STAG: Smart Tools and Applications in Graphics (2021)
P. Frosini, D. Giorgi, S. Melzi, and E. Rodolà (Editors)

LengthNet: Length Learning for Planar Euclidean Curves

Barak Or [†] and Ido Amos [‡]

ALMA Technologies LTD, Haifa, Israel

Abstract

In this work, we used a deep learning (DL) model to solve a fundamental problem in differential geometry.
One can find many closed-form expressions for calculating curvature, length, and other geometric properties in the literature. As we know these properties, we are highly motivated to reconstruct them by using DL models. In this framework, our goal is to learn geometric properties from many examples. The simplest geometric object is a curve, and one of the fundamental properties is the length. Therefore, this work focuses on learning the length of planar sampled curves created by a simulation. The fundamental length axioms were reconstructed using a supervised learning approach. Following these axioms, a DL-based model, we named LengthNet, was established. For simplicity, we focus on the planar Euclidean curves.

1. Introduction

The calculation of curve length is a significant component in many classical and modern problems involving numerical differential geometry [GP90, HC98]. Several numerical constraints affect the quality of the length calculation; additive noise, discretization error, and partial information. One use case for length calculation is a handwritten signature. It involves the computation of the length along the curve [OTPH16]. This use case and many others should handle the mentioned numerical constraints. Hence, a robust approach to handle it is required.

Recent works explore the possibilities of using classical machine learning (ML) or deep learning (DL) based approaches as they achieved great success in solving many classification, regression and anomaly detection tasks [LBH15]. Evidence of the effectiveness of DL in solving such tasks has been shown repeatedly in recent years [LBH15]. An efficient DL architecture finds intrinsic properties by using a convolutional operator (and some more sophisticated and nonlinear operators) and generalizes them. Their success is related to the enormous amount of data and their capability to optimize complicated models by high computational available resources.

Related papers in the literature mainly address a higher level of geometric information by DL approach [BBL*17, BN18]. Saying that, a fundamental property was reconstructed by DL model in [PWK16], where a curvature-based invariant signature was learned by using a Siamese network configuration [Chi20]. They presented the advantages of using the DL model to reconstruct the curvature signature, which mainly results in robustness to noise and sampling errors.

As we know the powerful functionality of DL models, we are highly motivated to use them to reconstruct fundamental geometric properties. Specifically, we focus on the length property reconstruction for curves in the two-dimensional Euclidean domain by designing a DL-based model. The task was formulated in a supervised learning setup. There, a data-dependent learning-based approach was applied by feeding each example at a time through our DL-based model and by minimizing a unique loss function that satisfies the length axioms. For that, we created four anchor shapes, and applied translations, rotations, and additional operations to cover a wide range of geometric representations. The resulting trained DL model was called LengthNet. It obtains a 2D vector as an input, representing samples of a planar Euclidean curve, and outputs their respective length.

The main contribution of this work is to reconstruct the length property. For that, a DL architecture was designed. This architecture is based on the classical Convolutional Neural Networks (CNNs).

The remainder of the paper is organized as follows: Section 2 summarizes the geometric background of the length properties. Section 3 provides a detailed description of the learning approach where the two architectures are presented. Section 4 presents, the results followed by the discussion. Section 5 gives the conclusions.

2. Geometric Background of Length

In this section, the length properties are presented and the discretization error is reviewed.

[†] barak@almatechnologies.com

[‡] ido@almatechnologies.com

DOI: 10.2312/stag.20211472

2.1. Length Properties

Consider a planar parametric differential curve in the Euclidean space, $C(p)=\{x(p),y(p)\}\in\mathbb{R}^2$, where x and y are the curve coordinates parameterized by parameter $p\in[0,N]$, where N is a partition parameter. The Euclidean length of the curve, is given by

$$l(p)=\int_0^p |C_{\tilde{p}}(\tilde{p})|\,d\tilde{p}=\int_0^p \sqrt{x_{\tilde{p}}^2+y_{\tilde{p}}^2}d\tilde{p}, \tag{1}$$

where $x_p=\frac{dx}{dp}, y_p=\frac{dy}{dp}$. Summing all the increments results in the total length of C, given by

$$\mathcal{L}=\int_0^N |C_{\tilde{p}}(\tilde{p})|\,d\tilde{p}. \tag{2}$$

Following the length definition, the main length axioms are provided.
Additivity: The length additives with respect to concatenation, where for any C_1 and C_2 the following holds

