DOI: 10.1111/cgf.14358
Eurographics Symposium on Geometry Processing 2021
K. Crane and J. Digne
(Guest Editors)

Volume 40 (2021), Number 5

Blending of hyperbolic closed curves

A. Ikemakhen and T. Ahanchaou

Faculty of Science and Technology, Cadi Ayyad University (UCA), Marrakesh, Morocco

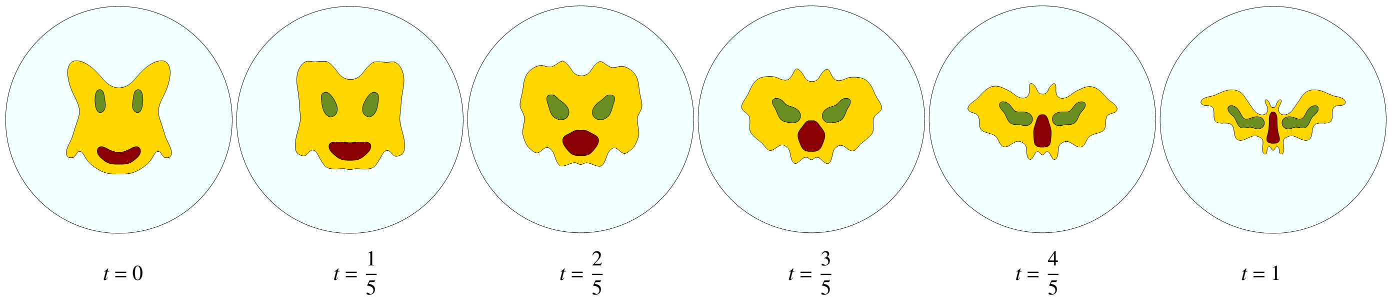


Figure 1: *Morphing sequence between a wolf's face and a bat on the Poincaré disc.*

Abstract

In recent years, game developers are interested in developing games in the hyperbolic space. Shape blending is one of the fundamental techniques to produce animation and videos games. This paper presents two algorithms for blending between two closed curves in the hyperbolic plane in a manner that guarantees that the intermediate curves are closed. We deal with hyperbolic discrete curves on Poincaré disc which is a famous model of the hyperbolic plane. We use the linear interpolation approach of the geometric invariants of hyperbolic polygons namely hyperbolic side lengths, exterior angles and geodesic discrete curvature. We formulate the closing condition of a hyperbolic polygon in terms of its geodesic side lengths and exterior angles. This is to be able to generate closed intermediate curves. Finally, some experimental results are given to illustrate that the proposed methods generate aesthetic blending of closed hyperbolic curves.

CCS Concepts

*• **Theory of computation** → **Computational geometry;** • **Mathematics of computing** → **Interpolation;** • **Computing methodologies** → **Animation;***

1. Introduction

Shape blending (or morphing) consists in making a continuous sequence of transformations from a source object to a target one. Morphing has wide practical use in areas such as computer graphics, animation and modeling. The blending between two closed curves plays an important role in the area of generation of animation and especially computer games [CWKBC13, BBCW10]. Recently, game developers are interested in developing games in the hyperbolic space [GMV15, KCC17].

Our investigation in this paper is to deal with the blending problem of closed curves in the hyperbolic plane. There are several models of hyperbolic plane. The most famous is the Poincaré disc, whose boundary represents infinity and in which the geodesics consist of all circular arcs contained within that disc that are orthogonal to the boundary of the disc, plus all diameters of the disc. Most existing closed curve blending methods were studied in the Euclidean plane [DSL15, SSHS14, SGWM93]. They consist in approximating the source and target closed curves by inscribed closed polygons. And use the linear interpolation approach. Precisely, they interpolate the geometric invariants of the inscribed polygons (side lengths, exterior angles or discrete curvatures). Then they reconstruct the intermediate polygon from the linear interpolation of these characteristics using the exterior angle blending method or the curvature blending one. Intermediate polygons are not necessarily closed. The closing is done by numerical optimization methods.

Another study of the blending of closed plane curves was made by Masahiro Hirano et al. [HWI17]. Their approach is to formulate the

closed curve blending problem in terms of curvature flow.
The closing condition of an oriented plane polygon can be easily expressed as a function of its side lengths and its exterior angles. But for a hyperbolic polygon (witch is a sequence of hyperbolic geodesic connecting a set of vertices), the closing condition is not easy to obtain. Because we do not have the notion of vector on the hyperbolic plane, but only the notion of hyperbolic geodesics and the Möbius group that act on the hyperbolic plane. This paper is organized as follows. In Section 3, we derive the necessary and sufficient conditions for a hyperbolic polygon to be closed in terms of its hyperbolic side lengths and exterior angles by using Möbius transformations and the geometry of the Poincaré disc. In Section 4, we give two algorithms for morphing between two closed hyperbolic polygons. Namely, the curvature blending algorithm and the exterior angle blending one. The first algorithm is based on linear interpolation of hyperbolic side lengths and discrete geodesic curvature of the source and target polygons. While the second, is based on linear interpolation of the exterior angles instead of discrete geodesic curvature. In Section 5, we give some experimental examples to illustrate that the proposed methods generate aesthetic blending of closed hyperbolic curves, and also a comparison between the two methods. We end this section by giving some substantial applications.

2. Related work

Many studies on blending between two Euclidean planar closed curves have been carried out. Most of these methods consist in discretizing the curves and the problem of blending between closed curves becomes then a problem of blending between closed polygons. Not that even the source and target curves are closed, this is not so for the intermediate curves. Sederberg et al. [SGWM93] proposed an algorithm based on linearly interpolating edge lengths and exterior angles of the source and target polygons. To ensure that the intermediate polygon is closed, they change its edge lengths by solving an optimization problem to find the perfect ones that allow to close the polygon. They were able to give an explicit solution to their problem. Another approach was given in [SE02]. The authors linearly interpolate the signed curvatures of the source and the target curves, and used it to construct the intermediate curves. The results presented in [SE02] have a natural appearance without artificial shrinks and distortions. Unfortunately, the method presented does not guaranty for intermediate curves to have no self-intersection. In both papers [SSHS14, DSL15], the authors approximate the source and target curve by their inscribed polygons, then they interpolate the discrete geodesic curvature and the edge lengths to reconstruct intermediate curves. Again, the intermediate polygon is not usually closed. to ensure that, they modify its intrinsic parameters and solve an optimization problem. Recently, Hirano et al. gave a new method to blend between two given curves [HWI17]. They use the notion of curvature flow to derive a linear condition for closed curves, which makes the optimization part faster. And the results of these methods seem to be perfect and visually pleasing. All these methods mentioned above work for Euclidean planar curves and there is no way to use them in hyperbolic geometry for some reasons:
1. The notion of a planar polygon must be replaced by the geodesic polygon.
2. The Euclidean planar closure condition of a polygon must be replaced by another condition that involves geodesics and the transformations of the Poincaré disc.
3. The intermediate curves must lie within Poincaré disc.
Our work will answer these questions, using the hyperbolic geometry of the Poincaré disc.

