ISSN 2464-4617 (print)
ISSN 2464-4625 (online)
Computer Science Research Notes - CSRN
http://www.wscg.eu
WSCG 2025 Proceedings

An Inception-based Variational Autoencoder for Curves Generation and Interpolation

Saeedeh Barzegar Khalilsaraei
Graz University of Technology
Inffeldgasse 16c
8010 Graz, Austria
s.barzegarkhalilsaraei@tugraz.at

Ursula Augsdörfer
Graz University of Technology
Inffeldgasse 16c
8010 Graz, Austria
augsdoerfer@tugraz.at

ABSTRACT

In this work, we introduce a novel approach for synthesizing new curves by leveraging Variational Autoencoder (VAE) latent space interpolation. Our method encodes existing ordered point sequences representing curves into a compact latent representation, enabling smooth and meaningful transitions between different curve shapes. By performing controlled interpolations in the learned latent space, we generate diverse, high-quality smooth curves that maintain structural coherence and geometric consistency. The proposed method is particularly useful for applications in shape design, procedural modeling, and data augmentation in geometric learning.

Keywords

Shape synthesis, Variational autoencoders, B-spline curves, Generative design

1 INTRODUCTION

Curve generation plays a fundamental role in numerous fields, including computer-aided design (CAD), animation, computational geometry, and data-driven shape synthesis [Ju2005, Wang2020]. From designing smooth, aesthetically pleasing contours in industrial design to generating natural-looking trajectories in robotics and motion planning, the ability to create novel curves from existing data is essential. In a typical design pipeline, for example, in the automotive industry, the designer creates an initial clay model of the car body. Using scanner devices, the model is digitized, and different techniques are applied to extract the curves and surfaces of the model for further manipulation and design purposes. Exploring some variations of an existing model while preserving its key features is a common aspect of the design process [de2020]. In addition, Deep Neural Networks (DNN) have been applied to process 2D curves for modeling, optimization and reconstruction [Laube2018, Gao2019, Barzegar2024]. However, the quality of results are highly dependent on good quality training data, which is often not available.

Traditionally, curve synthesis relies on parametric representations such as Bézier curves, B-splines, or procedural methods, which require explicit control points and manual adjustments. This human design process is also time-consuming and tedious. In addition, while these techniques offer precision, they often lack flexibility in generating diverse shapes. Integrating deep learning methods facilitates automated shape exploration by reducing its computational complexity and minimizing the need for human involvement [Regenwetter2022].

Recent advances in machine learning, particularly deep generative models, offer a promising alternative for curve synthesis by learning underlying shape distributions from existing data. Generative models such as Diffusion models [ho2020], Variational Autoencoders (VAE) [kingma2013] and Generative Adversarial Networks (GAN) [Goodfellow2014] have shown promising results in geometric deep learning for 2D and 3D shape generation [Hui2022, Koo2023, Chen2018], reconstruction and completion [Mittal2022, Cheng2023] and are frequently explored in automating design exploration.

Generative models are capable of extracting features and patterns from data, learning a meaningful representation, and generating new samples with similar characteristics. VAEs in particular have proven effective in capturing compact latent representations of complex geometric structures, enabling smooth and meaningful interpolations.

In this work, we propose a novel approach for generating new curves from ordered point sets using VAE latent space interpolation. By encoding curves into a continuous, low-dimensional space, our method facilitates controlled shape transformations, allowing for the

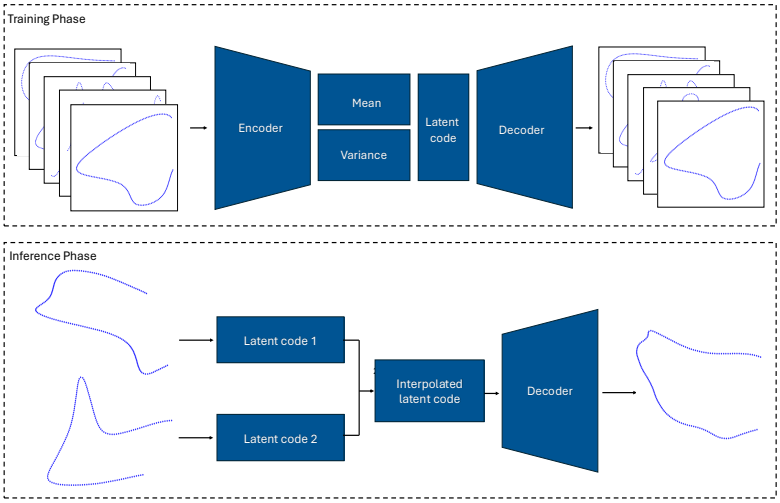


Figure 1: The pipeline of the proposed method: In the training phase, ordered points sampled from smooth 2D cubic B-spline curves are fed to a Variational Autoencoder to learn the data distribution and generate reconstructed curves. At inference, arbitrary curves are fed to the network to extract their latent vector. The latent vectors are linearly interpolated and fed to the decoder to synthesize new curves.

synthesis of realistic and structurally coherent curves. In this paper, we investigate the latent space interpolation within a VAE trained on 2D curve data. We demonstrate the effectiveness of our method for generating high-quality interpolated curves that maintain key geometric properties. Our contributions include (1) a novel pipeline for encoding and interpolating curves in a learned latent space, (2) a framework for generating smooth and diverse curves with minimal user input and (3) a coarse and continuous latent representation which enables manipulation for design exploration.

