Image-driven Robot Drawing with Rapid Lognormal Movements

Daniel Berio[1], Guillaume Clivaz[2], Michael Stroh[3], Oliver Deussen[3]
Réjean Plamondon[4], Sylvain Calinon[2] and Frederic Fol Leymarie[1]

Abstract— Large image generation and vision models, combined with differentiable rendering technologies, have become powerful tools for generating paths that can be drawn or painted by a robot. However, these tools often overlook the intrinsic physicality of the human drawing/writing act, which is usually executed with skillful hand/arm gestures. Taking this into account is important for the visual aesthetics of the results and for the development of closer and more intuitive artist-robot collaboration scenarios. We present a method that bridges this gap by enabling gradient-based optimization of natural human-like motions guided by cost functions defined in image space. To this end, we use the sigma-lognormal model of human hand/arm movements, with an adaptation that enables its use in conjunction with a differentiable vector graphics (DiffVG) renderer. We demonstrate how this pipeline can be used to generate feasible trajectories for a robot by combining image-driven objectives with a minimum-time smoothing criterion. We demonstrate applications with generation and robotic reproduction of synthetic graffiti as well as image abstraction.

I. INTRODUCTION

Collaborative robots are increasingly becoming a consumer technology, making them accessible to both professional and amateur artists as tools for creating physical artworks such as paintings, drawings, and calligraphy [18], [38]. Unlike standard machines like printers and plotters, articulated robotic arms introduce both new possibilities and challenges, such as the ability to mimic skilled, dynamic brushstrokes and adapt to various tools and surfaces.

In parallel, while AI-driven image processing systems are becoming a widely adopted tools by both artists and designers, physical and embodied AI (a.k.a. robots) currently remains a largely unexplored territory. Recent advances in differentiable vector graphics (DiffVG) are enabling to bridge this gap by extending deep learning image-generation systems from pixel-only outputs to vector-based representations. Specifically, this enables gradient-based optimization of curve and stroke parameters using image-based objectives [1]–[4], ranging from pixel-wise similarity to a reference image to complex objectives driven by the features of large vision and image generation models.

While the outputs of current DiffVG systems can be executed by a robot, these methods typically rely on para-metric curve representations [45] that do not account for the kinematics of the gestures a human would use to produce such strokes. At the same time, drawing, painting, and writing are human motor activities that often require rapid and skillful arm movements. These movements are learned through extensive practice and their kinematic qualities influence how ink or material is deposited on a canvas or a surface, ultimately affecting the aesthetic perception of the resulting trace [5].

In this paper, we show how DiffVG technology can be used to directly optimize the parameters of a computational model of human handwriting and drawing movements. We show how this can be used to constrain the generation of drawing gestures with a minimum-time smoothness criterion [6] that is consistent with motor control principles, while also enabling the generation of motions that are kinematically feasible by a robot. This is especially useful for robotic mark-making applications where rapid and human-like movement is desirable.

To describe the kinematics of hand and arm movements, we rely on the sigma-lognormal model [7], which describes hand/arm movements as the superposition of aiming sub-movements, each characterized with a speed profile mathematically modeled using a lognormal function. We provide a differentiable formulation of the model that can be easily integrated into existing DiffVG stroke generation pipelines. The output of our system produces trajectories that do not require an intermediate re-parameterization step and are feasible for a robot by only scaling and resampling according to given velocity and acceleration limits. We demonstrate how this is useful with three practical applications that combine a minimum-time trajectory smoothing criterion with a generation task: (i) fitting trajectories to images of graffiti tags, (ii) generating "pseudo" graffiti, and (iii) stylizing images by using a pre-trained multimodal model.

Our focus on reproduction of graffiti stems from the premise that it represents a form of "urban calligraphy" [8], where various stylizations and abstractions are applied to letterforms. Graffiti developed on the "moving canvas" of the New York City subway in the 1960s [9] and its aesthetics depend on the execution of rapid and skillful movements, making it a particularly interesting art form to investigate in a robotic setting.

II. BACKGROUND AND RELATED WORK

Differentiable rendering [10] enables the propagation of gradients through the transformation of 2D and 3D parametric primitives into pixels. In the 2D domain, DiffVG

[1]Goldsmiths, University of London, United Kingdom. `daniel.berio@gold.ac.uk, ffl@gold.ac.uk`

[2]Idiap Research Institute and École Polytechnique Fédérale de Lausanne (EPFL), Switzerland. `guillaume.clivaz@idiap.ch, sylvain.calinon@idiap.ch`

[3]University of Konstanz, Germany. `od@uni.kn, michael.stroh@uni-konstanz.de`

[4]Ecole Polytechnique de Montréal, Canada. `rejean.plamondon@polymtl.ca`

rendering [11] supports most primitives in the Scalable Vector Graphics (SVG) format, a capability that has facilitated the development of a wide variety of vector graphics generation methods that leverage the power of large pre-trained image generation models such as stable diffusion [12] or multimodal text-vision models such as Contrastive Language–Image Pretraining (CLIP) [13]. This has enabled a variety of applications ranging from image vectorization [14], to stylization and abstraction [4], [15], as well as sketch generation [2], [3], [16].

