Computing Midcurve with Multi-layer and Convolutional Neural Networks

Prashanth Sreenivasan
AI Engineer and Data Scientist
Pune, India
prashanthsrn11@gmail.com

Yogesh H. Kulkarni
AI Advisor
Pune, India
yogeshkulkarni@yahoo.com

***Abstract*—Midcurve computation: the dimension reduction of 2D thin polygon shapes to 1D curves is important for several Computer-Aided Design (CAD) and Computer-Aided Engineering (CAE) applications, particularly in finite element analysis and robot path planning. We advance the state-of-the-art by building on previous work on MidcurveNN, through two new neural architectures: a multi-layer dense network and a CNN-based architecture with skip connections. Through comparative analysis, we demonstrate that while the dense network shows limited improvements, the CNN-based architecture with skip connections achieves a tenfold reduction in average loss compared to existing methods. The proposed approach improves the quality of the generated Midcurves.**

***Keywords*—Mid-curve computation, dimension reduction, convolutional neural networks, dense neural networks**

I. Introduction

Computer-Aided Design (CAD) and Computer-Aided Engineering (CAE) applications often require dimensionality reduction of complex geometric shapes to make analysis and processing simpler. One important such reduction is the Midcurve - a one-dimensional curve representation of 2D thin polygon shapes. A Midcurve captures the essential geometry of the shape while reducing the complexity significantly. As illustrated in Fig. 1, the dimensional reduction process transforms 3D objects into simplified 1D Midcurve representations while preserving essential geometric features.

Midcurves have diverse applications, including Finite Element Analysis (FEA), character animation, and robot path planning. Chang [1] demonstrates the use of Midcurve computation based on Medial Axis Transform (MAT) for robot path planning, where it effectively dilates the robot's workspace and widens narrow passages.

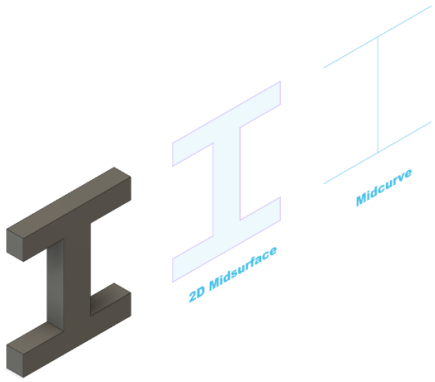

Fig. 1. Visualization of the dimensional reduction process demonstrating: the original 3D CAD model, intermediate 2D projection, and final 1D Midcurve representation preserving key geometric features.

II. Related Work

Previous research has explored multiple approaches for Midcurve and Midsurface extraction, including Voronoi diagram-based, thinning-based, tracing-based, decomposition-based, pairing-based, and mesh-based methods [2]. The evolution of these methods reflects the growing complexity of dimensional reduction challenges in CAD/CAE applications. Fig. 2 compares the outcomes of different traditional Midcurve extraction approaches.

The Medial Axis Transform (MAT), based on Voronoi diagram principles, provides mathematically accurate results by tracking the center of the largest inscribed circle [2]. As illustrated in Fig. 3, when the circle's radius approaches zero at sharp corners of the boundary, MAT accurately captures these features in the resulting Medial Axis Curve. While MAT excels in precision, the computational complexity and high sensitivity to boundary noise limit its practical applications. The Chordal Axis Transform (CAT) improves the efficiency by using mesh-based processing but remains highly dependent on mesh quality [2].

Thinning-based approaches are straightforward to implement but face challenges with non-uniform thickness and rough edges [2]. Tracing-based and Pairing-based methods perform efficiently for simple shapes but struggle with branches and junctions [2]. Decomposition-based methods handle complex shapes through systematic breakdown but produce suboptimal results at connecting interfaces and require significant processing time [2].

Traditional algorithms face three main problems in real-world applications. First, computational scalability is a significant concern as time and memory requirements increase with shape complexity. Second, they are very sensitive to minor irregularities or variations in the shape boundary, often requiring significant preprocessing for usage [3]. Third, edge cases like branching structures, junction points, and shapes with internal features present significant challenges to traditional algorithms.