$$\mathcal{L}(C_1)+\mathcal{L}(C_2)=\mathcal{L}(C_1\cup C_2). \tag{3}$$

Invariance: length is invariant with respect to rotation (**R**) and translation (**T**),

$$\mathcal{L}(\mathbf{T}[\mathbf{R}[C]])=\mathcal{L}(C). \tag{4}$$

Monotonic: length is monotone, where for any C_1 and C_2 the following holds

$$\mathcal{L}(C_1)\leq\mathcal{L}(C_2) \quad C_1\subseteq C_2. \tag{5}$$

Non-negativity: The length of any curve is non-negative,

$$\mathcal{L}(C)\geq 0. \tag{6}$$

2.2. Discretization Error

In order to reconstruct the length property by the DL model, a discretization of the curve should be applied. As a consequence, it is prone to errors. The curve C lies on a closed interval $[\alpha,\beta]$. In order to find the length by a discretized process, a partition of the interval is done, where

$$\mathcal{P}=\{\alpha=p_0<p_1<p_2<\cdots<p_N=\beta\}. \tag{7}$$

For every partition $\mathcal{P}$, the curve length can be represented by the sum

$$s(\mathcal{P})=\sum_{n=1}^{N}|C(p_n)-C(p_{n-1})|. \tag{8}$$

The discretization error is given by,

$$\begin{aligned} e_d &= \mathcal{L}-s(\mathcal{P}) \\ &= \textstyle\int_0^N |C_p(p)|\,dp-\sum\limits_{n=1}^{N}|C(p_n)-C(p_{n-1})|. \end{aligned} \tag{9}$$

where obviously, $e_d\to 0$ when $N\to\infty$ (for further reading, the reader refers to [DC16]). Fig. 1 illustrates a general curve with A their discretized representation for better error visualization.

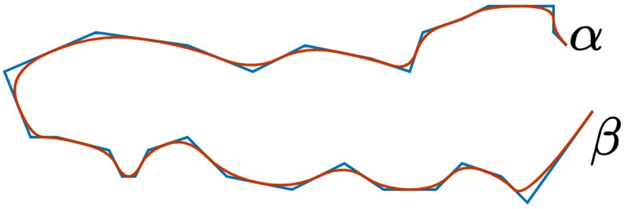

Figure 1: *Discretization.*

3. Learning Approach

3.1. Motivation

The motivation for using the DL model for this task lies in the core challenge of implementing equations (1) and (2) in real-life scenarios. These equations involve non-linearity and derivatives. Poor sampling and additive noise might lead to numerical errors [QW93]. The differential and integral operators can be obtained by using convolution filters [PWK16] and the summation can be represented using linear layers, which are highly common in DL models. The differential invariants can be interpreted as a high pass filter and the integration as a low pass filter. Hence, it is convenient to use a CNN alike model for our task. Another approach to deal with this task involves the Recurrent Neural Network (RNN), where the curve is considered a time-series [TB97, SJ19]. Our suggested architecture is based on a simplified CNN. As we aim to reconstruct the length axioms, (3)-(6), each of them is considered in the model establishment pipeline: from unique dataset generation **3.2**, through loss function design **3.3**, and the architecture structure **3.4**.

3.2. Dataset Generation

The reconstruction of the length properties was made in a supervised learning approach, where many curve examples with their lengths as labels were synthetically created. Each curve is represented by $2\times N$ vector for the x and y coordinates and a fixed number of points N. We created a dataset with $500,000$ to enable DL-based model establishment. This large number of examples aimed to cover curve transformations and to satisfy different patterns. These curves were created by considering four standard anchor geometric shapes (circles, straight lines, triangles, and rectangles), as shown in Figure 2. We scaled, rotated, translated, and then segmented them randomly into two segments, as shown in Figure 3. We performed 10 different splits for each anchor curve and sampled it in the forward and backward directions. In order to increase the dataset variety, an oscillating vector with random frequency and amplitude was added. This vector direction was defined as perpendicular to the shape's bounding curve. These steps were applied on the curve parametric analytic representation for a uniform sampling, defined on the $[0,1]$ interval. To enforce smoothness, the curve is convolved with a Gaussian kernel, creating differentiable curves even for the case of a rectangle and triangle. The ground truth length (label) of the shapes was calculated via 1^{st} order approximation between every two successive samples, where

