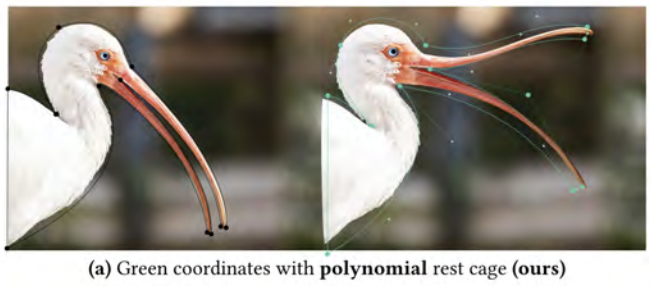
Variational Green and Biharmonic Coordinates for 2D Polynomial Cages

ÉLIE MICHEL, Adobe Research, France
ALEC JACOBSON, University of Toronto, Canada and Adobe Research, Canada
SIDDHARTHA CHAUDHURI, Adobe Research, United States of America
JEAN-MARC THIERY, Adobe Research, France


(a) Green coordinates with **polynomial** rest cage **(ours)**

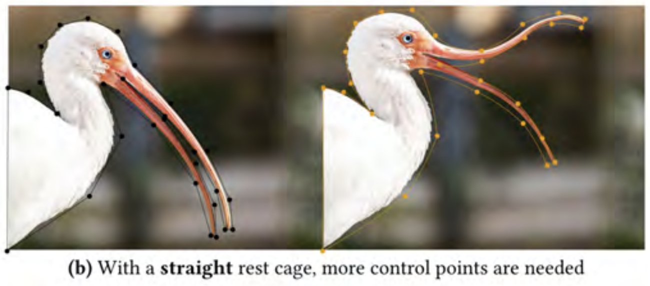

(b) With a **straight** rest cage, more control points are needed

Fig. 1. We enable the use of rest cages made of polynomial curves for Green cage-based deformation (a). Previous methods were limited to straight rest cages, making it cumbersome to fit some rest shapes like the bird's beak (b). Furthermore, we provide the derivatives of our coordinates so they can be used in variational solvers. Note that Green coordinates foster locally-conformal regularity over strict interpolation of the cage transform.

We present closed-form expressions for Green and biharmonic coordinates with respect to polynomial curved 2D cages, enabling reliable cage-based image deformation both to and from a curved cage. We further provide closed-form expressions for first- and second-order derivatives of these coordinates with respect to the encoded position. This enables the use of variational solvers for interacting with the 2D shape at arbitrary points while keeping the fast decoding strength of cage-based deformation, which we illustrate for a variety of elastic deformation energies.

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

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

ACM Reference Format:
Élie Michel, Alec Jacobson, Siddhartha Chaudhuri, and Jean-Marc Thiery. 2025. Variational Green and Biharmonic Coordinates for 2D Polynomial Cages. *ACM Trans. Graph.* 44, 4, Article 76 (August 2025), 20 pages. https://doi.org/10.1145/3731421

1 Introduction

Generalized barycentric coordinates can be used to define deformation of points within a polygon of an image as the weighted sum of deformed polygon corners. For non-triangular polygons, the menu of generalized barycentric coordinate flavors is seemingly endless. Different varieties spanning trade-offs of smoothness, non-negativity, locality, closed-formness. Generalized barycentric coordinates have been deployed as cage-deformers in popular software such as Krita and Blender. However, the vast majority of research in generalized barycentric coordinates has limited its consideration to rest polygons with *straight edges* mapped to similarly straight-edge deformed polygons. Unfortunately, at deformation time this means that regions near the cage edges deform linearly or otherwise reveal the cage discretization: At binding time, a straight cage edges must weave their way around the region of interest, often requiring many segments to closely follow curved boundaries.

