Polynomial Cauchy Coordinates for Curved Cages

ZHEHUI LIN, University of Science and Technology of China, China
RENJIE CHEN*, University of Science and Technology of China, China

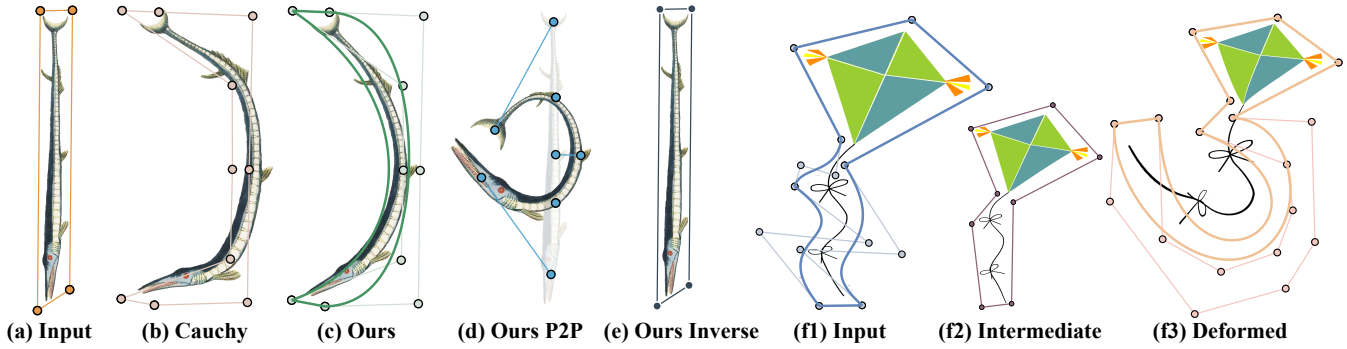

Fig. 1. (a) Input shape and its corresponding polygonal cage; (b) Deformation result of Cauchy coordinates [Weber et al. 2009]; (c) Deformation result of our Polynomial Cauchy coordinates; (d) P2P deformation result using our coordinates; (e) Inverse mapping obtained using our method to recover the original shape from (c). (f1-3) Deformation between curved cages using our method.

Barycentric coordinates are widely used in computer graphics, especially in shape deformation. Traditionally, barycentric coordinates are defined for polygonal domains. In this work, we relax this requirement by representing the boundary of the domain using a Bézier spline and extend the complex-valued Cauchy barycentric coordinates [Weber et al. 2009] to the Bézier case. Compared to the latest polynomial 2D Green coordinates [Michel and Thiery 2023], we obtain equivalent results. We further derive a numerical integration formula for the inverse mapping based on Cauchy's integral formula, enabling deformation between curved cages through an intermediate step. Notably, our approach allows curved cages as input. We also provide expressions for the nth-order derivatives of the coordinates, which facilitate constrained deformations with position constraints. Through extensive experiments, we demonstrate the versatility of our coordinates for interactive deformation.

CCS Concepts: • **Computing methodologies** → **Parametric curve and surface models**; **Shape modeling**.

Additional Key Words and Phrases: cage-based modeling, 2D shape deformation, conformal deformations, curve-based deformations, interactive deformations

*Corresponding author

Authors' Contact Information: Zhehui Lin, University of Science and Technology of China, Hefei, Anhui, China, linzhehui626@ustc.edu.cn; Renjie Chen, University of Science and Technology of China, Hefei, Anhui, China, renjiec@ustc.edu.cn.

SA Conference Papers '24, December 3–6, 2024, Tokyo, Japan

ACM ISBN 979-8-4007-1131-2/24/12
https://doi.org/10.1145/3680528.3687654

ACM Reference Format:
Zhehui Lin and Renjie Chen. 2024. Polynomial Cauchy Coordinates for Curved Cages. In *SIGGRAPH Asia 2024 Conference Papers (SA Conference Papers '24), December 3–6, 2024, Tokyo, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3680528.3687654

1 INTRODUCTION

Barycentric coordinates are a powerful mathematical tool that allows us to interpolate data within a region based on scalar or vector fields defined on the boundary of that region. Barycentric coordinates have found wide applications in computer graphics, including shape deformation and image cloning. In its most basic form, shape deformation is achieved by the user moving cage vertices from $\{v_i\}$ to $\{f_i\}$. Interior points are then repositioned as linear combinations of the cage vertices: $f(\eta) = \sum_i \lambda_i(\eta) f_i$, with $\lambda_i(\eta)$ being the barycentric coordinates of η satisfying $\eta = \sum_i \lambda_i(\eta) v_i$. This computation is often fast enough to enable real-time interaction.

Considerable progress has been made in the field of barycentric coordinates, leading to the development of diverse properties and functionalities. Traditionally, barycentric coordinates have been defined within polygonal domains, where the boundary is represented by a collection of linear edges. However, there have been increased interests in using curves to represent domain boundaries given its flexibility and fidelity. Polynomial 2D Green coordinates [Michel and Thiery 2023] (PGC) is a new barycentric coordinate scheme that allows the target cage to be polynomial curves, but this does not extend to the source cage. The complex-valued Cauchy coordinates [Weber et al. 2009] have been shown to be equivalent to Green coordinates [Lipman et al. 2008]. This inspires us to generalize Cauchy coordinates to the case of curved boundaries.

In this work, we first generalize Cauchy coordinates, allowing the boundary of the target cage to be represented by a Bézier spline. Compared to the PGC of Michel and Thiery [2023], we obtain effectively equivalent coordinates with an alternative formulation

based on Bézier splines, offering improved interactivity for the user. Additionally, we explore several extensions of our barycentric coordinates. We derive expressions for the nth-order derivatives of our coordinates and present a numerical integration formula for the inverse mapping using Cauchy's integral formula. This enables us to recover the original shape from the deformation result, leading to a deformation algorithm for curved cages, which has seen limited research. Furthermore, leveraging our barycentric coordinates and their closed-form derivatives, we demonstrate the application of our coordinates in constrained deformation. We showcase the results of our method on both simple and complicated cages. The closed-form expressions, coupled with the curved-to-curved mapping capability, make our coordinates a powerful and versatile tool for shape deformation and image editing.