The DiffVG method currently supports piecewise quadratic and cubic Bézier curves, a representation that, without additional constraints, does not provide high-order continuity that characterizes human movements [17]. Indeed, most existing sketch/drawing generation in this domain optimize the control points of many short curve segments. Schaldenbrand et al. [18] developed a custom neural rasterizer that takes into account the sim-to-real gap when executing brushstrokes with a robot. However, their method only supports combinations of short brush strokes. In this paper, we demonstrate how an alternate representation can be used to generate long continuous paths that are similar to the ones that would be typically done by a human when drawing or writing.

Human hand and arm movements are intrinsically smooth and appear to follow a series of "invariants" [19]. Speed and curvature of human movements tend to show an inverse relation, which for certain types of movements takes the form of a power law [20], [21]. Intuitively, high curvature requires higher precision and leads to slower movement [22]. Movement time appears to be approximately independent of movement extent, a principle that is commonly referred to as *isochrony* [23], [24]. Human movements are smooth [25] and appear to follow optimality principles [26]. This has been explicitly modeled with cost functions such as minimizing the square magnitude of high-order positional derivatives [17], as well as movement duration [6]. Target-directed movements are characterized by a bell-shaped speed profile, and more complex movements can be described as a space-time superposition of multiple such sub-movements [27]. The *Kinematic Theory of Rapid Human Movements* (Kinematic Theory for short) has been developed by Plamondon et al. over the years [28], with various applications in handwriting analysis and generation [29], as well as computer-aided design of calligraphy and graffiti [30].

Similarly to our work, Wolniakowski et al. [31] reproduce $\Sigma\Lambda$ kinematics with an industrial robot, while Berio et al. [32] use the properties of the $\Sigma\Lambda$ model to incrementally adjust and reproduce graffiti-like trajectories with a 7 DoF compliant robot. We adopt a similar extension of the $\Sigma\Lambda$ model, enabling 2.5D motions with respect to the drawing pad. We use the vertical direction to model brush width and to vary tool pressure. A similar approach to brush width modeling is used by Fujioka et al. [33], who use uniform B-splines with a minimum jerk cost to reproduce Japanese calligraphy with a robot. A number of works have been developed for robotic reproduction of portraits with a variety of tools [34]–[37], a goal that we share in this paper. Over a series of works, Tresset et al. [38] use a low-cost planar robot arm for pen-based robotic portraits that are made with rapid pen-strokes that are incrementally added in a visual-feedback loop. Deussen et al. [39], Stroh et al. [40], and Schaldenbrand et al. [18] also use a visual feedback loop to place painterly brush strokes for robot painting. While we do not cover visual feedback scenarios in this work, our method is potentially applicable in this setting. While most of these works produce drawings consisting of many strokes, Singhal et al. [41] generate one-line portraits with a robot by creating paths with "TSP-art" [42], a method that combines weighted Voronoi sampling [43] of an image and paths generated by solving a traveling salesman problem (TSP). We use a similar strategy as an initialization for our optimization procedure.

III. METHOD

Similar to existing works using the $\Sigma\Lambda$ model for robotics applications [31], [32], we generate trajectories using a 2.5D extension of the model, enabling the reproduction of strokes with a varying thickness profile.

The $\Sigma\Lambda$ model describes the planar evolution of a handwriting trajectory through the space-time superposition of a discrete number of stroke primitives, each having the 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 $(\boldsymbol{p}_1, \ldots, \boldsymbol{p}_m)$ (Fig. 1a). Each virtual target describes a ballistic stroke aimed along a vector $\boldsymbol{d}_i = \boldsymbol{p}_i - \boldsymbol{p}_{i-1}$. Assuming movements done with rotations of the wrist or elbow, the planar component of each stroke follows a circular arc geometry determined by a turning angle parameter δ_i (Fig. 1c).

A. Trajectory construction

The time parametrized kinematics $\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t)$ of a trajectory can be computed as linear combinations

$$\begin{bmatrix} \boldsymbol{x}(t) \\ \dot{\boldsymbol{x}}(t) \\ \ddot{\boldsymbol{x}}(t) \end{bmatrix} = \begin{bmatrix} \boldsymbol{p}_0 \\ \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix} + \sum_{i=1}^{m} \begin{bmatrix} \boldsymbol{x}_i(t) \\ \dot{\boldsymbol{x}}_i(t) \\ \ddot{\boldsymbol{x}}_i(t) \end{bmatrix} \quad (1)$$

of m sub-movements with position $\boldsymbol{p}_0 + \boldsymbol{x}_i(t)$, velocity $\dot{\boldsymbol{x}}_i(t)$ and acceleration $\ddot{\boldsymbol{x}}_i(t)$.

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

To maintain compatibility with an automatic differentiation pipeline, we compute the lognormal with
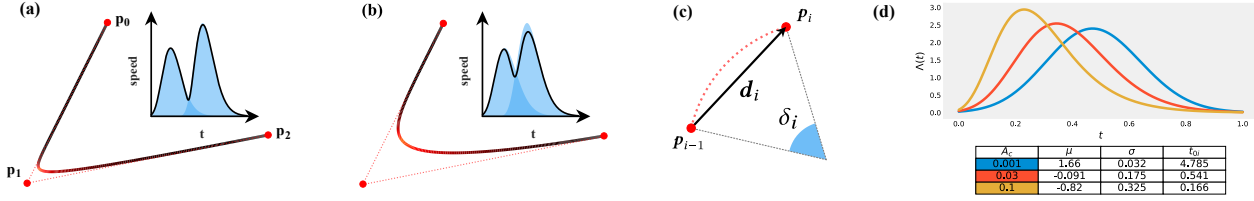
Fig. 1. **(a)** and **(b)**: The effect of different time overlaps for two lognormals, where a greater time overlap results in a smoother trajectory. The motor plan is made up of virtual targets (red dots) linked by their distance separation (dotted red). In blue, the resulting speed after executing the plan and black the generated trajectory. **(c)**: Sub-movement direction d_i and turning angle δ_i. The latter determines the internal angle of the assumed circular arc. **(d)**: Controlling lognormal shape/asymmetry using the intermediate parameter A_c and $T = 1$.

