Example-Based Authoring of Expressive Space Curves

Jiří Minarčík

Independent Researcher, Praha, Czech Republic

Jakub Fišer

Adobe Research, 1 Old Street Yard, London, EC1Y 8AF, United Kingdom

Daniel Sýkora

Czech Technical University in Prague, Faculty of Electrical Engineering, Karlovo náměstí 13, Praha 2, 121 35, Czech Republic

Abstract

In this paper we present a novel example-based stylization method for 3D space curves. Inspired by image-based arbitrary style transfer (e.g., Gatys et al. [1]), we introduce a workflow that allows artists to transfer the stylistic characteristics of a short exemplar curve to a longer target curve in 3D—a problem, to the best of our knowledge, previously unexplored. Our approach involves extracting the underlying, unstyled form of the exemplar curve using a novel smoothing flow. This unstyled representation is then aligned with the target curve using a modified Fréchet distance. To achieve precise matching with reduced computational cost, we employ a semi-discrete optimization scheme, which outperforms existing methods for similar curve alignment problems. Furthermore, our formulation provides intuitive controls for adjusting stylization strength and transfer temperature, enabling greater creative flexibility. Its versatility also allows for the simultaneous stylization of additional attributes along the curve, which is particularly valuable in 3D applications where curves may represent medial axes of complex structures. We demonstrate the effectiveness of our method through a variety of expressive stylizations across different application contexts.

Keywords: Style Transfer, Curve Analogies, Space Curves, Ribbons
2020 MSC: 53A04, 65D17, 35K55, 49M41

1. Introduction

Example-based style transfer to images has become a prominent research area, largely driven by advancements in deep neural networks [1]. These techniques transfer the style of one image onto the content of another, effectively recreating the content image with the stylistic attributes of the style image while preserving its underlying structure. While significant progress has been made with raster images, example-based style transfer for 3D vector curves re-

Email addresses: jiri@minarcik.com (Jiří Minarčík), fiser@adobe.com (Jakub Fišer), sykorad@fel.cvut.cz (Daniel Sýkora)

URL: https://minarcik.com/ (Jiří Minarčík), https://research.adobe.com/person/jakub-fiser/ (Jakub Fišer), https://dcgi.fel.cvut.cz/home/sykorad (Daniel Sýkora)

mains largely unexplored. A foundational work in this area is Curve Analogies by Hertzmann et al. [2], which extends the concept of Image Analogies [3] to 2D curves. Given a stylized curve and its unstyled counterpart, this method can transfer the style to a target curve. Despite its impressive 2D results, Curve Analogies has not, to our knowledge, been extended to 3D. While Hertzmann et al. suggest the potential for extending their approach to 3D surfaces (as later demonstrated by Zelinka and Garland [4]), the concept of 3D curve analogies has not been specifically addressed. A key challenge in extending this approach to 3D is the requirement of an unstyled counterpart for each stylized curve. Even when such a counterpart exists, establishing a robust correspondence between the stylized and unstyled versions to avoid transfer artifacts is a complex problem.

In this paper, we formulate an alternative approach to example-based stylization of 3D curves that requires only the stylized and target curve as an input—like in the domain of arbitrary style transfer to raster images [1]. To avoid the need for an unstyled version of the stylization exemplar, we introduce a new smoothing flow specifically designed for 3D curves. This flow uses smooth tangent redistribution to enhance numerical stability, effectively generating a suitable unstyled counterpart. To establish precise correspondence between the unstyled and stylized curves, we develop a variant of the Fréchet distance that better captures overall matching quality. We minimize this distance using a semi-discrete optimization scheme that retains the style curve in the continuous domain. This enables us to achieve analytically precise matching and significantly reduce the computational overhead compared to the discrete method of Hertzmann et al. [2]. Our novel formulation also introduces two new intuitive user controls: (1) *style strength*, which scales the displacement map to modulate the stylization effect, and (2) *transfer temperature*, which influences the optimization scheme, allowing the users to better express their creative vision. Furthermore, our method can be extended to incorporate additional metric-based information, such as thickness, frame orientation, or drawing speed. This capability is particularly relevant in 3D applications, where curves can represent medial axes of tubes or ribbons. The efficacy of the proposed method is demonstrated through a range of expressive stylizations applied within diverse contexts.

2. Related Work

While research specifically addressing the style transfer to 3D curves remains limited, we leverage the extensive literature on stylization of raster images, 2D curves, and meshes. We focus on the methods that are most relevant to our work, specifically those that deal with example-based style transfer.

Image analogies, introduced by Hertzmann et al. in their seminal work [3], offered a novel and intuitive approach to image stylization. By using a single pair of images called *source* and *exemplar* the user can define a style transformation that can be subsequently applied to a given *target* image. While initially demonstrated on raster images, the concept was later extended to 2D curves [2]. In this method curves are modelled as polylines, representing style through curve statistics sampled at regular intervals along their length. However, extension to 3D was not addressed. One of the reasons for this omission can be the fact that specifying source and exemplar curves in 3D can be challenging for users. Moreover, automatic estimation of correspondences between source and exemplar curves is a delicate process that is further complicated by higher dimensionality. Inaccuracies in this step can introduce noticeable artifacts.

The follow-up approach of Zelinka and Garland [4] extends the original idea of curve analogies into 3D to modify the geometry of meshes, however, still the curve examples are mostly planar. Wu et al. [5] use curve analogies for motion editing and transfer, operating in a limited setting of space-time curves representing motion trajectories. Lang and Alexa [6] utilize a double chain Markov model [7] to stylize curves in an online fashion, where the output curve is generated by sampling from a distribution learned on a pair of user-supplied input and output curves. Their approach is still 2D, however, they allow for line overshoots, gaps, overdrawing, and support folding avoidance.

Numerous approaches such as the one by Bhat et al. [8] divert from curves and use filtered and unfiltered versions of meshes as source-exemplar pairs to transfer geometric texture detail. Similarly, Liu and Jacobson [9] use mesh surface normals to capture and transfer geometric detail from one mesh to another, and Berkiten et al. [10] use metric learning to find a combination of geometric features predicting detail-map similarities on the source mesh, and use their combination to drive the detail transfer.

Ma et al. [11] uniformly sample the model surfaces and find correspondences between the source and example, then compute a set of dominant similarity transformations between the source and the target, and, finally, use multi-label optimization to compute the best source-to-target analogies. This works well for man-made or engineered shapes but is not well-suited for more organic shapes.

Stylized curves can also be generated by segmenting the input curve and then rearranging the resulting pieces. Each segment undergoes some preprocessing before this rearrangement to create the desired stylized effect. Merrell et al. [12] use a graph data structure to represent how the parts of the curve connect and allows for branching, while Roveri et al. [13] rely on point samples with attributes extracted from the input structures, including curves. A particularly relevant scenario in this context is iterative application of a small texture sample along a curved path. Zhou et al. [14] use dynamic programming to find a globally optimal solution that can preserve the geometric continuity of structured patterns even in areas of high curvature, albeit at the cost of occasional local divergence from the prescribed curve geometry. Although the above mentioned approaches can work with 3D structures and do not require the specification of base curves, they keep the exemplar parts unchanged and just apply shape-preserving deformation on them.

Our approach draws inspiration from and shares similarities with the 2D raster stroke synthesis technique of Lukáč et al. [15] that employs randomized patch graph traversal and multi-level blending to generate seamless, non-repetitive textures. We demonstrate how to adapt this concept to the domain of example-based stylization of 3D vector curves, which can eliminate the requirement to specify source-exemplar pairs.

3. Preliminaries

In this work we consider open space curves Γ parametrized by a function $\gamma\colon I \to \mathbb{R}^3$ on a closed interval $I \subset \mathbb{R}$. We assume that the parameterization is regular (*i.e.*, $\|\partial_u \phi\|$ is strictly positive) and the curve has appropriate regularity (at least $\mathcal{C}^2$). When γ is parametrized by its arclength, we denote the parameter s instead of u and use ∂_s. We also define the family of curve $\{\Gamma_t\}_{t\in[0,t_m]}$ evolving on time interval $[0, t_m]$ with $t_m > 0$, which is defined by the joint parametric function $\gamma\colon I \times [0, t_m] \to \mathbb{R}^3$. We use $\partial_t \gamma$ to denote the partial derivative *w.r.t.* the time parameter t.