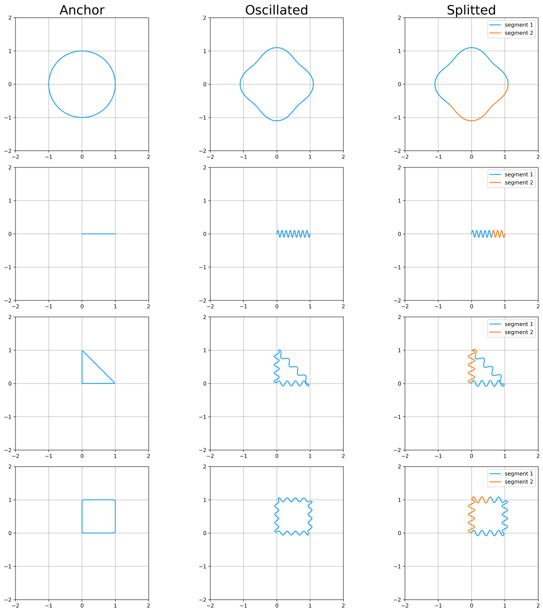

Figure 2: *Dataset generation: Curve examples of four anchor shapes.*

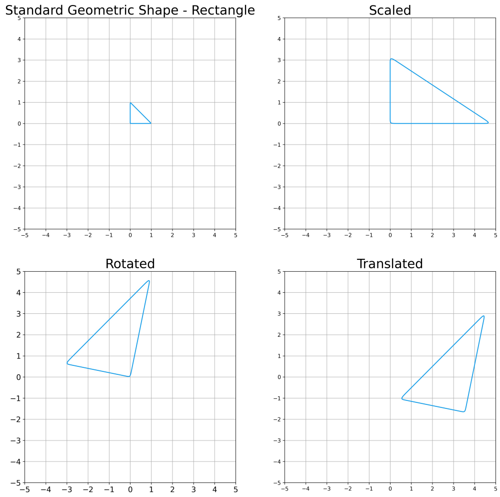

Figure 3: *Curve transformation example.*

we applied oversampling of 2 × 100N . This process allows the reconstruction of **Additivity** and **Invariance** axioms.

3.3. Loss function

The optimization problem was formulated as a supervised learning task. The loss function was design to meet the **Additivity** axiom. For each example, we aim to minimize the following

$$J_k = \|\mathcal{L}(s_1) - O(s_2) - O(s_3)\|_k^{\delta} + \lambda \left\|\Theta_{ij}\right\|_2^2, \tag{10}$$

where s_1, s_2, and s_3 are the input curves that hold the equality $\mathcal{L}(s_1) = \mathcal{L}(s_2) + \mathcal{L}(s_3)$, O is the DL model output, k is the example index, λ is a regularization parameter, Θ_{ij} are the various model weight, and $\|\cdot\|_2^{\delta}$ is the norm, where $\delta \in [1,2]$. The optimization task is to reconstruct the equality:

$$\mathcal{L}(s_1) = O(s_2) + O(s_3) \tag{11}$$

Minimizing J_k by passing many examples from the dataset, **3.1**, through the model, tuning its weights iteratively where eventually (10) and (11) coincide on the test set. The optimized resulting model is characterized by the optimal weights provided by

$$\Theta^*_{ij} = \arg\min_{\Theta_{ij}} \sum_k J_k. \tag{12}$$

In this work, we considered two options of the loss function norm, δ: the L_1 norm and the L_2 norm. The L_2 norm is very common to use for minimizing the mean-squared error (MSE) between the true label and the predicted one. The L_1 norm (Manhattan Distance) is very common to use for minimizing the Least Absolute Deviations (MAD) between the true label and the predicted one. We trained two models, one with L_1 norm and the second with L_2 norm. The motivation for including the L_1 norm in the loss is our interest in a model that calculates length. Hence, the relative error is of our concern. Our hypothesis was that L_1 outperforms the L_2, as we indeed obtained at **3.5**.

3.4. LengthNet Architecture

A simplified CNN was designed as a predictor for this supervised learning task. The baseline model receives as an input a 200 × 2 array representing discrete samples along the curve. It is inserted into two one-dimensional convolution layers (Conv1D). Both with a small kernel of size 3. The first with 24 filters and the second with 12 filters. They are connected with the Rectified Linear Unit (ReLU) activation function. Both Conv1D layers are followed by another ReLU and a linear layer of 2,352 neurons which finally outputs the length. The architecture is shown in Figure 4.

