Differentiable rasterization of minimum-time sigma-lognormal trajectories

Daniel Berio
Goldsmiths, Univ. of London
London, United Kingdom
daniel.berio@gold.ac.uk

Sylvain Calinon
Idiap Research Institute
Martigny, Switzerland
sylvain.calinon@idiap.ch

Réjean Plamondon
Ecole Polytechnique de Montréal
Montréal, Canada
rejean.plamondon@polymtl.ca

Frederic Fol Leymarie
Goldsmiths, Univ. of London
London, United Kingdom
ffl@gold.ac.uk

***Abstract*— We present an adaptation of the sigma-lognormal model to generate and fit smooth trajectories in conjunction with a differentiable vector graphics (DiffVG) rendering pipeline and with parameter selection driven by a minimum-time smoothing criterion. This approach enables the incorporation of the "Kinematic Theory of Rapid Human Movements" into modern image-based deep learning systems. We demonstrate its utility through various applications, including fitting handwriting trajectories to an image and generating trajectories using guidance from a large multimodal model.

Keywords*— *Sigma-lognormal model, Differentiable Vector Graphics, DiffVG, curve generation.

I. Introduction and background

Recent advances in automatic differentiation [1] and differentiable rendering [2] have enabled the development of numerous methods that use complex image-based costs to optimize parameters for 2D and 3D parametric shape primitives. In the 2D computer graphics domain, differentiable vector graphics (DiffVG) rasterization [3] supports most primitives in the Scalable Vector Graphics (SVG) format, allowing gradients to propagate through the transformation of these primitives into pixels of an image. This capability has facilitated various methods for image vectorization [4], stylization and abstraction [5, 6], as well as sketch generation [7, 8, 9].

One limitation of the SVG format for representing drawings, sketches and handwriting is its reliance on cubic Bézier curves. With this representation, longer curves are constructed by joining multiple polynomial segments at their endpoints, a representation that struggles to capture both the degree of continuity and intrinsic variability that characterize human movement. In previous work [10, 11], we demonstrated that the sigma-lognormal ($\Sigma\Lambda$) model provides a valid alternative to Bézier curves to model handwriting and drawing traces. Additionally, we showed that the model's formulation can be adapted to enable an intuitive user interface similar to those typically used in computer-aided curve design.

In this paper, we show that with minimal adaptations, a similar approach can be integrated with a DiffVG engine. This integration is particularly valuable in combination with various deep-learning techniques, enabling a wide range of potential applications using modern image generation and processing methods to drive the creation of $\Sigma\Lambda$ trajectories. We demonstrate two practical examples combining DiffVG with a minimum-time smoothing criterion [12]: (i) an approach that infers $\Sigma\Lambda$ parameters based on image similarity, similarly to existing methods [10, 13, 14], and (ii) a method for generating handwriting-like trajectories using a pre-trained multimodal (text and image) model [9].

II. Method

On the basis of the "Kinematic Theory of Rapid Human Movements" [15] we describe a planar trajectory using a weighted-displacement formulation of the $\Sigma\Lambda$ model [16]. The model describes the planar evolution of a handwriting trajectory through the space-time superposition of a discrete number of stroke primitives, each having a characteristic "bell-shaped" speed profile modeled with a two-parameter lognormal. The spatial layout of a trajectory is described with a high-level motor plan consisting of an initial position $\boldsymbol{p}_0$ followed by a sequence of "virtual targets" $(\boldsymbol{p}_1, \dots, \boldsymbol{p}_m)$ (Fig. 1a and 1b). Each virtual target describes a ballistic stroke aimed along the orientation θ_i of the vector $\boldsymbol{p}_i - \boldsymbol{p}_{i-1}$. Assuming movements are performed with rotations of the wrist or elbow, each stroke follows a circular arc geometry determined by a turning angle parameter δ_i (Fig. 1c). The time parametrized kinematics $\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t)$ of a trajectory can be computed as linear combinations

$$\boldsymbol{x}(t) = \boldsymbol{p}_0 + \sum_{i=1}^{m} \boldsymbol{x}_i(t) \quad \text{and} \quad \dot{\boldsymbol{x}}(t) = \sum_{i=1}^{m} \dot{\boldsymbol{x}}_i(t) \tag{1}$$

of m submovements with position $\boldsymbol{p}_0 + \boldsymbol{x}_i(t)$ and velocity $\dot{\boldsymbol{x}}_i(t)$.