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

where we use a rectifier linear unit $\varphi(t) = \mathrm{ReLU}(t - \epsilon) + \epsilon$ with a small value 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.

We compute the curvilinear evolution using the integral of $\Lambda_i(u)$, which can be computed explicitly using the error function giving

$$w(t) = \frac{1}{2}\left[1 + \mathrm{erf}\left(\frac{\ln(\varphi(t-t_{0i})) - \mu_i}{\sigma_i\sqrt{2}}\right)\right], \quad (3)$$

with $w(t) \in [0, 1]$. Each sub-movement has velocity

$$\dot{x}(t) = \Lambda_i(t)\boldsymbol{H}\boldsymbol{R}[-\delta_i/2 + \delta_i w(t)]\boldsymbol{d}_i, \quad (4)$$

with rotation matrix

$$\boldsymbol{R}[\theta] = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \text{and where} \quad (5)$$

$$\boldsymbol{H} = \begin{bmatrix} h & 0 & 0 \\ 0 & h & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{with} \quad h = \mathrm{sinc}\left(\frac{\delta_i}{2\pi}\right)^{-1} \quad (6)$$

uses the normalized sine cardinal or sinc function to adjust for the circular arc length, while avoiding numerical errors as the turning angle δ_i approaches zero. The position along each sub-movement can be computed explicitly as a displacement with respect to the initial position $\boldsymbol{p}_0$ and without requiring numerical integration of (4) with

$$\boldsymbol{x}_i(t) = \boldsymbol{Z}\boldsymbol{d}_i + \boldsymbol{R}[\delta_i w(t) - \delta_i]\left(\frac{\boldsymbol{d}_i}{2} - \boldsymbol{M}\boldsymbol{d}_i\right) + \boldsymbol{M}\boldsymbol{d}_i,$$

$$\text{if } |\delta_i| > \epsilon,$$

$$\boldsymbol{x}_i(t) = w_i(t)\boldsymbol{d}_i, \quad \text{otherwise, with}$$

$$\boldsymbol{M} = \frac{1}{2\tan\frac{\delta_i}{2}}\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \boldsymbol{Z} = \mathrm{diag}([\frac{1}{2}, \frac{1}{2}, w_i(t) - 0.5]).$$

The stroke acceleration is given by

$$\ddot{\boldsymbol{x}}_i(t) = \delta_i\Lambda_i(t)\boldsymbol{H}\dot{\boldsymbol{R}}[\delta_i w(t) - \frac{\delta_i}{2}]\boldsymbol{d}_i$$

$$+ \dot{\Lambda}_i(t)\boldsymbol{H}\boldsymbol{R}[\delta_i w(t) - \frac{\delta_i}{2}]\boldsymbol{d}_i \quad (7)$$

where

$$\dot{\boldsymbol{R}}[\theta] = \begin{bmatrix} -\sin\theta & -\cos\theta & 0 \\ \cos\theta & -\sin\theta & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and}$$

$$\dot{\Lambda}_i(t) = \Lambda_i(t)\frac{\mu_i - \sigma_i^2 - \ln(t - t_{0i})}{\sigma^2(t - t_{0i})}.$$

Time parametrization. The parameters μ_i and σ_i influence the lognormal's 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 [44]. The $\Sigma\Lambda$ parameters $\{\mu_i, \sigma_i, t_{0i}\}$ are then computed with

$$\sigma_i = \sqrt{-\ln(1 - A_{c_i})}, \quad \mu_i = 3\sigma_i - \ln(\frac{-1 + e^{6\sigma_i}}{T_i}),$$

$$t_{0i} = t_{0i-1} + \Delta t_i \sinh(3\sigma_i). \quad (8)$$

With this intermediate formulation, the parameter Δt_i explicitly determines the overlap between consecutive lognormals and consequently the trajectory smoothness in the proximity of a virtual target: lower values produce a greater overlap with the previous lognormal and a smoother/faster trajectory. Furthermore, a strictly positive Δt_i guarantees an ordering of the lognormals, which allows us to compute the end time of the trajectory with

$$T_{\mathrm{end}} = t_{0m} + e^{\mu_m + 3\sigma_m}, \quad (9)$$

which facilitates the definition of a minimum-time cost in the following optimization procedure.

B. Differentiable rasterization

Cubic Bézier curves can be defined using the Hermite formulation [45] consisting of point-tangent pairs. This makes it straightforward to convert a $\Sigma\Lambda$ trajectory to a format compatible with DiffVG by considering the trajectory positions $\boldsymbol{x}(t)$ and velocities $\dot{\boldsymbol{x}}(t)$ and defining multiple Bézier curve segments between consecutive trajectory samples t_i and $t_i + dt$. The control points for each segment are given by:

$$\left[\boldsymbol{x}(t_i), \ \boldsymbol{x}(t_i) + 3dt\dot{\boldsymbol{x}}(t_i),\right.$$

$$\left.\boldsymbol{x}(t_i + dt) - 3dt\dot{\boldsymbol{x}}(t_i + dt), \ \boldsymbol{x}(t_i + dt)\right]. \quad (10)$$

We compute a fixed number n of samples for each rendered trajectory using $dt = T_{\text{end}}/n$. We find that using $n = 5m$ gives a sufficiently good approximation for our use cases. With this approach, the number of trajectory samples is fixed for each optimization step, while the trajectory duration changes depending on the varying values of Δt_i.