Moving Frames. The Frenet frame is denoted by $\{\mathbf{t}, \mathbf{n}, \mathbf{b}\}$ with $\mathbf{t}$, $\mathbf{n}$ and $\mathbf{b}$ standing for the tangent, normal and binormal vectors, *resp.* Unlike in the planar case, the Frenet normal vector $\mathbf{n}$ is not always well-defined and it can discontinuously change direction at inflection points where the curvature vanishes [16]. In some parts of our method, we thus adopt an alternative Rotation Minimising Frame (RMF) $\{\mathbf{t}, \mathbf{n}', \mathbf{b}'\}$ from [17].

Augmented Curves. We use the term augmented curve to denote the tuple (Γ, ϕ), where Γ is a space curve and $\phi\colon I \to \Omega_\phi$ is a map defined on the same domain I as the base curve Γ. The map ϕ returns values from an associated metric space Ω_ϕ with metric denoted by ρ_ϕ. For convenience, the notation of the arclength parameter s and arc length derivative ∂_s is extended to ϕ, *i.e.*, $\partial_s \phi = \|\partial_u \gamma\|^{-1} \partial_u \phi$.

Style Analogies. We define a finite collection of curves as a scene $\mathcal{S} = \{\Gamma^i\}_{i=1}^n$. The core of our method involves solving the problem of scene analogies defined, in the spirit of image analogies [3], as:

$$\begin{array}{ccccccc} \text{Guide scene} & & \text{Target scene} & & \\ \downarrow & & \downarrow & & \\ \mathcal{S}_G & : & \mathcal{S}_S & :: & \mathcal{S}_T & : & \mathcal{S}_R \\ & & \uparrow & & \uparrow & & \\ & & \text{Style scene} & & \text{Result scene} & & \end{array}$$

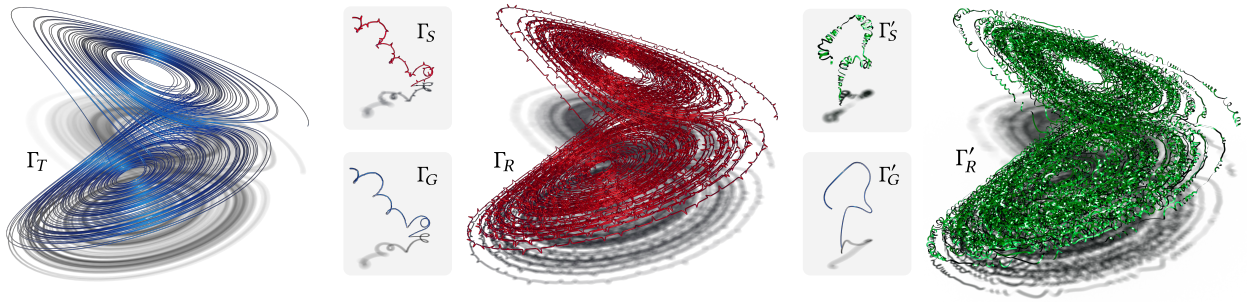


Figure 1: Results from our curve stylization pipeline (Alg. 1) applied to a large-scale scene using the Lorenz attractor as the target curve Γ_T. The guide curve Γ_G, automatically generated from the style curve Γ_S (Sec. 4.1), enables the stylized centerline Γ_R *(center)*. The stylized ribbon Γ'_R *(right)* demonstrates an extension of our method as described in Sec. 5.

	Domain	Name	Sec.
α	$(0,1)$	length-preserving coeff.	4.1
β	$(0,1)$	separation weight	4.2
p	$\mathbb{N}$	patch size	4.4
σ	$\mathbb{R}^+$	transfer magnitude	4.5.3
τ	$\mathbb{R}^+$	temperature	4.4.1

Table 1: An overview of adjustable parameters from Sec. 4.

The style and target scenes must be provided by the user, the guide scene is optional and otherwise computed automatically.

4. Method

This section describes our solution to the problem of scene analogies with curves that can be augmented by a general map ϕ. The core of our approach, sketched in the pseudocode below, is inspired by Lukáč et al. [15], where patches of texture from a stylized image are generated along a target curve by finding the shortest path between the style image patches using [18]. Instead of texture patches, our method is synthesizing a new displacement geometry along the target shape. To eliminate the need for a guide scene, we propose several preprocessing steps that rely solely on the style exemplar. These steps can be carried out only once and reused for multiple different targets, making the application of the style suitable for interactive environments. To simplify the notation we explain most parts of the algorithm for a single curve Γ and invoke the notion of scenes $\mathcal{S}$ only when necessary.

Algorithm 1 Scene Analogies

In: Style and target scene $\mathcal{S}_S$ and $\mathcal{S}_T$. (Guide scene $\mathcal{S}_G$ optional.)
Out: Result scene $\mathcal{S}_R$. (Optionally guide scene $\mathcal{S}_G$ as a byproduct.)

▽ **Preprocessing (target-independent)**
if $\mathcal{S}_G$ not provided **then**
  $\mathcal{S}_G \leftarrow \text{CreateGuide}(\mathcal{S}_S)$ ▷*Sec. 4.1*
end if
$\mathcal{S}_G \leftarrow \text{MatchGuide}(\mathcal{S}_G, \mathcal{S}_S)$ ▷*Sec. 4.2*
$\mathcal{S}_\delta \leftarrow \text{ComputeDisplacement}(\mathcal{S}_G, \mathcal{S}_S)$ ▷*Sec. 4.3*
$\mathcal{G} \leftarrow \text{PreparePatchGraph}(\mathcal{S}_S, \mathcal{S}_\delta)$ ▷*Sec. 4.4*
△ **End of preprocessing**
$\mathcal{S}_R \leftarrow \text{ApplyStyle}(\mathcal{S}_T, \mathcal{G}, \mathcal{S}_\delta)$ ▷*Sec. 4.5*
return $\mathcal{S}_R, (\mathcal{S}_G)$

4.1. Automatic Guide Creation

A key added value of our method in contrast to the approach of Hertzmann et al. [2] is that we can automatically generate the guide curve Γ_G from the style curve Γ_S and thus move beyond the analogy-based framework to one more akin to style transfer [1]. This capability is particularly beneficial in the context of 3D curve modeling where supplying the guide curve manually can be notably more complicated than in 2D. When generating Γ_G we assume that shape style information is encapsulated in higher-frequency components. We thus smooth out the original curve to

obtain its sort of medial axis and then find an appropriate matching between the two curves. This step has to be executed carefully since inaccuracies in the guide shape can compromise the following steps. We specifically need to avoid discretization degradation, unnecessary shrinking, and other artifacts. Basic smoothing algorithms such as FFT-based low-pass filter tend to create spurious oscillations and artifacts at the curve endpoints. An alternative could be to use a Gaussian smoothing filter or a curve shortening flow (CSF) given by $\partial_t\gamma = \kappa\mathbf{n} = \partial_s^2\gamma$. Nonetheless, for a strong enough smoothing both methods tend to introduce undesired artifacts caused by excessive shrinking. To alleviate this drawback, we formulate a novel variant of length-preserving curvature flow [19] for curves in $\mathbb{R}^3$.