We tested several variations upon this baseline, e.g., models with additional batch normalization layers. Still, we found that for this simple task, a simple and shallow model achieves satisfying results.

3.5. LengthNet Training

Our training set consisted of 400,000 examples created from 400 shapes (100 for each anchor shape). Each specific shape was rotated and segmented in various ways to achieve invariance over those transformations creating the final dataset of size 400,000. The test set consisted of 100,000 examples created from 100 different shapes in the same manner. Training of this architecture for both loss functions, (10), was done using the ADAM optimizer [KB14] with a learning rate of 1e-3 with a constant decay rate of 0.99 and a batch size of 128, which were set after parameter tuning. Both models (with L_1 norm and L_2 norm) were trained by passing many examples in small batches with a back-propagation method. The training process was carried out in batches of 128 examples for 400

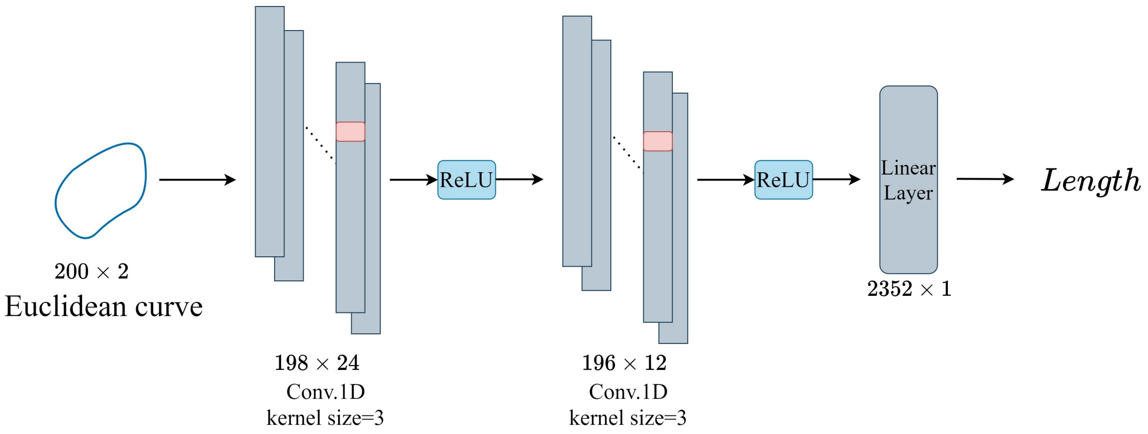

Figure 4: *LengthNet architecture: A general curve is inserted into the model for their length estimation. There, two 1D convolutional layers, followed by one linear layer were considered with two ReLU activation functions.*

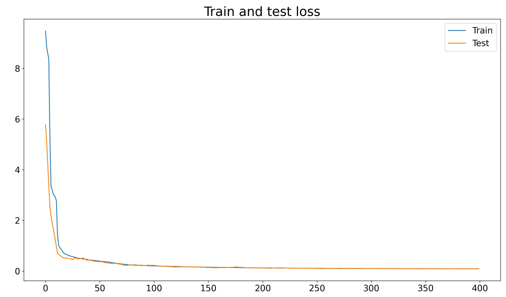

Figure 5: *LengthNet training*

epochs. The chosen CNN-based architecture with L_1 norm for the loss function was named LengthNet. Various parameters are provided in Table 1. Figure 5 shows a graph of the train and test losses as a function of the number of epochs.

Table 1: *Learning Parameters*

Description	Symbol	Value
Nunber of examples	K	500,000
Train/test ratio	-	80/20
Regularization parameter	λ	0.01
Partition parameter	N	200
Batch size	-	128
Learning rate	η	0.001
Decay rate	-	0.99
Epochs	-	400

4. Results and Discussion

The LengthNet was well established after 400 epochs. In order to validate the LengthNet, we used the Root MSE (**RMSE**) measure, as also the RMSE-Over-Length (**ROL**) measure, defined as

$$\mathbf{ROL} = \frac{\mathbf{RMSE}}{\mathcal{L}}, \tag{13}$$

This measure provides a normalized error with respect to the curve's length. As we deal with various curves of different lengths, we must appropriately weigh their errors. Figure 6 provides the ROL histograms of L_1 norm based model and L_2 norm based model. As shown, L_1 loss function based-model clearly outperforms L_2 loss function based-model according to the ROL measure.