3. 2D hyperbolic geometry and the Poincaré disc model

Throughout this paper, we take the Poincaré disc as a model of the hyperbolic geometry. In this section we recall the geometry of the Poincaré disc. And we give the expression of the exponential map and the one of the logarithm function. We also give the expression of the geodesic connecting two points in the Poincaré disc. For more details see [IB92, ET97].
A Hyperbolic geometry is a non-Euclidean geometry having constant sectional curvature -1. In hyperbolic geometry, the sum of angles of a triangle is less than π, and triangles with the same angles have the same areas. Furthermore, not all triangles have the same angle sum. The best-known model of 2-dimensional hyperbolic geometry is the Poincaré disc, also called the conformal disc model.

3.1. The Poincaré disc

The Poincaré disc is the open unit disc $\mathbb{D} = \{z \in \mathbb{C} \mid |z| < 1\}$ equipped with the Riemannian metric

$$g = 4\frac{|dz|^2}{\left(1-|z|^2\right)^2}$$

where $dz = dx + idy$ and $|dz|^2 = dx^2 + dy^2$. The hyperbolic distance between two given points $z_1, z_2 \in \mathbb{D}$ is given by

$$\cosh(d(z_1,z_2)) = 1 + \frac{|z_1 - z_2|^2}{(1-|z_1|^2)(1-|z_2|^2)}.$$

Proposition 3.1 The geodesics in the Poincaré disc are the line segments through the origin and the circular arcs that intersect the boundary orthogonally.

If we set

$$SU(1,1) := \left\{ \begin{pmatrix} a & b \\ \bar{b} & \bar{a} \end{pmatrix} \mid a,b \in \mathbb{C} \mid a\bar{a} - b\bar{b} = 1 \right\}.$$

we have

Proposition 3.2 The projective special unitary group $PSU(1,1) := SU(1,1)/\pm I$ acts transitively on the Poincaré disc $\mathbb{D}$ as following:

$$\begin{array}{rcccc} \rho & : & PSU(1,1)\times\mathbb{D} & \to & \mathbb{D}, \\ & & \left(\begin{pmatrix} a & b \\ \bar{b} & \bar{a} \end{pmatrix}, z\right) & \mapsto & \dfrac{az+\bar{b}}{\bar{b}z+\bar{a}}. \end{array} \quad (1)$$

And the isotropy group of the origin O is $\left\{ \begin{pmatrix} e^{i\theta} & 0 \\ 0 & e^{-i\theta} \end{pmatrix} \mid \theta \in \mathbb{R} \right\}$.
Moreover, for each $z \in \mathbb{C}$, there exists a matrix $A_z = \frac{1}{\sqrt{1-|z|^2}} \begin{pmatrix} -i & iz \\ -i\bar{z} & i \end{pmatrix} \in PSU(1,1)$ such that

$$\rho_z(0) := \rho(A_z, 0) = z.$$

Let $z \in \mathbb{D}$ and $T \in \mathbb{C}$ such that $|T| = 1$. Then the geodesic γ passing through z and oriented by T has the following expression :

$$\gamma(t) = \rho_z\Big(\tanh(\frac{t}{2})T\Big), \quad \text{for } t \in \mathbb{R}. \tag{2}$$

Proposition 3.3 Let $z_1, z_2 \in \mathbb{D}$ and $V \in T_{z_1}D = \mathbb{C}$. The exponential map $\exp_{z_1} : T_{z_1}\mathbb{D} \to \mathbb{D}$ is given by

$$\exp_{z_1}(V) := \rho_{z_1}\Bigg(\tanh(\frac{\sqrt{g(V,V)}}{2})\frac{V}{|V|}\Bigg). \tag{3}$$

And the unit vector tangent at z_1 which generates the geodesic joining z_1 to z_2 is

$$\log_{z_1}(z_2) := \frac{1}{\tanh\left(\frac{d}{2}\right)}\rho_{z_1}^{-1}(z_2), \tag{4}$$

where $d = d(z_1, z_2)$.

Now, from the above, we can define the geodesic connecting two given points $z, w \in \mathbb{D}$ by

$$\gamma_{z,w}(t) := \rho_z\Big(\tanh(\frac{t}{2})\log_z(w)\Big), \quad \text{for } t \in [0, d]; \tag{5}$$

where $d = d(z, w)$.

Definition 3.1 The rotation around the origin O by angle θ is

$$R(\theta) := \begin{pmatrix} e^{\frac{i\theta}{2}} & 0 \\ 0 & e^{\frac{-i\theta}{2}} \end{pmatrix}.$$

And the translation of length d along the geodesic that maps -1 to 1 is

$$L(d) := \begin{pmatrix} \cosh(\frac{d}{2}) & \sinh(\frac{d}{2}) \\ \sinh(\frac{d}{2}) & \cosh(\frac{d}{2}) \end{pmatrix}$$

.

4. Closure condition

In this section, we give the condition for a hyperbolic polygon to be closed in terms of its hyperbolic side lengths and exterior angles. Let $P := [z_0, ..., z_n]$ be a hyperbolic polygon that is a sequence of geodesics γ_k joining z_k and z_{k+1} for $k \in \{0, ..., n-1\}$. We denote by $d_k := d(z_k, z_{k+1})$ the length of the geodesic γ_k. We denote $\delta_k := \sphericalangle(\log_{z_k}(z_{k+1}), -\log_{z_k}(z_{k-1}))$ the exterior angle at z_k (see Fig. 2). Note that the angle is positive, if it is clockwise.