Length-Preserving Flow. The original length-preserving curvature flow defined by Ma and Zhu [19] reads $\partial_t\gamma = (\kappa - \mathcal{F})\mathbf{n}$ where the forcing term $\mathcal{F}$ preventing unwanted shrinking is given by

$$\mathcal{F} = \left(\int_{\Gamma_t} \kappa \mathrm{ds}\right)^{-1} \int_{\Gamma_t} \kappa^2 \mathrm{ds}. \tag{1}$$

A key complication here is that for curves in $\mathbb{R}^3$, $\mathcal{F}$ is not defined at the inflection points where $\kappa = 0$ and $\mathbf{n}$ does not exist. Note, that the original CSF $\partial_t\gamma = \kappa\mathbf{n}$ is well-defined because when $\kappa = 0$, the direction $\mathbf{n}$ is not needed and the velocity term $\kappa\mathbf{n}$ can be continuously extended. We use this observation and propose $\partial_t\gamma = \kappa(1 - \kappa\mathcal{F}')\mathbf{n}$ with force

$$\mathcal{F}' = \left(\int_{\Gamma_t} \kappa^3 \mathrm{ds}\right)^{-1} \int_{\Gamma_t} \kappa^2 \mathrm{ds}.$$

We further modify the flow by adding a coefficient $\alpha \in (0, 1)$ that controls the extent to which is the curve allowed to shrink. To show that this motion law indeed preserves length $L(\Gamma_t)$ when $\alpha = 1$, we use the general evolution of local quantities from [20]. Specifically, using the formula for the local commutator $[\partial_s, \partial_t] = \partial_t\partial_s - \partial_s\partial_t = \kappa^2(1 - \alpha\kappa\mathcal{F}')$ we can differentiate the length in time:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{dt}} \int_{\Gamma_t} \mathrm{ds} &= -\int_{\Gamma_t} \kappa^2 \mathrm{ds} + \alpha\mathcal{F}' \int_{\Gamma_t} \kappa^3 \mathrm{ds} \\ &= (\alpha - 1) \int_{\Gamma_t} \kappa^2 \mathrm{ds}. \end{aligned} \tag{2}$$

Tangential Redistribution. To prevent accumulation of points at high-curvature regions and avoid associated numerical instabilities, we also add a tangential component $\vartheta_\mathbf{t}\mathbf{t}$ to the velocity, *i.e.*, we consider $\partial_t\gamma = \kappa(1 - \alpha\kappa\mathcal{F}')\mathbf{n} + \vartheta_\mathbf{t}\mathbf{t} =: \vartheta_\mathbf{n}\mathbf{n} + \vartheta_\mathbf{t}\mathbf{t}$. This term does not affect the shape but leads to the redistribution of points along the curve. We use the same principle as in, *e.g.*, [21, 22] but adapt it to our flow by setting the tangential velocity $\vartheta_\mathbf{t}$ to

$$\vartheta_\mathbf{t}(s) = \vartheta_\mathbf{t}(0) + \int_0^s \vartheta_\mathbf{n} \mathrm{ds}' + s(1 - \alpha) \fint_{\Gamma_t} \kappa^2 \mathrm{ds}', \tag{3}$$

where $\vartheta_\mathbf{t}(0)$ is set by $\int_{\Gamma_t} \vartheta_\mathbf{t} \mathrm{ds} = 0$, as

$$\vartheta_\mathbf{t}(0) = \fint_{\Gamma_t} \left(\int_0^s \kappa\vartheta_\mathbf{n} \mathrm{ds}' + s(1 - \alpha)\mathcal{F} \right) \mathrm{ds},$$

and $\fint_{\Gamma_t}$ indicates the curve integral normalized by the length $L(\Gamma_t)$.

Proposition 1. *The modified flow with tangential velocity $\vartheta_\mathbf{t}$ from Eq. 3 preserves the relative local length $L(\Gamma_t)^{-1}\|\partial_u\gamma\|$.*

Proof. With $\vartheta_\mathbf{t}$, the parametrization rate evolves as

$$\partial_t\|\partial_u\gamma\| = (-\kappa^2(1 - \alpha\kappa\mathcal{F}') + \partial_s\vartheta_\mathbf{t})\|\partial_u\gamma\|.$$

Using a similar calculation as in Eq. 2, we derive

$$\frac{\mathrm{d}}{\mathrm{dt}} \frac{\|\partial_u\gamma\|}{L(\Gamma_t)} = \frac{\|\partial_u\gamma\|}{L(\Gamma_t)} \left[-\kappa\vartheta_\mathbf{n} + \partial_s\vartheta_\mathbf{t} - (1 - \alpha) \fint_{\Gamma_t} \kappa^2 \mathrm{ds} \right].$$

Substitution of $\partial_s\vartheta_\mathbf{t}$ from Eq. 3 indeed shows that the right-hand side is zero and the relative local length is constant. □

As a consequence of Prop. 1, when the initial curve $\Gamma_0 = \Gamma_S$ is equidistantly discretized, the points will tend to be equidistantly spaced along Γ_t at each

time step t. Altogether we arrive at the following initial value problem for the parametrization γ:

$$\begin{aligned} \partial_t \gamma &= \kappa(1-\alpha\kappa\mathcal{F}')\mathbf{n} + \vartheta_{\mathbf{t}}\mathbf{t} \quad \text{on } S^1 \times [0, t_m], \\ \gamma|_{t=0} &= \gamma_S \qquad\qquad\qquad\qquad \text{in } S^1. \end{aligned} \tag{4}$$

In summary, we propose three key modifications to the original CSF: *(1)* a well-defined forcing term that preserves length, *(2)* an α parameter that controls the rate at which the length decreases, and *(3)* an optimal tangential velocity $\vartheta_{\mathbf{t}}$ adapted for our flow.

Numerical Integration. To numerically integrate the flow, we use the Euler method with finite difference approximation of the right-hand side. The integrals from $\mathcal{F}$ and $\vartheta_{\mathbf{t}}$ are approximated using the discrete curvature based on Steiner formula with line segment corner expansion from [23]. For examples showcased in our results, we use $\alpha = 0.5$.

4.2. Guide Curve Matching

Robust and precise guide matching is another crucial step behind the success of our method. We formulate the energy inspired by Fréchet distance but more tailored to our problem. To minimize this energy we developed a semi-discrete approach where the style curve is continuous. This leads to both more precise matching and notably lower computational overhead.

Continuous Formulation. Given Γ_G and Γ_S we aim to construct an optimal increasing bijection $\delta\colon [0, L_G] \to [0, L_S]$ with fixed endpoints $\delta(0) = 0$ and $\delta(L_G) = L(\gamma_S)$. The optimal matching δ is obtained by minimizing the following energy functional:

$$\mathcal{E}[\delta] = \beta\mathcal{E}_s[\delta] + (1-\beta)\mathcal{E}_d[\delta],$$

where $\beta \in (0,1)$ is a prescribed coefficient. The separation energy $\mathcal{E}_s[\delta]$ and the distortion energy $\mathcal{E}_d[\delta]$ are defined as

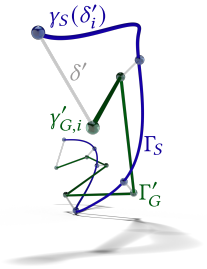
$$\begin{aligned} \mathcal{E}_s[\delta] &= \int_{\Gamma_G} \|\gamma_\delta(s)\|^2 \mathrm{ds}, \\ \mathcal{E}_d[\delta] &= \int_{\Gamma_G} \left(\delta(s) - \tfrac{L_S}{L_G}s\right)^2 \mathrm{ds}, \end{aligned}$$

where $\gamma_\delta = \gamma_S \circ \delta - \gamma_G$.

Discrete Matching. One way to solve the optimization problem $\delta = \mathrm{argmin}_\delta \mathcal{E}[\delta]$ is to use the connection with the problem of computing the discrete Fréchet distance between two curves given by

$$\rho_F(\gamma_G, \gamma_S) = \inf_\delta \max_s \|\gamma_\delta(s)\|.$$

This can be done by modifying the dynamic programming approach of Alt and Godau [24], adjusting their energy function to remember the best path and ensure strictly increasing sequences to obtain a bijection. To achieve a precise match, one could first redistribute points along Γ_S to a fine mesh and then apply the algorithm with a fixed discretization of Γ_G.



Semi-Discrete Matching. For curves with Γ_S and Γ_G with N and M points, respectively, both the computational and space complexity of the dynamic programming approach is $\mathcal{O}(NM)$ due to the computation of the energy matrix. As our use case requires large N to obtain a precise match, we propose an alternative optimization-based approach that lets the matching points on Γ_S move freely along its smooth interpolation. This way, we both increase the precision (measured by the final achieved energy $\mathcal{E}$) as well as make the algorithm much faster and more memory efficient.