C. Executing $\Sigma\Lambda$ trajectories with a robot

We first transform the trajectories from the image coordinate system (expressed in pixels) to a rectangle on the horizontal plane of a 3D coordinate system (expressed in meters). For 2.5D trajectories with a varying width profile, we convert the stroke radius to a perpendicular distance from the drawing pad with a linear mapping. We compute this mapping empirically, but this can be done automatically using visual feedback and a method such as the one proposed by Gülzow et al. [46]. We then resample the trajectories using a time step that ensures predefined speed and acceleration limits $v_{\max}$ and $a_{\max}$. Finally, we transform the trajectory into the drawing pad's coordinate system. We determine the drawing area and plane by manually guiding the robot to the position of four points on the drawing pad.

An optimal control problem formulation is used to plan for joint angle trajectories, by using an iterative linear quadratic regulator (iLQR) problem formulation solved with a Gauss–Newton iterative scheme [47], by considering end-effector orientation control and joint limits avoidance. The resulting joint angle reference trajectory is tracked by an impedance controller, resulting in a feedback controller with torque commands controlling the robot at a 1kHz rate.

IV. RESULTS

All the steps described to this stage, including the conversion to Bézier, consist of differentiable operations that are seamlessly handled by standard automatic differentiation engines such as the one provided by PyTorch [48]. This, combined with differentiable rasterization, allows us to optimize the motor plan positions $\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_m$, the time offsets Δt_i, the curvature parameters δ_i and optionally the shape parameters A_c with respect to a loss that trades off trajectory smoothness in a geometric sense with an objective expressed in pixel space. For all the examples that follow, we consider a compound loss

$$\mathcal{L} = \lambda\mathcal{L}_{\text{smooth}} + \mathcal{L}_{\mathbb{I}}, \tag{11}$$

where $\mathcal{L}_{\text{smooth}}$ 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. We formulate the smoothing term

$$\mathcal{L}_{\text{smooth}} = T_{\text{end}} + w_\sigma \text{Var}[\boldsymbol{\Delta t}], \tag{}$$

where the second term $\text{Var}[\boldsymbol{\Delta t}]$ is the variance of the Δt_i parameters and encourages uniform spacing in time between lognormals, with a weight w_σ that we set to 50 in our experiments. We observe that this is particularly important
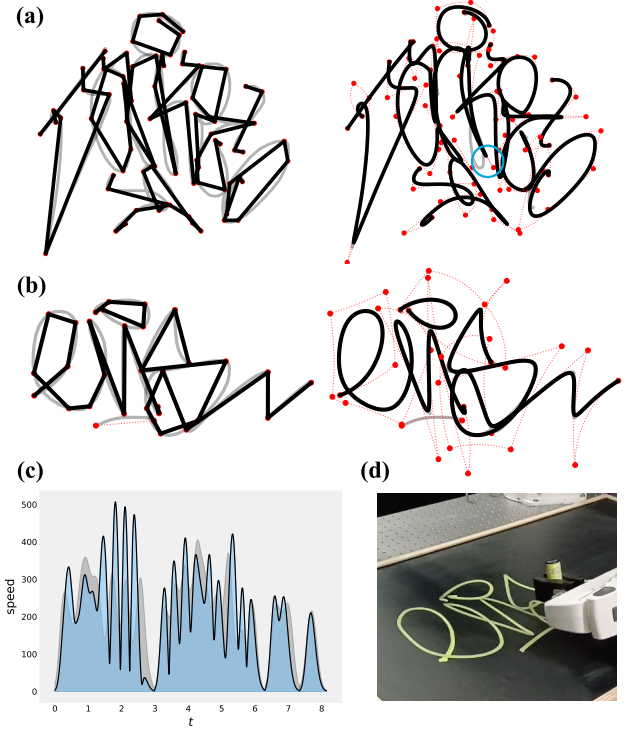

Fig. 2. Image-driven trajectory reconstruction using (12). **(a)** Reconstructing an input trajectory (grey) starting from the motor plan (red) and trajectory (black) on the left. We start with values of $\Delta t_i = 1$ and $\delta_i = 0$ and the motor plan and trajectory initially match. Here, we optimize also the shape parameters A_c resulting in the motor plan and trajectory on the right. In this instance, a significant reconstruction error (blue circle) occurs on the lower part of the "K", likely due to the presence of an inflection. **(b)** A second reconstruction keeping A_c fixed. **(c)** the generated speed profile for **(b)** with the corresponding lognomals in blue and the original (Gaussian smoothed) speed profile in grey. The latter is not taken into consideration, but the minimum time cost produces a similar number and location of peaks. **(c)** Reproduction with a 7-axis Franka robot and a chalk marker.

to maintain stability for long trajectories (with a high m), while encouraging a form of isochrony, which is consistent with empirically observed motor control principles [24] and also with the Kinematic Theory, which predicts a decrease in variance as the motor control increases [49]. In the following sections, we demonstrate two examples of variants of $\mathcal{L}_{\mathbb{I}}$ for different applications, while always maintaining the same smoothing cost $\mathcal{L}_{\text{smooth}}$.