The speed profile of each stroke follows a lognormal function (Fig. 1d), and the overall trajectory results from the superposition of multiple strokes over time. The smoothness of the trajectory depends on the activation time and overlap of consecutive lognormals (Fig. 1a and 1b). If a new lognormal starts before the previous one is zero, the superposition creates a smooth transition between the two strokes. A greater overlap results in a smoother transition.

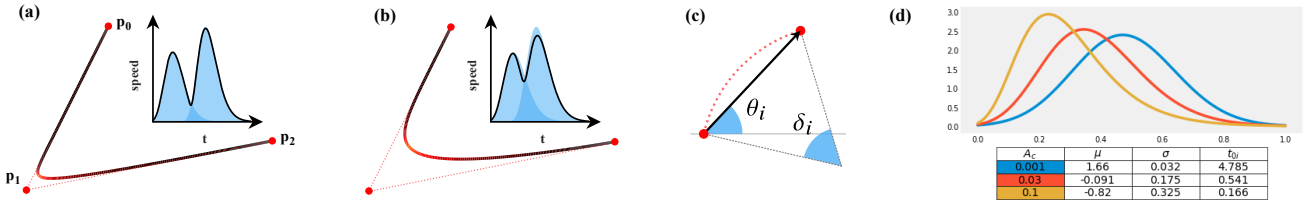

Fig. 1. **(a)** and **(b)**: The effect of different time overlaps for two lognormals, where a greater time overlap results in a smoother trajectory. The degree of overlap is indicated along the black trajectory with a red shade and the trajectory speed is filled in blue. The motor plan is made up of virtual targets (red dots) linked by their distance separation (dotted red). **(c)**: Stroke orientation θ_i and turning angle δ_i. The latter determines the internal angle of the assumed circular arc. **(d)**: Controlling lognormal shape and asymmetry using the intermediate parameter A_c and $T = 1$. The lognormals are shifted by $-t_{0i}$ for visualization, so that their onset matches at $t = 0$.

To maintain compatibility with an automatic differentiation pipeline we compute the lognormal with:

$$\Lambda_i(t) = \frac{1}{\sigma_i\sqrt{2\pi}\,\varphi(t-t_{0i})} \exp\left(-\frac{1}{2{\sigma_i}^2}(\ln(\varphi(t-t_{0i})) - \mu_i)^2\right), \tag{2}$$

where we use a rectifier linear unit $\varphi(x) = \text{ReLU}(x-\epsilon)+\epsilon$, using a small ϵ to avoid values ≤ 0 in the logarithm. The parameters t_{0i}, μ_i and σ_i, respectively determine the activation time, delay and response time of the lognormal.

To compute the curvilinear evolution of each stroke we use the following integral:

$$w(t) = \int_0^t \Lambda_i(u)du = \frac{1}{2}\left[1 + \text{erf}\left(\frac{\ln\left(\varphi\left(t-t_{0i}\right)\right) - \mu_i}{\sigma_i\sqrt{2}}\right)\right] \quad \in [0,1]\,, \tag{3}$$

which can be computed explicitly using the error function (erf). The velocity of each stroke is then given by:

$$\dot{\boldsymbol{x}}(t)_i = D_i\Lambda_i(t)\begin{bmatrix}\cos(\delta_i w(t) - \frac{\delta_i}{2} + \theta_i)\\ \sin(\delta_i w(t) - \frac{\delta_i}{2} + \theta_i)\end{bmatrix} \qquad \text{with} \qquad D_i = \left\|\boldsymbol{p}_i - \boldsymbol{p}_{i-1}\right\| \text{ sinc}\left(\frac{\delta_i}{2\pi}\right)^{-1}, \tag{4}$$

where D_i determines the distance traveled by a submovement along a circular arc while the use of the normalized sine cardinal (aka sinc) function avoids numerical errors as the turning angle approaches zero. The position along each stroke can be computed explicitly as a displacement with respect to the initial position $\boldsymbol{p}_0$ and without requiring numerical integration of Eq. 4 with:

$$\boldsymbol{x}_i(t) = D_i\begin{bmatrix}\frac{\cos(\theta_{0i}+\delta_i w_i(t))-\cos(\theta_{0i})}{2\sin(\delta_i/2)}\\ \frac{\sin(\theta_{0i}+\delta_i w_i(t))-\sin(\theta_{0i})}{2\sin(\delta_i/2)}\end{bmatrix} \quad \text{if } |\delta_i| > \epsilon, \quad \boldsymbol{x}_i(t) = D_i\begin{bmatrix}\cos(\theta_i)w_i(t)\\ \sin(\theta_i)w_i(t)\end{bmatrix} \quad \text{otherwise, and} \quad \theta_0 = \theta - \frac{\pi+\delta_i}{2}.$$

III. Reparameterization

While the lognormal function works remarkably well in describing the form of human movement speed profiles [17], its parameters μ_i and σ_i influence its mode and shape while also influencing the time occurrence of the activation parameter t_{0i}. To facilitate the optimization procedure that follows, we reparameterize each submovement with: (i) a stroke duration T_i, (ii) a relative time offset Δt_i with respect to the previous stroke time occurrence and duration, and (iii) a shape parameter $A_{c_i} \in (0,1)$, which defines the skewness of the lognormal [18]. The $\Sigma\Lambda$ parameters $\{\mu_i, \sigma_i, t_{0i}\}$ are then computed with:

$$\sigma_i = \sqrt{-\ln(1-A_{c_i})}, \qquad \mu_i = 3\sigma_i - \ln(\frac{-1+e^{6\sigma_i}}{T_i}), \qquad t_{0i} = t_{0i-1} + \Delta t_i \sinh(3\sigma_i). \tag{5}$$

With this formulation, the parameter Δt_i explicitly determines the smoothness of a trajectory near a virtual target, with lower values producing a greater overlap with the previous lognormal and a consequently smoother trajectory near the corresponding virtual target. Furthermore, a strictly positive Δt_i guarantees an ordering of the lognormals, which allows us to compute the end time of the trajectory with:

$$T_{\text{end}} = t_{0m} + e^{\mu_m + 3\sigma_m}\,. \tag{6}$$

This, in turn, allows us to easily compute a minimum-time cost for the optimization procedure that follows.

IV. Differentiable rasterization

On the basis of DiffVG's support for cubic Bézier curves, we can render an approximation of the $\Sigma\Lambda$ trajectory by leveraging the equivalent Hermite formulation of curves [19], where a cubic polynomial curve is described by two point and tangent pairs. Given the planar trajectory positions $\boldsymbol{x}(t)$ and velocities (tangents) $\dot{\boldsymbol{x}}(t)$ along with two consecutive samples t_i and $t_i + dt$, the four control points for the corresponding Bézier curve segments are simply obtained as:

$$[\boldsymbol{x}(t_i),\ \boldsymbol{x}(t_i) + 3dt\dot{\boldsymbol{x}}(t_i),\ \boldsymbol{x}(t_i + dt),\ \boldsymbol{x}(t_i + dt) - 3dt\dot{\boldsymbol{x}}(t_i + dt)] \quad ,$$

where the factors $3dt$ ensure that the control points are positioned to match the desired endpoint velocities.

We compute a fixed number of samples, n, for each rendered trajectory using $dt = T_{\text{end}}/n$. We find that, in practice, using five times as many samples as targets ($n = 5m$) gives a sufficiently good approximation for our use cases. Note that even though the number of samples is fixed, the trajectory duration changes depending on the values of Δt_i.