We consider discretization of the guide curve $\Gamma'_G = \{\Gamma'_{G,i}\}_{i=1}^N \subset \mathbb{R}^3$ and its length L'_G and define the semi-discrete displacement map as $\delta'\colon \mathbb{N}_{\leq N} \mapsto [0, L'_S]$ with fixed endpoints $\delta'_1 = 0$ and $\delta'_N = L'_S$ where δ'_i denotes $\delta'(i)$ for convenience. The assumption here is that Γ'_G is discretized equidistantly and Γ_S is a smooth interpolation of the discretized Γ'_S. Then a semi-discrete version of the energy $\mathcal{E}[\delta]$ can be defined as $\mathcal{E}'[\delta'] = \beta\mathcal{E}'_s[\delta'] + (1-\beta)\mathcal{E}'_d[\delta']$ where

$$\begin{aligned} \mathcal{E}'_s[\delta'] &= \sum_{i=2}^{N-1} \|\gamma'_{G,i} - \gamma_S(\delta'_i)\|^2, \\ \mathcal{E}'_d[\delta'] &= \sum_{i=2}^{N-1} \left(\delta'_i - \tfrac{L_S}{L'_G}i\right)^2. \end{aligned}$$

We can compute the gradients *w.r.t.* $\delta'(i)$ as

$$[\nabla \mathcal{E}'_s]_i = -2\left\langle \gamma'_{G,i} - \gamma_S(\delta'_i), \mathbf{t}_S(\delta'_i)\right\rangle,$$
$$[\nabla \mathcal{E}'_d]_i = 2\left(\delta'_i - \tfrac{L_S}{L'_G} i\right),$$

where $\mathbf{t}_S$ is the tangent vector of Γ_S. The gradient of the full energy is then obtained from chain rule $\nabla \mathcal{E}' = \beta \nabla \mathcal{E}'_s + (1-\beta)\nabla \mathcal{E}'_d$ and the Hessian for the full energy reads $\mathbb{H}\mathcal{E}' = \mathbb{H}\mathcal{E}'_s = \mathbb{H}\mathcal{E}'_d = 2\mathbb{I}$.

Optimization. To ensure valid one-to-one mapping between the guide curves Γ'_G and Γ_S during optimization, we implement constrained gradient descent. For each update of δ' we compute the upper and lower bounds of each point to ensure that the neighboring points cannot reverse their order. Any excessive changes are then clipped to keep δ non-decreasing.

4.3. Transform Displacement Vectors

To reproduce the local geometric features of Γ_S, we consider the displacement map $\gamma_\delta = \gamma_S \circ \delta - \gamma_G$. The use of displacement vectors is crucial as it enables variable length of the curves as well as non-trivial topology changes like loops. We make the transfer of geometric details rotation invariant by using the following displacement map

$$\mathcal{R}^{\#}\gamma_\delta(u) = \begin{pmatrix} \langle \gamma_\delta(u), \mathbf{t}_G(u)\rangle \\ \langle \gamma_\delta(u), \mathbf{n}'_G(u)\rangle \\ \langle \gamma_\delta(u), \mathbf{b}'_G(u)\rangle \end{pmatrix}$$

where $\mathcal{R} := \{\mathbf{t}_G, \mathbf{n}'_G, \mathbf{b}'_G\}$ is the local basis for Γ_G.

4.4. Patch Distance Matrix & Graph

To enable the use of the algorithm of Lukáč et al. [15] we compute the matrix M with distances between patches of the Frenet displacements as well as the associated graph $\mathcal{G}$ (see Figures 2 and 3). The former is given by $M_{i,j} = \rho_p(i, j-1)$ where $2p+1$ is the patch size and

$$\rho_p^2(i,j) = \sum_{k=-p}^{p} \|\gamma'_{\delta,i+k} - \gamma'_{\delta,j+k}\|^2 \qquad (5)$$

for $i, j \in [p, N-p]$. Otherwise $\rho_p(i,j) = +\infty$ for $i \neq j$ and $\rho_p(i,j) = 0$ for $i = j$. One can also multiply the summands by a bump function centered around $k = 0$ to emphasize the center of the patch. However, the changes to the results are usually insignificant. The patch graph $\mathcal{G} = (V, E)$ is a weighted graph with nodes $V = \mathbb{N}_{\leq N}$ and edges $E \subset V \times V \times \mathbb{R}^+$ constructed from the patch matrix as

$$E = \{(i, j, M_{i,j}) \colon i, j \in V;\ M_{i,j} < +\infty\}.$$

Even if the scene $\mathcal{S}$ contains multiple curves, only one patch matrix and graph are constructed (see Fig. 3).

4.4.1. Temperature

Because the elements $M_{i,i+1} = \rho_p(i,i)$ are zero, the reconstruction tends to exactly copy parts of the exemplar displacement. This is typically desired as it leads to the method's robust performance. However,

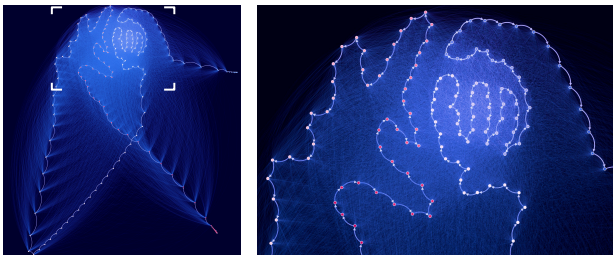


Figure 2: Patch graph visualized using Gephi's force-directed layout [25]. Nodes represent points along the curve, with edge size and opacity determined by patch similarity. The visualization places strongly connected (consecutive) nodes close together, forming prominent edges, while weaker edges between structurally similar but non-adjacent patches create the intricate, crinkled structure. This structure highlights the algorithm's preference for consecutive patches but also its flexibility to occasionally jump to similar, distant patches.

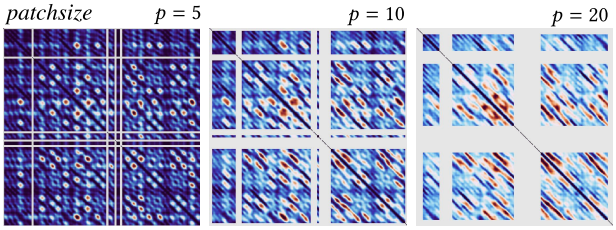


Figure 3: Patch matrix from a scene consisting of four curves with patch size of 5 *(left)*, 10 *(center)* and 20 *(right)*. Gray regions correspond to value $+\infty$.

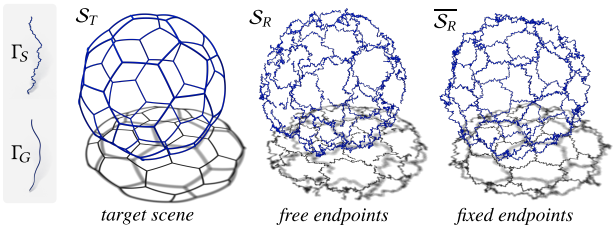


Figure 4: Style Γ_S applied to the target scene $\mathcal{S}_T$ *(left)*, a spherical tiling based on the truncated icosahedron. Shown with *(right)* and without *(center)* the endpoint fix.

to increase the variation of the outputs we can set these values to a constant $M_{i,i+1} = \tau$. This new parameter τ acts like a temperature enabling the user to adjust the extent to which the applied style can deviate from the original exemplar.

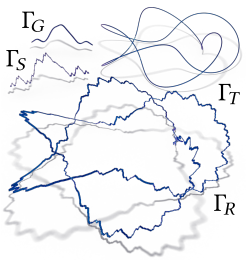
4.5. Displacement Reconstruction

All previous steps were independent of the target curve Γ_T and can be precomputed, cached, and reused for any future target scene. The Frenet displacement map Γ_Δ for Γ_T is constructed by selecting random nodes and using Dijkstra's algorithm [18] to find shortest paths as in [15]. The resulting Γ_R is then obtained by adding the displacement to Γ_T, *i.e.*,

$$\gamma_R(u) = \gamma_T(u) + \mathcal{R}^{\#}\gamma_\Delta(u),$$

where $\mathcal{R}$ denotes the local frame associated with the target Γ_T. To prevent a *cul-de-sac*, where the Dijkstra's algorithm would get stuck, the random destinations are sampled between p-th and $(N-p)$-th point along the curve.