Our key contributions include:

- Extension of complex Cauchy barycentric coordinates to the polynomial domain, offering closed-form expressions..
- Formulation of coordinates using Bézier representation, enabling intuitive, interactive editing of target curves via control polygons.
- Derivation of nth-order derivatives for our coordinates, facilitating point-to-point constrained and smooth energy-driven deformations.
- Development of inverse barycentric mapping expressions, allowing deformations between curved cages through an intermediate step.

2 RELATED WORK

Barycentric coordinates have been extensively studied in the academic community. Different barycentric coordinates exhibit various properties, e.g. interpolation or shape preservation. Barycentric coordinates can be expressed using real or complex scalars, and some coordinates are provided in closed-form.

Interpolating coordinates. Mean-Value coordinates [Floater 2003; Hormann and Floater 2006; Ju et al. 2005] (MVC) are developed based on the mean value theorem for harmonic functions, allowing interpolation of scalar functions at any point in space. While MVC offers a closed-form expression with interpolation properties, it may produce negative values. MVC has been applied to image editing and cloning [Farbman et al. 2009]. To address the issue of negative values in MVC, Lipman et al. [2007] proposed Positive Mean-Value coordinates, which however lacks a closed-form expression. Harmonic coordinates [Joshi et al. 2007] provides more intuitive shape editing behavior due to their non-negative nature and magnitude decrease with distance measured within the cage. Weber et al. [2012] proposed Biharmonic coordinates, enabling the interpolation of boundary derivative data. Local barycentric coordinates [Zhang et al. 2014] enhances the locality of barycentric coordinates through convex optimization of the total variation. More recently, variational barycentric coordinates [Dodik et al. 2023] are proposed to offer additional control by leveraging the capabilities of neural fields.

Conformal coordinates. Green coordinates [Lipman et al. 2008] (GC) are derived from Green's third identity, and can be used for deformations. Unlike previously mentioned methods, GC does not possess interpolation properties but instead exhibits conformal properties. The Cauchy coordinates proposed by Weber et al. [2009] are complex-valued coordinates and equivalent to GC. Weber et al. [2011] analyzed 2D barycentric mappings with a complex view, demonstrating the flexibility of complex barycentric coordinates over real-valued coordinates. It has been shown that Cauchy coordinates are useful for harmonic shape deformation and interpolation [Chen and Weber 2017; Chien et al. 2016; Shi and Chen 2022].

Coordinates for curved cages. While barycentric coordinates for polygonal domain have been extensively studied, research for curved boundaries remains comparatively limited. Cubic MVC [Li et al. 2013] generalizes MVC to cubic polynomial curves, offering interpolation properties but lack support for higher-order curves. PGC [Michel and Thiery 2023] extends GC to accommodate arbitrary-order polynomial curve targets, though input cages remain restricted to polygons. Transfinite Barycentric Coordinates can also facilitate deformations between smooth cages [Belyaev 2006; Belyaev and Fayolle 2017; Dyken and Floater 2009]. However, the barycentric mapping induced by these coordinates involves an integral and requires numerical approximation in practice, as they assume no properties beyond continuity for the cage.

3 BACKGROUND

Before diving into our method, we first introduce several closely related concepts utilized in subsequent discussions.

3.1 Cauchy's integral formula

The Cauchy's integral formula states that the value of any holomorphic function within a closed region is completely determined by its values on the boundary of that region. Furthermore, it provides a method for calculating the integral of any order derivative at every point within the region. Cauchy's integral formula holds for any closed region D under certain conditions:

- The region D must be simply connected, which means its boundary curve ∂D consists of a single branch without holes or self-intersections.
- The function $f(z)$ being integrated must be analytic within the region D, i.e. holomorphic.
- The integration path ∂D is the counter-clockwise oriented boundary curve of D.

Cauchy's integral formula provides a method to compute the Cauchy transform of a function. The details regarding its computation and the Cauchy transform will be discussed in Section 3.2.

3.2 Cauchy-Green coordinates

The Cauchy transform is a fundamental mathematical operation with wide-ranging applications in various fields, including complex analysis, potential theory, and image processing. It produces a holomorphic functions u on a domain D when given a continuous function $f(z)$ on the boundary ∂D:

$$u(z) = \frac{1}{2\pi i} \int_{\partial D} \frac{f(w)}{w - z} dw, \quad z \in D \tag{1}$$

Function u is holomorphic within domain D, and if f corresponds to the boundary values of a holomorphic function, then the Cauchy transform will reproduce f.

Cauchy coordinates [Weber et al. 2009] are a discretization of Cauchy's integral formula (1). If we discretize the boundary into a polygon with a series of vertices $\{z_j\}_{j=1}^n$, and function f at these boundary vertices is given as $f_j = f(z_j)$, and gets linearly interpolated along the boundary edges, then we can compute a complex number $C_j(z)$ associated with f_j. Consequently, the integral (1) can be expressed as the following summation:

$$u(z) = \sum_{j=1}^{n} C_j(z) f_j, \quad z \in D, \tag{2}$$

where domain D is the interior of the polygon, and z is a point inside D. $C_j(z)$ is given by:

$$C_j(z) = \frac{1}{2\pi i}\left(\frac{B_{j+1}}{A_{j+1}} \log \frac{B_{j+1}}{B_j} - \frac{B_{j-1}}{A_j} \log \frac{B_j}{B_{j-1}}\right), \quad z \in D \tag{3}$$

with $A_j = z_j - z_{j-1}$ and $B_j = z_j - z$. Weber et al. [2009] derived analytic expressions for the first and second derivatives of the coordinates and proved the following theorem:

Theorem 3.1. *Lipman's 2D Green coordinates [Lipman et al. 2008] are identical to discrete Cauchy coordinates.*

3.3 Bézier curve

The Bézier curve is a widely used parameterized curve representation in computer graphics. It is a smooth and continuous curve defined by a control polygon. Assuming the control polygon is given by a $n+1$ point sequence $[\mathbf{b_0}, \mathbf{b_1}, ..., \mathbf{b_n}]$, then the degree n (order $n+1$) Bézier curve is given by:

$$x(t) = \sum_{i=0}^{n} B_i^n(t)\mathbf{b_i}, \tag{4}$$

where $B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$ are the Bernstein basis functions. This formulation will be used in the derivation of our coordinates.