All the trajectory construction procedures described to this stage, including conversion to Bézier, result in a graph of differentiable operations that are seamlessly handled by standard automatic differentiation engines such as the one provided by PyTorch [1]. This, combined with the ability to rasterize an approximation of the trajectories in a differentiable manner, allows us to optimize the motor plan positions $\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_m$, the time offsets Δt_i, the turning angle parameters δ_i and, as an option, the shape parameters A_c. The latter are set with respect to a compound loss that balances trajectory smoothness, in a geometric sense, with an image-based objective. For all the examples that follow, we consider an optimization objective determined by the following compound loss:

$$\mathcal{L} = \lambda T_{\text{end}} + \mathcal{L}_{\mathbb{I}},$$

where T_{end} acts as a smoothing term by rewarding overlap between consecutive lognormals, $\mathcal{L}_{\mathbb{I}}$ is an image-based cost depending on the application, and λ is a user-defined and application-dependent scalar weight that controls the tradeoff between the two terms. In the following sections, we demonstrate two examples of variants of $\mathcal{L}_{\mathbb{I}}$ for different applications. We implement the optimization procedure in PyTorch and use the Adam optimizer for 300 steps. To aid convergence, we find it beneficial to apply a cosine annealing schedule to the learning rate and to linearly increase the value of λ from zero to its user-specified value over the first 30 steps, ensuring that the initial optimization steps are principally focused on the image-space term.

A. *Fitting trajectories to an image*

As a baseline test for our method, we test fitting $\Sigma\Lambda$ trajectories to an image, resulting in a procedure similar to the geometry-based fitting methods described by Berio et al. [10] and Ferrer et al. [14]. We start by using tag traces from the Graffiti Analysis database [20], a dataset of graffiti tags recorded with low-cost DIY tracking devices. We render the input trajectory into a black-and-white image and then create an initial motor plan for generating a $\Sigma\Lambda$ trajectory by applying the discrete curve evolution (DCE) polyline simplification method [21] to the input trace (Fig. 2). DCE simplifies polylines by

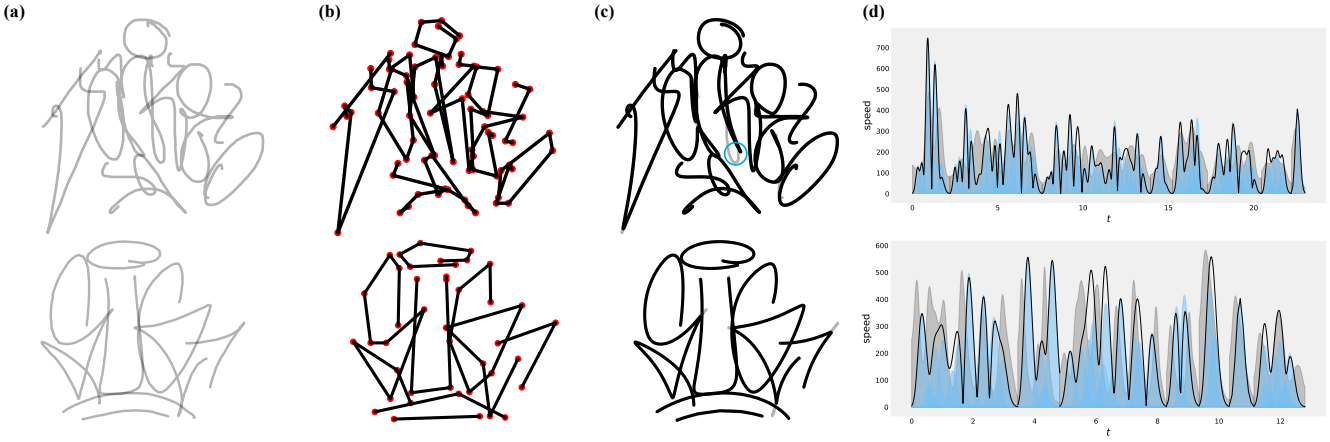

Fig. 2. Image-based trajectory reconstruction using Eq. 7. **(a)** Reconstructing an input trajectory (grey) for two graffiti tags ("JANKE" and "SUG"). **(b)** We start from the motor plan (red) and trajectory (black) computed with initial values of $\Delta t_i = 1$ and $\delta_i = 0$. The motor plan and trajectory initially match. **(c)** Trajectory optimization results . For the top row we optimize also the shape parameters A_c while for the bottom row we keep A_c fixed. A significant reconstruction error (top row, 3rd image, blue circle) occurs on the lower part of the "K", likely due to the presence of an inflection. **(d)** The generated speed profiles with the corresponding lognomals in blue and the original (Gaussian smoothed) speed profiles in grey. The latter are not taken into consideration, but the minimum time cost produces a similar number and location of peaks.