4.5.1. Closed Curves



One of the issues of the proposed approach is that there is no control over the exact location of the last point of the generated curve. Thus creating closed curves is not supported by default. However, since the displacement magnitude of a given style is bounded, one can enforce the closure by setting

$$\overline{\gamma}_R(s) = \gamma_R(s) + \frac{s}{L_R}\left[\gamma_R(0) - \gamma_R(L_R)\right],$$

which ensures that $\overline{\gamma}_R(L_R) = \overline{\gamma}_R(0)$. For a large enough scene, the introduced distortion to the style and shape of Γ_R is negligible (see a closed "knotted fractal" example in the *inset*).

4.5.2. Filament Networks

To stylize the curved filament networks with prescribed connections, the positions of endpoints for all connected curves must be enforced (see Fig. 4).

$$\overline{\gamma}_R(s) = \gamma_R(s) + \frac{L_R - s}{L_R}\left[\gamma_T(0) - \gamma_R(0)\right] + \\ + \frac{s}{L_R}\left[\gamma_T(L_T) - \gamma_R(L_R)\right].$$

This ensures both that $\overline{\gamma}_R(0) = \gamma_T(0)$ and $\overline{\gamma}_R(L_R) = \overline{\gamma}_T(L_T)$. It is crucial for stylizing complex scenes like wireframe architectural or design models.

4.5.3. Magnitude & Scale

Our method allows the user to set arbitrary magnitude and scale of the transfer. Different scales can be achieved by rescaling the style, and the magnitude can be changed by setting

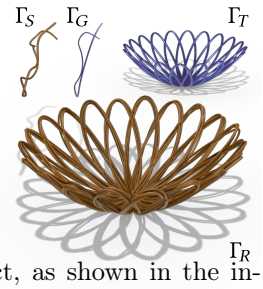
$$\gamma_R(u) = \gamma_T(u) + \sigma\mathcal{R}^{\#}\gamma_\Delta(u),$$

where $\sigma \in \mathbb{R}^+$ dictates the magnitude of displacement (see Fig. 8).

4.5.4. Multiresolution

To simultaneously transfer details on several scales, one can simply sum the similarity matrices for several patch sizes p. Alternatively, the sum may be weighted to boost the impact of a particular scale.

4.5.5. Multistroke



Given a closed target curve, the style can be applied multiple times to achieve a multistroke artistic effect. For instance, this can be used for creating visually interesting vine root arrangements or braided effect, as shown in the inset. This example also shows how the method can be used for larger scale changes, not just for adding

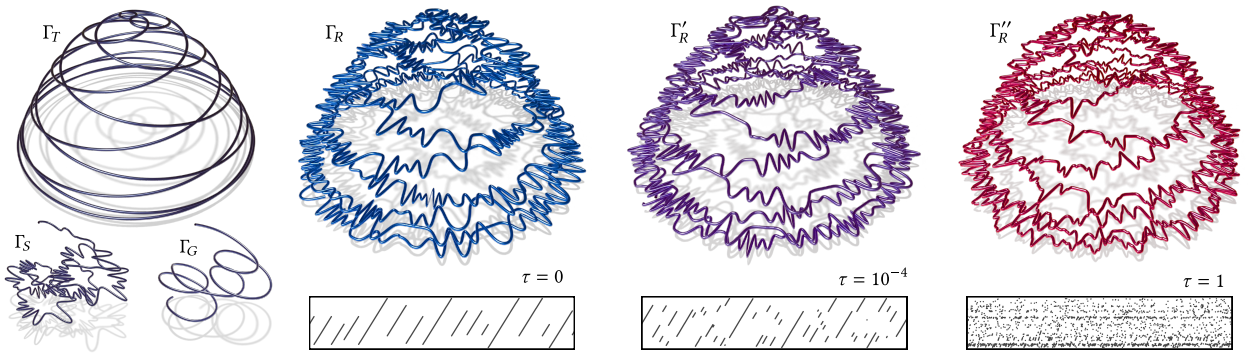


Figure 5: The temperature parameter $\tau \in \mathbb{R}^+$ enables the user to specify the extent to which the method should precisely copy parts of the precomputed style displacement, *i.e.*, how "creative" or "chaotic" should the transfer be. The scatter plots under figures show the results of Dijkstra's path finding algorithm, with the x-axis representing points along the target curve Γ_T and the y-axis representing the indexes of chosen style patches. Long diagonal lines, such as those found on the plot with the lowest setting $\tau = 0$, represent exact displacement map copies.

high-frequency details, while still following the target shape. Note that our approach does not guarantee that the generated curves or tubes do not intersect.

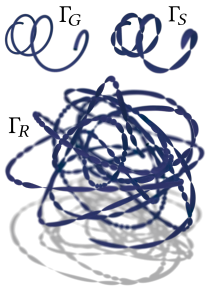
5. Extensions

The algorithm presented in Sec. 4 can be generalized for use cases beyond the stylization of infinitesimal curve shape by incorporating additional attributes along the curve. This is achieved by introducing a function ϕ that maps each point to a value in a metric space Ω_ϕ, enabling applications such as tubular structures (where ϕ represents the radius), ribbons (where ϕ encodes an orientation), or other augmented trajectories. To integrate these attributes, we define a metric ρ_ϕ on Ω_ϕ and extend the patch distance ρ_p from Eq. (5) as follows:

$$\rho_{p,\phi}(i,j) = \rho_p(i,j) + \rho_\phi(\phi(\delta'_i), \phi(\delta'_j)).$$

Here, ρ_ϕ quantifies the difference in ϕ values along the curve, up to a scaling factor that serves as a tunable parameter and allows the user to steer the path generation to emphasize either stroke shape or the values of the augmented map. Our approach enables this by following the displacement patches along the curve instead of tracking the neighborhood or using rasterized images.

5.1. Tubular Neighborhoods



We first consider a tube represented as an augmented curve with a variable thickness $h_i = \phi(\delta'_i) \in \mathbb{R}^+$. The inset figure shows an example result (Γ_R, ϕ) with simultaneous transfer of centerline displacement and the tube thickness. In this case, the metric space Ω_ϕ is $\mathbb{R}^+$ equipped with the Euclidean metric ρ. Thus the right-hand-side of Eq. (5) is modified by

$$\sum_k \rho^2_\phi(\phi(\delta'_{i+k}), \phi(\delta'_{j+k})) = \sum_k |h_{i+k} - h_{j+k}|^2.$$

Tubes with a more complicated cross-section can be considered by modifying Ω_ϕ. For example, the shape of a tube with an elliptical profile can be captured by $\Omega_\phi = \mathbb{R}^+ \times \mathbb{R}^+$. The orientation of the ellipse can be further modulated by considering $\Omega_\phi = \mathbb{R}^+ \times \mathbb{R}^+ \times S^1$, where $S^1 = \mathbb{R}/2\pi\mathbb{Z}$ is the unit circle.

5.2. Ribbon Curves

In many applications such as the stylization of camera trajectories or modeling of a stylized roller coaster, it is useful to not only apply the style of the curve centerline but also the curve orientation. This

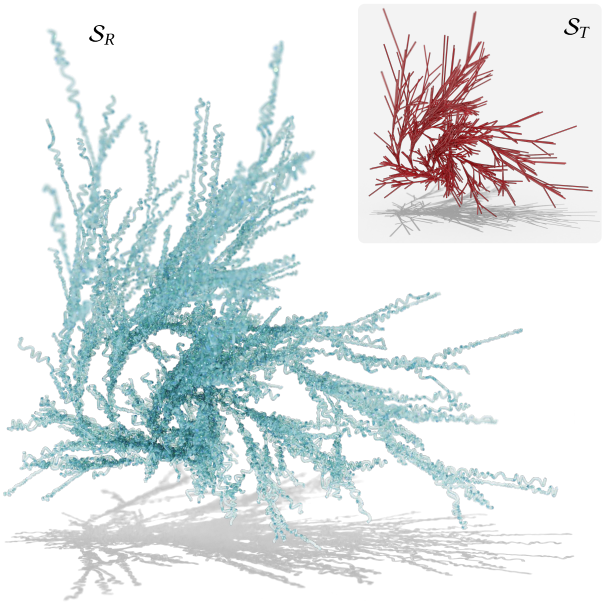


Figure 6: This example showcases stylization $\mathcal{S}_S$ of an embedded tree graph structure $\mathcal{S}_T$ representing several iterations of a 3D L-system. The stylization is enabled by our use of RMF, as the target scene $\mathcal{S}_T$ consists of straight lines with undefined Frenet vectors. To achieve the expected connectivity in $\mathcal{S}_S$, we used the locking mechanism described in Sec. 4.5.2.

