# MidcurveNN: Neural Network for Computing Midcurve of a Thin Polygon

Yogesh H. Kulkarni[1]

[1]Pune, India, yogeshkulkarni@yahoo.com

Corresponding author: Yogesh H. Kulkarni, yogeshkulkarni@yahoo.com

**Abstract.** Multiple applications demand lower dimensional representation of geometric shapes. Midsurface is such two-dimensional(2D) i.e surface representation of a three-dimensional(3D) thin-walled solid shape. It is used in applications such as animation, shape matching, retrieval, finite element analysis, etc. 2D counterpart of 3D Midsurface is called Midcurve. So, Midcurve is one-dimensional(1D) i.e curve representation of a two-dimensional(2D) sketch profile shape. Methods available to compute midcurves vary based on the type of the input shape (images, sketches, etc.) and processing (thinning, Medial Axis Transform (MAT), Chordal Axis Transform (CAT), Straight Skeletons, etc.).

This paper talks about a novel method called MidcurveNN which uses Encoder-Decoder neural network for computing midcurve from images of 2D thin polygons in supervised learning manner. This dimension reduction transformation from input 2D thin polygon image to output 1D midcurve image is learnt by the neural network, which can then be used to compute midcurve of an unseen 2D thin polygonal shape.

**Keywords:** Midcurve, Encoder-Decoder, Neural Network
**DOI:** https://doi.org/10.14733/cadaps.2022.aaa-bbb

## 1 INTRODUCTION

Computer-aided Design (CAD) models of thin-walled solids such as sheet metal or plastic parts are often reduced dimensionally to their corresponding midsurfaces for quicker and fairly accurate results of Computer-aided Engineering (CAE) analysis (Fig. 1).

A midsurface is a surface lying midway of (and representing) the input shape (Fig. 2).

Computation of the midsurface is still a time-consuming and mostly, a manual task due to lack of robust-automated approaches. Many of the existing automatic midsurface generation approaches result in some kind of failures such as gaps, missing patches, overlapping surfaces, etc. It takes hours or even days to correct such errors with manual intervention.

In CAD, thin solid shapes are typically created by extruding profile. Thus, generating Midsurface is same as extruding Midcurve of the profile (Fig. 3a).

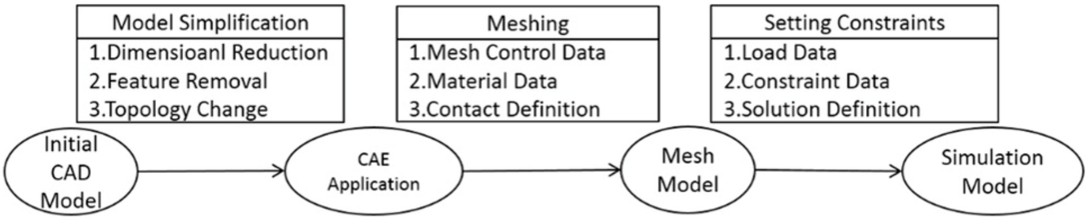



**Figure 1**: Computer-aided PDP (Source: Tierney[2])

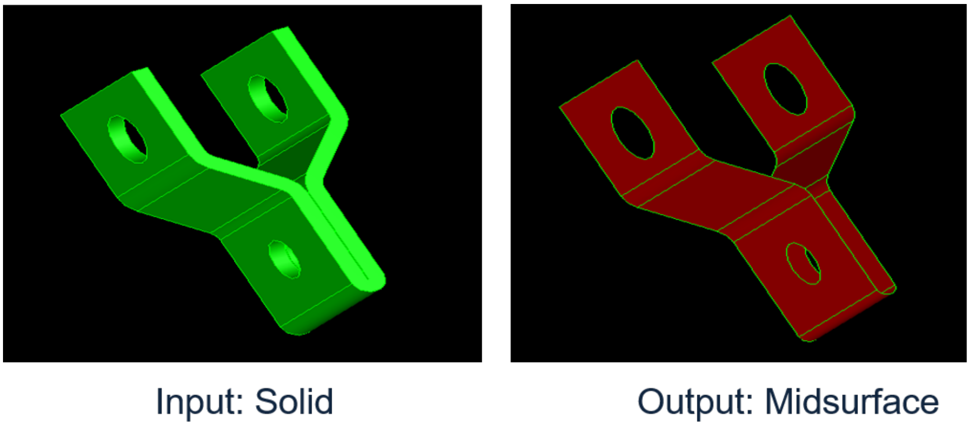
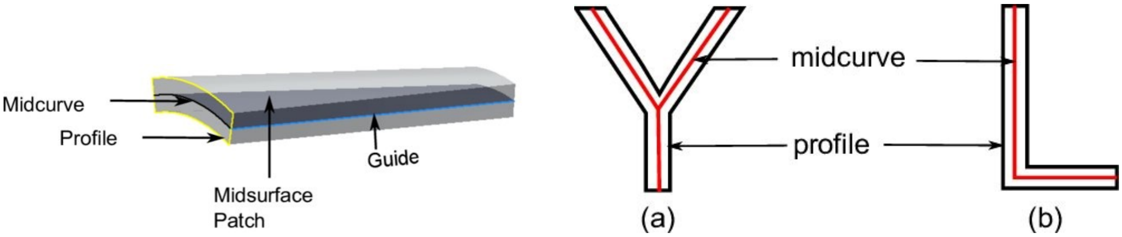