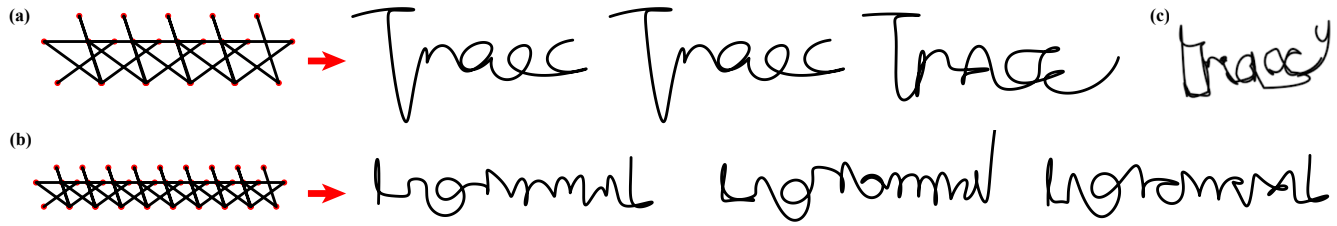

Fig. 3. CLIP-guided trajectory generation. **(a)** and **(b)**, left: We start from a motor plan (red) and a trajectory (black) with $\Delta t_i = 1$, which initially overlaps almost exactly with the motor plan. Here, the motor plan is created by concatenating multiple copies of a self intersecting polygon; however, different user-defined polygons can be used. **(a)** and **(b)**, right: Trajectories resulting from different optimizations using Eq. 8 and the target words "TRACE" and "lognormal". **(c)**, top right: Optimizing the control points of cubic Bézier curves instead of the $\Sigma\Lambda$ model parameters to generate the word "TRACE" also does not produce a fully legible output and further results in a trace with multiple discontinuous overlapping segments.

identifying significant curvature extrema, with a result that is similar to that of Berio et al. [10] and De Stefano et al. [22]. We then set $\mathcal{L}_{\mathbb{I}}$ to a multi-scale mean squared error (MSE) between the rendered image $\mathbb{I}$ and the target $\hat{\mathbb{I}}$:

$$\mathcal{L}_{\mathbb{I}} = \sum_{s \in S} \text{MSE}(\mathbb{I}_s, \hat{\mathbb{I}}_s) \ . \tag{7}$$

The MSE term consists of multiple downscaled and blurred versions $\mathbb{I}_s$ and $\hat{\mathbb{I}}_s$ of the input, where $s \in S$ represents different scaling factors applied to the images. We find that this multiscale approach improves convergence while decreasing the risk of ending in local minima. Note that the optimization infers a plausible motion from the input geometry and that is done differently from most state-of-the-art $\Sigma\Lambda$ fitting methods [13, 23], as our method does *not attempt* to reproduce the kinematics of the input.

B. CLIP-driven handwriting generation

In a second application of our method, we test an approach similar to ClipDraw [8] to guide the generation of a trace that is consistent with a given text caption or "prompt" (Fig. 3). We set

$$\mathcal{L}_{\mathbb{I}} = -\langle f(\mathbb{I}), f(c) \rangle \ , \tag{8}$$

where $\langle \cdot \rangle$ denotes the cosine similarity measured between the embeddings $f(\mathbb{I})$ of the rendered image and the embeddings $f(c)$ of the text caption. To construct $f(\cdot)$, we use the finetuned CLIPAG model [9], which has been shown to give significantly better results than the original CLIP model for image generation tasks. We use a caption "{WORD}, neatly handwritten" where {WORD} is a word we wish the trace to be similar to. Interestingly, while the generated text is not fully legible, it possesses a structure that resembles the desired word shape, suggesting that the method partially captures the high-level features of handwriting style and word formation.