<<<<< HEAD Multiple Bézier curves can be composited into a Bézier spline which is at least C^0 continuous at the endpoints of each Bézier curve. In our notations, we use "Bézier curve" to denote a single curve, while "Bézier spline" refers to a cage formed by multiple curves, which are C^0 continuous at each end point. ======= Multiple Bézier curves can be composited into a Bézier spline which is at least C^0 continuous at the endpoints of each Bézier curve. In our notations, we use "Bézier curve" to denote a single curve, while "Bézier spline" refers to a cage formed by multiple curves, which are C^0 continuous at each end point. >>>>> fe11ab1 (.)

4 METHOD

We aim to extend Cauchy coordinates of Weber et al. [2009] to curved cages. We begin by addressing the fundamental case of mapping from a polygonal cage to a curved cage. In this process, we generalize Cauchy coordinates to derive coordinates functionally equivalent to PGC [Michel and Thiery 2023], but with novel closed-form expressions. Employing a similar methodology, we develop formulas for the derivatives of these coordinates.

Building on these barycentric coordinates and their derivatives, we introduce a method to compute the inverse barycentric mapping. This innovation enables deformations from curved cages to polygonal cages. By synthesizing these advancements, we present a comprehensive framework for mapping between curved cages. To demonstrate the practical utility of our approach, we showcase a technique for Point-to-Point (P2P) constrained mappings using our newly developed coordinates.

4.1 Formulation: Polygonal Cage to Curved Cage

Consider a simple case: suppose the source cage is a polygon formed by several line segments in counterclockwise (CCW) order, while the target cage is a curve represented by a Bézier spline. Both the source cage and the target cage have the same number of edges or segments. Our goal is to derive new barycentric coordinates by utilizing Cauchy's integral formula.

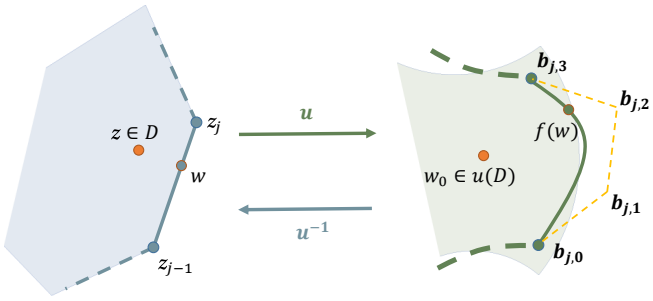

Fig. 2. Implementation of our coordinates, $f(w)$ is defined by Bézier curves on the cage. A holomorphic function u will be constructed.

We aim to compute the discretized form of Cauchy's integral formula and distribute it to each Bézier control point. Since our coordinates is derived from this formula, it is important to ensure that our input satisfies the conditions stated in Section 3.1. Specifically, we require that D is a simply connected region.

Denote the region enclosed by the polygonal cage by D, then the integral formula (1) can be transformed into the following form:

$$u(z) = \frac{1}{2\pi i} \sum_{e_j \in \partial D} \int_{e_j} \frac{f(w)}{w - z} dw, \quad z \in D, \tag{5}$$

where $f(w) = \sum_{m=0}^{n_j} B_m^{n_j}(t)\mathbf{b_{j,m}}$ represents a Bézier curve in the cage. Based on this, we can compute the coordinates on each edge e_j individually. In our formulation, the source edge e_j is defined by two points, z_j and z_{j+1}, while the target curve is defined by a Bézier control polygon $\{\mathbf{b_{j,m}}\}_{m=0}^{n_j}$, where n_j is the degree of current Bézier curve. Since the edge e_j is a line segment, we can express t as $t = \frac{w - z_{j-1}}{z_j - z_{j-1}}$. So

$$\begin{aligned} u(z) =& \frac{1}{2\pi i} \sum_{e_j \in \partial D} \int_{e_j} \frac{\sum_{m=0}^{n_j} B_m^{n_j}(t)\mathbf{b_{j,m}}}{w - z} dw \\ =& \frac{1}{2\pi i} \sum_{e_j \in \partial D} \sum_{m=0}^{n_j} \left(\int_{e_j} \frac{B_m^{n_j}\left(\frac{w - z_{j-1}}{z_j - z_{j-1}}\right)}{w - z} dw \right) \mathbf{b_{j,m}}. \end{aligned}$$

And if we write

$$\text{Integral}(z, e_j, m, n_j) = \int_{e_j} \frac{B_m^{n_j}(\frac{w-z_{j-1}}{z_j-z_{j-1}})}{w-z} dw,$$

then the coordinates associated with point $\mathbf{b_{j,m}}$ can be expressed as

$$C_{j,m}(z) = \frac{1}{2\pi i}\,\text{Integral}(z, e_j, m, n_j). \tag{6}$$

$$u(z) = \sum_{e_j \in \partial D} \sum_{m=0}^{n_j} C_{j,m}(z)\mathbf{b_{j,m}} \tag{7}$$

Let $A_j = z_j - z_{j-1}$ and $B_j = z_j - z$. By utilizing the binomial expansion theorem, we can derive a closed-form expression for the aforementioned integral. The detailed derivation can be found in the supplemental material, and the derived result is as follows:

$$\begin{aligned}&\text{Integral}(z, e_j, m, N)\\ =&\frac{\binom{N}{m}}{(A_j)^N}\Big[\sum_{\substack{k=0\\ N-m-l+k\neq 0}}^{m}\sum_{l=0}^{N-m}\frac{\binom{m}{k}\binom{N-m}{l}(-1)^{N-k-l}}{N-m-l+k}\Big((B_j)^{N-m+k}(B_{j-1})^{m-k}\\ &-(B_j)^l(B_{j-1})^{N-l}\Big)+(-B_{j-1})^m(B_j)^{N-m}\log\frac{B_j}{B_{j-1}}\Big].\end{aligned} \tag{8}$$

Here, we use N to simplify the notation for n_j, as our derivation is focused on a single segment. The subsequent formulas also employ the same simplified notation. Thus, we have obtained the closed-form expression for our coordinates. We evaluate the expression (6), denoted as $C_{j,m}(z)$, for all edges e_j and assign the resulting values to their corresponding Bézier control point $\mathbf{b_{j,m}}$.