There has been some continual progress on methods which consider curved *deformed* cages: Langer and Seidel [2008] convert any coordinates into linear-blend skinning like weights which afford derivative control at cage-vertices (i.e., allowing cage edges to bend). The resulting weights are not *coordinates* in the usual sense; neither are, e.g., bounded biharmonic weights [Jacobson et al. 2011], which similarly define skinning weights for cage-deformers with tangent control at vertices. Beatson et al. [2018] extend Mean Value Coordinates [Floater 2003] to control tangents *along* (straight) cage edges. Manson and Schaefer [2010], [Smith and Schaefer 2015], and Beatson et al. [2018] define or extend other coordinates to similarly control tangents *along* deformed cage edges. Most recently, Michel and Thiery [2023] extend Green Coordinates [Lipman et al. 2008] to cages with polynomial (e.g., Bézier) cage-edges, *but* their derivations assume that the rest cage edges are straight. Their seminal work has since then inspired several works on the use of polynomial 2D cages for shape manipulation. Lin and Chen [2024] have demonstrated that the coordinates can be obtained from the Cauchy formula, and have obtained closed-form expressions for their gradients and Hessians, thus allowing for variational 2D shape deformation using polynomial cages (that are still constrained to be made of segments at binding time). They allow considering curved cages as input to

Authors' Contact Information: Élie Michel, Adobe Research, France, emichel@adobe.com; Alec Jacobson, University of Toronto, Canada and Adobe Research, Canada, alecjacobson@adobe.com; Siddhartha Chaudhuri, Adobe Research, United States of America, sidch@adobe.com; Jean-Marc Thiery, Adobe Research, France, jthiery@adobe.com.

ACM 1557-7368/2025/8-ART76
https://doi.org/10.1145/3731421

some extent, but require a straightening step (similar in spirit to the one of [Li et al. 2013]) to conform the cage to a state where their formulas can be used – which is not always feasible, as disclaimed and analyzed in their paper. In an unpublished preprint, Liu et al. [2024] have extended the formalism of Michel and Thiery by providing closed-form expressions for Green coordinates for input polynomial 2D rest cages directly, based on residual computations, making both the derivation and the evaluation of the formulas non-trivial (e.g., requiring special cases for different root multiplicities).

In this paper, we extend those works and "complete the picture" on this topic. We make the following contributions:

(1) We introduce Green and biharmonic coordinates for input/deformed polynomial cages,
(2) we derive their gradients and Hessians, and
(3) we present several variational methods using our various polynomial subspaces (Green, harmonic, biharmonic).

2 Related work

While there are many different ways to deform 2D/3D shapes (e.g., free-form [Sederberg and Parry 1986], skeleton-based [Jacobson et al. 2011], brush-based [De Goes and James 2017; Manson and Schaefer 2010]), we focus here on cage-based 2D deformation methods, and refer the interested reader to [Ströter et al. 2024] that surveys cage-based 3D ones and [Jacobson et al. 2014] that compares cage-based approaches to other popular methods (e.g., skeleton-based). In these methods, each rest position is encoded once into cage coordinates that describe it w.r.t. the rest cage, and may then be efficiently decoded back from a deformed cage. Cage-based methods can be classified in many ways (e.g., either *interpolating* or *approximating*); here we expose the most related literature by classifying those methods as the ones compatible with *straight polygonal cages* (the majority) and the ones compatible with *curved cages*.