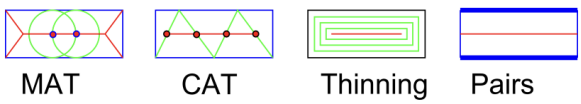

Fig. 2. Comparison of traditional Midcurve extraction approaches. From left to right: Medial Axis Transform (MAT) showing center-line extraction, Chordal Axis Transform (CAT) demonstrating mesh-based reduction, Thinning-based approach showing progressive erosion, and Pairing-based method illustrating boundary matching [2]

979-8-3315-1296-5/25/$31.00 ©2025 IEEE

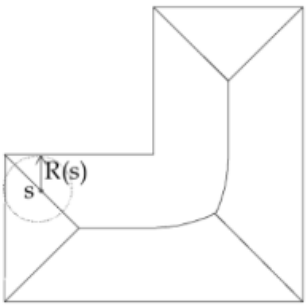

Fig. 3. Demonstration of Medial Axis Transform (MAT) capturing sharp corners. MAT captures features when the inscribed circle's radius approaches zero at the boundaries. [2]

The MidcurveNN [3] paper introduced several key innovations, by converting the Midcurve computation into an image-to-image transformation problem. The shift to image-based representation was driven by limitations in handling geometric shapes: they can't be treated like simple sequences, some shapes branch in multiple directions, and the inputs and outputs can have different lengths. Graph structures focus more on connectivity than spatial accuracy. Converting shapes to fixed images solves this problem and works well with neural networks. This approach also allows easy data augmentation, by rotation, mirroring, and scaling. Though this approach loses some precision by converting exact geometry to pixels, it offers a practical solution that works well with standard CNN encoder-decoder networks.

MidcurveNN [3] used a simple encoder-decoder architecture [4] for dimensional reduction and supervised learning with training data pairs. This method was successful in capturing mid-curves of 2D closed shapes, but it had certain limitations. It followed a simple architecture design with basic dense layers and was limited in its capability to capture spatial relations. It also lacked modern neural network optimizations. The simple encoder-decoder approach often produced noisy results as shown in Fig. 4.

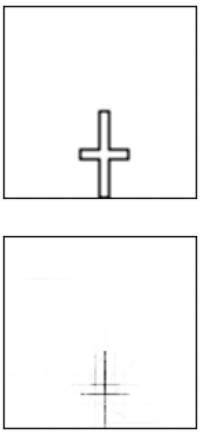

Fig. 4. Example outputs showing noise in Midcurve generation using basic encoder-decoder architecture. The results demonstrate specific limitations: extra branches in curve generation, discontinuities in curve structure, and inconsistent line thickness affecting overall accuracy.

Our proposed architecture addresses these limitations through two approaches: a dense encoder-decoder architecture for deeper feature representations and a CNN-based architecture that preserves spatial information through skip connections. Both proposed architectures inherit the advantage of the image-based approach introduced in the MidcurveNN [3] paper. By converting a geometric problem into fixed-dimension images, the edge cases in the first approach like irregular boundaries, intersections, and complex topologies, are naturally handled. For example, our approach successfully handles cases such as shapes with multiple intersection points and varying boundary thickness, which traditionally require special case handling in geometric approaches.

While the Image-based approach handles geometric edge cases, the conversion from image to geometric representations remains non-trivial. This presents an important trade-off in this approach, gaining robustness in the computation of Midcurves at the cost of precision in the final geometric reconstruction.

III. Proposed Architectural Improvements

The original MidcurveNN framework introduced the method of tackling the problem of Midcurve computation as a supervised learning problem. This implementation demonstrated feature learning and Midcurve generation from input images. The MidcurveNN paper [3] indicates that a *Simple single-layer encoder and Decoder* network can learn the dimension reduction function well, as shown in Fig. 5.

A. Updated Architectural Approaches

This paper introduces two new frameworks that use the recent advancements in deep learning to better capture the essential features for dimension reduction. This paper advances the current state-of-the-art in neural network-based Midcurve computation through these new frameworks with comparative analysis of performance.

This section goes into the architecture behind the two frameworks, the hypothesis behind why they would perform better than the original model, and the results.

Both these frameworks have a common preprocessing pipeline. This normalizes the images into 128x128 pixels and scales the pixels to a [0,1] range. This ensures the network behaves consistently across varied inputs.

The transformation to an image-to-image approach addresses generalizability by moving from a geometrical rule-based approach to a standardized image representation, making this approach theoretically applicable to any thin polygon shape in an image format. While testing across various arbitrary shapes remains future work, the main limitations are those related to image-based deep learning, like resolution dependence, rather than geometric complexity.

B. Dense Encoder-Decoder Architecture

The first proposed architecture uses a fully connected deep structure. The hypothesis was to expand the representational capacity of the original MidcurveNN model using gradual dimension reduction through multiple dense layers to enable better feature extraction.