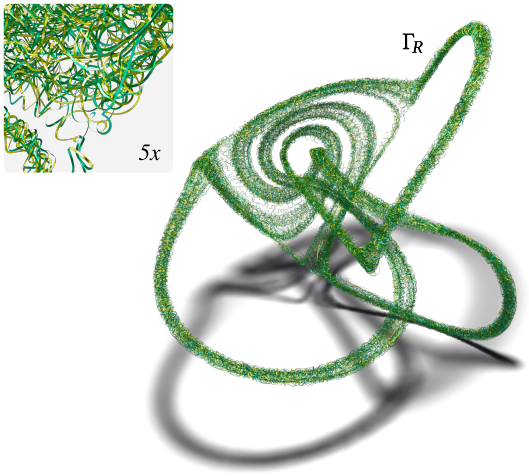


Figure 7: Thomas attractor stylized as two-sided ribbons. The unstyled target curve is shown in Fig. 12. This extension is made using the associated metric space $\Omega_\phi = S^1$. The attractor has approximately 369,400 points and the stylization process takes 51.25 seconds, demonstrating the capability of our method for large scene stylization.

can be enabled in two different modes; either as a two-sided ribbon with $\Omega_\phi = S^1$ (see Fig. 7) or a one-sided ribbon with $\Omega_\phi = \mathbb{RP}^1$, which denotes the real projective line. In both cases, the associated metric ρ_ϕ is given by the length of the shortest arc length between two points. Stylized ribbons with variable thickness are also possible with $\Omega_\phi = S^1 \times \mathbb{R}^+$ or $\mathbb{RP}^1 \times \mathbb{R}^+$ for the one-sided version.

6. Applications

Our approach has numerous potential applications in interactive environments, including virtual reality drawing (e.g., Tilt Brush [26]) or centerline stylization for swept volumes (e.g., Adobe Medium [27, 28]). Other possible applications include styling strands of hair or fur (see Fig. 8), organic structures such as stems or branches (see Fig. 6), confetti (see Fig. 1), wicker (see inset in Sec. 4.5.5), or architectural design (see Fig. 9). Our method can also be utilized to stylize the trajectories of moving objects by adding displacement to their animation curves and changing the local velocity, e.g., for character motion stylization, converting a laminar flow into a turbulent field (see Fig. 10), or preparation of animated construction of line drawings [29]. In addition, existing trajectories could also be extended with an associated sound encoded as a spectrogram (e.g., pencil handwriting).

7. Results and Evaluation

We provide several comparisons and explore different settings and properties of the proposed method.

Comparison with Curve Analogies. The most relevant comparison to our method is *Curve Analogies* [2], originally designed for planar curves. To enable a fair evaluation, we extended it to 3D, but this naive generalization exposed significant limitations. Its neighborhood alignment is sensitive to torsion and inflection points, leading to severe artifacts

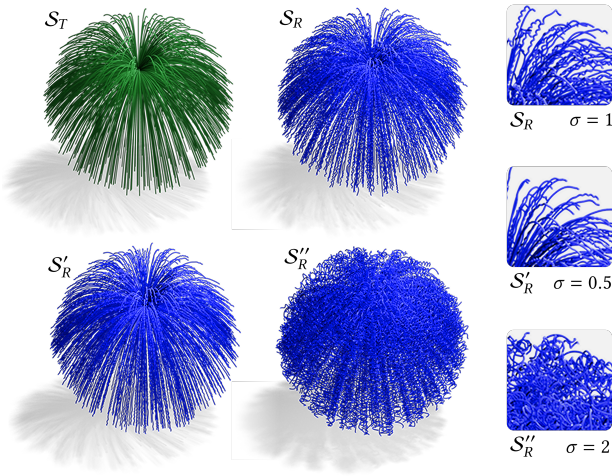


Figure 8: Unlike [2], our method allows the user to adjust the transfer magnitude via parameter σ. This enables a continuous transition from unstyled to stylized and can be used to precisely adjust artistic intent. The figure also demonstrates a potential application for hair stylization and the stylization of large scenes with many curves, here 2000 hair strands.

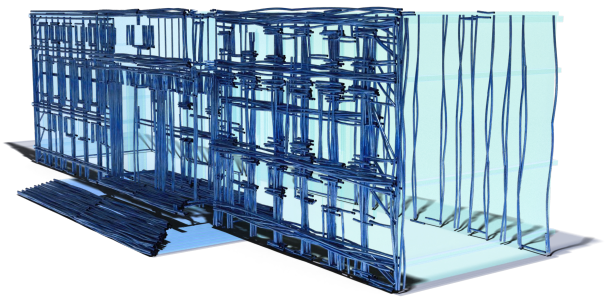


Figure 9: Application of our method to an architectural wireframe. The target curves are progressively stylized by increasing the magnitude parameter σ from left to right, resulting in a sketch-like, expressive style. This approach can be used for visualization of architectural drafts or for generating organic, hand-crafted assets in games and animation.

(Fig. 11). In contrast, our approach explicitly incorporates the local frame of Γ_T, ensuring more stable and expressive style transfer, including 2D-to-3D stylization (Fig. 16).

Efficiency is another key advantage. *Curve Analogies* relies on costly per-point neighborhood sampling and comparisons, making it impractical for large curves—such as the Thomas attractor (Fig. 7) with 369,400 points. Our method overcomes this by precomputing a reusable patch graph, enabling fast shortest-path queries via Dijkstra's algorithm. When applying the same style curve to different targets, the graph can be cached and reused for further speedup. For a fair comparison, we used the predefined guide curve from Fig. 11 and measured only stylization times (excluding guide curve creation). *Curve Analogies* required 7.52 seconds, while our method completed in 1.98 seconds. With a precomputed patch graph, the runtime dropped further to 0.51 seconds. Additionally, our approach offers tunable *temperature* and *magnitude* parameters for finer artistic control, along with natural extensions such as ribbons and variable thickness.

Framing Experiments. The method is flexible and can apply style in any local frame, with RMF used by default. Using the Frenet frame caused artifacts near inflection points (Fig. 12), so switching to RMF improved consistency in the style transfer. RMF also enabled stylizing straight-line segments in the L-system from Fig. 6, where Frenet vectors were undefined. Additionally, the ability to apply 2D styles to 3D targets (Fig. 16 and our supplementary video) shows our versatility compared to neighborhood-matching methods like [2].

Tunable Parameters. Our approach offers additional tunable parameters, listed in Tab. 1, providing greater flexibility compared to [2]. The style transfer magnitude σ parameter, described in Sec. 4.5.3 and showcased in Fig. 8, allows continuous interpolation between unstyled and styled versions of the curve. The temperature parameter τ, introduced in Sec. 4.4.1, has a similar effect as in autoregressive language models. Fig. 5 and our supplementary video show how increasing its value can lead to results ranging from predictable to more creative style transfer.

Closed Curves and Curve Networks. In several experiments, we addressed the closing and connecting curve endpoints, and showed how the method is able to transfer style while preserving topological consistency. For closed curves, we enforced closure by adjusting the displacement magnitude, as detailed in

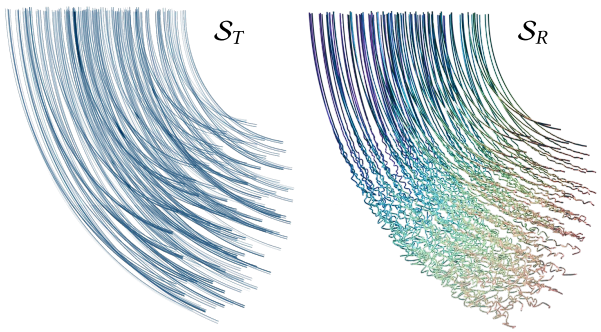


Figure 10: Stylization of laminar trajectories $\mathcal{S}_T$ *(left)* into turbulent-like flows $\mathcal{S}_R$ *(right)* by modulating the magnitude parameter $\sigma \in [0, 1]$. The method introduces spatially varying detail and complexity without physical simulation, enabling controllable flow-line aesthetics for animation, visual effects, and interactive design.