Traditional polygonal cages. Cage-based deformations can be seen as an extension of Free-form deformations, that define explicitly the volumetric region to deform using lattices [Sederberg and Parry 1986]. Cage-based methods have initially raised interest in the scientific community due to their ability to define a deformation field everywhere inside a volume solely from its boundary. Mean-Value coordinates were introduced in several works [Floater 2003; Hormann and Floater 2006; Ju et al. 2005] to present a solution to the *boundary value problem.* While they allow defining generalized barycentric coordinates in closed-form for arbitrary points, inside *or outside* the domain, they result as a direct consequence in coordinates that may be negative, possibly resulting in non-natural deformation behavior (e.g., moving parts of the cage to the left results in motion of parts of the shape to the right). To remedy this issue, Harmonic coordinates [Joshi et al. 2007] and Positive Mean-Value coordinates [Lipman et al. 2007] were introduced. Those two sets of coordinates require resorting to approximate solvers for their computation, and do not allow for computing coordinates outside the cage, but ensure positivity of the coordinates. While all previously-cited methods lead to interpolation on the cage boundary, approximating coordinates were rendered popular by Lipman et al., who introduced Green coordinates [Lipman et al. 2008] allowing for angle-preserving 2D deformations under closed-form. As conformal deformations are transformations whose Jacobian J is the product of a rotation matrix and a uniform *positive* scale ($J = sR, s \in \mathbb{R}^+, R \in SO_3$), and the method of Lipman et al. can produce local degenerate transformations (i.e., s might become null or negative), we term their transformations angle-preserving instead of conformal. Those coordinates were used for variational deformations [Ben-Chen et al. 2009] that relied on their *shape-awareness* property, allowing for well-behaved deformations even in situations where separate cage limbs are close-by in the encoding rest state. Weber et al. used those coordinates as 2D deformation subspace and bounded the stretch through constrained boundary analysis [Weber and Gotsman 2010]), effectively guaranteeing conformality of the deformations. This work has inspired following works on bounded-distortion 2D mappings, such as [Chen and Weber 2015, 2017]. To go beyond what harmonic and conformal deformations can provide to artists in terms of flexibility and expressivity, Weber and colleagues introduced biharmonic cage-based deformations in 2D [Weber et al. 2012] and demonstrated the superiority of biharmonic functions over harmonic ones, in particular in variational setups. This work was extended for 3D deformations more than ten years later by Thiery and colleagues [2024], who presented analytical formulas for the coordinates and their derivatives.

Curved cages. While some of the previously-cited methods allow for highly-smooth deformations (e.g., harmonic, biharmonic, Green: angle-preserving, conformal), the polygonal nature of the cage remained a limitation in some deformation scenarios, as the continuous-only nature of the cage itself may influence the local smoothness of the function near cage vertices. Cage-based approaches have been extended to allow for *curved elements* for this reason, and for the additional design flexibilities they offer. In 2D, Li and colleagues have presented Cubic Mean-Value coordinates [Li et al. 2013], which allowed artists to use 2D cages made of cubic arcs with a Mean-Value interpolant. They allowed *to some extent* considering curved cages as input, the only requirement being that they can be put in a straightened state where encoding can be defined (and that the inversion function allows fitting the input image to deform in this state in a compatible manner). In 3D, cage-based deformations using non-triangular facets have been proposed, either for Mean-Value coordinates [Langer et al. 2006; Thiery et al. 2018] or Green coordinates [Thiery and Boubekeur 2022]. Regarding 2D deformations, Michel and Thiery have introduced Green coordinates for input polygonal cages allowing transforming into arbitrary polynomial ones [Michel and Thiery 2023]. A major limitation of this work is that it did not allow considering *input rest* cages made of polynomial curves. This work was extended in several ways since then. As already stressed, Lin and Chen have shown that those coordinates can be equivalently computed using complex coordinates [2024]. They have also introduced formulas for their derivatives, allowing for variational deformations, and they have proposed an inversion mechanism allowing to use curved polynomial cages through a straightening step akin to the one of Li and colleagues [2013]. As the straightening step is still required, not all input polynomial cages can be considered, as disclaimed in their paper. Recently, Liu et al. [2024] have shown in an unpublished preprint that the computations suggested by Michel and Thiery can be extended to allow for arbitrary (non-intersecting) input polynomial cages. As disclaimed earlier, our work follows in these footsteps, and extends[Michel and Thiery 2023] by offering: **i)** using curved polynomial cages as input, **ii)** closed-form expressions for Green coordinates and for biharmonic coordinates in this setup, and **iii)** closed-form expressions for the gradients and Hessians of both sets of coordinates, allowing for variational shape deformation.

3 Background

We begin with a minimal review of 2D Green coordinates [Lipman et al. 2008], necessary boundary integrals used by [Michel and Thiery 2023], and 2D biharmonic coordinates [Weber et al. 2012].