The remainder of this paper is structured as follows: Section 2 provides an overview of the latest related works. Section 3 presents the proposed method in detail, including the neural network architecture and interpolation process. The evaluation of the proposed method is provided in Section 4 followed by the conclusion in Section 5.

2 RELATED WORKS

Generative models, Generative Adversarial Networks (GAN) in particular, are ideal when employed for curve synthesis. Every GAN contains two components trained in a competitive manner: a generator to generate new examples and a discriminator which is a classifier to recognize real samples from the generated fake ones. After training, the discriminator will be discarded and the generator will be utilized for new data generation.

There have been numerous works on using GANs to automate the shape exploration of curves describing airfoils: BézierGAN [Chen2018] is a Generative Adversarial Network where the generator produces the control points, weights, and parameter variables of rational Bézier curves. A custom Bézier layer is then applied to derive the discrete points on a 2D curve from the predicted parameters. Their method allows for meaningful interpolation between different shapes to explore a design space. In their paper they focus on the design of airfoils. An extension of BézierGAN is the BSplineGAN [Du2020], where the Bézier layer is replaced with a B-spline parameterization layer in the generator. These techniques are mostly applied in aerodynamic product design, to enable fast and interactive shape exploration before assessing the aerodynamic properties of a shape. The adversarial scheme causes sensitivity to parameter changes in GAN, which leads to mode collapse and makes it difficult to find a balance between the generator and discriminator [Creswell2018, Gonog2019]. Tan et al. [Tan2022] present a conditional GAN for airfoil shape optimization. They smooth the results using the Savitzky-Golay algorithm [Press1990] and B-spline interpolation. The manual parameter selection limits the generalization capability of this approach. Yonekura and Suzuki [Yonekura2021] propose a conditional VAE for 2D curve generation. Again, their focus is on design space exploration of airfoils. Wang et al. [Wang2023] applied a hybrid generative model, VAEGAN [Larsen2016] to take advantage of both VAE and GAN for generating and optimizing airfoil shapes. With VAE, they learn a continuous and structured latent vector, followed by a GAN to generate realistic output. Wada et al. [Wada2024] introduce a new airfoil shape synthesis approach which combines GAN with external physic equations. Their approach can create new shapes that are beyond the training set. NURBS-OT [Yang2025] is a generative curve model which integrates an optimal transform distance

[Villani2009] technique with a lightweight neural network to learn NURBS parameterization. Due to the complexity of NURBS, their model encounters training instability.

Recent works have used generative models to augment training data. Kim et al. [Kim2021] propose a time series augmentation technique using an autoencoder and adversarial augmentation method. Chen et al. [Chen2021] introduce a cooperative training with latent space masking-based data augmentation for medical image segmentation. LatentAugment [Tronchin2023] is a data augmentation method to explore the latent space for enhancing the diversity of images generated by GANs.

In contrast, we are aiming to generate a general dataset with comparatively large shape variations.

3 METHODOLOGY

We design a VAE to learn a continuous and coarse representation, referred to as the latent representation, from samples taken from 2D B-spline curves. We then apply interpolation over the latent representation to create new polygonal curves that have similar characteristics.

The pipeline of the proposed method is shown in Figure 1. In the training phase, the input is discrete set of 2D points sampled from open cubic B-spline curves. The network is a Variational Autoencoder, where the encoder part includes convolutional layers and inception modules [Szegedy2015] to learn a coarse and continuous representation from the input data. The decoder is the counterpart of the encoder and reconstructs the input data from the latent representation. The system is trained to output the same points along a curve as the input. If the output is an accurate reconstruction of the input curve, it demonstrates that the encoding is good and that the encoding part produces a latent vector which can be considered a good representative of the input curve. At inference, our trained model can produce a representative latent code in two parts: a mean distribution and a standard deviation of the input points samples. We synthesize new curves by interpolating the mean distributions derived by the encoding of the VAE network.

3.1 Dataset

For training a neural network, the first step is providing a sufficient dataset. We utilize the approach described in [Barzegar2024] to generate 2D open cubic B-spline curves with 10 control points. To generate random control points, we consider a circle with 64 radius in the center of a 2D plane of size 128×128. From the zero angle, the circle is divided into ten non-uniform regions and in each region a random control point is generated. Using the generated control points, we derive open cubic B-spline curves. The input data is derived by uniformly sampling from 70000 planar generated open, uniform B-spline curves without intersections. Each curve data created in this way contains 161 ordered sample points.

3.2 VAE training

Variational Autoencoders are a type of generative models that are designed to learn the probability distribution from a dataset and generate new examples with similar characteristics. Like any autoencoder, it has an encoder-decoder architecture. The encoder maps the input data into a continuous lower-dimensional latent vector. Unlike standard autoencoders, VAEs enforce a structured latent space by ensuring the latent variables follow a known probability distribution. This makes its learned latent space more meaningful.

The encoder is responsible for extracting features and providing probability distributions over the latent attributes from input data c. Instead of directly outputting a latent vector z, the encoder predicts a mean $\mu(c)$ and a variance $\sigma(c)$, defining a distribution over possible latent vectors. A latent vector z is sampled from the outputs of the encoder

$$z = \mu(c) + \sigma(c) \cdot \varepsilon. \tag{1}$$

where ε is taken from the standard normal distribution, $\mathcal{N}(0,1)$. The latent vector z is the input to the decoder.