We implement the optimization procedure in PyTorch and use the Adam optimizer for 300 steps running on a NVIDIA GeForce RTX 3060. We test our trajectories with different painting tools, using a 7-axis Franka robot in two different configurations (inclined and horizontal drawing planes), see accompanying video.

A. Fitting trajectories to an image

As a baseline test for our method, we test the fitting $\Sigma\Lambda$ trajectories to an image (Fig. 2), resulting in a procedure similar to the geometry-based fitting methods described by [7] and [50]. Similar to [7], we use tag traces from the Graffiti Analysis database [51], a dataset of graffiti tags recorded with low-cost DIY tracking devices. We render the input trajectory
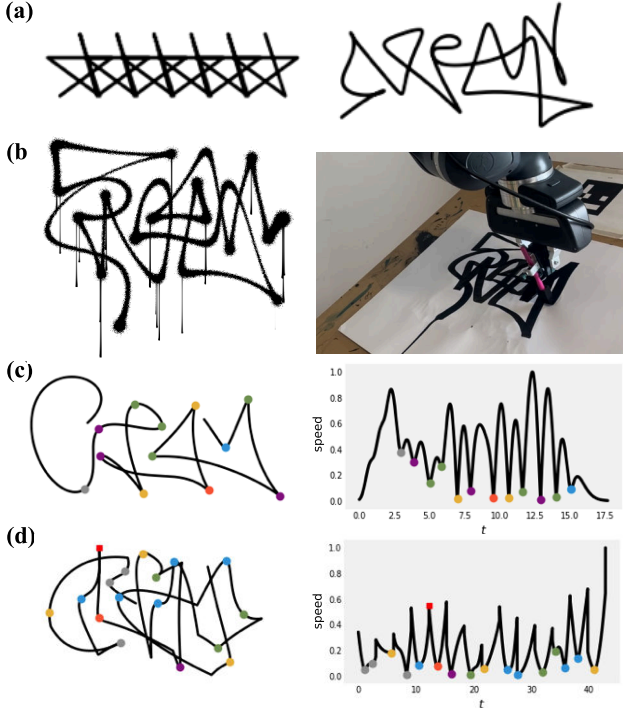
Fig. 3. CLIP-guided "pseudo"-tag generation. **(a)** Left, initial trajectory with $\Delta t_i = 1$; right, a trajectory resulting from the optimization using (13) with the word "CREAM". **(b)** Left: using the speed profile to simulate a trace made with spray paint using Gaussian samples along the trajectory; the standard deviation and density of samples are inversely proportional to speed; drips are added for effect near speed minima. Right: the same trajectory executed with a robot equipped with a sponge brush. **(c)** Speed minima (color coded) along a $\Sigma\Lambda$ trajectory generated with our method. **(d)** Speed minima for a piecewise cubic Bézier curve generated with the same method but without time-based smoothing. Note that certain curvature extrema (e.g. the one indicated with the red square) do not correspond to speed maxima, which would lead to an unnatural motion [20].

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 [52] to the input trace (Fig. 2). DCE simplifies polylines by identifying significant curvature extrema, with a result that is similar to the approaches of [7] and [53]. We then set

$$\mathcal{L}_{\mathbb{I}} = \sum_{s \in S} \text{MSE}(\mathbb{I}_s, \hat{\mathbb{I}}_s) \qquad (12)$$

to a multi-scale mean squared error (MSE) between the rendered image $\mathbb{I}$ and the target $\hat{\mathbb{I}}$. 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 getting stuck in local minima. Note that the optimization infers a plausible motion from the input geometry. Different from most state-of-the-art $\Sigma\Lambda$ fitting methods [29], [54], our method does not attempt to reproduce the kinematics of the input.

B. CLIP-driven graffiti tag generation

In a second application of our method, we test an approach similar to ClipDraw [2] to guide the generation of a trace that
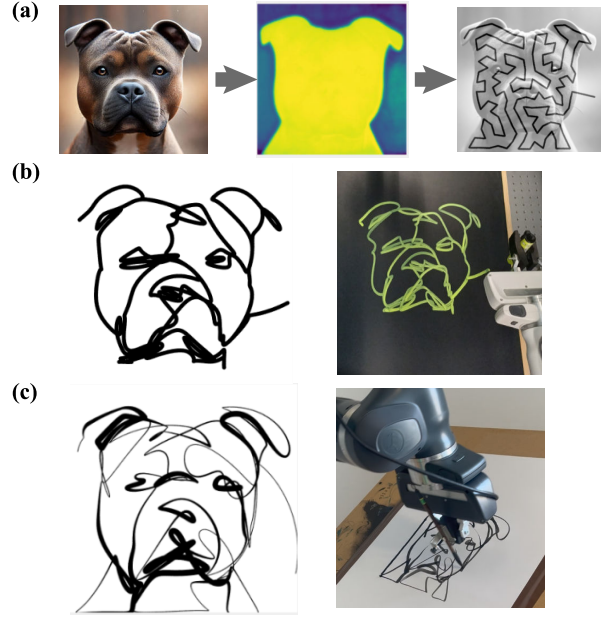

Fig. 4. Single path CLIP-guided image abstraction. **(a)** Left: an input image of a dog (AI-generated); middle: saliency map for the image; right: TSP initialization path with 150 virtual targets. **(b)** A fixed-width image abstraction and the resulting reproduction with a chalk marker. **(c)** A varying width image abstraction and the resulting reproduction using a brush dipped in Indian ink.

is consistent with a given text caption or *prompt* (Fig. 3) provided by the user. We set

$$\mathcal{L}_{\mathbb{I}} = -\langle f(\mathbb{I}), f(c) \rangle, \qquad (13)$$

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. We use a caption "{WORD}, a stylish graffiti tag", 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 and appears similar to instances of existing graffiti.

