similarity between the sports car templates, demonstrating that the final generated shape (S_n) belongs to the sports car cluster. Thus, this example illustrates the usability of our framework for guiding design exploration by suggesting personalized latent variable slider ranges to navigate in the latent manifold for modifying the initial shape to a 3D shape representing the target class.

https://doi.org/10.1017/pds.2022.177 Published online by Cambridge University Press

4.2. Performance Predictions of 3D Designs

The second experiment demonstrates the applicability of our framework to provide real-time structural or functional performance estimation of the 3D designs by projecting the performance measure value onto each point of the 3D point cloud as color maps (Section 3.3, second scenario). (Figure 6) illustrates the workflow of the framework combining both, the trained PC-VAE and the MLP as a surrogate model. In our experiment, we considered the MLP (Section 3.2) trained on the latent manifold of the PC-VAE for predicting performance measures of 3D designs. The two outputs of the MLP are normalized volume (V) measures and normalized drag coefficients (F_x) between 0 and 1. Based on the selection of performance of interest by the designer, the pre-trained models provide a performance measure (V or F_x) feedback to the designer through our user interface (Figure 6).

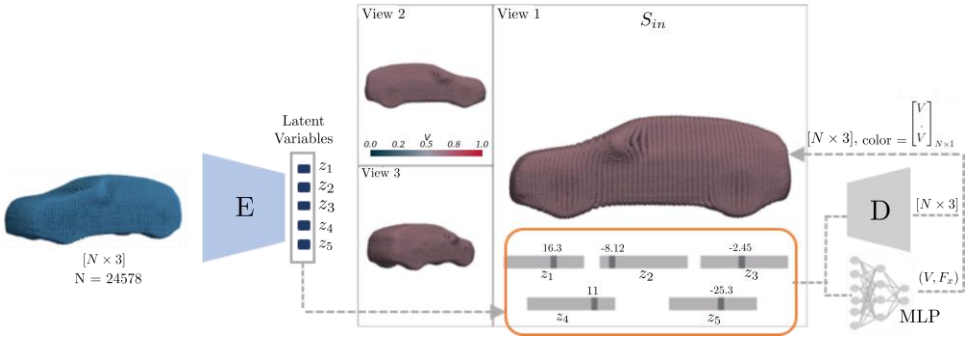

Figure 6. Schematic overview of the interactive framework for 3D shape volume prediction.

Additionally, to visualize the organization of the latent manifold concerning performance measures, we considered 600 shapes from the dataset along with their performance metrics. We embedded our 5D latent representation of 600 shapes into a 2D representation using UMAP (Figure 7) and observed that the 2D latent representation is organized with respect to the performance measures, showing that exploring the distinct regions in the latent manifold provides designs with target performance measures.

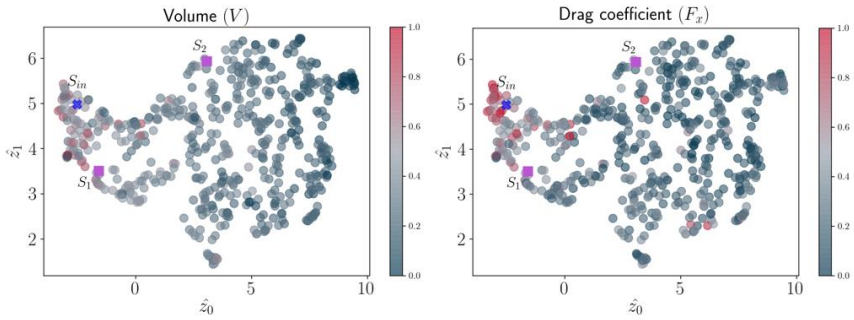

Figure 7. 2D latent representation of 3D car shapes organized with respect to two performance measures: a) Volume and b) Drag coefficient.

After starting the GUI interface, the user observes the point cloud representation of the initial chosen design (S_{in}), in which the normalized volume (V) is projected as color-map, along with sliders at the bottom of the GUI, which represent the latent variables of S_{in}. To illustrate the assistance of our framework not only for faster shape exploration on the latent manifold but also to predict the performances of the 3D designs from the latent manifold, we modify only one latent variable (z_2) of the initial design (S_{in}) gradually to generate the updated shape S_1 and S_2 (Figure 8). We observe the maximum shrink in the back of the car roof in S_1 and S_2 (Figure 8. b) and c), indicating that the volume of S_1 reduced from S_{in}, with further reduction in S_2. We visualized the position of the 3 shapes (S_{in}, S_1 and S_2) in the 2D latent representation (Figure 7.a), and observed that all 3 shapes are in distinct regions in the 2D latent representation verifying the difference in volume color maps for the shapes. Thus, based on the real-time performance feedback, the designer can explore any region of the latent manifold to

https://doi.org/10.1017/pds.2022.177 Published online by Cambridge University Press

achieve a targeted performance design, which indicates that the system not only predicts the performance of the learned dataset but also predicts the performances of extrapolated designs from the latent manifold.