The sample from the latent vector distribution is the input of the decoder. The continuous latent vector enables meaningful interpolation and the generation of new data. The decoder output $\tilde{c}$ is the reconstructed version of the input data:

$$z \sim \text{Enc}(c) = q(z \mid c), \quad \tilde{c} \sim \text{Dec}(z) = p(\tilde{c} \mid z). \tag{2}$$

$q(z \mid c)$ is the distribution of latent vector z given the 2D curve c and $p(\tilde{c} \mid z)$ is the distribution of reconstructed curve $\tilde{c}$ given z.

3.3 The Loss Function

The loss function in a neural network computes the difference between target and predicted outputs and serves as a guide for optimizing the parameters of the network.

The VAE is trained using a loss that consists of two parts: a reconstruction loss (4) and a regularization loss (5). The reconstruction loss ensures that the reconstructed output, $\tilde{c}$, is similar to the original input c and is given by the mean squared difference between the input curve data c and the reconstructed curve $\tilde{c}$. The regularization loss is Kullback-Leibler (KL) divergence loss which ensures that the latent distribution produced by

the encoder is similar to a standard normal distribution. The loss function, (3), is given by:

$$\mathscr{L}_{\text{VAE}} = \mathscr{L}_{\text{MSE}} + \mathscr{L}_{\text{KL}}. \tag{3}$$

$$\mathscr{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(c_i - \tilde{c}_i)^2. \tag{4}$$

$$\mathscr{L}_{\text{KL}} = D_{\text{KL}}(q(z \mid c) \parallel p(z)). \tag{5}$$

The output of the proposed VAE should either be utilized as synthetic data to train deep learning models on data sampled from smooth curves or in context of design exploration. In both cases, the generated curves should be smooth and free from noise. For some test examples, the model prediction is not as smooth as the input curve. Therefore, we introduce a new term in the loss function (3) to ensure the predicted curve also has the same second-order derivative properties as the input curve. The Laplacian loss, (7), is given by the mean absolute difference between the second-order derivatives of the input curve c and the reconstructed curve $\tilde{c}$. Figure 2 shows the prediction of two VAEs, namely with and without Laplacian loss.

The final loss function for the proposed VAE is given by:

$$\mathscr{L}_{\text{VAE}} = \mathscr{L}_{\text{MSE}} + \mathscr{L}_{\text{KL}} + \mathscr{L}_{\text{Laplacian}}. \tag{6}$$

$$\mathscr{L}_{\text{Laplacian}} = \sum_{i} \left|\text{Lap}(c_i) - \text{Lap}(\tilde{c}_i)\right|. \tag{7}$$

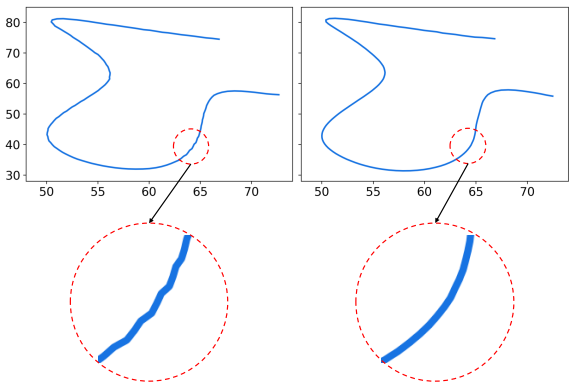


Figure 2: The reconstruction result of VAE without (left side) and with (right side) Laplacian loss function.

3.4 Smoothness

The output of our proposed VAE is a discrete set of 2D points that lie on a curve resembling a B-spline curve in style but does not exhibit the same smoothness of a B-spline curve. The Laplacian term in the loss function guarantees the smoothness of the output curves. We evaluate the smoothness of the generated curves via analyzing the tangent vector behavior. To calculate the unit tangent vectors, we use the following equation where p denotes the points lie on the curve:

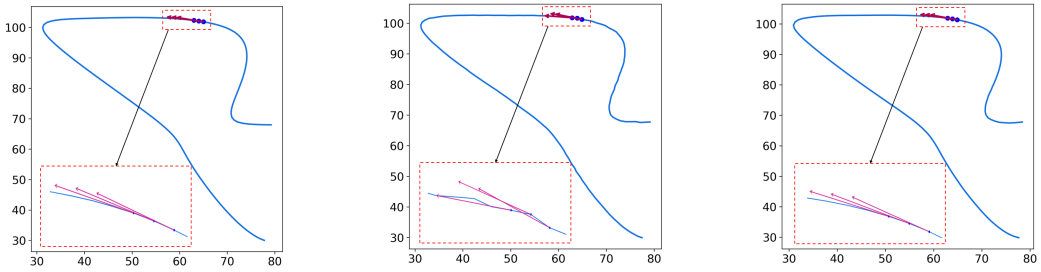
$$t_i = \frac{p_{i+1} - p_i}{\|p_{i+1} - p_i\|}. \tag{8}$$

In a smooth curve, the tangent direction changes gradually while in a noisy curve it might show abrupt changes. We demonstrate the tangent direction of some consecutive points of the smooth input curve and the VAE output with and without Laplacian term in Figure 3. The tangent vectors are visualized as arrows at selected points along the curve. Compared to the VAE-generated curve without Laplacian term, the tangent vectors on the VAE-generated curve with Laplacian term exhibit smoother directional transitions.