C. CLIP-driven image abstraction

In a second CLIP-driven task, we plug our method into the *CLIPasso* image abstraction method proposed by Vinker et al. [4], enabling the generation and robotic execution of *single stroke image abstractions*. We use the L2 norm between the activations of selected internal CLIP layers CLIP_l:

$$\mathcal{L}_{\mathbb{I}} = \sum_l \lambda_l \left\| \text{CLIP}_l(\hat{\mathbb{I}}) - \text{CLIP}_l(\mathbb{I}_\theta) \right\|_2^2, \qquad (14)$$

where $\mathbb{I}$ is the rendered image and $\hat{\mathbb{I}}$ is the target image. Similarly to CLIPasso, we start with a saliency map of the input to initialize points, but we use a different path initialization strategy (Fig. 4a). We use a saliency map derived from the last-layer activations of the OneFormer panoptic image segmentation model [55] trained on the COCO dataset [56]. We then initialize a user-defined number of points using
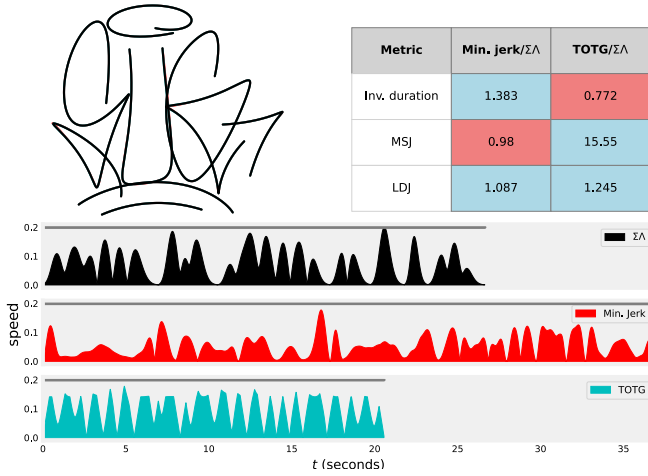
Metric	Min. jerk/ΣΛ	TOTG/ΣΛ
Inv. duration	1.383	0.772
MSJ	0.98	15.55
LDJ	1.087	1.245

Fig. 5. Comparing a reconstruction using the ΣΛ model with minimum jerk [57] and TOTG [58]. Left: the ΣΛ trajectory and its reparametrizations superposed. The trajectories are nearly identical geometrically, so only the ΣΛ trajectory is visible in the figure. Middle: the speed profiles for ΣΛ , minimum jerk, and TOTG parametrizations. Note that while TOTG is faster, the resulting speed has triangular and trapezoidal segments that will produce discontinuous acceleration. Right: table comparing inverse trajectory duration, mean square jerk (MSD) [59] and log-dimensionless jerk (LDJ) [25] for the three parameterizations using the ratios minimum jerk/ΣΛ and TOTG/ΣΛ . Values > 1 show a better performance of ΣΛ in the metric.

weighted Voronoi sampling [43] on the saliency map. Then, we connect the resulting points using a TSP [42]. We find that setting all layer weights to zero with the exception of $\lambda_2 = 0.5, \lambda_3 = 0.5, \lambda_6 = 1.0$ to work well for our use cases.

D. Evaluation

We performed a quantitative comparison (Fig. 5) with two different well-known trajectory parameterizations: (i) time-optimal trajectory generation (TOTG) [58], which is widely used to plan efficient motions in robotic applications and (ii) the path-constrained minimum jerk model [57], a generalization of the minimum jerk model [17] to arbitrarily complex paths, well known in the motor control literature. Both methods take a path as an input and produce a reparametrization of the path. While TOTG produces a trajectory subject to the given maximum velocity and acceleration limits, minimum jerk does not. As a result, we resample the resulting path according to the same limits.

We compared all methods in terms of trajectory duration (given the imposed limits) and smoothness according to two trajectory smoothness measures: mean squared jerk (MSJ) [59] and log-dimensionless jerk (LDJ) [25]. The latter has been proposed as a robust measure of trajectory smoothness due to its invariance to scale and duration. Our results show that TOTG is faster than ΣΛ but significantly less smooth according to both smoothness measures. At the same time, ΣΛ trajectories are less smooth than min. jerk according to the MSJ measure but smoother according to the more recently proposed LDJ measure. This suggests that ΣΛ provides a good tradeoff for movements that must be rapid but smooth, making it a strong candidate for applications such as drawing and writing, where these two aspects are both important.

V. CONCLUSION

We demonstrated how a 2.5D extension of the ΣΛ model can be integrated into a DiffVG to enable image-guided optimization of smooth trajectories with kinematics that can be tracked by a drawing/painting robot. The ΣΛ model construction provides a straightforward way to compute trajectory duration, which enables the implementation of a smoothing cost in terms of minimum time. Minimum-time is a widely used optimization objective [60] in robotic path planning, and the ΣΛ formulation provides a representation that effectively supports this measure while producing movements that are kinematically similar to those made by a human [29]. In quantitative comparisons, we showed that ΣΛ provides a good tradeoff for movements that must be rapid but smooth. Evaluating the perceived naturalness and smoothness of the trajectories is an interesting avenue of future research, which links to existing work that evaluates the relation between the perceived naturalness of drawing movements and the aesthetic appreciation of the resulting traces [5].