4.1. Architectures Comparison

We compared the LengthNet performance with several architectures. A Batch-Norm was added to the LengthNet architecture, after every convolutional layer. Once with L1 loss and once with L2 loss. This modification did not improve the test loss, relatively to LengthNet test loss. We also tried the long short term memory (LSTM) architecture, which are commonly used for sequential data. The LengthNet outperformed LSTM with both L1 and L2 loss. Results are summarized in Table 2.

The with Batch-Norm, and obtained large relative error of

Table 2: *Architectures Comparison*

Model	Test loss	ROL
LengthNet (CNN+L1 loss)	0.239	0.021
CNN+L2 loss	0.254	0.029
CNN+L1 loss + Batch-Norm	0.360	0.039
CNN+L2 loss + Batch-Norm	11.36	0.230
LSTM +L1 loss	1.768	0.166
LSTM +L2 loss	2.370	0.099

4.2. Monotonic Property

A linear relation was established between the true length and the LengthNet (Fig.7). The x-axis represents the ground truth length,

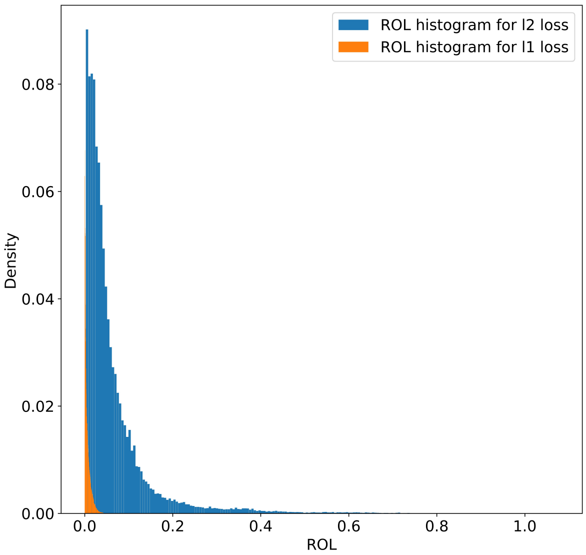

Figure 6: *ROL histograms. L_1 norm vs. L_2 norm comparison. L_1 loss function based-model clearly outperforms L_2 loss function based-model*

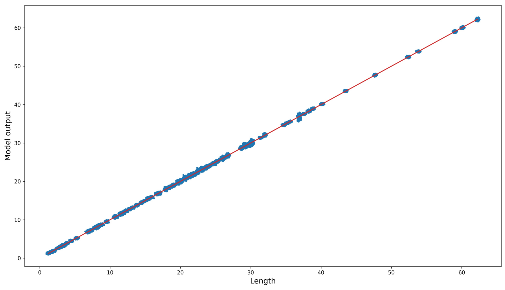

Figure 7: *LengthNet monotonic property assessment*

and the y-axis is the predicted length by the LengthNet. This result shows the generalization capability and, in particular, shows the success of reconstructing the (**Monotonic**) axiom.

4.3. Comparison to first-order Spline interpolation

A classical approach for length calculation uses the first order spline interpolation (yielding a length calculation equivalent to (8)). A comparison between the 1st order spline interpolation and the LengthNet was made. The length of all test set examples were calculated, once by the 1st order spline interpolation and once by the LengthNet. The results are presented in Figure 8, where the relative error histograms are shown for each. The LengthNet histogram is mostly below 2.5%, while for the 1st order linear spline, half of the relative error is concentrated around 2.5%, relatively wide spread.