3.5 Investigating VAE architectures

We did numerous experiments to develop an appropriate VAE architecture. The ordered set of points sampled from 2D cubic B-spline curves is the input to the VAE. Convolutional Neural Networks (CNN) are a good choice for processing such spatially structured data [Komarichev2019]. The experiments are started with a shallow network (CNN_1), including two convolutional layers with kernel sizes 3 in the encoder part. Activation functions introduce nonlinearity in the neural networks, which allows them to learn complicated patterns and solve sophisticated tasks. The Rectified Linear Unit (ReLU) activation function is utilized in all the convolutional layers. A BatchNormalization layer [Ioffe2015] is included to stabilize and facilitate the learning process, followed by a max pooling layer to reduce the dimension of extracted feature maps. In a separate experiment, the kernel sizes are changed from 3 to 5 (CNN_2). It improves the performance of the network, however, using a mix of both kernel sizes (CNN_3), 5 for the first convolutional layer and 3 for the second one, has further improved the performance.

The shallow networks are able to capture the overall shape of the curve, however, they show poor performance in regions with high curvature. Increasing the number of convolutional kernels or layers is a promising way to improve the performance of a neural network [Szegedy2016]. In two separate experiments, we double the number of convolution kernels (CNN_4) and increase the number of convolutional layers from two to three (CNN_5). Increasing the number of convolutional layers demonstrates a greater improvement compared to increasing the number of convolutional kernels. The reconstruction still suffered distortions, however, adding more convolutional layers followed by max pooling layers lead to negative dimensions. Inspired by InceptCurves [Barzegar2024], we introduced inception modules between convolutional layers to improve the performance of the proposed VAE (CNN_6).



(a) (b) (c)

Figure 3: The tangent direction of selected points on (a) the original input curve, (b) the output of the VAE without Laplacian loss function and (c) the output of VAE with Laplacian loss function.

In an inception module, convolutional layers with different kernel sizes extract features from the same input in parallel and concatenate the results to feed to the next layer, see Figure 4. It enables the network to extract multi-scale features concurrently and increase the representational power of the encoder.

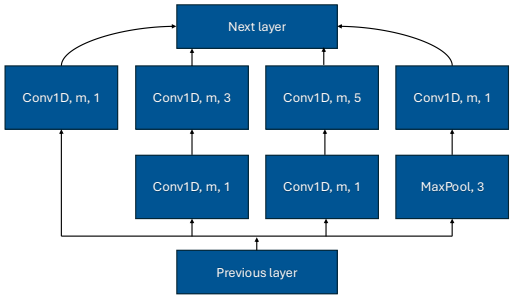


Figure 4: Inception module. The number of convolutional filters is denoted by m.

Table 1 presents the evaluation of the above-mentioned networks on a test set including 1000 open cubic B-spline curves using mean squared difference between target and predicted curves.

Network	Average MSE
CNN_1	1.018
CNN_2	0.696
CNN_3	0.549
CNN_4	0.364
CNN_5	0.250
CNN_6	0.099

Table 1: The performance of different VAE architectures are analyzed for curve reconstruction.

The reconstruction error, Equation (4), is averaged over 1000 curve reconstructions and is shown to reduce with adding complexity to the network. Using the inception module in the architecture reduces the reconstruction loss considerably. The inception-based VAE is able to capture the overall shape of the curve and learn the fine details in the input curve.

The size of the latent code will also influence the reconstruction accuracy of the proposed inception-based VAE. We tried different latent dimensions, such as 8, 16, 32, and 64. Increasing the size of the latent code improves the reconstruction of the input curve. Table 2 presents the evaluation of VAEs with different latent code sizes using the average mean squared error between target and predicted curves on a test set consisting of 1000 examples. We set the size to 32 to have a balance between reconstruction accuracy and computational complexity.

Latent code size	Average MSE
8	12.075
16	0.670
32	0.099
64	0.090

Table 2: The mean squared reconstruction error for VAE with different latent code sizes.

The architecture of this network is depicted in Figure 5. In the encoder part, three convolutional layers and two inception modules are employed. The number of kernels in the first inception module is 64, while the second inception module contains 128 kernels. The decoder has a similar architecture, except in the beginning, a dense layer is employed to transform the low-dimensional latent representation into a higher-dimensional feature space.

For training this network, the gradient-based Adam optimizer [Kingma2014] with a learning rate of 10^{-3} is utilized. The batch size is 128, and the network is trained for 900 epochs. All the experiments are performed using Tensorflow on a Linux server equipped with a GeForce RTX 4090 GPU.

Figure 6, shows two different test curves with their corresponding reconstruction via our proposed method. Below each figure the mean squared difference between target and predicted curves is calculated. With the Laplacian smoothing loss function, this network is able to generate a smooth curve as output. It can also pre-

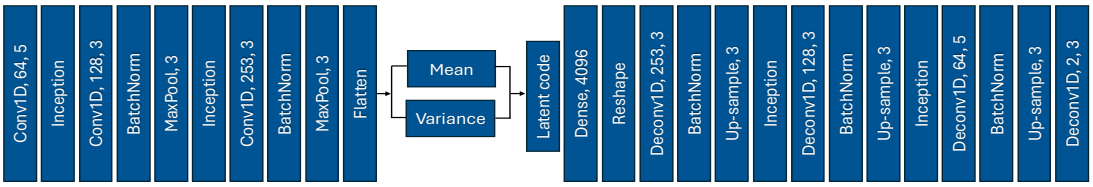
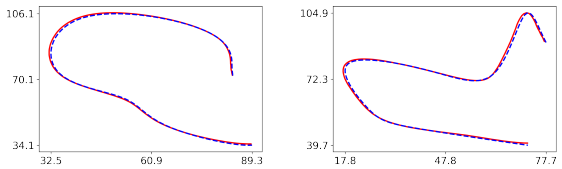


Figure 5: The architecture of the proposed network. In each convolutional layer, the first number is the number of convolution filters and the second is the size of filters. The architecture of the inception module is presented in Figure 4.

serve the overall shape of the input curve and reconstruct it accurately.