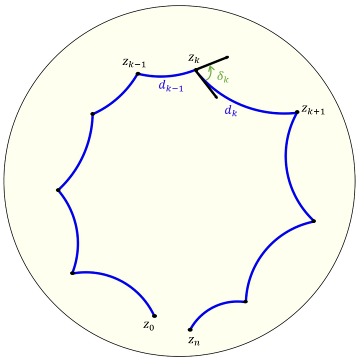


Figure 2: *Hyperbolic polygon*

If $z_0 = z_n$, then P is closed and δ_0 is the exterior angle at z_0. Otherwise P is open.

We define the discrete geodesic curvature of P at the vertex z_k (see its meaning in the appendix (A)) by:

$$\kappa_k = \frac{2\delta_k}{d_{k-1} + d_k}. \tag{6}$$

If P is closed, the discrete geodesic curvature at z_0 is defined by:

$$\kappa_0 = \frac{2\delta_0}{d_{n-1} + d_0}.$$

This notion of discrete geodesic curvature will be used in the curvature blending algorithm in Section 4.

4.1. Hyperbolic triangle

Let $\Pi = [z_0 = z_3, z_1, z_2, z_3]$ be a closed hyperbolic triangle with hyperbolic sides d_0, d_1, d_2 and exterior angles δ_0, δ_1, δ_2. Without loss of generality we can assume that $z_0 = 0$ and z_2 is a point on the geodesic that send -1 to 0 (see Fig. 3a).

The fact that $z_0 = z_3$ is equivalent to the following equation

$$\rho(R(\delta_0)L(d_0)R(\delta_1)L(d_1)R(\delta_2)L(d_2), z_k) = z_k, \qquad \text{for } k = 0, ..., 3. \tag{7}$$

Which means that

$$R(\delta_0)L(d_0)R(\delta_1)L(d_1)R(\delta_2)L(d_2) = \pm I. \tag{8}$$

Set $A * \Pi := \rho(A, .)$, for $A \in PSU(1,1)$.

The explanation of Eq. (8) is as follows. The first action of $R(\delta_2)L(d_2)$ on the triangle Π gives a new triangle Π_1, where $L(d_2)$ moves z_2 to z_0. In this way z_2 is the origin of the disc $\mathbb{D}$. While the rotation $R(\delta_2)$ brings z_1 to a point in the geodesic that connects -1 to 0 (see Fig. 3b). For the second action $R(\delta_1)L(d_1)$, the translation $L(d_1)$ moves z_1 to z_2 (z_1 is the origin of the disc $\mathbb{D}$), while the rotation $R(\delta_1)$ sends z_0 to a point in the geodesic that connecting -1 and 0 (see Fig. 3c). Finally, the translation $L(d_0)$ brings z_0 to its origin position, then the rotation $R(\delta_0)$ returns all the vertices at their original positions (see Fig. 3d). This sequence of actions let the triangle Π invariant. Which means that, after this sequence of transformations, we get $\Pi_3 = \Pi$. That explains Eq. (8).

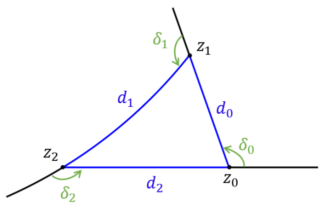
4.2. Hyperbolic polygon

Now, let $P := [z_0, ..., z_n]$ be a hyperbolic polygon, with vertices $z_0, ..., z_n$. With the same observations for the hyperbolic triangle case, the condition for P to be closed is

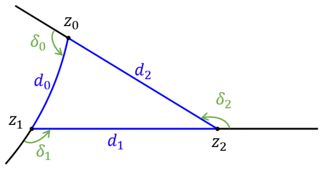
$$R(\delta_0)L(d_0)R(\delta_1)L(d_1)\cdots R(\delta_{n-2})L(d_{n-2})R(\delta_{n-1})L(d_{n-1}) = \pm I. \tag{9}$$

In other words, if we put $S := \prod_{k=0}^{n-1} R(\delta_{n-1-k})L(d_{n-1-k})$, then Eq. (9) can be written:

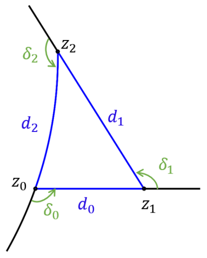
$$S = \pm I. \tag{10}$$



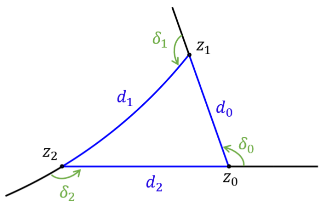
(a) *Triangle* Π



(b) $\Pi_1 = R(\delta_2)L(d_2) * \Pi$



(c) $\Pi_2 = R(\delta_1)L(d_1) * \Pi_1$



(d) $\Pi_3 = R(\delta_0)L(d_0) * \Pi_2$

Figure 3: *Illustration of closure condition for hyperbolic triangle.*

Now, if we look at P as an open polygon, it has no exterior angle δ_0 and no geodesic curvature κ_0 at z_0. We define the last as follows:

$$\delta_0 := \sphericalangle(\log_{z_0}(z_1), -\log_{z_0}(z_{n-1})); \tag{11}$$

and

$$\kappa_0 = \frac{2\delta_0}{d(z_{n-1}, z_0) + d_0}.$$

According to the above remarks, we get

Proposition 4.1 Let $P := [z_0, ..., z_n]$ be a hyperbolic polygon having $d_1, ..., d_{n-1}$ as hyperbolic side lengths, and $\delta_1, ..., \delta_{n-1}$ as exterior angles. We denote by δ_0 the exterior angle at z_0 defined in Eq. (11). Then the following propositions are equivalent:

1. P is closed.

2. $$S = \pm I. \tag{12}$$

3. $$\left\{ \begin{array}{rcl} |\,tr(S)\,| & = & 2, \\ det(S) & = & 1, \\ s_2\bar{s}_2 & = & 0. \end{array} \right. \tag{13}$$

Where $S := \begin{pmatrix} s_1 & s_2 \\ \bar{s}_2 & \bar{s}_1 \end{pmatrix}$.