Note that the endpoint of the current Bézier curve coincides with the starting point of the next curve ($\mathbf{b_{j,n_j}} = \mathbf{b_{j+1,0}}$), we need to compute their barycentric coordinates separately ($C_{j,n_j}(z)$ and $C_{j+1,0}(z)$). Depending on the data structure used in the implementation, it may be required to sum up the two obtained barycentric coordinates. In our implementation, we treat these two points as the same point because we require the target Bézier control polygons to enclose a closed region.

We represent the coordinates in matrix form, with $C_{j,m}(z)$ being row vectors and $\mathbf{b_{j,m}}$ being column vectors, and $u(z)$ can be obtained through matrix multiplication.

$$\begin{aligned}\mathbf{C} &= [C_{1,0}(z), C_{1,1}(z), ..., C_{1,n_1-1}(z), C_{2,0}(z), ...]\\ \mathbf{b} &= [\mathbf{b_{1,0}}, \mathbf{b_{1,1}}, ..., \mathbf{b_{1,n_1-1}}, \mathbf{b_{2,0}}, ...]^T\\ u(z) &= \mathbf{C}\cdot\mathbf{b}\end{aligned} \tag{9}$$

4.2 Derivatives of our coordinates

The derivatives of the classic Cauchy barycentric coordinates are obtained by directly differentiating $C_j(z)$, as it is relatively simple. However, we employ a novel approach to compute our derivatives.

Cauchy's integral formula holds for any simply connected closed region under certain conditions, which align with our assumptions. According to Cauchy's integral formula, the derivatives of the barycentric coordinates of any order can be expressed using an integral formula. This allows us to derive the derivatives of our barycentric coordinates following the same approach.

The nth-order derivative of our coordinates can be calculated by:

$$u^{(n)}(z) = \frac{n!}{2\pi i}\int_{\partial D}\frac{f(w)}{(w-z)^{n+1}}dw \tag{10}$$

Since it is a polynomial integration, the derivation is similar to that of our coordinates. Coefficients $C_{j,m}^{(n)}(z)$ for $\mathbf{b_{j,m}}$ are given by:

$$C_{j,m}^{(n)}(z) = \frac{n!}{2\pi i}\int_{e_j}\frac{B_m^N(\frac{w-z_{j-1}}{z_j-z_{j-1}})}{(w-z)^{n+1}}dw$$

The derivation is straightforward. The detailed derivation and closed-form expressions of the nth-order derivatives can be found in the supplementary material. We provide Alg. 1 for calculating the nth-order derivative. This pseudocode can also handle the case of $n = 0$, i.e. the expression of our coordinates. In fact, the derivatives of any order can be computed within the same function.

Algorithm 1: Calculate n-th Derivative of Our Coordinates

Input: Interior point z, edge e_j, index of Bézier control points m, degree of current Bézier curve N, derivative order n
Output: result as $C_{j,m}^{(n)}(z)$

```
result = 0;
for k = 0:m do
    for l = 0:N-m do
        if N-m-l+k == n then
            result += C(m,k) C(N-m,l) (-1)^{N-k-l} (B_j)^l (B_{j-1})^{m-k} log(B_j/B_{j-1});
        else
            result += C(m,k) C(N-m,l) (-1)^{N-k-l} / (N-m-l+k-n) ((B_j)^{N-m+k-n} (B_{j-1})^{m-k} -
            (B_j)^l (B_{j-1})^{N-l-n});
        end
    end
    result ×= n! C(N,m) / (A_j)^N / (2πi) ;
end
```

Expression (8) contains two nested summation operations, each denoted by the symbol Σ. Typically, the curve order is not high, so the direct evaluation of the sums does not incur significant overhead. However, in some applications with large N, the growth in computational complexity could be unpredictable. We provide a method to speedup the computation. By isolating variables k and l and precomputing the following two matrices $T1$ and $T2$ in $O(N^2)$ time, we can speedup the computation process, ensuring that the average runtime of our coordinates for each control point does not exceed $O(N)$. The details can be found in the supplementary material.

$$T1[N-m,k] = \sum_{l=0}^{N-m}\frac{\binom{N-m}{l}(-1)^{-l}}{N-m-l+k} \qquad k = 0, 1, ..., m \tag{11}$$

$$T2[m,N-l] = \sum_{k=0}^{m}\frac{\binom{m}{k}(-1)^{-k}}{N-m-l+k} \qquad l = 0, 1, ..., N-m. \tag{12}$$

With this optimization, the expression can be obtained with a pre-computation step of $O(N^2)$ and a per-point coordinates calculation of $O(N)$, significantly reducing the complexity from the original $O(N^3)$ for each point and $O(N^4)$ for a Bézier curve.