The encoder pipeline gradually reduces the input dimension from 10,000 to 100, through intermediate representations of 2048 and 1024 neurons. Each layer has ReLU activation functions. The decoder pipeline is symmetric to the encoder structure and progressively expands the compressed representation to its original dimensions. Finally, the last layer employs a sigmoid activation function to produce the output. Fig. 6 illustrates the dense encoder-decoder architecture, which tries to improve the representational capacity with multiple layers.

979-8-3315-1296-5/25/$31.00 ©2025 IEEE

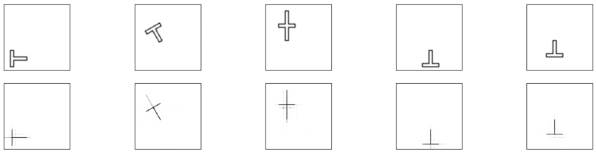

Fig. 5. Performance evaluation of the original MidcurveNN model showing comparative results between input test shapes (top row) and their corresponding generated Midcurves (bottom row), demonstrating the model's capability in dimension reduction.

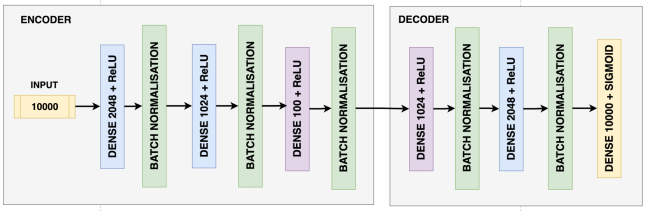

Fig. 6. Architectural diagram of the proposed dense encoder-decoder network. The structure shows gradual dimension reduction from 10,000 to 100 neurons through intermediate layers of 2048 and 1024 neurons with ReLU activation, except for the final sigmoid layer.

We incorporated several training strategies to enhance training stability and convergence. The model is trained using an Adam Optimizer with a custom fine-tuned learning rate of 0.0001. The training is monitored through validation performance, with early stopping to prevent overfitting [5]. We decided on a batch size of 32 to balance between computational efficiency and stability.

Despite using multiple layers for gradual dimension reduction, the dense encoder-decoder model showed higher average loss and MAE compared to the original single encoder-decoder. As shown in Fig. 7, despite the increased architectural complexity, the model's outputs showed limited improvements over the baseline. Performance comparisons with consistent batch size and epochs, as shown in Fig. 14, led us to explore a CNN-based architectural approach.

C. *CNN-based Architecture*

The second proposed architecture uses a convolutional network structure that preserves spatial relationships in the inputs. This architecture tackles a fundamental drawback in fully connected networks: the inability to capture spatial hierarchies [6].

The CNN encoder extracts features through four convolutional blocks, as illustrated in Fig. 8, each progressively reducing spatial dimensions while increasing depth from 32 to 256 filters. This allows the network to capture both the fine details and the overall structure. Each layer is followed by batch normalization. This improves the stability and normalization of features.

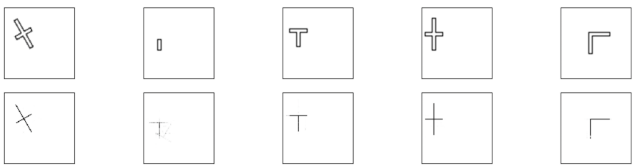

Fig. 7. Dense encoder-decoder model results on test cases. Original input shapes at the top, generated Midcurves below, showing modest improvements in curve consistency.

One of the key features in this architecture is the use of skip connections between encoder and decoder layers [7]. These connections preserve information and gradients that might otherwise be lost or diluted by passing through multiple layers. The decoder pathway uses transposed convolutions for upsampling, with skip connections at each level to preserve spatial accuracy.

The training process implements dynamic learning rate adjustment, halving the rate when loss plateaus for five consecutive epochs. This approach, combined with early stopping, ensures better results while preventing overfitting.

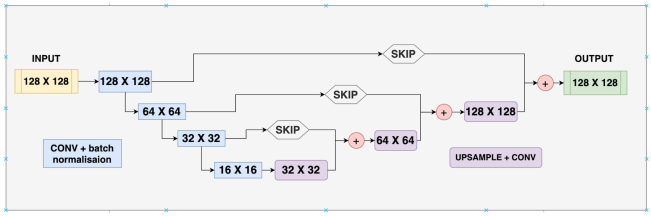

Fig. 8. Architectural diagram of the proposed CNN network. The structure shows: An encoder pathway with four convolutional blocks and increasing filter depths, skip connections to retain spatial information, and a decoder pathway with transposed convolutions.