**Figure 2**: Midsurface




(a) (b)

**Figure 3**: Midcurve

Midcurve is a lower dimensional entity which represents shape of its parent object. Fig. 3b shows examples of midurves.

**Definition:** *Midcurve is an aggregation of curve segments (where each segment corresponds to a pair of nonadjacent edges in the object that are closest to each other) that form a closed and connected set and that satisfy homotopy*

Midcurve, being simpler than the parent object, operations like pattern recognition, approximation, similarity estimation, collision detection, animation, matching and deformation can be performed efficiently on it than on the parent object.

## 2 RELATED WORK

Research on computing Midsurface and Midcurve is going on for decades. Fig. 4) shows important milestones along the journey.

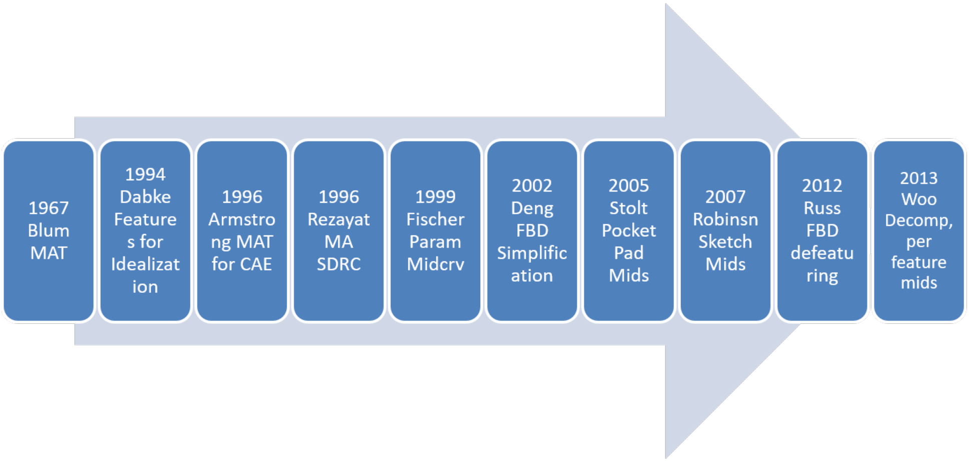



**Figure 4**: Milestones along journey of Midsurface computation research

Midcurve can be computed via various mathematical formulations/approaches such as Medial Axis Transform (MAT), Chordal Axis Transform (CAT), Pairing, Thinning etc. Figure 5 shows some of these. More detailed analysis can be found in the survey paper [3].

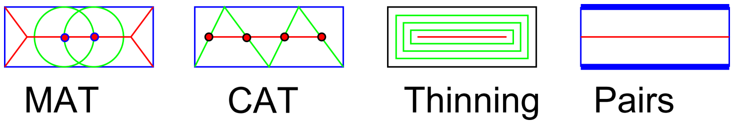



**Figure 5**: Medial Object computation methods

Following are some salient observations:

- Biggest strength of formal approaches like MAT is that it can be computed of any shape, thick or thin. Being formally defined, the converse or reversal process, meaning "given a MAT compute the original shape", is possible.
- Major drawback of MAT, Thinning methods is that it creates unnecessary branches and its shape is smaller than the original corresponding faces.
- MAT based approaches also suffer from robustness problem. A slight change in base geometry forces re-computation of MAT and the results could very well be different than the original.
- Although MAT approaches have been around for decades and are fairly mature, its usage in midcurve generation is still very complex and difficult to ensure appropriate topology.
- The major limitation of CAT approach is that mesh has to be generated beforehand. Creating constrained, single layer meshes on complicated 2D profiles are, at times, difficult.
- Thinning approaches are based on split events of the straight line skeleton gives counter-intuitive results if the polygon contains sharp reflex vertices.
- In Parametric approach, the two input curves or surfaces may not be in one-to-one form. In such cases maintaining continuity can be challenging.