5. Morphing of hyperbolic polygons

Let $P^0 = [z_0^0, ..., z_n^0]$ and $P^1 = [z_0^1, ..., z_n^1]$ be two hyperbolic polygons of $n+1$ vertices. A blending between P^0 and P^1 is a continuous function $t \to P^t \in \mathbb{D}$ defined for $t \in [0,1]$ such that it coincides with the source polygon P^0 for $t = 0$ and with the target polygon P^1 for $t = 1$. Our aim is to build the intermediate morph P^t. A natural approach to construct a morph between P^0 and P^1 is to linearly interpolate the intrinsic variables of source and target polygons.
In this section we propose two morphing algorithms: the curvature blending and the exterior angle blending one.
For the first algorithm, the intermediate morph P^t is a polygon with hyperbolic side lengths:

$$d_k^t = (1-t)d_k^0 + t\,d_k^1, \quad \text{for } k = \{0, ...n-1\}, \tag{14}$$

and with discrete geodesic curvatures:

$$\kappa_k^t = (1-t)\kappa_k^0 + t\,\kappa_k^1, \quad \text{for } k = \{0, ...n-1\}. \tag{15}$$

For the second algorithm, the intermediate morph P^t is the polygon of hyperbolic side lengths d_k^t defined in (14) and with exterior angles:

$$\delta_k^t = (1-t)\delta_k^0 + t\,\delta_k^1, \quad \text{for } k = \{0, ...n-1\}. \tag{16}$$

5.1. Construction of the intermediate polygon P^t

In this section we will give the process to construct the polygon P^t by using its intrinsic variables d_k^t and δ_k^t or κ_k^t.
$P^t := [z_0^t, ..., z_n^t]$ is constructed in two steps:

- Firstly, we construct the initial vertex z_0^t and the second one z_1^t.
- Secondly, we use the hyperbolic side lengths d_k^t and the exterior angles δ_k^t to complete the construction of P^t.

The details of construction are given in the following.

5.1.1. Construction of z_0^t and z_1^t

Let γ be the hyperbolic geodesic connecting the initial point z_0^0 of P^0 and the initial point z_0^1 of P^1. If we let $l_0 := d\left(z_0^0, z_0^1\right)$, we take

$$z_0^t := \gamma(t\,l_0). \tag{17}$$

Now, let α_0 (respectively α_1) be the angle between the x-axis and $T_0 := \log_{z_0^0}\left(z_1^0\right)$ (respectively $T_1 =: \log_{z_0^1}\left(z_1^1\right)$).
Set

$$\alpha_t := (1-t)\alpha_0 + t\,\alpha_1, \quad t \in [0,1]. \tag{18}$$

Let $T_t = \cos(\alpha_t) + i\sin(\alpha_t)$. Then by Eq (2), the geodesic β_t passing through z_0^t and oriented by T_t is

$$\beta_t(s) = \rho_{z_0^t}\left(\tanh(\frac{s}{2})T_t\right), \qquad \text{for } s \in \mathbb{R}.$$

We choose z_1^t to be in the geodesic β as it shown in Fig. 4 such that

$$z_1^t := \beta_t\left(d_0^t\right). \tag{19}$$

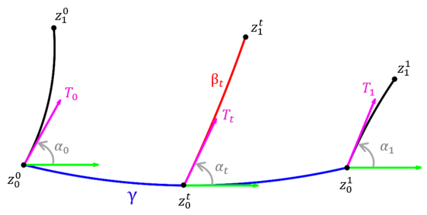


Figure 4: *Construction of z_0^t and z_1^t*

5.1.2. Construction of z_k^t for $k = \{2, \ldots n\}$

z_0^t, z_1^t are respectively constructed in (17)-(19), and δ_k^t is computed either directly by (16) or from κ_k^t in (15) using (6).

Now, by induction, we can construct the other vertices z_k^t of P^t using the edge lengths and the exterior angles as follows:

$$z_k^t = \rho_{z_{k-1}^t}\left(\tanh\left(\frac{d_{k-1}^t}{2}\right)T_{k-1}^t\right), \quad \text{for } k = \{2, \ldots n\} \tag{20}$$

where $T_{k-1}^t = -e^{\left(-i\delta_{k-1}^t\right)}\log_{z_{k-1}^t}\left(z_{k-2}^t\right)$.

5.2. Morphing of two hyperbolic closed polygons

Let P^0 and P^1 be two closed polygons, and $t \in [0,1]$. The intermediate morph P^t constructed for both methods in Section 5.1 is not necessary closed. For that, we will change the exterior angles δ_k^t in the smallest possible way to close P^t.
This means we seek $\epsilon_0, \ldots \epsilon_{n-1}$ such that the polygon $\bar{P}^t$, with hyperbolic side lengths d_k^t and exterior angles $\bar{\delta}_k^t := \delta_k^t + \epsilon_k$ will be closed and the norm $\| \bar{\kappa}^t - \kappa^t \|^2$ will be minimized.
Where κ^t (resp. $\bar{\kappa}^t$) denotes the vector of components κ_k^t (resp. $\bar{\kappa}_k^t := \dfrac{2\bar{\delta}_k^t}{d_{k-1}^t + d_k^t}$), and $\| \,.\, \|$ is the Euclidean norm in $\mathbb{R}^n$. In order to solve this, we minimize the following problem:

$$\min_{(\epsilon_0,\ldots\epsilon_{n-1})\in\mathbb{R}^n} \sum_{k=0}^{n-1} \left| \frac{4\epsilon_k^2}{\left(d_{k-1}^t + d_k^t\right)^2} \right|. \tag{21}$$

Subject to constraint equality given in Proposition 4.1:

$$\begin{cases} |\,tr(S)\,| &= 2, \\ det(S) &= 1, \\ s_2\bar{s}_2 &= 0. \end{cases} \tag{22}$$

where

$$S := \prod_{k=0}^{n-1} R(\delta_{n-1-k} + \epsilon_{n-1-k})L(d_{n-1-k}),$$

This will ensure the closure of the hyperbolic polygon $\bar{P}^t$.