(a) Error: 0.333 (b) Error: 0.366

Figure 6: Reconstructing two different curves with the proposed VAE. The target curve is represented in blue dashed line and the prediction of the network is depicted in red.

3.6 Interpolation

At inference, we consider interpolation between the latent vectors to generate new curve data.

Curve data sampled from two different B-spline curves are mapped to their respective mean distributions μ_1 and μ_2, using the encoder. A new latent vector $\tilde{\mu}$ is generated via an affine combination of μ_1 and μ_2:

$$\tilde{\mu} = w\mu_1 + (1-w)\mu_2. \tag{9}$$

where w is a weight controlling the influence of μ_1 and μ_2. For interpolation, we set the w to be in the interval $[0,1]$. This technique is generalizable to the triplet case, where the latent vectors of three curves are extracted and interpolated:

$$\tilde{\mu} = w_1\mu_1 + w_2\mu_2 + w_3\mu_3. \tag{10}$$

where $w_1 + w_2 + w_3 = 1$.

By interpolating latent means and feeding the newly generated latent representation to the decoder, it is possible to generate diverse and meaningful variations of existing 2D curve data. Thus, we can expand an existing dataset with realistic and new curves to avoid overfitting and improve the generalization capability of a deep learning model.

4 RESULTS

Our proposed VAE maps planar curve data to a low-dimensional space, the latent feature vectors, via the encoder. The curve can be reconstructed from the low-dimensional latent vector via the decoder.

The encoder encodes the training samples to an informative latent space, where the nearby points in this latent space are geometrically similar [Roberts2018]. Thus, the distance between points shows the difference of shapes in the dataset. t-distributed Stochastic Neighbor Embedding (t-SNE) is a dimensionality reduction technique to map the high-dimensional data to a 2D or 3D space while preserving the similarity between points. We use t-SNE [Van2008] to visualize the latent space in a lower dimension. In Figure 7, we show the t-SNE visualization of latent means from the training set. On the top row, some examples are provided to illustrate close points in this latent space correspond to curves with similar shapes. In the bottom, we illustrate two interpolations where in the left side, the latent means of the corresponding input curves are close in the latent space which leads to a smooth interpolation between these curves. On the right side, the latent means in the latent space are far from each other which deteriorate the interpolation results. The interpolated curves might have some features that are not existed in the input curves.

In order to synthesize a new curve, curve data sampled from two 2D open cubic B-spline curves are fed to the encoder and their corresponding mean latent vectors are extracted. We linearly interpolate between mean latent vectors of two smooth B-spline curves using Equation (9). Our objective is to generate 2D polygonal curves that preserve the geometric properties. The interpolated mean vectors are fed to the decoder. Figure 11 illustrates latent vector interpolation of two arbitrary curves and the new generated curves. We interpolate the latent vectors in five steps, but this number can be adjusted as desired. Figure 8 shows an example of triplet interpolation.

If the synthetic data is required in form of B-splines, we feed the generated curves to another data-driven approach, such as InceptCurves [Barzegar2024], to derive a coarse mathematical representation of the new generated curves and utilize it in the design pipeline for real-world object creation or manipulation. Because the newly generated synthetic curves closely resemble B-splines, the conversion yields good results. In Figure

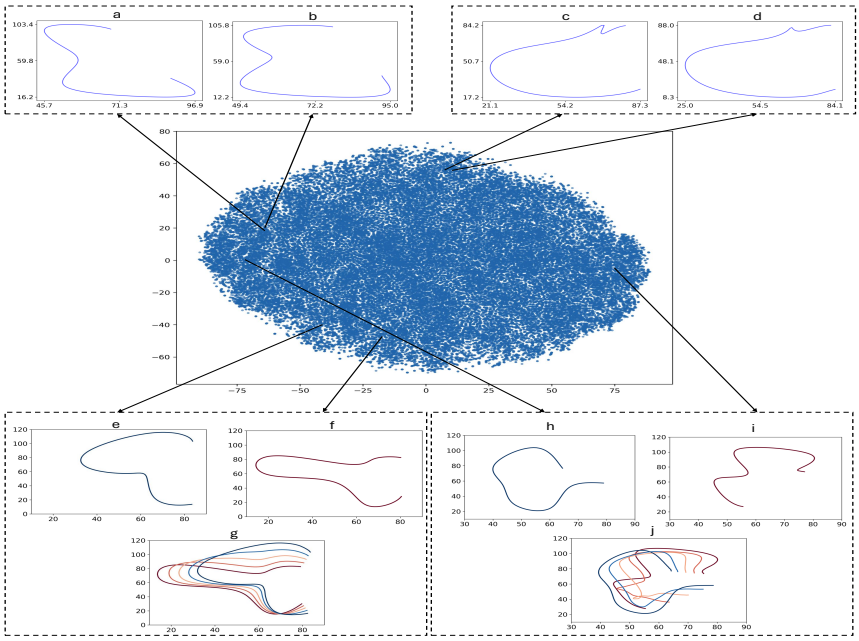
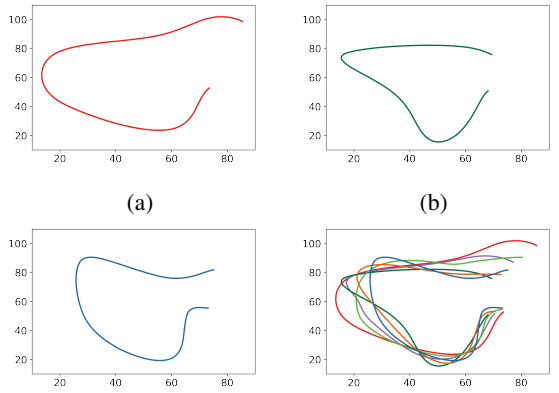