- Quality of surface generated by parametric approach depends on the sampling done to compute the midpoints.
- Midcurve by profile decomposition approach is not used widely. The decomposition can result in large number of redundant sub-shapes making it ineffective and unstable for further processing.

The survey paper [3] suggests to avoid formal methods such as MAT, CAD, Thinning and Parametric, for computing midcurve as they need heavy post-processing to remove unwanted curves. The heuristic method of decomposition, has been error prone and inefficient so far. Author's own past work (Fig. 6) demonstrates reasonable succes on simpler shapes, but being, rule-based approach, had limitation on scalability.

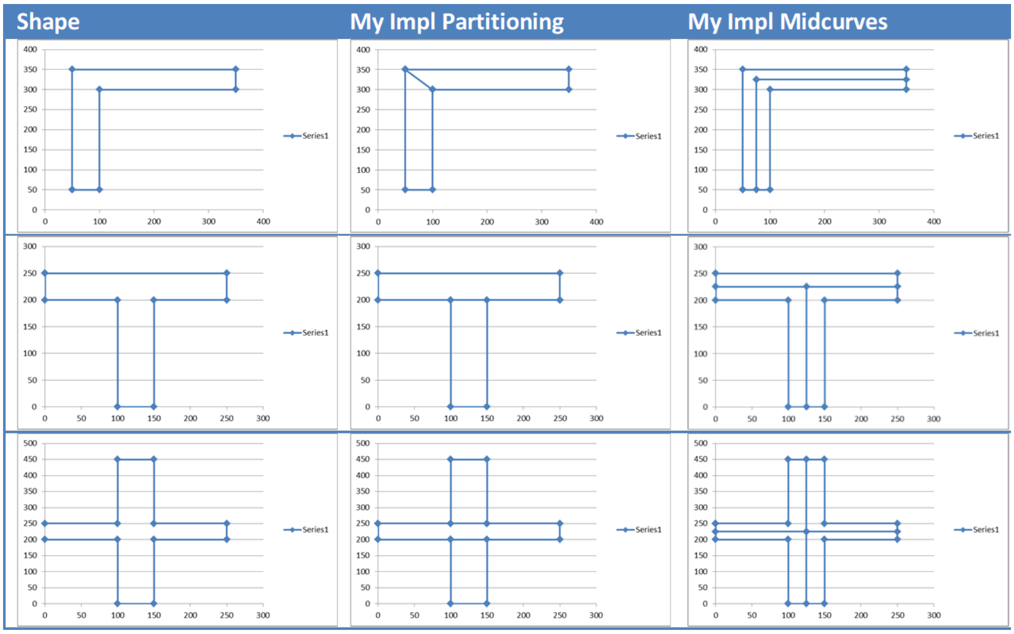



**Figure 6**: Midcurve Computation by Cellular Decomposition ([4])

Review of the approaches suggests approaches for computing midcruve are needed to take into account the application context, accuracy, characteristics and aspect ratio of the sub-polygons. For the present research work, the midcurve generation needs primitive shaped, thin sub-polygons. Non-primitive, skewed shapes would result in inappropriate midcurves.

The current paper proposes a novel method of computing Midcurve using Neural Networks i.e Deep Learning approach.

## 3 PROPOSED METHOD

In the current paper we focus on computing midcurve for 2D planar sketch profiles and not 3D skeletonal shapes. Even in 2D profiles, shapes vary enormously. As the first level of simplification, we would deal with 2D polygons only (with an assumption that curved shapes can be converted to polygonal shape by faceting). Figure 7 shows some of the input shapes which can be considered. English alphabets are chosen for easy understanding and verification of the proposed method.


**Figure 7**: 2D Thin Polygonal shapes

Computation of midcurve, in its original form, is transformation of a 2D thin closed, with/without-holes polygon to 1D open/closed/branched polyline. Paper [4] details one of the effective midcurve computation techniques, based on rule-based computational geometry approach. Such techniques have a shortcoming of not being scalable or generic enough to be able to handle variety of shapes. Deep Learning neural network architectures are showing potential of developing such generic models. This dimension reduction transformation should ideally be modeled as Sequence to Sequence (Seq2Seq) neural architecture, like Machine Translation.