V. Conclusion

We have demonstrated how $\Sigma\Lambda$ trajectories can be constructed and approximated as piecewise cubic Bézier curves through a sequence of differentiable operations. When combined with a DiffVG rendering pipeline, this approach enables the optimization of trajectory parameters using cost functions defined in image space. In this paper, we have presented two example applications: image-based trajectory fitting and CLIP-driven trajectory generation. Beyond these, numerous image-based objectives and metrics [e.g. as found in 5, 6, 24] can be used in a similar way, offering good opportunities for future research. For the fitting procedure, we rely on a correctly ordered starting trajectory. Combining our method with others that infer ordered traces from bitmap images [25] or shape outlines [26] is also an interesting avenue of future research.

Additionally, we have shown that the $\Sigma\Lambda$ parametrization allows for a straightforward computation of trajectory duration, which can be effectively used as a smoothing cost under the assumption of minimum-time as a performance criterion. The results suggest potential applications of the $\Sigma\Lambda$ model in path planning and robotics, where minimum-time is a widely used optimization objective [27].

While this paper has focused on differentiable rendering, a similar minimum-time cost function could be integrated with a geometric similarity measure for trajectory optimization. Although our initial goal was to enable creative applications of the $\Sigma\Lambda$ model, our results suggest that the proposed method is generally a useful way to reconstruct handwriting samples in terms of $\Sigma\Lambda$ parameters. In future developments, we plan to test our method on samples from handwriting signature databases with a higher sampling rate. This will allow to validate the reconstruction accuracy and efficiency of our method, while comparing it with state-of-the-art $\Sigma\Lambda$ reconstruction methods such as robust XZERO [23] and iDeLog [14].

ACKNOWLEDGEMENTS

This work has been partly funded by the EACVA (Embodied Agents in Contemporary Visual Art: eacva.org) project jointly led by Goldsmiths — under grant no. AH/X002241/1 from the UKRI/AHRC — and the University of Konstanz — under grant no. 508324734 from the Deutsche Forschungsgemeinschaft (DFG).