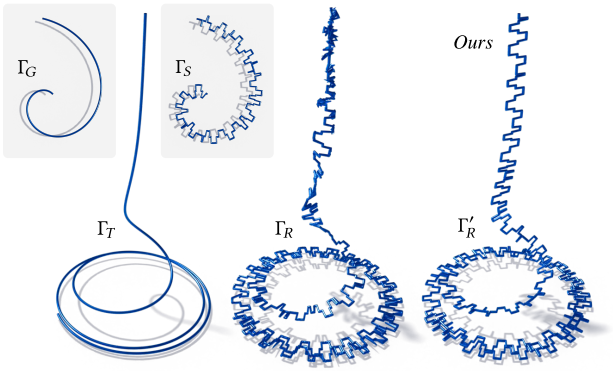
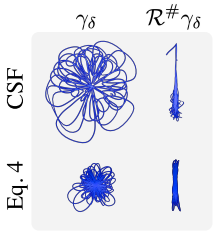


Figure 11: Comparison with *Curve Analogies* [2]. When generalized to 3D, their method fails in regions with high torsion and near inflection points, leading to instability in the resulting curve Γ_R. In contrast, our approach explicitly handles local frames, ensuring stable style transfer and greater artistic control. We ensured a fair comparison by using the same predefined guide curve, Γ_G, for both methods.

Sec. 4.5.1, and demonstrated with the "knotted fractal" inset. For curve networks, we ensured endpoint connections using the method described in Sec. 4.5.2. We applied this technique to a branching L-System (Fig. 6) and a truncated icosahedron (Fig. 4).

Length-Preserving Flow. Our modified length-preserving flow avoids the artifacts introduced by excessive curve shrinking. The CSF displacement map *(inset top)* and the displacement map in the Frenet frame show unwanted distortions. In contrast, the length-preserving flow *(inset bottom)* generates a smooth and accurate guide curve.



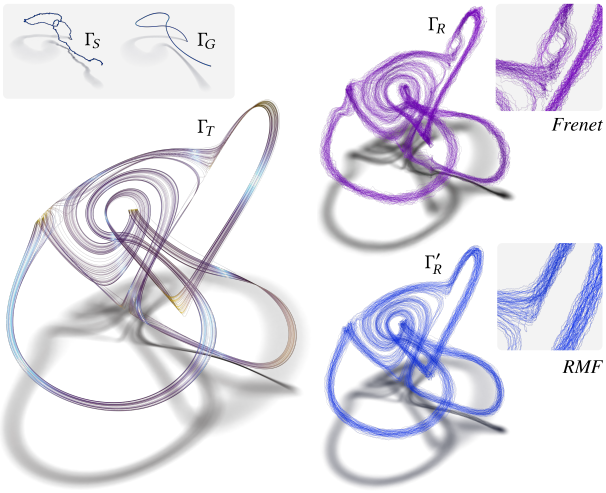


Figure 12: Using the Frenet frame during displacement reconstruction leads to undesired artifacts near inflection points of the target curve (see the resulting curve Γ_R). The example shows how changing the framing to RMF in Γ'_R changes the style transfer with the same style curve Γ_S and target curve Γ_T.

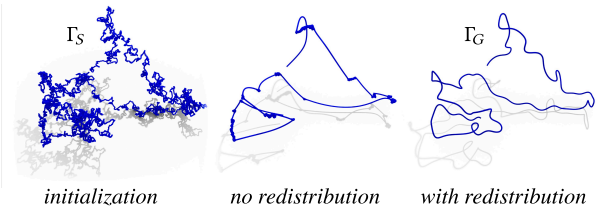


Figure 13: Without tangential redistribution, the flow is unstable due to point accumulation in high-curvature regions.

Tangential Redistribution. We added a tangential component to the velocity to prevent point accumulation in high-curvature regions, as described in Prop. 1. This adjustment leads to a more uniform point distribution and thus enhances numerical sta-

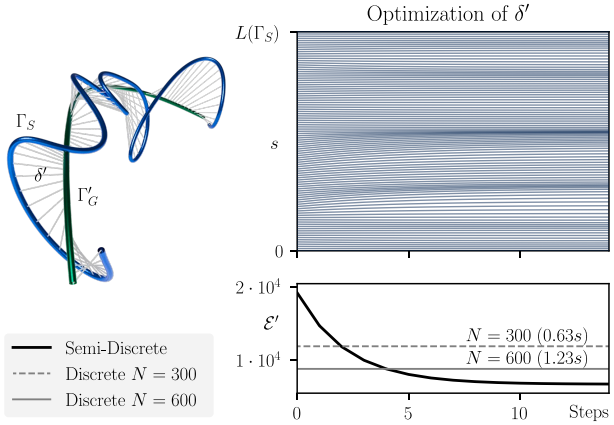


Figure 14: Comparison of semi-discrete and discrete guide curve matching. *Left:* Visualization of optimal matching δ' between discretized guide curve Γ'_G *(green)* and continuously interpolated style curve Γ_S *(blue). Right (top)*: Evolution of matching points along curve lengths s during optimization. *Right (bottom):* Energy $\mathcal{E}'$ of the semi-discrete method versus discrete matching at resolutions $N = 300$ and $N = 600$. The semi-discrete approach not only achieves significantly lower total energy $\mathcal{E}'$ but also drastically outperforms the discrete method in computational efficiency (semi-discrete: 0.0241s after 15 steps; discrete: 0.629s for $N = 300$, 1.23s for $N = 600$).

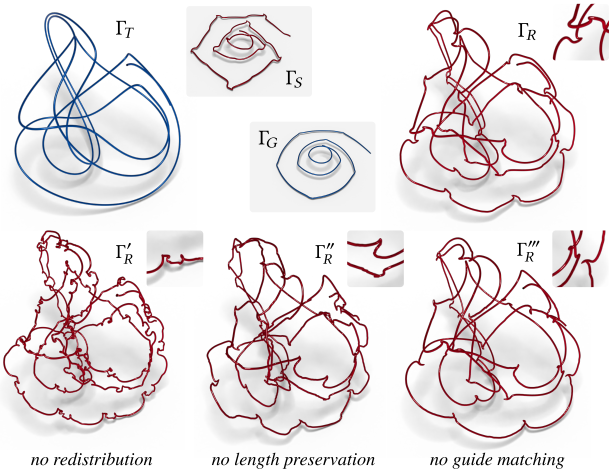


Figure 15: Ablation study of our method. *Top row:* Target curve Γ_T, style curve Γ_S, generated guide curve Γ_G, and result curve Γ_R using the full algorithm. *Bottom row:* Results with ablated components. Γ'_R (no tangential redistribution, causing point clumping near high-curvature regions and incorrect displacements around spikes), Γ''_R (no length-preserving flow with $\alpha = 0$, leading to curve shortening and staggered displacement artifacts), and Γ'''_R (no guide matching, using trivial equidistant matching, resulting in spike shape deformation).

bility, as shown in Fig. 13.

Guide Matching Comparison. We evaluated our semi-discrete guide matching method against the discrete approach (Sec. 4.2). Fig. 14 demonstrates that even for relatively small curves (100 points), the semi-discrete scheme significantly outperforms the discrete scheme in terms of both final achieved energy and computational efficiency.

Ablation Experiments. We conducted ablation experiments to evaluate the contribution of each component in our method, with the following execution times for the full pipeline: guide creation (0.0097 seconds), guide matching (0.0502 seconds), patch matrix and graph creation (5.4556 seconds), and transfer application (1.4943 seconds). Fig. 15 illustrates the impact of removing individual components-tangential redistribution, length-preserving flow, and guide matching-on the resulting curve. Omitting these components introduces artifacts, such as point clumping, curve shortening, and shape deformation.

8. Limitations and Future Work

The curve style transfer parameters depend highly on the scale of the style scene. Our method does not natively support all stroke topologies like dashed lines, although we can achieve a similar effect by using an additional case-specific metric such as signed distance to the nearest stroke-hole boundary. The transfer may alter topological properties, potentially causing self-intersections; however, this can be controlled by adjusting the σ value and style scale. Future work may include applying style transfer to full tubular meshes, stylizing Gaussian splat neighborhoods along the curve, and adding texture to tubular meshes.