4.3 Inverse mapping

We have previously derived barycentric coordinates for deformation using a curved cage, specifically for cases where the input cage is polygonal. We now extend this work to enable the inverse mapping $u^{-1}(z)$ from a curved cage to a polygonal cage, given the expression for $u(z)$. Employing Cauchy's integral formula with $u(D)$ satisfying the conditions outlined in Section 3.1 and w_0 located inside $u(D)$ (Fig. 2), we perform a change of variables $w = f(z)$ to derive:

$$u^{-1}(w_0) = \frac{1}{2\pi i}\int_{\partial u(D)} \frac{u^{-1}(w)}{w - w_0} dw = \frac{1}{2\pi i}\int_{\partial D} \frac{z\, u'(z)}{u(z) - w_0} dz.$$

This inverse can be numerically integrated if $u(z)$ and $u'(z)$ are known on the boundary, requiring an additional derivation of our polynomial Cauchy coordinates at the boundary. To determine $u(z)$ on the boundary, we take the limit of our coordinates expression, similar to the approach used for Cauchy coordinate derivatives [Segall and Ben-Chen 2016, Appendix A]. Special care is needed for endpoints where the logarithmic function may become infinite. We address this by specifically computing $C_{j,N}(z_j)$ and $C_{j+1,0}(z_j)$, allowing infinite terms to cancel out when summed. The resulting unified expressions are:

$$C_{j,m}(z_{j-1}) = \binom{N}{m}(-1)^{N-m} T1[N-m, m] \tag{13}$$

$$C_{j,m}(z_j) = \left(\log\left(\frac{B_{j+1}}{B_{j-1}}\right) + 2\pi i\right) I_{\{m=N\}} - \binom{N}{m} T2[m, N]. \tag{14}$$

Note that the complex logarithm is a multi-valued function, and we take the principal branch. The $2\pi i$ term in (14) is due to the following limit

$$\lim_{\substack{z\in e_j \\ z\to z_j}} \log\frac{B_{j+1}}{B_j} = \log\left|\frac{B_{j+1}}{B_j}\right| + \pi i = \log\left(-\frac{B_{j+1}}{B_j}\right) + \pi i.$$

While $C(z)$ can be obtained through limits on the boundary and at vertices, the limit of $D(z)$ at vertices does not exist. In practice, we approximate the limit of $D(z)$ at the vertices by evaluating $D(z)$ for points inside the region close to the vertices.

For numerical integration, we sample the cage boundary and evaluate $u(z)$ and $u'(z)$ at midpoint of line segments. This approach ensures computation accuracy and circumvents the non-existence of the limit of $D(z)$ at vertices. Our numerical results validate the efficacy of this sampling method.

4.4 Formulation: Between Curved Cages

We've developed a method for curved-cage deformation using our proposed coordinates and inverse maps. Inspired by Cubic MVC [Li et al. 2013], our approach involves an intermediate step and addresses two typical scenarios:

Case 1. Known deformation results and corresponding cages: We restore the original shape using the inverse map (section 4.3), then apply deformation as described in section 4.1.

Case 2. One shape with user-specified source and intermediate cages: The input may not meet conditions for inversion initially, as our inverse is valid only for the interior of $u_1(D)$, which may differ from the region enclosed by the Bézier spline, since our coordinates are not interpolating. The user can interactively adjust the Bézier control points of the input cage so that the containing condition is met. Alg. 2 outlines this simple process.

During the inversion process, we sample interior points of $u_1(D)$. It is less straightforward to sample this curved region, compared to polygonal domains. We can either sample within the polygonal region D and apply u_1, or sample the entire plane, compute the inverse, and filter out points mapped near 0 (those outside $u_1(D)$).

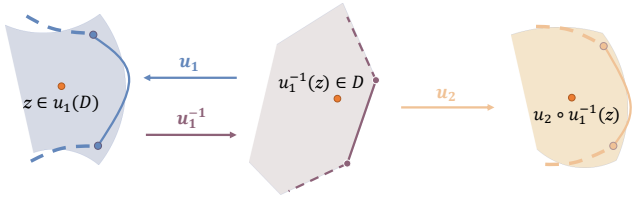

Fig. 3. Implementation of our algorithm between curved cages. u_1 will be firstly calculated, and then follows u_1^{-1} and u_2.

Algorithm 2: Find the Mapping Between Curved Cages

Input: Source Shape, Source curved cage, Intermediate polygonal cage, Target curved cage
Output: Deformed Shape
1. Evaluate u_1 at samples on the intermediate cage;
2. Check whether the entire shape is contained inside $u_1(D)$ and otherwise perform user interaction;
3. Sample the interior of $u_1(D)$ for the mapping;
4. Evaluate u_1^{-1} for sampled points;
5. Evaluate u_2 for final result;

4.5 Point-to-Point Constrained Map

With our coordinates and its derivatives, we can also design energy functions and formulate an optimization problem for deformation. Using the matrix form (9) of our coordinates, the P2P energy can be written as follows:

$$E_{\text{P2P}}(\hat{\mathbf{b}}) = ||\mathbf{C}_{\text{P2P}}\,\hat{\mathbf{b}} - z_{\text{dst}}||^2,$$

where $\hat{\mathbf{b}}$ is the free variable, which represents the column vector of the Bézier control points of the cage, $\mathbf{C}_{\text{P2P}}$ is our coordinates for $\{z_{\text{src}}\}$, indicating that $\{z_{\text{src}}\}$ should be mapped to $\{z_{\text{dst}}\}$. In addition, to promote smoothness of the mapping, we define a smoothness energy. Utilizing the previously given second derivatives of our coordinates, the smoothness energy is defined as:

$$E_{\text{Smooth}}(\hat{\mathbf{b}}) = ||\mathbf{C}^{(2)}\,\hat{\mathbf{b}}||^2.$$

Combining these two energies, we obtain the following mapping energy for optimization,

$$E_{\text{Def}}(\hat{\mathbf{b}}) = E_{\text{P2P}}(\hat{\mathbf{b}}) + \lambda\, E_{\text{Smooth}}(\hat{\mathbf{b}}),$$

where λ is a user specified weight.

We note that the optimization variables are $\hat{\mathbf{b}}$, the Bézier control points of the cage. Since the energy function is in the form of sum of squares, the optimization problem can be easily solved using a linear solver.

5 RESULTS AND COMPARISONS

Fig. 4 showcases the deformation from a polygonal cage to a curved cage using our coordinates, juxtaposed with results from Cauchy coordinates and PGC. This comparison highlights our method's conformal, smooth, and controllable nature, attributed to the Bézier representation of the cage. Notably, our results are visually indistinguishable from PGC outcomes. Fig. 5 illustrates the inverse mapping computed by our method. When the mapped boundary is free of self-intersections, we successfully reconstruct the original shape from its deformed state. It's worth noting that the quality of this inversion is influenced by the density of boundary sampling. Fig. 6 presents deformation results between curved cages, including the intermediate mapping stage. Fig. 7 and Fig. 8 further demonstrate the breadth of our method's applications, including Point-to-Point (P2P) deformation.