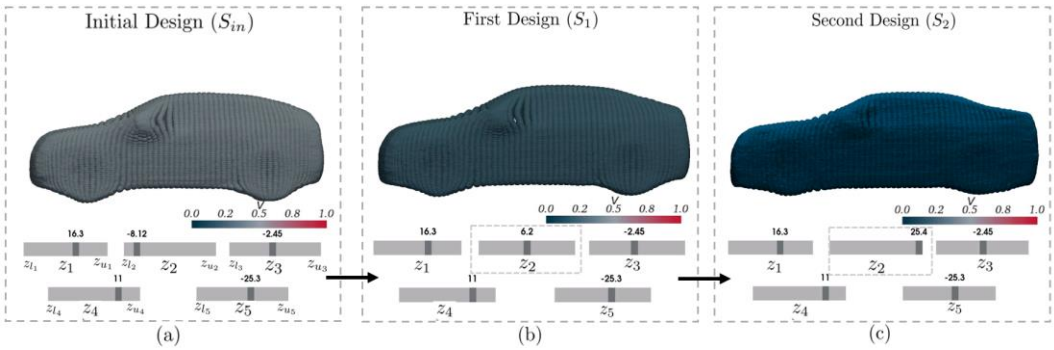

Figure 8. Visualization of the interactive framework for predicting volume (V) value and its projecting onto the 3D point cloud representation as color-maps: a) Initial point cloud design S_{in} with V= 0.58. b) First design S_1 generated by moving the latent variable z_2, with V = 0.4. c) Second design S_2 generated by further moving the latent variable z_2, with V=0.28.

We also evaluated the framework for drag coefficient estimation (F_x), which has higher significance for critical decision-making in computational design studies. (Figure 9) shows the initial design (S_{in}) color-coded with normalized drag (F_x), and S_1 and S_2 generated by modifying only the z_2 latent variable, while keeping other latent variables similar to the initial design (S_{in}). Additionally, to verify the accuracy of the MLP, we calculated the R-square (R2) score of the true and predicted volume (V) and drag coefficient (F_x) measures for shapes (S_{in}, S_1 and S_2). We observed a higher R2 score for volume (0.82) compared to that of drag estimation (0.70), suggesting that volume estimation is an easier task compared to predicting non-linear drag coefficient, which is in line with our finding in (Saha *et al.*, 2021a). Thus, this experiment illustrates the usability of our framework for performance-guided design exploration for generating shapes with target performance measures.

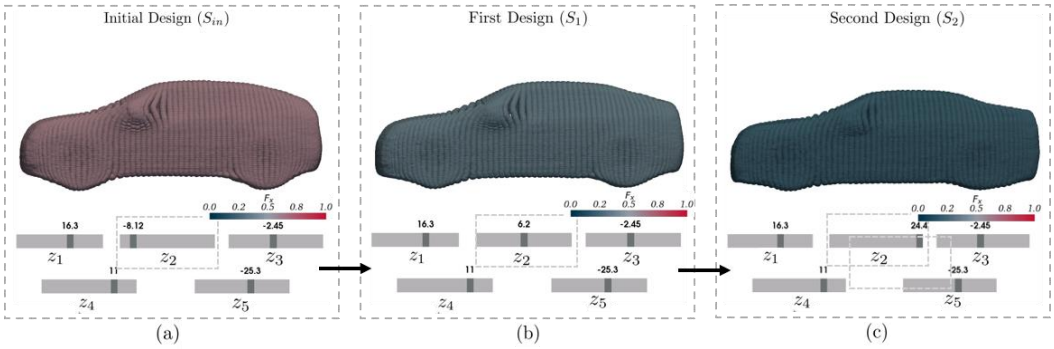

Figure 9. Visualization of the interactive framework for predicting normalized aerodynamic drag (F_x) value and projecting onto the 3D point cloud representation as color-maps: a) Initial point cloud design S_{in} with F_x= 0.72. b) First design S_1 generated by moving the latent variable z_2, estimating F_x = 0.51. c) Second design S_2 generated by further moving the latent variable z_2, estimating F_x = 0.38.