3.1 Green coordinates

Any harmonic function f can be defined in a bounded 2D domain Ω from its boundary conditions thanks to Green's third identity as

$$f(\eta) = \underbrace{\int_{\xi\in\partial\Omega} f(\xi)\frac{\partial G}{\partial n_\xi}(\xi,\eta)d\xi}_{:=f_{\mathcal{D}}(\eta)} + \underbrace{\int_{\xi\in\partial\Omega} -G(\xi,\eta)\frac{\partial f}{\partial n_\xi}(\xi)d\xi}_{:=f_{\mathcal{N}}(\eta)}, \tag{1}$$

with $G(\xi,\eta) := \frac{1}{2\pi}\log(\|\xi-\eta\|)$ solution to $\triangle_1 G(\xi,\eta) = \triangle_2 G(\xi,\eta) = \delta(\|\xi-\eta\|)$ and n_ξ the unit normal of the cage at point ξ.

The term $f_{\mathcal{D}}(\eta)$ corresponds to the contribution given by the diffusion of the *Dirichlet* boundary condition, i.e., the constraints on the value of f on the boundary $\partial\Omega$ of the integration domain. The term $f_{\mathcal{N}}(\eta)$ corresponds to the diffusion of *Neumann* boundary condition, which imposes the normal derivative of f on $\partial\Omega$.

We denote in the following the boundary $\partial\Omega$ as the *cage* made of a non-intersecting set of curves c_j (oriented Counter-Clock-Wise by convention) connected at vertices $v_i \in \mathcal{V}$. We note deformed quantities with a bar $(\bar{\cdot})$ and rest-pose ones without it. For $\delta = (x, y)$, we note $\delta^\perp := (y, -x)$ the clock-wise rotation of δ by $\pi/2$.

3.1.1 Polygonal cages. Lipman et al. have introduced Green coordinates for polygonal cages with vertices V and edges E by considering affine Dirichlet and constant Neumann conditions on each edge e_j, resulting in harmonic coordinates $\{\phi_i(\eta), \psi_j(\eta)\}$:

$$\phi_i(\eta) = \int_{\xi\in\partial\Omega} \Gamma^i(\xi)\frac{\partial G}{\partial n_\xi}(\xi,\eta)d\xi \tag{2}$$

$$\psi_j(\eta) = \int_{\xi\in e_j} -G(\xi,\eta)d\xi\ , \tag{3}$$

$\Gamma^i(\xi)$ denoting the "hat" basis function of vertex i, that takes value δ_i^j on vertex v_j and is affine on each edge of the polygonal cage.

They have demonstrated that their deformations are angle-preserving in 2D iff the following Neumann constraint is set on each edge (linking the Neumann and Dirichlet conditions):

$$\frac{\partial f}{\partial n_\xi}(\xi) = \sigma_j \bar{n}_j\ \forall\xi\in e_j \tag{4}$$

$$\sigma_j = \frac{\|\bar{e}_j\|}{\|e_j\|}\ , \tag{5}$$

σ_j capturing the stretch of edge j and $\bar{n}_j$ denoting the unit normal of $\bar{e}_j$. This leads to the angle-preserving deformation function:

$$f(\eta) = \sum_{i\in V}\phi_i(\eta)\bar{v}_i + \sum_{j\in E}\sigma_j\psi_j(\eta)\bar{n}_j. \tag{6}$$

Equivalent formulation from Cauchy integral. It is worth noting that the exact same deformations can be obtained by mapping the real plane $\mathbb{R}^2$ to the complex plane $\mathbb{C}$, and using the Cauchy integral:

$$f(z) = \frac{1}{2\pi i}\int_{\partial\Omega}\frac{f(w)}{w-z}dw. \tag{7}$$

Weber and colleagues used this formulation to derive Cauchy coordinates [2009], which were proven to be equivalent to Green coordinates when using per-edge stretch factors σ_j as in Eq. (5).