5.3. Algorithms

In the procedure of Section 4.1, we preserve the side lengths, and we modify the exterior angles to ensure the closure of the polygon $\bar{P}^t$. Unfortunately, this leads to a non-linear optimization problem (21) - (22). To overcome it, we use the off-the-shelf solver fmincon in Matlab. So we can construct the polygon $\bar{P}^t$ by applying the following algorithms.

Algorithm 1: Exterior angle blending

```
Input:
Source and target polygons $P^0$,$P^1$.
A time-step $t$ for which we evaluate $P^t$.
Output:
A closed polygon $\bar{P}^t$.
Do:
    Compute the lengths $d_k^t$ and exterior angles $\delta_k^t$ using
Eqs. (14)-(16),
    Compute $\epsilon_i$, $i \in \{0, \ldots, n-1\}$ solutions of (21)-(22) and
then $\bar{\delta}_i^t$,
    Reconstruct the polygon $\bar{P}^t$ using Eqs. (17)-(19)-(20).
```

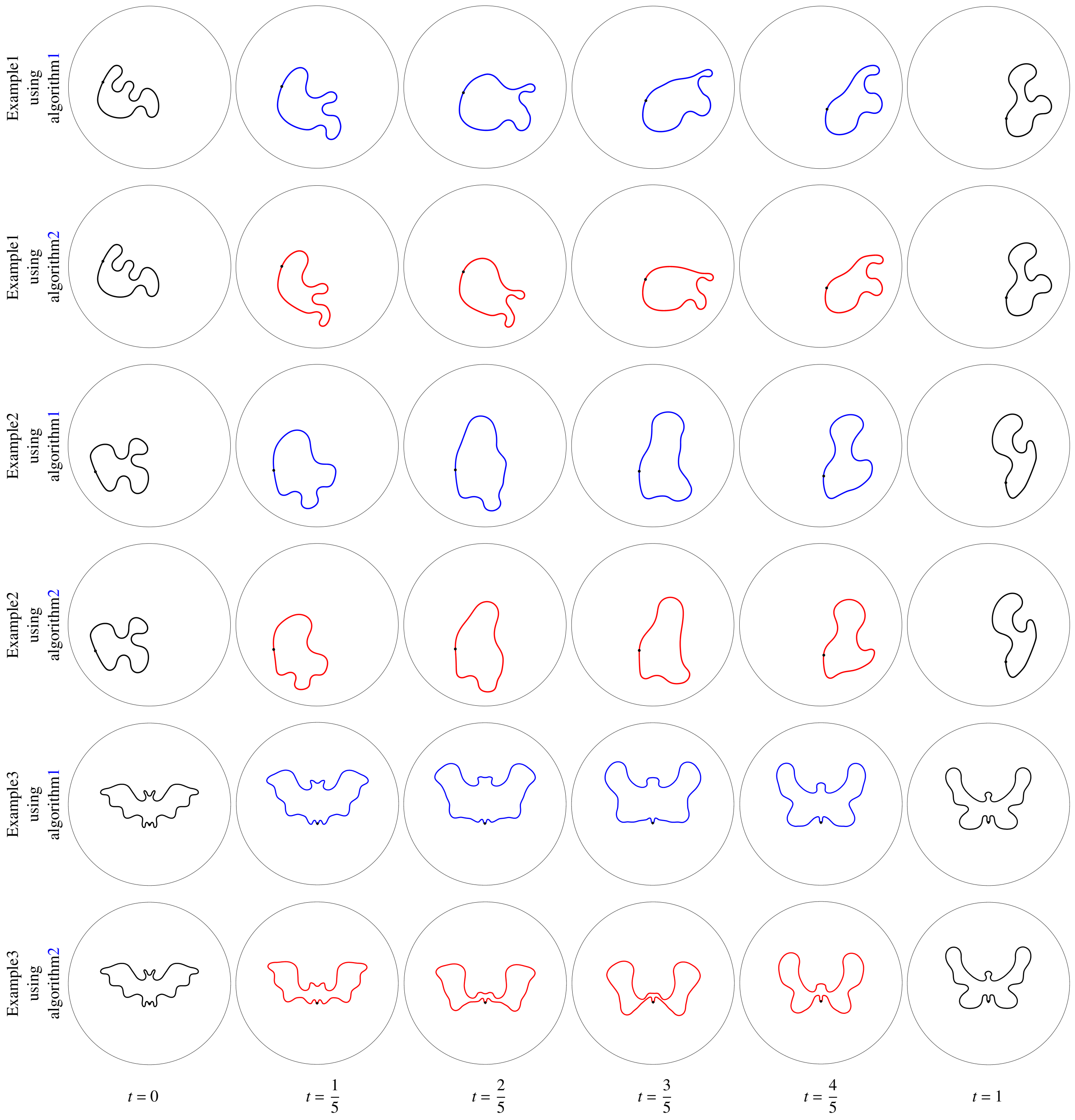
Algorithm 2: Curvature blending

```
Input:
Source and target polygons $P^0$,$P^1$.
A time-step $t$ for which we evaluate $P^t$.
Output:
A closed polygon $\bar{P}^t$.
Do:
    Compute the lengths $d_k^t$ and discrete geodesic
curvatures $\kappa_k^t$ using Eqs. (14) - (15),
    Compute $\delta_k^t$ using (6),
    Compute $\epsilon_i$, $i \in \{0, \ldots, n-1\}$ solutions of (21)-(22) and
then $\bar{\delta}_i^t$,
    Reconstruct the polygon $\bar{P}^t$ using Eqs. (17)-(19)-(20).
```

6. Results

In this section, we provide evaluation of our hyperbolic curve morphing algorithms for different closed curves. All methods were implemented in MATLAB R2016a and all the experiments were run on a Intel(R) Core(TM) i7-4500U CPU @ 1.80 GHz 2.40 GHz machine. As we mentioned above, the optimization part was done using the standard fmincon solver in Matlab with all its defaults settings except 'MaxFunctionEvaluations' which we change to 500000.