9. Conclusions

We have proposed a novel method for curve stylization that enables example-based style transfer between 3D vector curves without requiring a

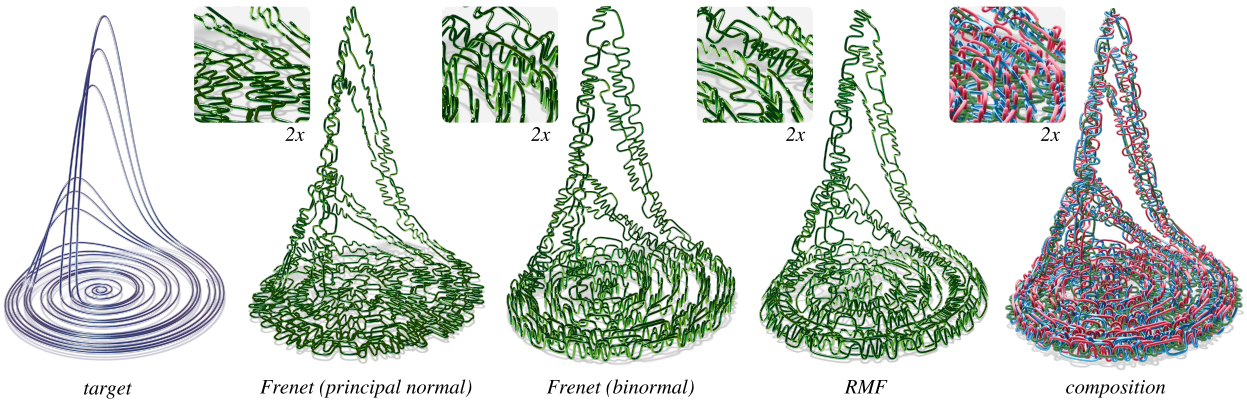


Figure 16: Our approach can also process 2D style Γ_S and apply it to a 3D target Γ_T. The style displacement can be applied with respect to any local basis $\mathcal{R}$ of the target curve Γ_T. This choice gives the user a flexibility, that is not possible with methods based on neigbourhood matching such as [2]. The composition of all three different normal directions *(right)* demonstrates the potential for generating intricate, knitted, art-like patterns. While our approach allows for rich geometric detail, preventing self-intersections remains a challenge.

pre-defined unstyled version of the style exemplar. We achieve this by formulating a new smoothing flow that infers the unstyled curve and a modified Fréchet distance to guarantee precise correspondence between stylized and unstyled forms. Our semi-discrete optimization scheme significantly improves efficiency compared to existing methods, reducing computational cost while maintaining analytical accuracy. Furthermore, adjustable stylization strength and style transfer temperature offer the user enhanced creative control. The versatility of our approach allows also for the incorporation of additional information and associated metrics, making it applicable to the stylization of complex 3D structures like tubes, ribbons, and motion curves, where information such as velocity can be incorporated. Given the prevalence of 3D curves across diverse applications, we anticipate this work will stimulate further research in this area.

References

[1] L. A. Gatys, A. S. Ecker, M. Bethge, Image style transfer using convolutional neural networks, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2414–2423.

[2] A. Hertzmann, N. Oliver, B. Curless, S. M. Seitz, Curve analogies, in: Proceedings of Eurographics Workshop on Rendering, 2002, pp. 233—-246.

[3] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, D. H. Salesin, Image analogies, in: SIGGRAPH Conference Proceedings, 2001, pp. 327–340.

[4] S. Zelinka, M. Garland, Mesh modelling with curve analogies, in: Proceedings of Pacific Conference on Computer Graphics and Applications, 2004, pp. 94–98.

[5] Y. Wu, H. Zhang, C.-Y. Song, H. Bao, Spacetime curve analogies for motion editing, in: Proceedings of Geometric Modeling and Processing Conference, 2008, pp. 437—-449.

[6] K. Lang, M. Alexa, The Markov pen: Online synthesis of free-hand drawing styles, in: Proceedings of International Symposium on Non-Photorealistic Animation and Rendering, 2015, pp. 203–215.

[7] A. Berchtold, The double chain markov model, Communications in Statistics - Theory and Methods 28 (11) (1999) 2569–2589.

[8] P. Bhat, S. Ingram, G. Turk, Geometric texture synthesis by example, in: Proceedings of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, 2004, pp. 41–44.

[9] H.-T. D. Liu, A. Jacobson, Normal-driven spherical shape analogies, Computer Graphics Forum 40 (5) (2021) 45–55.

[10] S. Berkiten, M. Halber, J. Solomon, C. Ma, H. Li, S. Rusinkiewicz, Learning detail transfer based on geometric features, Computer Graphics Forum 36 (2) (2017) 361–373.

[11] C. Ma, H. Huang, A. Sheffer, E. Kalogerakis, R. Wang, Analogy-driven 3D style transfer, Computer Graphics Forum 33 (2) (2014) 175–184.

[12] P. C. Merrell, D. Manocha, Example-based curve synthesis, Computers & Graphics 34 (2010) 304–311.

[13] R. Roveri, A. C. Öztireli, S. Martin, B. Solenthaler, M. Gross, Example based repetitive structure synthesis, Computer Graphics Forum 34 (5) (2015) 39–52.

[14] S. Zhou, C. Jiang, S. Lefebvre, Topology-constrained synthesis of vector patterns, ACM Trans. Gr. 33 (6) (2014) 215.

[15] M. Lukáč, J. Fišer, J.-C. Bazin, O. Jamriška, A. Sorkine-Hornung, D. Sýkora, Painting by Feature: Texture boundaries for example-based image creation, ACM Trans. Gr. 32 (4) (2013) 116.

[16] R. Bishop, There is more than one way to frame a curve, American Mathematical Monthly 82 (1975) 246–251.

[17] W. Wang, B. Jüttler, D. Zheng, Y. Liu, Computation of rotation minimizing frames, ACM Trans. Gr. 27 (1) (2008) 2.

[18] E. W. Dijkstra, A note on two problems in connexion with graphs, Numerische Mathematik 1 (1959) 269–271.

[19] L. Ma, A. Zhu, On a length preserving curve flow, Monatshefte für Mathematik 165 (2008) 57–78.

[20] M. Beneš, M. Kolář, D. Ševčovič, Qualitative and numerical aspects of a motion of a family of interacting curves in space, SIAM Journal on Applied Mathematics 82 (2) (2022) 549–575.

[21] T. Y. Hou, J. S. Lowengrub, M. J. Shelley, Removing the stiffness from interfacial flows with surface tension, Journal of Computational Physics 114 (2) (1994) 312–338.

[22] M. Kimura, Numerical analysis of moving boundary problems using the boundary tracking method, Japan Journal of Industrial and Applied Mathematics 14 (1997) 373–398.

[23] K. Crane, M. Wardetzky, A glimpse into discrete diff erential geometry, Notices of the American Mathematical Society 64 (2017) 1153–1159.

[24] H. Alt, M. Godau, Computing the fréchet distance between two polygonal curves, International Journal of Computational Geometry & Applications 5 (1 & 2) (1995) 75–91.

[25] M. Bastian, S. Heymann, M. Jacomy, Gephi: An open source software for exploring and manipulating networks, Proceedings of the International AAAI Conference on Web and Social Media 3 (1) (2009) 361–362.

[26] A. Doronichev, Tilt brush: Painting from a new perspective, online; accessed 28-Jan-2025 (2016).

[27] S. Sellán, N. Aigerman, A. Jacobson, Swept volumes via spacetime numerical continuation, ACM Trans. Gr. 40 (4) (2021) 55.

[28] Z. Marschner, S. Sellán, H.-T. D. Liu, A. Jacobson, Constructive solid geometry on neural signed distance fields, in: SIGGRAPH Asia 2023 Conference Papers, 2023, p. 121.

[29] H. Fu, S. Zhou, L. Liu, N. J. Mitra, Animated construction of line drawings, ACM Trans. Gr. 30 (6) (2011) 133.