Figure 7: The latent space of our proposed VAE is visualized using t-SNE algorithm. Some examples on the top shows that close points in the latent space correspond to curves with similar shapes. The bottom shows the interpolation of two close (left) and far (right) points in the latent space.

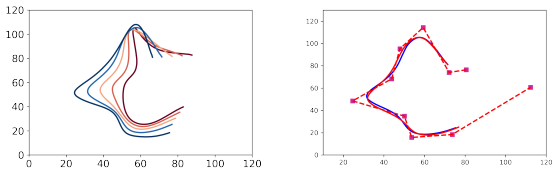


(a) (b) (c) (d)

Figure 8: Three input curves (a), (b) and (c) are interpolated (d).

9, new curves are generated with our proposed VAE (a) and fed to IncepCurves to produce a coarse polygon (b) for each new generated curve.

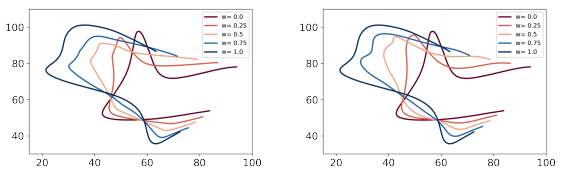
There are different techniques for interpolation which leads to different augmentation. A naive approach is to interpolate two input curves data points. However, latent interpolation yields results which are more smooth. Figure 10 shows an example of interpolating curves vs interpolating their corresponding latent codes using our proposed VAE. As can be seen from Figure 10 interpolating latent codes results generates new curve data which is smoother than interpolating the curves directly.



(a) (b)

Figure 9: Predicting the position of control points of new generated curve (the light blue one in (a)). The control polygon with the corresponding B-spline curve is depicted in red and the generated curve from VAE is depicted in blue.

In many design workflows, especially in product design, the shape of a 3D part is often driven by a 2D curve. Our proposed model supports this process by allowing a designer to sketch two input curves representing desired shapes (see Figure 12 a and c). Using our model, we generate various curves that smoothly



(a) (b)

Figure 10: Interpolating curves (a) vs interpolating the mean latent vectors of the same curves using the proposed VAE (b).

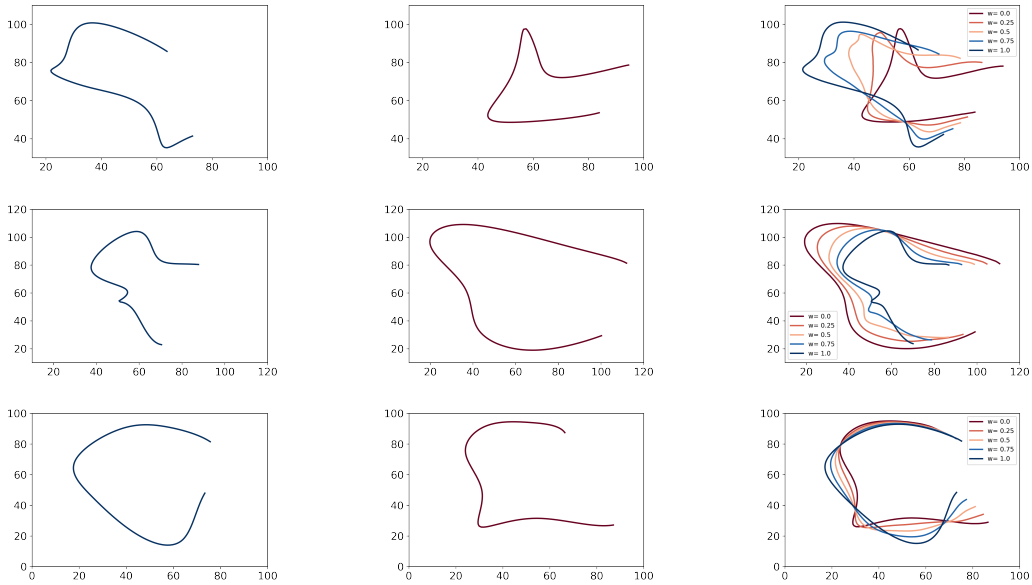


Figure 11: In each row, the first two columns demonstrate curves for latent vectors interpolation. The last column shows 5 steps of linear interpolation.

transition between the two inputs (we show one example in Figure 12 b). These generated curves offer designers a range of alternative shapes that lie between their original ideas, serving as a source of inspiration and creativity. The variations are not random, but rather geometrically plausible, enabling designers to explore the latent space of possible designs with minimal effort. Each generated 2D curve can then be used to form a full 3D shape. This capability provides a valuable tool for early-stage design, where creative exploration is critical but time-consuming. Instead of manually sketching multiple options, the designer can use our system to quickly produce consistent shape variants that maintain key geometric properties. We demonstrate an example in Figure 12 for designing the body of a vase using our proposed method.