Approximation of curves in $\mathbb{D}$. Let $\gamma : [a,b] \to \mathbb{D}$ be a closed curve in the Poincaré disc $\mathbb{D}$. We can approximate γ by a closed hyperbolic polygon $P := [z_0, \ldots, z_n]$ with $n+1$ vertices by taking $z_k = \gamma(t_k)$, where $t_k = a + k\dfrac{|\,b-a\,|}{n}$ is an uniform subdivision of $[a,b]$. P is an inscribed hyperbolic polygon in γ. It gives a good approximation of γ as long as the integer n is large enough. Furthermore, we show that the discrete hyperbolic geodesic of P at



Example1 using algorithm1
Example1 using algorithm2
Example2 using algorithm1
Example2 using algorithm2
Example3 using algorithm1
Example3 using algorithm2
$t = 0$
$t = \frac{1}{5}$
$t = \frac{2}{5}$
$t = \frac{3}{5}$
$t = \frac{4}{5}$
$t = 1$

Figure 5: *Comparison of the results using both algorithms. The dot indicates the start and end point of the curves.*

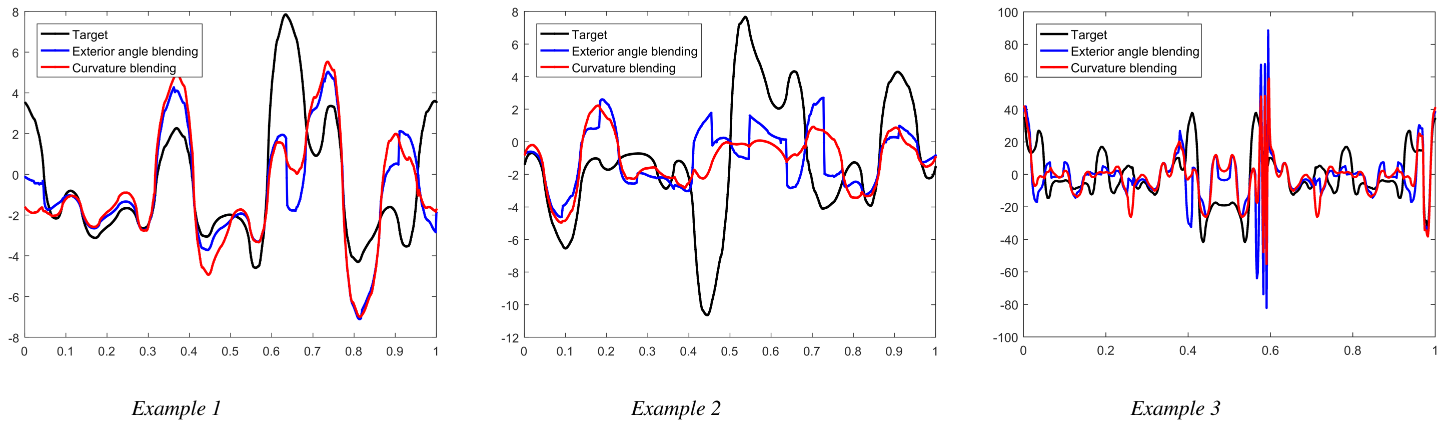


Figure 6: *Plots of geodesic curvature of the target curves and intermediate curves at time* $t = \frac{1}{2}$ *used in Fig. 5.*

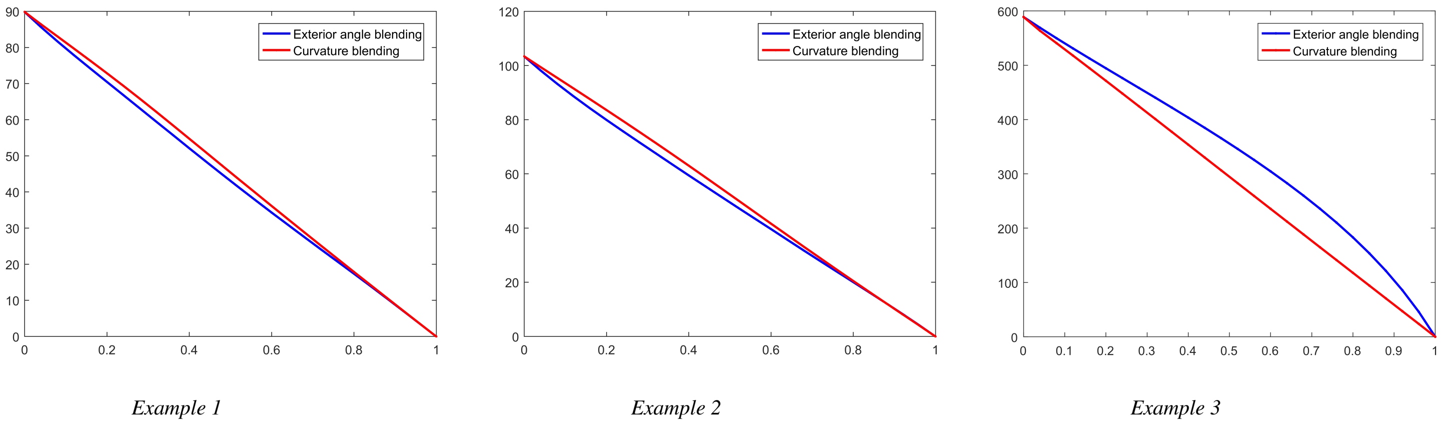


Figure 7: *Norm of the difference between the geodesic curvature of intermediate curves and the target one of examples in Fig. 5.*

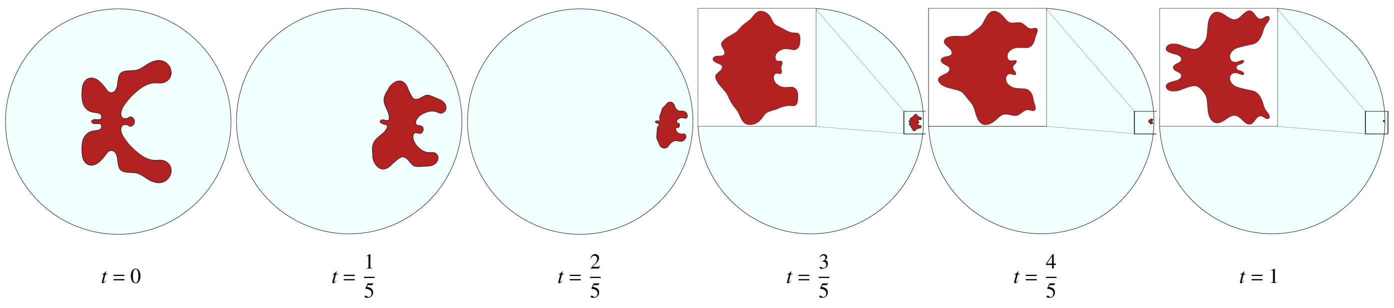


Figure 8: *Morphing sequence between a butterfly and a bat at infinity on the Poincaré disc.*