3.1.2 Polynomial deformed cages. Michel and Thiery [2023] have extended the 2D Green coordinates of Lipman et al. [2008] and have shown that if the rest curves c and deformed curves $\bar{c}$ are any C^1 arcs (rather than straight edges e_j), one can obtain angle-preserving deformations using polynomial cages by baking directly Lipman et al's Neumann condition, resulting in the following contribution of curve c to the Dirichlet and Neumann terms:

$$f_{\mathcal{D}}^c(\eta) = \int_{t=0}^{1}\frac{(c(t)-\eta)\cdot c'(t)^\perp}{2\pi\|c(t)-\eta\|^2}\bar{c}(t)dt \tag{8}$$

$$f_{\mathcal{N}}^c(\eta) = \int_{t\in 0}^{1}\frac{-1}{2\pi}\log(\|c(t)-\eta\|)\,\bar{c}'(t)^\perp dt. \tag{9}$$

While they derive this general formula for arbitrary polynomials for the rest-pose and the deformed cages, they obtain closed-form expressions for *input polygonal cages only* (i.e., each input rest-pose curve c is a segment). Noting the deformed curves $\bar{c}(t) = \sum_{k=0}^{\bar{N}_c} t^k\bar{c}_k$, they derive several per-curve coordinates $\{\phi_k^c(\eta), \psi_k^c(\eta)\}$:

$$\phi_k^c(\eta) = \int_{t=0}^{1}\frac{(c(t)-\eta)\cdot c'(t)^\perp}{2\pi\|c(t)-\eta\|^2}t^k dt \tag{10}$$

$$\psi_k^c(\eta) = \int_{t\in 0}^{1}\frac{-1}{2\pi}\log(\|c(t)-\eta\|)\,kt^{k-1}dt, \tag{11}$$

and their final deformation function reads therefore as:

$$f(\eta) = \sum_{c\in\partial\Omega}\sum_{k=0}^{\bar{N}_c}\left(\phi_k^c(\eta)\bar{c}_k + \psi_k^c(\eta)\bar{c}_k^\perp\right) \tag{12}$$

We effectively extend this method by providing closed-form expressions for the more general case where each input rest-pose curve c is an arbitrary polynomial (see Section 4.1), and we also present closed-form expressions for the derivatives of our new coordinates, allowing their use for variational deformations (see Section 7).

Cauchy-based formulation. Again, it is feasible to derive those coordinates from the Cauchy integral. Note that multiplication by $i\in\mathbb{C}$ results in a counter-clock-wise rotation by $\pi/2$ in the complex plane. We can therefore identify the coordinates of Michel and

Thiery (Eq. (12)) with the ones that can be obtained from the Cauchy formulation, which has been presented in [Lin and Chen 2024]:

$$f(z) = \sum_{c \in \partial\Omega} \sum_{k=0}^{\bar{N}_c} \left(\phi_k^c(z) - i\psi_k^c(z)\right) \bar{c}_k \tag{13}$$

3.2 Biharmonic coordinates for polygonal cages

A biharmonic function is fully described from its additional 3^{rd} and 4^{th} order boundary conditions, using the following identity:

$$f(\eta) = \int_{\xi\in\partial\Omega} f(\xi)\frac{\partial G}{\partial n_\xi}(\xi,\eta)d\xi + \int_{\xi\in\partial\Omega} -G(\xi,\eta)\frac{\partial f}{\partial n_\xi}(\xi)d\xi$$
$$+ \underbrace{\int_{\xi\in\partial\Omega} \triangle f(\xi)\frac{\partial g}{\partial n_\xi}(\xi,\eta)d\xi}_{=:f_b(\eta)\text{ (third order)}} + \underbrace{\int_{\xi\in\partial\Omega} -g(\xi,\eta)\frac{\partial \triangle f}{\partial n_\xi}(\xi)d\xi}_{=:f_B(\eta)\text{ (fourth order)}},$$

with $g(\xi,\eta) := \frac{\|\xi-\eta\|^2}{8\pi}(\log(\|\xi-\eta\|-1)$ the Green biharmonic kernel, whose derivatives are:

$$\nabla_1 g(\xi,\eta) = -\nabla_2\, g(\xi,\eta) = \frac{2\log(\|\xi-\eta\|)-1}{8\pi}(\xi-\eta)$$
$$H_1 g(\xi,\eta) = H_2 g(\xi,\eta) = \frac{2\log(\|\xi-\eta\|)-1}{8\pi}I_2 + \frac{(\xi-\eta)(\xi-\eta)^T}{4\pi\|\xi-\eta\|^2}$$
$$\triangle_1 g(\xi,\eta) = \triangle_2 g(\xi,\eta) = G(\xi,\eta).$$