5. Conclusion

In automotive design development, CAD/E tools provide a means for altering shapes and simulating performance measures of 3D designs. However, these tools do not follow a data-driven approach to assist and guide the designer in a digital design development process. In this paper, we address this challenge by proposing a basic data-driven-based cooperative design system (CDS) that serves as an assistance system for the designers to explore design ideas. We use the advantage of data-driven generative models to create a low-dimensional data-manifold (latent manifold) for easy explorations

https://doi.org/10.1017/pds.2022.177 Published online by Cambridge University Press

of 3D designs through our proposed framework. Further, the CDS provides assistance in two forms: First, it provides direction indications for exploring the latent manifold to generate target-oriented designs. Second, it predicts performance measures of the generated shape based on the position in the latent manifold, enabling the designers to explore designs taking into account the estimated performances of different domains. However, it is not necessary to follow the recommendation at each step of the design modification and the designer can explore any region of the latent manifold.

Thus, the present results show a promising step toward using geometric deep learning-based systems as an additional tool in the automotive design exploration phase, which provides the benefit of combining knowledge from data-driven AI models and human experience. However, there are open challenges that need to be addressed. From a representation point of view, designers may need to recreate CAD/E models from the point cloud representation which is a reverse engineering process, e.g. through shrink-wrapping methods, but still may require manual work. Also, it is important to consider that data-driven methods rely on existing past data and as a consequence limit the space of exploration. Here, transfer learning methods may offer further options to allow for optimizing in a wider design search space.

Acknowledgement

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

References

Achlioptas, P., Diamanti, O., Mitliagkas, I. and Guibas, L. (2018), "Learning representations and generative models for 3d point clouds", *International Conference on Machine Learning (ICML)*, Vol. 80, pp. 40–49. https://arxiv.org/abs/1707.02392.

Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., et al. (2015), ShapeNet: An Information-Rich 3D Model Repository, available at: http://arxiv.org/abs/1512.03012.

Ha, D. and Eck, D. (2018), "A Neural Representation of Sketch Drawings", *6th International Conference on Learning Representations, ICLR*. https://arxiv.org/abs/1704.03477.

Lee, Y.J., Zitnick, C.L. and Cohen, M.F. (2011), "ShadowDraw: Real-time user guidance for freehand drawing", *ACM Transactions on Graphics*, Vol. 30 No. 4, pp. 1–10. https://doi.org/10.1145/2010324.1964922.

Qi, C.R., Su, H., Mo, K. and Guibas, L.J. (2017), "PointNet: Deep learning on point sets for 3D classification and segmentation", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 652–660. https://arxiv.org/abs/1612.00593.

Rios, T., Bäck, T., van Stein, B., Sendhoff, B. and Menzel, S. (2019), "On the Efficiency of a Point Cloud Autoencoder as a Geometric Representation for Shape Optimization", *IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 791–798. https://doi.org/10.1109/SSCI44817.2019.9003161.

Saha, S., Menzel, S., Minku, L.L., Yao, X., Sendhoff, B. and Wollstadt, P. (2020), "Quantifying The Generative Capabilities Of Variational Autoencoders For 3D Car Point Clouds", *IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1469–1477. https://doi.org/10.1109/SSCI47803.2020.9308513.

Saha, S., Minku, L.L., Yao, X., Sendhoff, B. and Menzel, S. (2021a), "Exploiting Linear Interpolation of Variational Autoencoders for Satisfying Preferences in Evolutionary Design Optimization", *IEEE Congress on Evolutionary Computation (CEC)*, pp. 1767–1776. https://doi.org/10.1109/CEC45853.2021.9504772.

Saha, S., Rios, T., Minku, L.L., van Stein, B., Wollstadt, P., Yao, X., Bäck, T., Sendhoff, B. and Menzel, S. (2021b), "Exploiting Generative Models for Performance Predictions of 3D Car Designs", *IEEE Symposium Series on Computational Intelligence (SSCI)*. pp. 1-9. https://doi.org/10.1109/SSCI50451.2021.9660034.

Saltelli, A., Annoni, P., Azzini, I., Campolongo, F., Ratto, M. and Tarantola, S. (2010), "Variance based sensitivity analysis of model output. Design and estimator for the total sensitivity index", *Computer Physics Communications*, Elsevier B.V., Vol. 181 No. 2, pp. 259–270. https://doi.org/10.1016/j.cpc.2009.09.018.

Sederberg, T.W. and Parry, S.R. (1986), "Free-form deformation of solid geometric models", *ACM Special Interest Group on Computer Graphics and Interactive Techniques*, Vol. 20 No. 4, pp. 151–160. https://doi.org/10.1145/15886.15903.

Umetani, N. (2017), "Exploring generative 3D shapes using autoencoder networks", *SIGGRAPH Asia 2017 Technical Briefs*, Vol. 4, pp. 1–4. https://dl.acm.org/doi/10.1145/3145749.3145758.