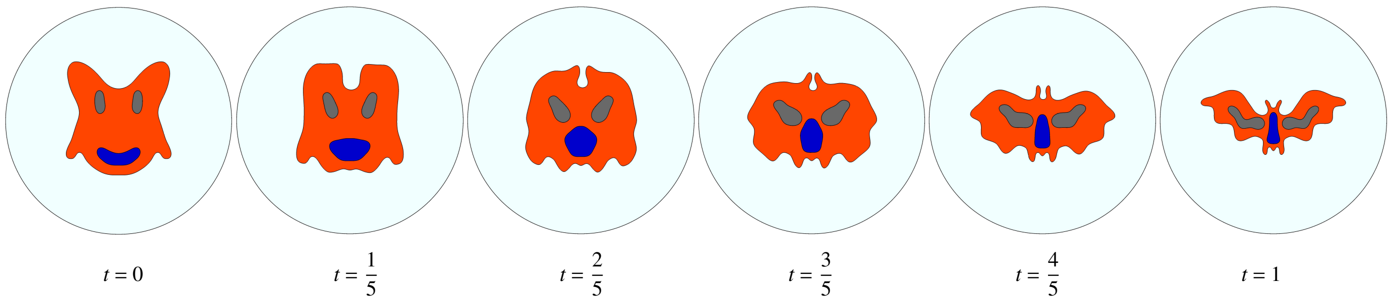


Figure 9: *Morphing sequence between a wolf's face and a bat on the Poincaré disc using algorithm 2.*

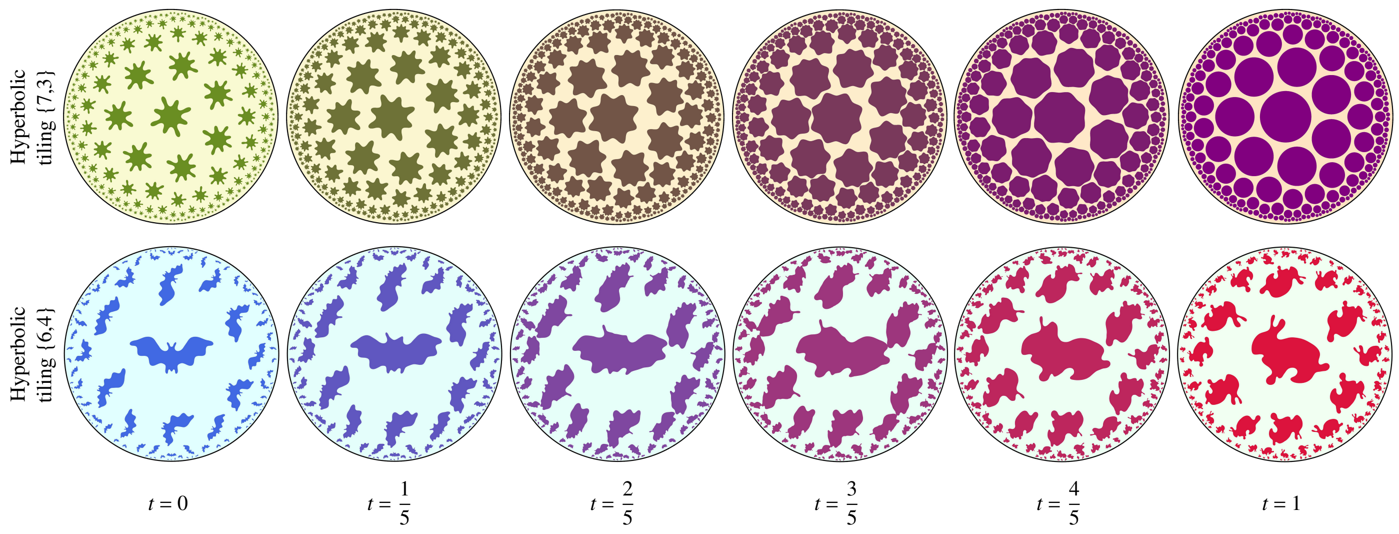


Figure 10: *Tiling the Poincaré disc using intermediate curves generated by blending two given motifs using algorithm 1.*

	Example 1	example 2	Example 3
number of samples	353	353	529
runtime using 1	423,6039	776,2036	831,8603
runtime using 2	288,0755	187,5172	2,7087 10^3
distance between first and last point using 1	1,4355 10^{-6}	1,7103 10^{-6}	3,8671 10^{-6}
distance between first and last point using 2	3,2224 10^{-6}	1,1159 10^{-6}	2,8465 10^{-6}

Table 1: *Comparison of runtime (in seconds) for intermediate curve at time* $t = \frac{1}{2}$ *of examples in Fig. 5 and their closure error.*