Weber et al. [2012] presented biharmonic coordinates for 2D polygonal cages (each curve being a straight segment in the input and in the deformed states). In the spirit of [Lipman et al. 2008], they consider affine 3^{rd} order $\triangle f$ and constant 4^{th} order $\frac{\partial \triangle f}{\partial n_\xi}$ boundary conditions per edge, and derive biharmonic coordinates as

$$\Phi_i(\eta) = \int_{\xi\in F_1(i)} \Gamma^i(\xi)\frac{\partial g}{\partial n_\xi}(\xi,\eta)d\xi \tag{14}$$

$$\Psi_j(\eta) = \int_{\xi\in e_j} -g(\xi,\eta)d\xi \tag{15}$$

Complementary to the harmonic basis functions $\{\phi_i(\eta),\psi_j(\eta)\}$ associated with the input polygonal cage, $\{\Phi_i(\eta),\Psi_j(\eta)\}$ span a biharmonic function space. Considering 2D values $\{\bar{a}_i,\bar{b}_j,\bar{A}_i,\bar{B}_j\}$ associated with V and E, a biharmonic function can be defined as

$$f(\eta) = \sum_{i\in V}\phi_i\bar{a}_i + \sum_{j\in E}\psi_j\bar{b}_j + \sum_{i\in V}\Phi_i\bar{A}_i + \sum_{j\in E}\Psi_j\bar{B}_j \tag{16}$$

Weber et al. illustrated their use for 2D deformation in different scenarios, and have demonstrated their usefulness as additional DoFs complementary to harmonic ones. We extend their work in the sense that we allow biharmonic coordinates and their derivatives to be computed for rest-pose and deformed *polynomial* cages, when they were previously defined for rest-pose and deformed *polygonal* cages only. We derive those coordinates in Sec. 4.2 and illustrate their use for variational deformation in Sec. 7.

4 Cage coordinates for polynomial cages

We present in this section our main contributions which are the closed-form expressions for Green and biharmonic coordinates for polynomial cages, as well as their first and second order derivatives. To ease reading of the following two sections, we frame our main results to make them stand out from their mathematical derivations.

Notations for our curved cages. From now on, we consider cages made of polynomial curves of degree N_c at encoding (i.e., $c(t) = \sum_{k=0}^{N_c} t^k c_k$) being deformed into polynomial curves of degree $\bar{N}_c$ (i.e., $\bar{c}(t) = \sum_{k=0}^{\bar{N}_c} t^k \bar{c}_k$). We do not impose constraints on the degrees of the rest and deformed curves respectively, though obviously only *refined* curves (i.e., $\bar{N}_c \geq N_c$) allow reproducing the input shape, which is a common requirement in most cage-based modeling scenarios. By default, we use the same number of degrees of freedom in the input and deformed cages.

Our key insight is that all our coordinates (Green, harmonic, biharmonic) can be obtained in closed-form along with their gradient and Hessian, for both input (rest) and output (deformed) polynomial cages, from a single parametric integral F_K^c (Eq. (17)).

Parametric functional. We note $c_\eta(t) = c(t) - \eta$ (i.e., the curve c expressed in the coordinate system centered in η) and $P_c(t) := \|c_\eta(t)\|^2$ the polynomial of degree $2N_c$. We define the functional

$$\boxed{F_K^c[p] := \int_{t=0}^1 \frac{p(t)dt}{P_c(t)^K} = \int_{t=0}^1 \frac{p(t)dt}{\|c(t)-\eta\|^{2K}}.} \tag{17}$$

for any polynomial $p(X) := \sum_i p_i X^i$, whose derivation (Sec. 5) will be key to finding all our coordinates and their derivatives.