REFERENCES

[1] P. Schaldenbrand, Z. Liu, and J. Oh, "StyleCLIPDraw: Coupling Content and Style in Text-to-Drawing Translation," Feb. 2022.

[2] 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. Curran Associates, Inc., 2022, pp. 5207–5218.

[3] X. Xing, C. Wang, H. Zhou, J. Zhang, Q. Yu, and D. Xu, "Diffsketcher: text guided vector sketch synthesis through latent diffusion models," in *Proceedings of the 37th Intl. Conf. on Neural Information Processing Systems*, ser. NIPS '23. Curran Associates Inc., 2023.

[4] 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, no. 4, July 2022.

[5] R. Chamberlain, D. Berio, V. Mayer, K. Chana, F. F. Leymarie, and G. Orgs, "A dot that went for a walk: People prefer lines drawn with human-like kinematics," *British Journal of Psychol.*, vol. 113, no. 1, 2021.

[6] 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.

[7] 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*, ser. Series in Machine Perception and Artificial Intelligence, R. Plamondon, A. Marcelli, and M. Á. Ferrer, Eds., Dec. 2020, vol. 88, ch. 11, pp. 237–268.

[8] A. Arte, *Forms of Rockin': Graffiti Letters and Popular Culture*. Dokument Press, 2015.

[9] J. Kimvall, *The G-word*. Stockholm: Dokument, 2014.

[10] H. Kato, D. Beker, M. Morariu, T. Ando, T. Matsuoka, W. Kehl, and A. Gaidon, "Differentiable Rendering: A Survey," July 2020.

[11] T. Li, M. Lukáč, M. Gharbi, and J. Ragan-Kelley, "Differentiable vector graphics rasterization for editing and learning," *ACM Trans. on Graphics*, vol. 39, no. 6, pp. 1–15, Dec. 2020.

[12] A. Jain, A. Xie, and P. Abbeel, "Vectorfusion: Text-to-svg by abstracting pixel-based diffusion models," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*. IEEE, 2023, pp. 1911–1920.

[13] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proceedings of the 38th Intl. Conf. on Machine Learning*, vol. 139. PMLR, 18–24 Jul 2021, pp. 8748–8763.

[14] 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: IEEE, June 2022, pp. 16 293–16 302.

[15] S. Iluz, Y. Vinker, A. Hertz, D. Berio, D. Cohen-Or, and A. Shamir, "Word-as-image for semantic typography," *ACM Trans. Graph.*, vol. 42, no. 4, July 2023.

[16] 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: IEEE, Jan. 2024, pp. 3831–3841.

[17] T. Flash and N. Hogan, "The coordination of arm movements: An experimentally confirmed mathematical model," *The Journal of Neuroscience*, vol. 5, no. 7, pp. 1688–1703, July 1985.

[18] P. Schaldenbrand, J. McCann, and J. Oh, "Frida: A collaborative robot painter with a differentiable, real2sim2real planning environment," in *2023 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2023, pp. 11 712–11 718.

[19] D. A. Rosenbaum, *Human Motor Control*. Academic press, 2009.

[20] P. Viviani and R. Schneider, "A developmental study of the relationship between geometry and kinematics in drawing movements," *Journal of Experimental Psychol.: Human Perception and Performance*, vol. 17, no. 1, p. 198, 1991.

[21] R. Plamondon and W. Guerfali, "The 2/3 power law: When and why?" *Acta psychologica*, vol. 100, no. 1, pp. 85–96, 1998.

[22] E. Todorov and M. I. Jordan, "A minimal intervention principle for coordinated movement," in *Proceedings of the 15th Intl. Conf. on Neural Information Processing Systems*, ser. NIPS'02. Cambridge, MA, USA: MIT Press, 2002, pp. 27—34.

[23] M. I. Jordan and D. M. Wolpert, "Computational motor control," 1999.

[24] A. Thomassen and H.-L. Teulings, "Time, size and shape in handwriting: Exploring spatio-temporal relationships at different levels," in *Time, Mind, and Behavior*, J. Michon and J. Jackson, Eds. Springer, 1985, pp. 253–263.

[25] S. Balasubramanian, A. Melendez-Calderon, A. Roby-Brami, and E. Burdet, "On the analysis of movement smoothness," *Journal of NeuroEngineering and Rehabilitation*, vol. 12, no. 1, p. 112, Dec. 2015.

[26] S. E. Engelbrecht, "Minimum principles in motor control," *Journal of Mathematical Psychol.*, vol. 45, no. 3, pp. 497–542, 2001.

[27] P. Morasso, "Spatial control of arm movements," *Experimental Brain Research*, vol. 42, no. 2, pp. 223–7, 1981.

[28] R. Plamondon, "A kinematic theory of rapid human movements: Part i. movement representation and generation," *Biological Cybernetics*, vol. 72, no. 4, pp. 295–307, Mar 1995.

[29] R. Plamondon, C. O'Reilly, J. Galbally, A. Almaksour, and É. Anquetil, "Recent developments in the study of rapid human movements with the kinematic theory," *Pattern Recognition Letters*, vol. 35, pp. 225–35, 2014.