a point z_k defined in Eq. (6) estimates the geodesic curvature of γ at z_k.
Now let γ_0 and γ_1 be two closed smooth curves in $\mathbb{D}$ (γ_0 is the source and γ_1 is the target). Let $n \in \mathbb{N}$. We approximate γ_0, γ_1 respectively by two closed hyperbolic polygons P^0, P^1 with the same number of vertices $n+1$.
Fig. 5 indicates three different examples. In each example we plot intermediate curves between the source and target curves using algorithm 1 or 2. We remark that for both methods, the intermediate curves remain inside the disc and are closed and aesthetic. Fig. 6 shows, for the three examples in Fig. 5, that the geodesic curvatures for intermediate curves $t = \frac{1}{2}$ are close to the one of the target curve. Fig. 7 clearly shows for the two methods that the norm of the difference between the geodesic curvature of the intermediate curves and the target curve goes to 0 as t tends to 1. In addition, there are no "spikes" in the two plots (red and blue). Therefore, the intermediate curves do not have cusps.
Note that, the increase in the number of vertices of the source and target curves is done by the method of hyperbolic interpolatory geometric subdivision schemes. This schemes give G^1-continuous limit curves. So by construction, if the two given curves are G^1-continuous so are the intermediate curves independently of step-time. In addition, for each fixed time t, fmincon search the ε_i so as to find a closed curve by keeping the property of G^1-continuity. In conclusion, even if we decrease the step-time, our methods does not present a jump and they give aesthetic curves.
Table 1 contains the computational results that we used to produce curves at time $t = \frac{1}{2}$. It shows that the closure constraint has been reached. However, in runtime perspective, we remark that it depends on the number of vertices.
In practice, we remark the distance between the first point and the last one remains constant even if we reduce the constraint

tolerance. Another remark is that, in example 3 (Fig. 5), the shape generated by the curve is almost preserved by the first method during the blending while the second risks losing it at $t = \frac{3}{5}$.

Applications: In this part, we give three applications to our methods. It could be used to develop some computer games on hyperbolic plane.

Blending to infinity. This application consists to morph an object to another that is at infinity. Here infinity is represented by the border of the Poincaré disc. Fig. 8 illustrates an example of this kind of morphing.

Blending network curves. This second application consists in doing the blending between two network of curves with the same number of curves in Poincaré disc. That means, we blend each curve of the first network by its correspondent in the second network. Fig. 1 indicates network blending between wolf's face and a bat using algorithm 1. Likewise for Fig. 9 using algorithm 2.

Hyperbolic tiling animation. This third application consists to animate hyperbolic tiling, which produces a beautiful hyperbolic images (animations). In Fig. 10 two given motifs γ_0 and γ_1 are blended, and for each time t, we generate hyperbolic tiling using in-between motif γ_t. Fig. 10 indicates a sequence of pictures that could give a nice animation. We have used the hyperbolic tiling {7,3} and {6,4}.

7. Conclusion

In this paper, we have presented two novel algorithms for blending between two curves in the Poincaré disc, using their intrinsic variables. Both methods generate closed intermediate smooth curves by using the closure condition and by solving an optimization problem. In practice, the algorithm 2 failed when we deal with polygons with a large number of vertices ($(n \geqslant 300)$). It gives, sometimes, intermediate curves with self intersecting even if the source curve or the target one is not self intersecting. While the algorithm 1 works for any polygon with any number of vertices.

Limitation. Both algorithms take a long time to generate intermediate curves because of complexity of non linear constraint (matrix 2x2 $\in \mathbb{C}$). Therefore, these methods can't be applied for a real-time execution. The goal of our future work is to give a rapid blending method which reduces the runtime.

Appendix A: Geodesic curvature

In general, a C^2-curve σ (red one) on a Riemannian surface S can be approximate by an inscribed geodesic polygon P (green one) (see Fig. 11). And the geodesic curvature $\kappa(p)$ of σ at a vertex p can be approximate by the discrete geodesic curvature of P at p and it is not hard to show that:

$$\kappa(p) = \lim_{\substack{p_1,p_2 \to p \\ p_1,p_2 \in \Sigma}} \frac{2\,\delta}{d(p_1,p)+d(p,p_2)},$$

where Σ is the support of σ, δ the exterior angle of the triangle $[p_1,p,p_2]$ at p and d the distance on the surface S.

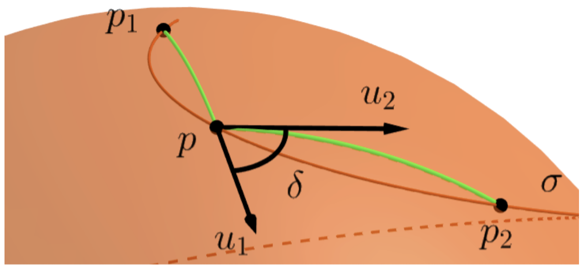


Figure 11: *Approximation of the geodesic curvature by the discrete one.*

References

[BBCW10] Bernhardt A., Barthe L., Cani M.-P., Wyvill B.: Implicit blending revisited. In *Computer Graphics Forum* (2010), vol. 29, Wiley Online Library, pp. 367–375. 1

[CWKBC13] Chen R., Weber O., Keren D., Ben-Chen M.: Planar shape interpolation with bounded distortion. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 1–12. 1

[DSL15] Dym N., Shtengel A., Lipman Y.: Homotopic morphing of planar curves. In *Computer Graphics Forum* (2015), vol. 34, Wiley Online Library, pp. 239–251. 1, 2

[ET97] Earp R., Toubiana E.: *Introduction à la géométrie hyperbolique et aux surfaces de Riemann*. Bibliothèque des sciences. Diderot Editeur Arts Sciences, 1997. 2

[GMV15] Guimaraes F., Mello V., Velho L.: Geometry independent game encapsulation for non-euclidean geometries. In *Proceedings of SIBGRAPI Workshop of Works in Progress* (2015). 1

[HWI17] Hirano M., Watanabe Y., Ishikawa M.: Rapid blending of closed curves based on curvature flow. *Computer Aided Geometric Design 52* (2017), 217–230. 1, 2

[IB92] Iversen B., Birger I.: *Hyperbolic geometry*, vol. 25. Cambridge University Press, 1992. 2

[KCC17] Kopczynski E., Celinska D., Ctrnáct M.: Hyperrogue: Playing with hyperbolic geometry. In *the proceedings of the Bridges Conference, July* (2017), pp. 27–31. 1

[SE02] Surazhsky T., Elber G.: Metamorphosis of planar parametric curves via curvature interpolation. *International Journal of Shape Modeling 8*, 02 (2002), 201–216. 2

[SGWM93] Sederberg T. W., Gao P., Wang G., Mu H.: 2-d shape blending: an intrinsic solution to the vertex path problem. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques* (1993), pp. 15–18. 1, 2

[SSHS14] Saba M., Schneider T., Hormann K., Scateni R.: Curvature-based blending of closed planar curves. *Graphical models 76*, 5 (2014), 263–272. 1, 2