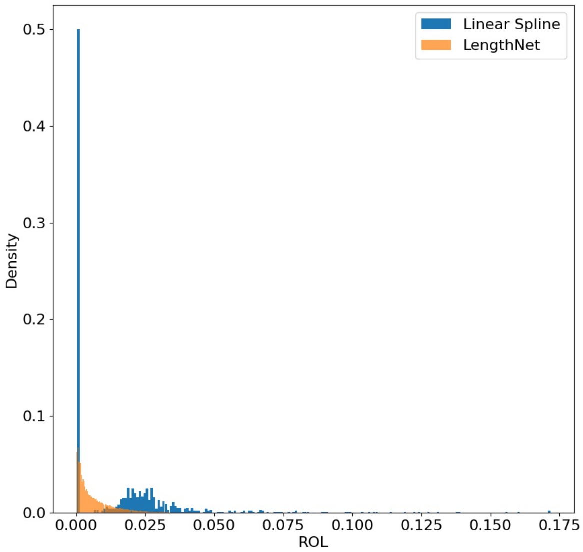

Figure 8: *LengthNet vs. spline interpolation relative error histograms.*

4.4. Noise Robustness

We check the capability of the LengthNet to estimate the length of curves with additive white Gaussian noise. We set the standard deviation of the noise to $\lambda\bar{d}$, where $\bar{d}$ is the mean distance between two successive points along the curve and $\lambda \in [0,1]$ is the noise magnitude parameter. Figure 9 shows two curves with their associated noisy curves, with $\lambda = 0.5$. We compared the performance of our model with linear spline interpolation. The LengthNet and the linear spline sensitivity to additive noise are presented in Figure 10 (blue and green plots). For a low level of additive noise, the LengthNet predicts the curve length property pretty well. Only for noise magnitude of 0.3 a relative error of over 10% is obtained. The relative error of the LengthNet is much lower than the linear approximation for most of the data. The ability to be robust to noise, even though the LengthNet didn't see any noisy example, is a good capability.

In addition, we add a low pass filter (LPF) to smooth the noise for both approaches (orange and purple in Figure 10). We obtained better results, where the suggested LengthNet outperforms the linear spline for noise magnitude up to $\lambda = 0.7$.

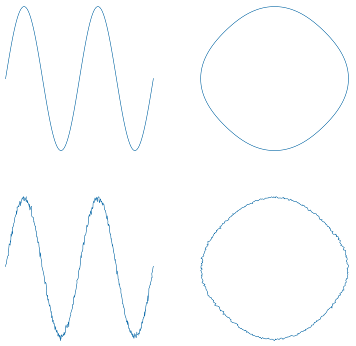

Figure 9: *Noisy vs. original shapes. Var parameters=0.5*

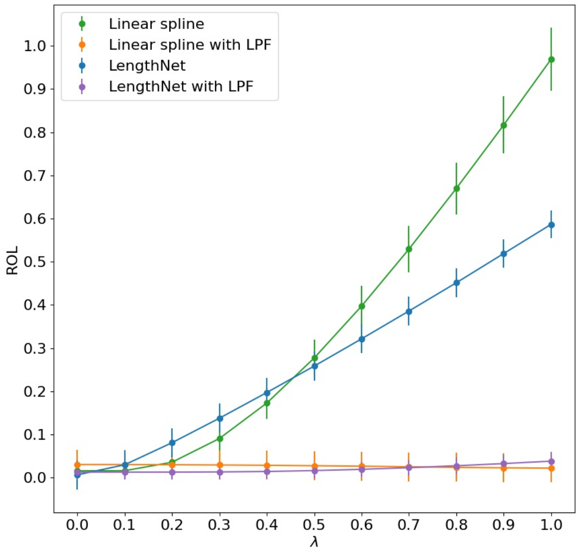

Figure 10: *LengthNet vs. linear approximation sensitivity to additive noise with and without LPF*

4.5. Generalization for unseen examples: Lame curves

In order to evaluated model generalization capabilities, we present the Lame curves family (super-ellipse), given by:

$$\left|\frac{x}{a}\right|^r + \left|\frac{y}{b}\right|^r = 1 \tag{14}$$

where a and b were set to 1, for simplicity, and r is the shape parameter. Note, when $r \leq 1$ the curve has non-differentiable points. Some curves from the family are provided in figure 11. We implemented the parametric equation for 39 different curves where $r \in [0.5, 10]$ with steps of 0.25. Then, we passed each curve trough the LengthNet. Results are shown in Figure 12.