[30] 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*, ser. EG. Goslar, DEU: Eurographics Assoc., 2018, pp. 33–36.

[31] A. Wolniakowski, J. J. Quintana, M. Diaz, K. Miatliuk, and M. A. Ferrer, "Towards human-like kinematics in industrial robotic arms: a case study on a ur3 robot," in *2021 Intl. Carnahan Conf. on Security Technology (ICCST)*, 2021, pp. 1–5.

[32] D. Berio, S. Calinon, and F. Fol Leymarie, "Learning dynamic graffiti strokes with a compliant robot," in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Daejeon, Korea, Oct. 2016.

[33] H. Fujioka, H. Kano, H. Nakata, and H. Shinoda, "Constructing and reconstructing characters, words, and sentences by synthesizing writing motions," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 36, no. 4, pp. 661–670, 2006.

[34] S. Calinon, J. Epiney, and A. Billard, "A humanoid robot drawing human portraits," in *5th IEEE-RAS Intl. Conf. on Humanoid Robots, 2005.*, 2005, pp. 161–166.

[35] T. Löw, J. Maceiras, and S. Calinon, "drozbot: Using ergodic control to draw portraits," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 728–11 734, 2022.

[36] L. Scalera, S. Seriani, A. Gasparetto, and P. Gallina, "Non-photorealistic rendering techniques for artistic robotic painting," *Robotics*, vol. 8, no. 1, p. 10, Feb. 2019.

[37] D. Song, T. Lee, and Y. J. Kim, "Artistic pen drawing on an arbitrary surface using an impedance-controlled robot," in *2018 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2018, pp. 4085–4090.

[38] P. Tresset and F. Fol Leymarie, "Portrait drawing by paul the robot," *Computers & Graphics*, vol. 37, no. 5, pp. 348–363, 2013.

[39] O. Deussen, T. Lindemeier, S. Pirk, and M. Tautzenberger, "Feedback-guided stroke placement for a painting machine," in *Proc. of 8th Symp. on Comp. Aesthetics in Graphics, Visualization, and Imaging*, 2012, pp. 25–33.

[40] M. Stroh, J. M. Gülzow, and O. Deussen, "Semantic image abstraction using panoptic segmentation for robotic painting." Eurographics Association, 2023, pp. 133–140.

[41] A. Singhal, A. Kumar, S. Thukral, D. Raina, and S. Kumar, "Chitrakar: robotic system for drawing jordan curve of facial portrait," *arXiv preprint arXiv:2011.10781*, 2020.

[42] C. S. Kaplan and R. Bosch, "TSP art," in *Renaissance Banff: Mathematics, Music, Art, Culture*, 2005, pp. 301–308.

[43] A. Secord, "Weighted Voronoi stippling," in *Proceedings of the 2nd Intl. Symp. on Non-photorealistic Animation and Rendering*. Annecy France: ACM, June 2002, pp. 37–43.

[44] R. Plamondon, C. Feng, and A. Woch, "A kinematic theory of rapid human movement. Part IV," *Biological Cybernetics*, vol. 89, no. 2, pp. 126–38, 2003.

[45] G. E. Farin, *Curves and Surfaces for CAGD: A Practical Guide*, 5th ed., ser. Morgan Kaufmann Series in Comp. Graphics and Geometric Modeling. San Francisco, CA: Morgan Kaufmann, 2001.

[46] J. M. Gülzow, L. Grayver, and O. Deussen, "Self-improving robotic brushstroke replication," in *Arts*, vol. 7, no. 4. MDPI, 2018, p. 84.

[47] W. Li and E. Todorov, "Iterative linear quadratic regulator design for nonlinear biological movement systems," in *Proc. Intl Conf. on Informatics in Control, Automation and Robotics (ICINCO)*, 2004, pp. 222–229.

[48] 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.

[49] R. Plamondon, *The Lognormality Principle: A Personalized Survey*, ser. Series in Machine Perception and Artificial Intelligence. WORLD SCIENTIFIC, 2020, vol. Volume 88, pp. 1–39.

[50] 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.

[51] E. Roth, T. Watson, C. Sugrue, T. Vanderlin, and J. Wilkinson, "An open databse for graffiti markup language (GML) files," 2009.

[52] L. J. Latecki and R. Lakämper, "Discrete approach to curve evolution," in *Mustererkennung 1998*. Springer, 1998, pp. 85–92.

[53] 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*. IEEE, 2004, pp. 124–129.

[54] 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.

[55] J. Jain, J. Li, M. Chiu, A. Hassani, N. Orlov, and H. Shi, "OneFormer: One Transformer to Rule Universal Image Segmentation," 2023.

[56] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Doll'a r, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[57] E. Todorov and M. I. Jordan, "Smoothness Maximization Along a Predefined Path Accurately Predicts the Speed Profiles of Complex Arm Movements," *Journal of Neurophysiology*, vol. 80, no. 2, pp. 696–714, Aug. 1998.

[58] T. Kunz and M. Stilman, "Time-optimal trajectory generation for path following with bounded acceleration and velocity," in *Robotics: Science and Systems VIII*. The MIT Press, 07 2013.

[59] N. Hogan and D. Sternad, "On rhythmic and discrete movements: Reflections, definitions and implications for motor control," *Experimental Brain Research*, vol. 181, no. 1, pp. 13–30, July 2007.

[60] 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.