Our proposed generative model enables new 2D curve generation which can either be utilized for learning purposes or entering the design pipeline. The coarse and continuous latent representation enables manipulation for design exploration. In addition, the latent code by our system can be used to gain understanding which features are extracted by a network. With deep neural networks, it is difficult to understand how the features are extracted or what features are more important in the final decision. Our proposed generative model can be used to explore how and which features are extracted in the latent code.

However, The proposed network is sensitive to its training set. The training set includes equidistributed curves which are generated using a particular strategy for control points selection. Thus, the model is limited in new curve generation. To generate new curves using our proposed method, the user can define the overall shape of the curves for interpolation, however, the shape should remain similar to those present in the training set. In future, we will work to expand and diversify the dataset to cover a broader range of curves and control points configuration. In addition, the input and output sizes of this model is fixed. We will improve the model to process input curves of variable size and outputs the same size as input.

5 CONCLUSION

In this work, we propose a new generative model for 2D polygonal curve generation and reconstruction. While this approach may find applications in numerous fields, including computer-aided design (CAD), animation, and robotics, this work focuses on data-driven shape synthesis. The network is a Variational Autoencoder designed with convolutional layers and inception modules to learn a coarse and continuous latent representation of 2D B-spline curves and reconstruct the input data or synthesize new examples via interpolation. The network is trained in a self-supervised manner, where it learns low-dimensional informative features without any labels required. This model is able to produce smooth curves without the need for any post-processing.

6 REFERENCES

[Barzegar2024] Barzegar Khalilsaraei, Saeedeh, Alexander Komar, Jianmin Zheng, and Ursula Augsdörfer. "InceptCurves: curve reconstruction using an inception network." The Visual Computer (2024): 1-11.

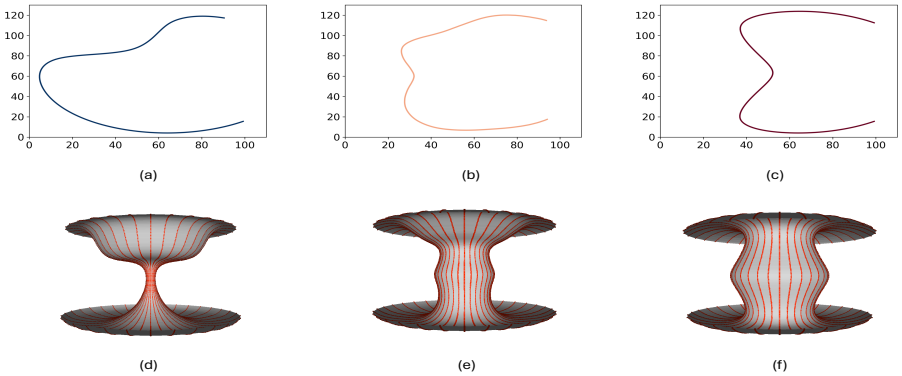


Figure 12: The application of the proposed method to design different shapes of a vase. Figures (a) and (c) depict design choices and one interpolated curve is Figure (b). Figures (d-f) are their corresponding rotational surfaces.

[Cheng2023] Cheng, Yen-Chi, Hsin-Ying Lee, Sergey Tulyakov, Alexander G. Schwing, and Liang-Yan Gui. "Sdfusion: Multimodal 3d shape completion, reconstruction, and generation." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4456-4465. 2023.

[Chen2018] Chen, Wei, and Mark Fuge. "BézierGAN: Automatic Generation of Smooth Curves from Interpretable Low-Dimensional Parameters." arXiv preprint arXiv:1808.08871 (2018).

[Chen2021] Chen, Chen, Kerstin Hammernik, Cheng Ouyang, Chen Qin, Wenjia Bai, and Daniel Rueckert. "Cooperative training and latent space data augmentation for robust medical image segmentation." In Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part III 24, pp. 149-159. Springer International Publishing, 2021.

[Creswell2018] Creswell, Antonia, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A. Bharath. "Generative adversarial networks: An overview." IEEE signal processing magazine 35, no. 1 (2018): 53-65.

[de2020] de Miguel Rodríguez, Jaime, Maria Eugenia Villafañe, Luka Piškorec, and Fernando Sancho Caparrini. "Generation of geometric interpolations of building types with deep variational autoencoders." Design Science 6 (2020): e34.

[Du2020] Du, Xiaosong, Ping He, and Joaquim RRA Martins. "A B-spline-based generative adversarial network model for fast interactive airfoil aerodynamic optimization." In AIAA scitech 2020 forum, p. 2128. 2020.

[Gao2019] Gao, Jun, Chengcheng Tang, Vignesh Ganapathi-Subramanian, Jiahui Huang, Hao Su, and Leonidas J. Guibas. "Deepspline: Data-driven reconstruction of parametric curves and surfaces." arXiv preprint arXiv:1901.03781 (2019).

[Gonog2019] Gonog, Liang, and Yimin Zhou. "A review: generative adversarial networks." In 2019 14th IEEE conference on industrial electronics and applications (ICIEA), pp. 505-510. IEEE, 2019.

[Goodfellow2014] Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." Advances in neural information processing systems 27 (2014).

[Hagg2021] Hagg, Alexander, Sebastian Berns, Alexander Asteroth, Simon Colton, and Thomas Bäck. "Expressivity of parameterized and data-driven representations in quality diversity search." In Proceedings of the Genetic and Evolutionary Computation Conference, pp. 678-686. 2021.