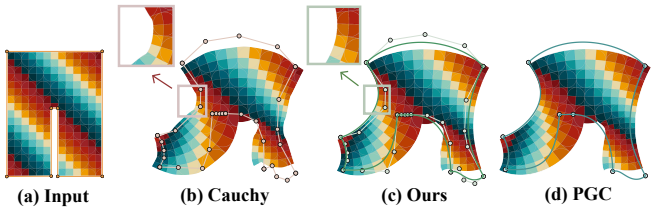

Fig. 4. When using a polygonal cage as input (a), our method and Cauchy coordinates both employ a linear combination of the control vertices. However, our approach (c) achieves smoother boundary results compared to Cauchy coordinates (b). Compared to PGC (d), our method (c) produces identical result, with differences only at the level of numerical precision (1e-8 under the L^∞ norm).

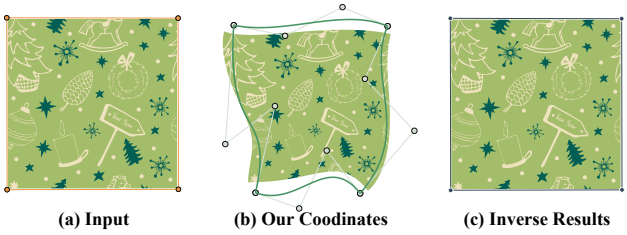

Fig. 5. Inverse map. We can see that the computed inverse map (c) is the same as the original shape (a). However, due to the nature of numerical integration, the obtained results are not accurate and exhibits numerical error. Table. 1 shows how boundary samples affects average numerical errors (L_1 norm), convergence order and running time for this example.

To ensure the practical applicability and reproducibility of our proposed method, several key implementation considerations must be taken into account.

Table 1. Statistics for the Inverse Mapping in Fig. 5

#Boundary Samples	Error	Order	Runtime(s)
3200	3.1914e-03	-	0.202
32000	3.2069e-04	0.9979	0.801
320000	3.2091e-05	0.9997	12.89
3200000	3.1971e-06	1.0016	150.20

- Inverse Mapping Accuracy: The fidelity of inverse mapping is contingent on boundary sampling density. Insufficient sampling can lead to noticeable deviations from the ground truth, with sparse sampling potentially causing significant numerical errors, especially at control polygon endpoints. This issue can be mitigated by increasing sampling density. Our implementation employs mid-point integration for boundary samples, achieving a convergence order of 1, as evidenced in Table 1. This convergence pattern remains consistent across our diverse set of examples, demonstrating the robustness of our approach.
- Content Preservation in Curved Cage Deformations: During curved cage deformations, the inverse map $u_1(D)$ may not encompass the entire shape, potentially resulting in content loss during inversion. This discrepancy arises from the non-interpolating nature of our proposed coordinates, unlike interpolating methods such as Cubic MVC. The difference between the actual inverse region $u_1(D)$ and the area enclosed by the source Bézier spline can be addressed through user interaction.
- Minimum Curve Requirement: The input cage should comprise more than two curves to avoid degeneracy in the intermediate polygonal cage. In practice, shape-aware cages typically satisfy this requirement naturally. For inputs with only one or two curves, this issue can be resolved by subdividing the Bézier curves.

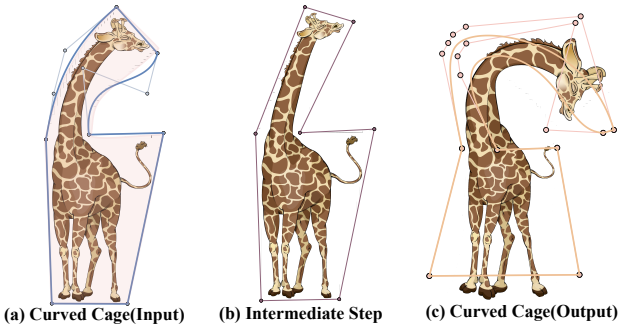

Fig. 6. Input with a curved cage. In this case, the user specifies the cages of both the intermediate map and the target.

Comparison with Cauchy Coordinates. Cauchy coordinates necessitate polygonal representations for both input and output cages, and require numerous segments to approximate curved boundaries. In contrast, our method achieves comparable results using only

a few Bézier curves. This not only simplifies the representation but also leads to more shape-preserving outcomes, mirroring the advantages PGC holds over GC. Our approach thus offers a more efficient and accurate alternative for handling curved boundaries in deformation tasks. ««««< HEAD When the Bézier curve degenerates to a line segment ($N = 1$), our barycentric coordinates and their derivatives are equivalent to Cauchy coordinates, ensuring consistency with existing methods for linear cages. This property allows our approach to seamlessly handle both curved and linear cage elements, providing a unified framework for deformation tasks across varying domain complexities. ======= When the Bézier curve degenerates to a line segment (N=1), our barycentric coordinates and their derivatives are equivalent to Cauchy coordinates, ensuring consistency with existing methods for linear cages. This property allows our approach to seamlessly handle both curved and linear cage elements, providing a unified framework for deformation tasks across varying domain complexities. »»»»> fe11ab1 (.)

Comparison with PGC. Weber et al. [2009] established the equivalence of Cauchy coordinates and Green coordinates, based on the relationship between Cauchy's integral formula and Green's third identity [Ahlfors 1979]. Our experimental results suggest this equivalence extends to polynomial boundaries, with our coordinates showing equivalence to PGC. Despite minor differences in boundary curve representation, the interconvertibility of Bézier and polynomial curves enables direct comparisons. While a formal proof of equivalence remains challenging, we believe Theorem 3 from Weber et al. [2009] could potentially be adapted, albeit with more complex derivations.

Our approach offers several key advantages over PGC.