The transformation to a 128x128 pixel representation allows consistent processing across various geometries. The complexity of the CNN-based model scales with this fixed input dimension rather than the geometric complexity of the shape. The computational complexity of the CNN architecture is $O(n^2 \cdot k^2 \cdot c)$ for an n×n input image, where k is the convolutional kernel size and c is the maximum channel depth.

This model shows the best results among the three models, as illustrated in Fig. 9, with significantly reduced noise and improved geometric accuracy.

IV. Experimental Setup

This section summarizes the dataset preparation and the evaluation metrics.

A. *Dataset Preparation*

This research uses the MidcurveNN dataset, comprising 2D thin polygons, including English alphabets and geometric shapes. These shapes are augmented using translations, rotations, mirroring, and adding noise to enhance model robustness and prevent overfitting. The training dataset, as shown in Fig. 10, consists of diverse 2D thin polygons, including alphabets and geometric shapes, along with their corresponding ground truth Midcurves.

B. *Evaluation Metrics*

Binary cross-entropy loss [5] measures the difference between predicted and actual pixel distributions in generated images. It is suitable for image-to-image tasks where output pixels are normalized between 0-1.

Mean absolute error, or MAE, measures the average absolute pixel difference between the generated and the actual image, providing a direct measure of reconstruction.

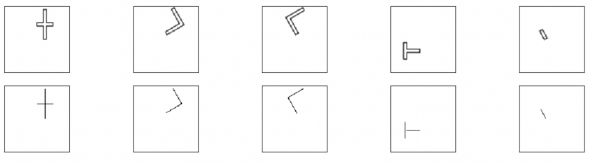

Fig. 9. CNN model results on test cases. Original input shapes at the top generated Midcurves below. Note the reduced noise compared to previous approaches.

979-8-3315-1296-5/25/$31.00 ©2025 IEEE

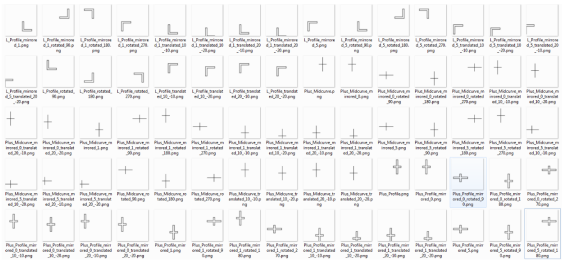

Fig. 10. Representative samples from the training dataset showing: input 2D thin polygons, corresponding ground truth Midcurves, and augmented variations including rotations, translations, and mirroring transformations [8].

V. Results

We evaluate the performance of all three architectures: simple encoder-decoder, dense network, and the CNN-based model—using consistent training parameters (100 epochs, batch size of 16) to ensure a fair comparison. Fig. 11 depicts the baseline performance of the simple encoder-decoder model, showing steady convergence in both Loss and MAE metrics. As seen in Fig. 12, despite the more complex architecture, the dense network shows higher validation loss compared to the baseline. Fig. 13 highlights the improved performance of the CNN-based architecture, showing more stable convergence and lower overall loss.

The underperformance of the dense model can be attributed to two factors. First, as shown in Fig. 12, the Dense architecture shows higher validation loss as compared to the Simple encoder-decoder, suggesting the increased number of parameters may be leading to overfitting. Second, the lack of spatial information preservation, which is evident compared to the CNN architecture, seems to be a limitation in the dense encoder-decoder approach. The integration of dropout to tackle overfitting, batch normalization to stabilize training, or alternate activation functions are some architectural improvements that can be explored in future work.

The CNN-based architecture performs better than the other models across all metrics, achieving a validation loss of 0.0005 compared to the simple architecture's 0.008, and dense architecture's 0.012. This superior performance can be attributed to three key factors. First, the convolutional layers naturally preserve spatial relationships. The four convolutional blocks in the encoder pathway, with progressively increasing filters, enable the model to capture both local geometric and global shape structure in the input shapes, enabling better geometric continuity in the generated Midcurves compared to the dense architecture's processing. Second, skip connections prove crucial for preserving spatial accuracy between the encoder and decoder. Third, the convolutional layer's translation invariance makes the model more robust to variations in shape position, and orientation. The CNN model's learning of spatial features, combined with skip connections, creates a more robust model as compared to the dense encoder-decoder architecture.

The comparison in Table I demonstrates the performance improvement with the CNN architecture, achieving consistently lower loss values across all metrics.