[ho2020] Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." Advances in neural information processing systems 33 (2020): 6840-6851.

[Hui2022] Hui, Ka-Hei, Ruihui Li, Jingyu Hu, and Chi-Wing Fu. "Neural wavelet-domain diffusion for 3d shape generation." In SIGGRAPH Asia 2022 Conference Papers, pp. 1-9. 2022.

[Ioffe2015] Ioffe, S., Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International conference on machine learning, 448-456 (2015).

[Ju2005] Ju, Tao, Joe Warren, James Carson, Gregor Eichele, Christina Thaller, Wah Chiu, Musodiq Bello, and Ioannis Kakadiaris. "Building 3D surface networks from 2D curve networks with application to anatomical modeling." The Visual Computer 21 (2005): 764-773.

[Kim2021] Kim, Beomsoo, Jang-Ho Choi, and Jaegul Choo. "Augmenting imbalanced time-series data via adversarial perturbation in latent space." In

Asian Conference on Machine Learning, pp. 1633-1644. PMLR, 2021.

[kingma2013] Kingma, Diederik P. "Auto-encoding variational bayes." arXiv preprint arXiv:1312.6114 (2013).

[Kingma2014] Kingma, Diederik P. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[Komarichev2019] Komarichev, Artem, Zichun Zhong, and Jing Hua. "A-cnn: Annularly convolutional neural networks on point clouds." In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 7421-7430. 2019.

[Koo2023] Koo, Juil, Seungwoo Yoo, Minh Hieu Nguyen, and Minhyuk Sung. "Salad: Part-level latent diffusion for 3d shape generation and manipulation." In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 14441-14451. 2023.

[Larsen2016] Larsen, Anders Boesen Lindbo, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. "Autoencoding beyond pixels using a learned similarity metric." In International conference on machine learning, pp. 1558-1566. PMLR, 2016.

[Laube2018] Laube, Pascal, Matthias O. Franz, and Georg Umlauf. "Deep learning parametrization for B-spline curve approximation." In 2018 International Conference on 3D Vision (3DV), pp. 691-699. IEEE, 2018.

[Mittal2022] Mittal, Paritosh, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. "Autosdf: Shape priors for 3d completion, reconstruction and generation." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 306-315. 2022.

[Piegl2012] Piegl, Les, and Wayne Tiller. The NURBS book. Springer Science & Business Media, 2012.

[Press1990] Press, William H., and Saul A. Teukolsky. "Savitzky-Golay smoothing filters." Computers in Physics 4, no. 6 (1990): 669-672.

[Regenwetter2022] Regenwetter, Lyle, Amin Heyrani Nobari, and Faez Ahmed. "Deep generative models in engineering design: A review." Journal of Mechanical Design 144, no. 7 (2022): 071704.

[Roberts2018] Roberts, Adam, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. "A hierarchical latent vector model for learning long-term structure in music." In International conference on machine learning, pp. 4364-4373. PMLR, 2018.

[Szegedy2015] Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1-9. 2015.

[Szegedy2016] Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. "Rethinking the inception architecture for computer vision." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2818-2826. 2016.

[Tan2022] Tan, Xavier, Dai Manna, Joyjit Chattoraj, Liu Yong, Xu Xinxing, Dao My Ha, and Yang Feng. "Airfoil Inverse Design using Conditional Generative Adversarial Networks." In 2022 17th International Conference on Control, Automation, Robotics and Vision (ICARCV), pp. 143-148. IEEE, 2022.

[Tronchin2023] Tronchin, Lorenzo, Minh H. Vu, Paolo Soda, and Tommy Löfstedt. "LatentAugment: Data Augmentation via Guided Manipulation of GAN's Latent Space." arXiv preprint arXiv:2307.11375 (2023).

[Van2008] Van der Maaten, Laurens, and Geoffrey Hinton. "Visualizing data using t-SNE." Journal of machine learning research 9, no. 11 (2008).

[Villani2009] Villani, Cédric. Optimal transport: old and new. Vol. 338. Berlin: springer, 2009.

[Wada2024] Wada, Kazunari, Katsuyuki Suzuki, and Kazuo Yonekura. "Physics-guided training of GAN to improve accuracy in airfoil design synthesis." Computer Methods in Applied Mechanics and Engineering 421 (2024): 116746.

[Wang2020] Wang, Xiaogang, Yuelang Xu, Kai Xu, Andrea Tagliasacchi, Bin Zhou, Ali Mahdavi-Amiri, and Hao Zhang. "Pie-net: Parametric inference of point cloud edges." Advances in neural information processing systems 33 (2020): 20167-20178.

[Wang2023] Wang, Yuyang, Kenji Shimada, and Amir Barati Farimani. "Airfoil GAN: encoding and synthesizing airfoils for aerodynamic shape optimization." Journal of Computational Design and Engineering 10, no. 4 (2023): 1350-1362.

[Yang2025] Yang, Shaoliang, Jun Wang, and Kang Wang. "NURBS-OT: An Advanced Model for Generative Curve Modeling." Journal of Mechanical Design 147, no. 3 (2025).

[Yonekura2021] Yonekura, Kazuo, and Katsuyuki Suzuki. "Data-driven design exploration method using conditional variational autoencoder for airfoil design." Structural and Multidisciplinary Optimization 64, no. 2 (2021): 613-624.