- User-friendly interaction: The Bézier spline representation for the boundaries allows for more user intuitive and interactive curve editing.
- Analytical derivatives: We provide closed-form expressions for nth-order derivatives of the coordinates, enabling efficient implementation of variational optimization techniques such as point-to-point deformation.
- Curved cage support: Our method extends to accommodate curved cages as input, significantly enhancing its versatility and expanding its potential applications.

6 CONCLUSION

We have generalized the Cauchy barycentric coordinates for Bézier spline cages, enabling seamless deformation from polygonal to curved cages via Bézier control polygons. Our derived closed-form derivatives of the coordinates facilitate interactive point-to-point deformation by minimizing mapping smoothness energy. We also introduce a method for computing the inverse mapping, which, when combined with our polynomial coordinates, enables curved-to-curved cage deformations. This advancement opens up a wide range of potential applications in computer graphics and geometric modeling.

Limitations and Future work. While our approach for curved-to-curved cage deformations relies on an intermediate map, future research will focus on more automated generation of this mapping. Our current inverse mapping technique doesn't prevent self-intersection of boundary curves in $u(z)$, potentially leading to unexpected results. Although this can be mitigated through user interaction, we aim to develop an automated solution. The iterative closest conformal map technique proposed by Segall and Ben-Chen [2016] presents a promising direction for addressing this challenge.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable comments. This work was supported by National Natural Science Foundation of China (62072422).

REFERENCES

L.V. Ahlfors. 1979. *Complex Analysis: An Introduction to the Theory of Analytic Functions of One Complex Variable.* McGraw-Hill.

Alexander Belyaev. 2006. On Transfinite Barycentric Coordinates. In *Symposium on Geometry Processing*, Alla Sheffer and Konrad Polthier (Eds.). The Eurographics Association. https://doi.org//10.2312/SGP/SGP06/089-099

Alexander G Belyaev and Pierre-Alain Fayolle. 2017. Transfinite barycentric coordinates. In *Generalized barycentric coordinates in computer graphics and computational mechanics*. CRC Press, 43–62.

Renjie Chen and Ofir Weber. 2017. GPU-accelerated locally injective shape deformation. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 1–13.

Edward Chien, Renjie Chen, and Ofir Weber. 2016. Bounded distortion harmonic shape interpolation. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–15.

Ana Dodik, Oded Stein, Vincent Sitzmann, and Justin Solomon. 2023. Variational barycentric coordinates. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–16.

Christopher Dyken and Michael S Floater. 2009. Transfinite mean value interpolation. *Computer Aided Geometric Design* 26, 1 (2009), 117–134.

Zeev Farbman, Gil Hoffer, Yaron Lipman, Daniel Cohen-Or, and Dani Lischinski. 2009. Coordinates for instant image cloning. *ACM Transactions on graphics (TOG)* 28, 3 (2009), 1–9.

Michael S Floater. 2003. Mean value coordinates. *Computer aided geometric design* 20, 1 (2003), 19–27.

Kai Hormann and Michael S Floater. 2006. Mean value coordinates for arbitrary planar polygons. *ACM Transactions on Graphics (TOG)* 25, 4 (2006), 1424–1441.

Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. 2007. Harmonic coordinates for character articulation. *ACM transactions on graphics (TOG)* 26, 3 (2007), 71–es.

Tao Ju, Scott Schaefer, and Joe Warren. 2005. Mean value coordinates for closed triangular meshes. *ACM Transactions on Graphics* 24, 3 (2005), 561–566.

Xian-Ying Li, Tao Ju, and Shi-Min Hu. 2013. Cubic mean value coordinates. *ACM Trans. Graph.* 32, 4 (2013), 126–1.

Yaron Lipman, Johannes Kopf, Daniel Cohen-Or, and David Levin. 2007. GPU-assisted positive mean value coordinates for mesh deformations. In *Proceedings of the fifth Eurographics symposium on Geometry processing*. 117–123.

Yaron Lipman, David Levin, and Daniel Cohen-Or. 2008. Green coordinates. *ACM transactions on graphics (TOG)* 27, 3 (2008), 1–10.

Élie Michel and Jean-Marc Thiery. 2023. Polynomial 2D Green Coordinates for Polygonal Cages. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–9.

Aviv Segall and Mirela Ben-Chen. 2016. Iterative closest conformal maps between planar domains. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 33–40.

Dongbo Shi and Renjie Chen. 2022. Harmonic Shape Interpolation on Multiply-connected Planar Domains. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, 1–11.

Ofir Weber, Mirela Ben-Chen, Craig Gotsman, et al. 2009. Complex barycentric coordinates with applications to planar shape deformation. In *Computer Graphics Forum*, Vol. 28. Citeseer, 587.

Ofir Weber, Mirela Ben-Chen, Craig Gotsman, and Kai Hormann. 2011. A complex view of barycentric mappings. In *Computer Graphics Forum*, Vol. 30. Wiley Online Library, 1533–1542.

Ofir Weber, Roi Poranne, and Craig Gotsman. 2012. Biharmonic coordinates. In *Computer Graphics Forum*, Vol. 31. Wiley Online Library, 2409–2422.

Juyong Zhang, Bailin Deng, Zishun Liu, Giuseppe Patanè, Sofien Bouaziz, Kai Hormann, and Ligang Liu. 2014. Local barycentric coordinates. *ACM Transactions on Graphics (TOG)* 33, 6 (2014), 1–12.