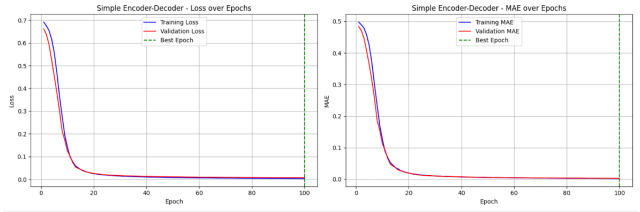

Fig. 11. Training metrics for the simple encoder-decoder model over 100 epochs showing: binary cross-entropy loss progression (left) and mean absolute error (right) evolution. Both metrics demonstrate consistent convergence behavior.

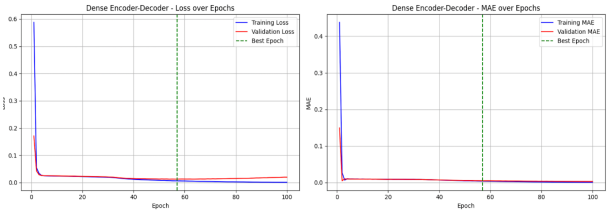

Fig. 12. Training performance of the dense encoder-decoder model showing: binary cross-entropy loss progression (left) and mean absolute error (right) across training epochs. This shows higher validation loss despite higher complexity

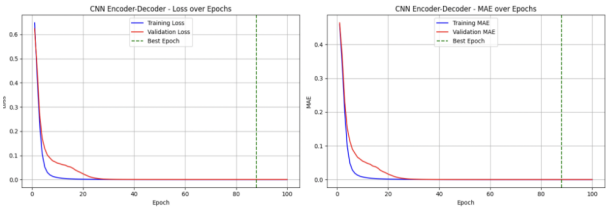

Fig. 13. Training metrics for the CNN-based architecture showing: binary cross-entropy loss progression (left) and mean absolute error (right), demonstrating more stable convergence and lower overall loss values compared to previous models.

TABLE I

Metric	Simple	Dense	CNN
Best Epoch	100	62	93
Training Loss	0.0034	0.0049	**0.0003**
Training MAE	0.0023	0.0032	**0.0003**
Validation Loss	0.0080	0.0121	**0.0005**
Validation MAE	0.0031	0.0042	**0.0003**
Training Time	48s	59s	2m 42s

VI. Conclusion

This paper demonstrates two deep neural network architectures for tackling the problem of Midcurve computation. Building on the MidcurveNN paper [3], the first architecture is a dense layer encoder-decoder. While the dense layer architecture failed to outperform the traditional single encoder-decoder model, our CNN-based architecture achieved a tenfold reduction in average loss with significantly reduced noise.

979-8-3315-1296-5/25/$31.00 ©2025 IEEE

VII. REFERENCES

[1] Chang, Y.C. & Saha, Mitul & Prinz, F. & Latombe, J.C. & Pinilla, Miguel. (2004). Medial Axis Transform assists path planning in configuration spaces with narrow passages.

[2] Kulkarni, Y.; Deshpande, S.: Medial object extraction - a state of the art. In International Conference on Advances in Mechanical Engineering, SVNIT, Surat, 2010

[3] Kulkarni, Yogesh. (2022). MidcurveNN: Neural Network for Computing Midcurve of a Thin Polygon. Computer-Aided Design and Applications. 19. 1154-1161. 10.14733/cadaps.2022.1154-1161.

[4] Christopher Tierney, T.R., Declan Nolan; Armstrong, C.: Using mesh-geometry relationships to transfer analysis models between cae tools. Proceedings of the 22nd International Meshing Roundtable, 2013.

[5] Ruby, Usha & Yendapalli, Vamsidhar. (2020). Binary cross entropy with deep learning technique for Image classification. International Journal of Advanced Trends in Computer Science and Engineering. 9. 10.30534/ijatcse/2020/175942020.

[6] Shen, W.; Zhao, K.; Jiang, Y.; Wang, Y.; Zhang, Z.; Bai, X.: Object skeleton extraction in natural images by fusing scale-associated deep side outputs, 2016.

[7] Chollet, F.: Building autoencoders in keras. https://blog.keras.io/building-autoencoders-in-keras.html, 2019.

[8] Y. H. Kulkarni, "Simple Encoder Decoder for MidcurveNN," Kaggle dataset, 2022. [Online]. Available: kaggle.com/yogeshkulkarni/simple-encode-decoder-for-midcurvenn

979-8-3315-1296-5/25/$31.00 ©2025 IEEE