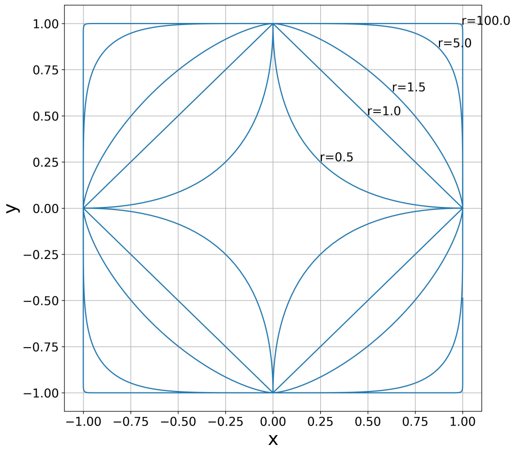

Figure 11: *Lame curves family.*

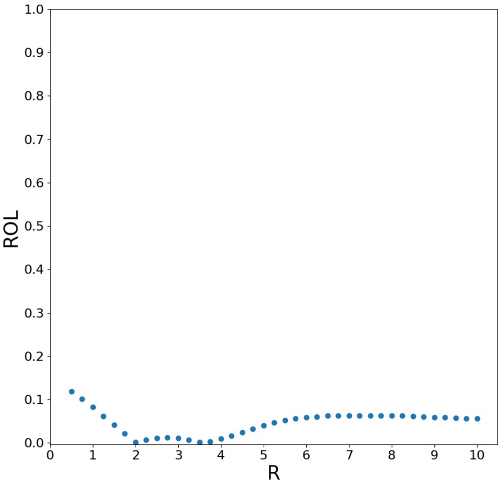

Figure 12: *Lame curves: Relative error of the LengthNet as a function of r.*

5. Conclusions

A learning-based approach to reconstruct the length of curves was presented. The power of the deep learning based model to reconstruct the fundamental axioms was demonstrated. There, a very simplified architecture was designed to deal with sequential data. We have shown that the norm L_1 is more appropriate for this problem than the common L_2 norm in the loss function formulation. Furthermore, by comparison to the linear approximation, we see how the LengthNet deals with noisy examples, even though it hasn't been trained on noisy data. Currently, the LengthNet does not deal well with high noise magnitude. Future challenges: we aim to generalize a DL model with the capabilities of taking a level set from a given image and performing accurate length calculation using more examples, such as the outline of the human figure. For that, we aim to formulate the problem as an unsupervised learning (or self-supervised learning) task. Also, we may include some transformations, such as affine, equi-affine, and homography transformations.

Acknowledgements

The first author would like to thanks Prof. Roni Kimmel, from the Technion - Israel Institute of Technology, for introducing him to this fascinating problem. Both authors would like to thank Dr. Maxim Freydin, from ALMA Technologies LTD, and Dr. Chaim Baskin from the Technion - Israel Institute of Technology, for assistance with editing this paper.

References

[BBL*17] Bronstein M. M., Bruna J., LeCun Y., Szlam A., Vandergheynst P.: Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine 34*, 4 (2017), 18–42.

[BN18] Berg J., Nyström K.: A unified deep artificial neural network approach to partial differential equations in complex geometries. *Neurocomputing 317* (2018), 28–41.

[Chi20] Chicco D.: Siamese neural networks: An overview. *Artificial Neural Networks* (2020), 73–94.

[DC16] Do Carmo M. P.: *Differential geometry of curves and surfaces: revised and updated second edition.* Courier Dover Publications, 2016.

[GP90] Guenter B., Parent R.: Computing the arc length of parametric curves. *IEEE Computer Graphics and Applications 10*, 3 (1990), 72–78.

[HC98] Hellweg H.-B., Crisfield M.: A new arc-length method for handling sharp snap-backs. *Computers & Structures 66*, 5 (1998), 704–709.

[KB14] Kingma D. P., Ba J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[LBH15] LeCun Y., Bengio Y., Hinton G.: Deep learning. *nature 521*, 7553 (2015), 436–444.

[OTPH16] Ooi S. Y., Teoh A. B. J., Pang Y. H., Hiew B. Y.: Image-based handwritten signature verification using hybrid methods of discrete radon transform, principal component analysis and probabilistic neural network. *Applied Soft Computing 40* (2016), 274–282.

[PWK16] Pai G., Wetzler A., Kimmel R.: Learning invariant representations of planar curves. *arXiv preprint arXiv:1611.07807* (2016).

[QW93] Qian S., Weiss J.: Wavelets and the numerical solution of partial differential equations. *Journal of Computational Physics 106*, 1 (1993), 155–175.

[SJ19] Smagulova K., James A. P.: A survey on lstm memristive neural network architectures and applications. *The European Physical Journal Special Topics 228*, 10 (2019), 2313–2324.

[TB97] Tsoi A. C., Back A.: Discrete time recurrent neural network architectures: A unifying review. *Neurocomputing 15*, 3-4 (1997), 183–223.