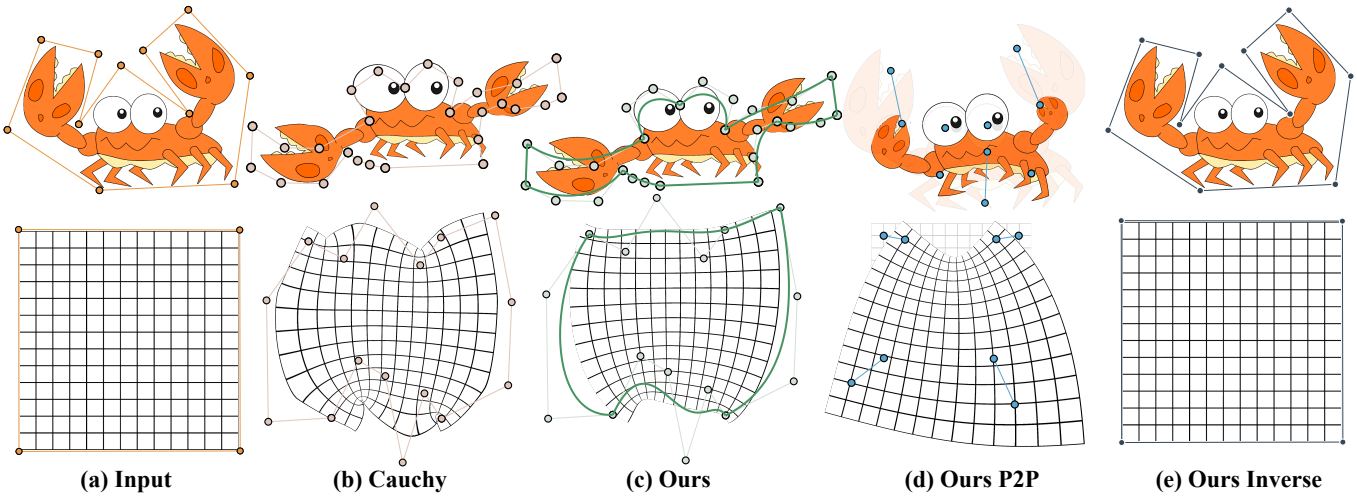

Fig. 7. Comparison of our deformation results with Cauchy coordinates [Weber et al. 2009]. In these examples, the target cage has segments with varying degrees, which can be observed from the number of Bézier control points in our results (c). The results of our P2P deformation are shown in (d), note that in the grid example, the source points we selected are all grid points. The results of using our method to compute the inverse mapping and restore the original shape are presented in (e).

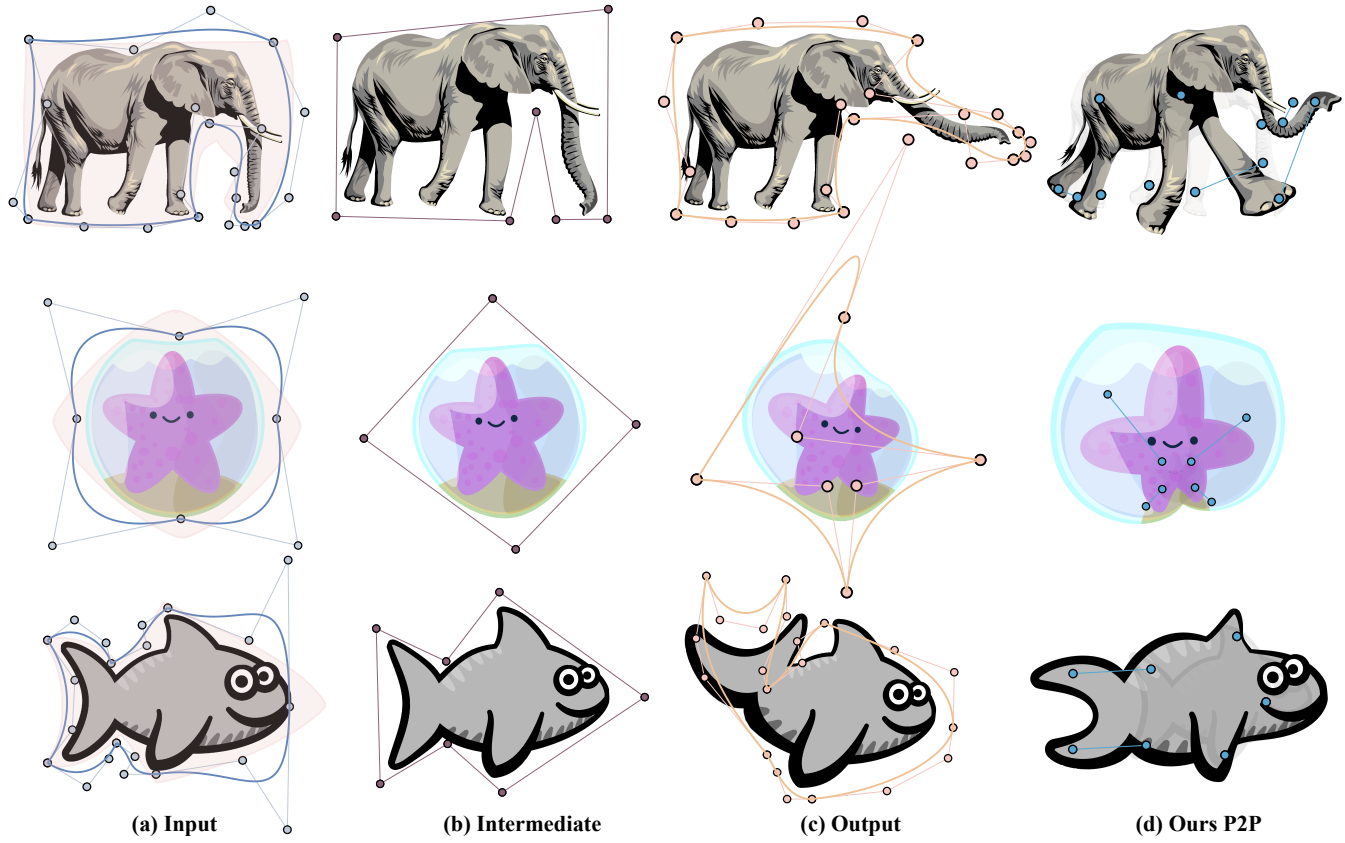

Fig. 8. Deformation results of our method, with curved cages as input. (a) the input, the interior of $u_1(D)$ is marked with a light red mask, (b) the intermediate step, (c) the deformed shape, (d) P2P deformation results of (b), with the original image displayed in the background to indicate the P2P constraints.