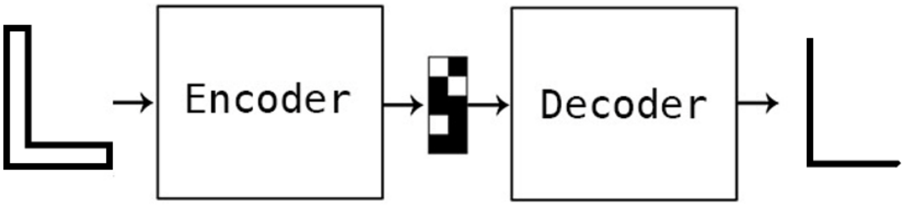



**Figure 8**: Encoder-Decoder Architecture

In the current problem, the input and the output sizes could be different not just in a single sample, but across all samples. Many current Seq2Seq neural networks need fixed size inputs and outputs, which if not present in data, are artificially managed by adding padding of certain improbable value. Such padding is not appropriate for the current midcurve computation problem, as the padding-value can inappropriately get treated as part of the valid input. In this initial phase, to circumvent the problem of variable size, image-based inputs and outputs are used, which are of fixed size. Both input and output polygonal shapes are rasterized into images of 100x100, thus making them fixed size for all samples, irrespective of the original shapes.

This paper proposes to use such network for midcurve computation in the form of image-to-image mode. Input images have thin polygonal shapes whereas output images have corresponding midcurve images. Figure 8 shows the Encoder-decoder architecture, called MidcurveNN.

Input and output geometries are rasterized into 100x100 size images. Transformations like translation, rotation and mirroring are applied to create diversity in the samples. MidcurveNN being a Supervised Learning approach, both input thin-polygons and corresponding output midcurve polylines are transformed simultaneously. Figure 9 shows some samples. This is training data.

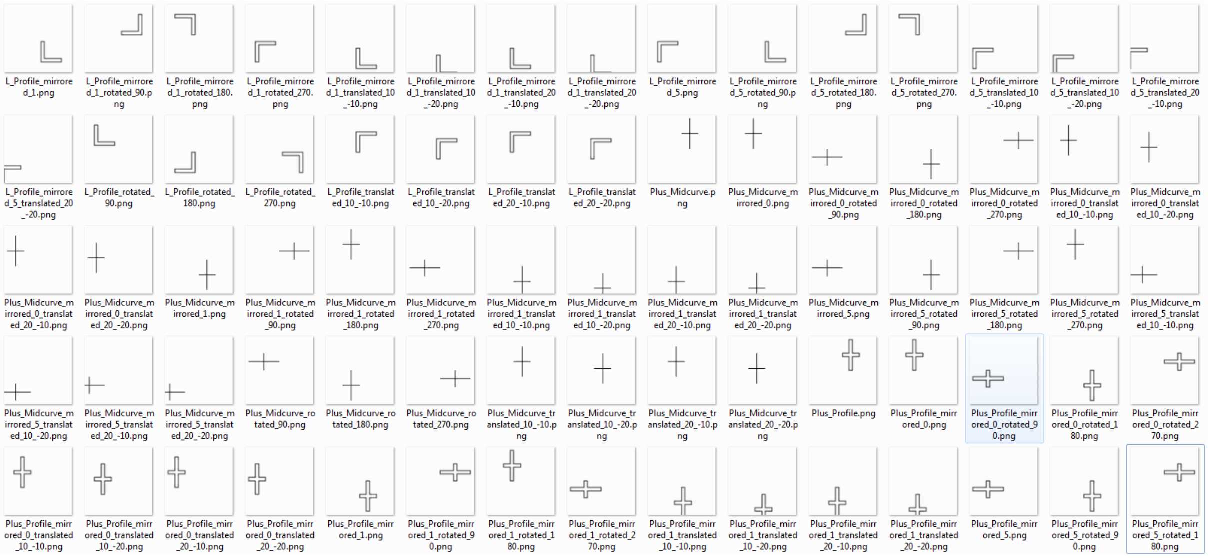


**Figure 9**: Training Data: Inputs (thin polygons) and outputs (midcurves)

MidcurveNN encoder-decoder has been implemented in Python programming with Keras library [1]. Abridged code listing is presented below (Code listing 1).