REFERENCES

[1] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in Proc. of the 33rd Intl. Conf. on Neural Information Processing Systems, Red Hook, NY, USA: Curran Associates Inc., 2019.
[2] H. Kato, D. Beker, M. Morariu, T. Ando, T. Matsuoka, W. Kehl, and A. Gaidon, "Differentiable Rendering: A Survey," July 2020.
[3] T. Li, M. Lukáč, M. Gharbi, and J. Ragan-Kelley, "Differentiable vector graphics rasterization for editing and learning," ACM Trans. on Graphics, vol. 39, pp. 1–15, Dec. 2020.
[4] X. Ma, Y. Zhou, X. Xu, B. Sun, V. Filev, N. Orlov, Y. Fu, and H. Shi, "Towards Layer-wise Image Vectorization," in 2022 IEEE/CVF Conf. on Comp. Vision and Pattern Recognition (CVPR), (New Orleans, LA, USA), pp. 16293–16302, IEEE, June 2022.
[5] S. Iluz, Y. Vinker, A. Hertz, D. Berio, D. Cohen-Or, and A. Shamir, "Word-As-Image for Semantic Typography," Mar. 2023.
[6] Y. Vinker, E. Pajouheshgar, J. Y. Bo, R. C. Bachmann, A. H. Bermano, D. Cohen-Or, A. Zamir, and A. Shamir, "CLIPasso: Semantically-aware object sketching," ACM Trans. Graph., vol. 41, July 2022.
[7] X. Xing, C. Wang, H. Zhou, J. Zhang, Q. Yu, and D. Xu, "DiffSketcher: Text Guided Vector Sketch Synthesis through Latent Diffusion Models," Jan. 2024.
[8] K. Frans, L. Soros, and O. Witkowski, "CLIPDraw: Exploring text-to-drawing synthesis through language-image encoders," in Advances in Neural Information Processing Systems (S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, eds.), vol. 35, pp. 5207–5218, Curran Associates, Inc., 2022.
[9] R. Ganz and M. Elad, "CLIPAG: Towards Generator-Free Text-to-Image Generation," in 2024 IEEE/CVF Winter Conf. on Applications of Comp. Vision (WACV), (Waikoloa, HI, USA), pp. 3831–3841, IEEE, Jan. 2024.
[10] D. Berio, F. F. Leymarie, and R. Plamondon, "Kinematics reconstruction of static calligraphic traces from curvilinear shape features," in The Lognormality Principle and Its Applications in E-Security, e-Learning and e-Health (R. Plamondon, A. Marcelli, and M. Ã. Ferrer, eds.), vol. 88 of Series in Machine Perception and Artificial Intelligence, ch. 11, pp. 237–268, Dec. 2020.
[11] D. Berio, F. F. Leymarie, and R. Plamondon, "Expressive curve editing with the sigma lognormal model," in Proc. of the 39th Annual European Assoc. for Comp. Graphics Conf.: Short Papers, EG, (Goslar, DEU), pp. 33–36, Eurographics Assoc., 2018.
[12] H. Tanaka, J. W. Krakauer, and N. Qian, "An optimization principle for determining movement duration," J. of Neurophysiology, vol. 95, no. 6, pp. 3875–3886, 2006.
[13] R. Plamondon, C. O'reilly, J. Galbally, A. Almaksour, and A. Anquetil, "Recent developments in the study of rapid human movements with the kinematic theory: Applications to handwriting and signature synthesis," Pattern Recognition Letters, vol. 35, pp. 225–235, Jan. 2014.
[14] M. A. Ferrer, M. Diaz, C. Carmona-Duarte, and R. Plamondon, "iDeLog: Iterative dual spatial and kinematic extraction of sigma-lognormal parameters," IEEE trans. on pattern analysis and machine intelligence, vol. 42, no. 1, pp. 114–125, 2018.
[15] R. Plamondon, C. O'Reilly, C. Remi, and T. Duval, "The lognormal handwriter: Learning, performing and declining.," Frontiers in Psychol., vol. 4, no. 945, 2013.
[16] D. Berio, F. Fol Leymarie, and P. R., "Computer aided design of handwriting trajectories with the kinematic theory of rapid human movements," in 18th Biennial Conf. of the Intl. Graphonomics Society, 2017.
[17] B. Rohrer and N. Hogan, "Avoiding spurious submovement decompositions II," Biological cybernetics, vol. 94, no. 5, pp. 409–14, 2006.
[18] R. Plamondon et al., "A kinematic theory of rapid human movement. Part IV," Biological Cybernetics, vol. 89, no. 2, pp. 126–38, 2003.
[19] G. E. Farin, Curves and Surfaces for CAGD: A Practical Guide. Morgan Kaufmann Ser. Comput. Graph. Geom. Model., 2001.
[20] E. Roth, T. Watson, C. Sugrue, T. Vanderlin, and J. Wilkinson, "An open databse for graffiti markup language (GML) files," 2009.
[21] L. J. Latecki and R. Lakämper, "Discrete approach to curve evolution," in Mustererkennung 1998, pp. 85–92, Springer, 1998.
[22] C. De Stefano, M. Garruto, and A. Marcelli, "A saliency-based multiscale method for on-line cursive handwriting shape description," in Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004. Ninth Intl. Workshop On, pp. 124–129, IEEE, 2004.
[23] C. O'Reilly and R. Plamondon, "Automatic extraction of sigma-lognormal parameters on signatures," in Proc. of 11th Intl. Conf. on Frontier in Handwriting Recognition (ICFHR), 2008.
[24] X. Xing, H. Zhou, C. Wang, J. Zhang, D. Xu, and Q. Yu, "SVGDreamer: Text Guided SVG Generation with Diffusion Model," Apr. 2024.
[25] R. Plamondon and C. M. Privitera, "The segmentation of cursive handwriting: An approach based on off-line recovery of the motor-temporal information," IEEE Trans. on Image Processing, vol. 8, no. 1, pp. 80–91, 1999.
[26] D. Berio, F. F. Leymarie, P. Asente, and J. Echevarria, "StrokeStyles: Stroke-based segmentation and stylization of fonts," ACM Trans. Graph., vol. 41, Apr. 2022.
[27] J. Bobrow, S. Dubowsky, and J. Gibson, "Time-optimal control of robotic manipulators along specified paths," The Intl. J. of Robotics Research, vol. 4, no. 3, pp. 3–17, 1985.