Listing 1: Encoder-Decoder

```
input_img = Input(shape=(input_dim,))

encoded = Dense(encoding_dim, activation='relu')(input_img)
decoded = Dense(input_dim, activation='sigmoid')(encoded)

autoencoder = Model(input_img, decoded)

encoder = Model(input_img, encoded)
encoded_input = Input(shape=(encoding_dim,))
decoder_layer = autoencoder.layers[-1]
decoder = Model(encoded_input, decoder_layer(encoded_input))

autoencoder.compile(optimizer='adadelta', loss='binary_crossentropy')
```

Encoder takes input image of size $100 \times 100 = 10000$, then comes Dense layer with size $100$ to form the encoded vector. Decoder takes encoded vector as input, then with a Dense layer expands back to $100 \times 100 = 10000$ size of the output image. Relu activation is used for Encoder whereas Sigmoid for the decoder. AdaDelta optimizer with binary cross entropy as loss function is used to compute the losses. Table 1 shows loss across number of epochs.

Some of the results are shown in Figure 10. Inputs are at the top and output midcurve at the bottom.

**Table 1**: Improvement in performance with epochs

| Epochs | Training loss | Validation loss |
|---|---|---|
| 50 | -17.6354 | -8.3223 |
| 200 | -16.8878 | -7.7672 |

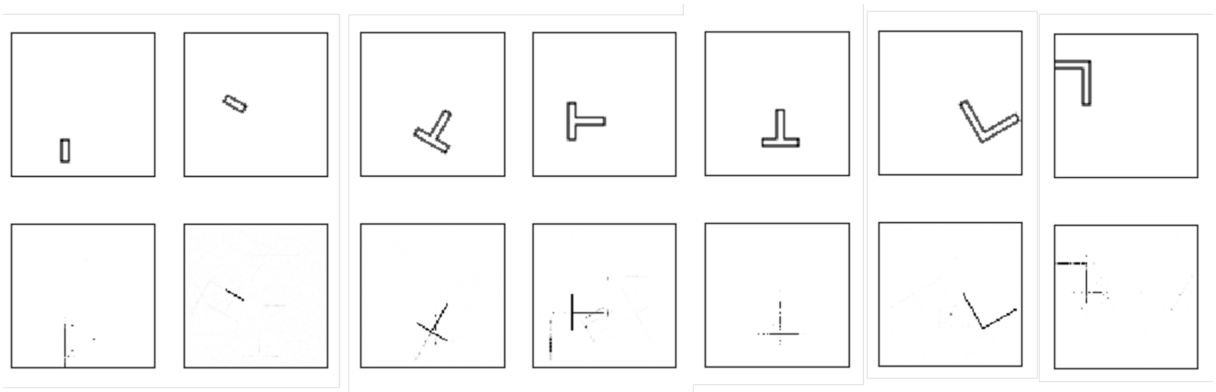


**Figure 10**: Predicted Data: Inputs (thin polygons) and outputs (midcurves)

Shape on the top is the input thin polygon whereas the corresponding shape at the bottom is the predicted midcurve. It can be clearly seen that the network is able to localize the shape and learn the dimension reduction function reasonably well. It is still not perfect or usable, as some stray points are still being wrongly classified as the part of output midcurve. A better network model and/or post processing is needed to make output midcurve practically usable.

## 4 CONCLUSIONS AND FUTURE WORK

Traditional methods of computing midcurves are predominantly rules-based and thus, have limitation of not developing a generic model which will accept any input shape. MidcurveNN, a novel Encoder-Decoder network attempts to build such a generic model. This paper demonstrates that simple single layer encoder and decoder network can learn the dimension reduction function reasonably well. Although more development is necessary in evolving a better neural architecture, the current results show positive potential.

Working on truly variable size inputs (thin polygon) and outputs (polyline) using dynamic graph of Encoder-Decoder network can be attempted in the future. More and highly diversified data can help improve the quality of the output. Developing a formal representation of polygonal shapes with variations such as open/closed, with-without loops, branched as a coherent sequence of points is also on the agenda.

*Yogesh H. Kulkarni* http://orcid.org/0000-0002-1686-7907

## REFERENCES

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

[2] 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.

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

[4] Kulkarni, Y.; Sahasrabudhe, A.; Kale, M.: Dimension-reduction technique for polygons. International Journal of Computer Aided Engineering and Technology, 9(1), 2017. http://doi.org/10.1504/IJCAET.2017.080772.