We make the trivial yet critically-important observation that F is a *linear* functional:

$$\boxed{F_K^c\left[\sum_i p_i X^i\right] = \sum_i p_i F_K^c\left[X^i\right].} \tag{LIN}$$

This effectively moves the complexity of evaluating all our cage coordinates to the one of evaluating $F_K^c[X^i]$, for which we detail in Sec. 5 an efficient implementation.

Our derivation of gradients and Hessians relies on the following differential property of F, for any scalar and vector polynomials p:

$$\boxed{\nabla_\eta^T F_K^c[p] = F_K^c\left[\nabla_\eta^T p\right] + 2K F_{K+1}^c\left[p c_\eta^T\right].} \tag{DIF}$$

We now formulate the harmonic coordinates ϕ, ψ (Sec. 4.1) and biharmonic ones Φ, Ψ (Sec. 4.2), as well as their derivatives, as linear combinations of $\{F_K^c[X^i]\}$. We defer derivations to Appendix A.

4.1 Green coordinates for polynomial curves

We build upon the formalism of Michel and Thiery [2023], and first derive Green coordinates for polynomial curves, effectively extending their formulation by allowing rest-pose cages to be made of non-straight polynomial curves.

4.1.1 Dirichlet term. $\phi_k^c(\eta)$ (Eq. (10)) can be rewritten as

$$\boxed{\phi_k^c(\eta) = \frac{1}{2\pi}F_1^c\left[X^k c_\eta \cdot c'^\perp\right].} \tag{18}$$

Derivatives. We easily differentiate $\phi_k^c(\eta)$ using (DIF):

$$\boxed{\nabla_\eta(\phi_k^c)(\eta) = -\frac{1}{2\pi}F_1^c\left[X^k c'^\perp\right] + \frac{1}{\pi}F_2^c\left[X^k\left(c_\eta \cdot c'^\perp\right)c_\eta\right].} \tag{19}$$

We repeat this process once again to express its Hessian:

$$\boxed{\begin{aligned}H_\eta(\phi_k^c)(\eta) = &-\frac{1}{\pi}F_2^c\left[X^k\left(c'^{\perp}c_\eta^T + c_\eta c'^{\perp T} + c_\eta\cdot c'^{\perp}I_2\right)\right]\\ &+\frac{4}{\pi}F_3^c\left[X^k\left(c_\eta\cdot c'^{\perp}\right)c_\eta c_\eta^T\right]\end{aligned}} \quad (20)$$

4.1.2 Neumann term. Using integration by parts (more details given in Appendix A), $\psi_k^c(\eta)$ (Eq. (11)) can be rewritten as

$$\boxed{\psi_k^c(\eta) = \frac{1}{2\pi}\left(-\log(\|c_\eta(1)\|) + F_1^c\left[X^k c_\eta\cdot c'\right]\right).} \quad (21)$$

As noted by Michel and Thiery [2023], $\psi_0^c(\eta) = 0$, which is trivial to see beforehand since $\bar{c}_0$ does not appear in the expression of $\bar{c}'(t)$.

Derivatives. The gradient of $\psi_k^c(\eta)$ is given by

$$\boxed{\nabla_\eta\psi_k^c(\eta) = \frac{k}{2\pi}F_1^c\left[X^{k-1}c_\eta\right].} \quad (22)$$

Differentiating $\nabla_\eta\psi_k^c(\eta)$ using (DIF), we obtain

$$\boxed{H_\eta\psi_k^c(\eta) = \frac{-k}{2\pi}F_1^c\left[X^{k-1}\right]I_2 + \frac{k}{\pi}F_2^c\left[X^{k-1}c_\eta c_\eta^T\right].} \quad (23)$$

4.1.3 Final expression. Considering the whole contour $\partial\Omega$, we obtain the following expression for the deformation function:

$$f(\eta) = \sum_{c\in\partial\Omega}\sum_{k=0}^{\bar{N}_c}\phi_k^c(\eta)\bar{c}_k + \psi_k^c(\eta)\bar{c}_k^{\perp} \quad (24)$$

Omitting (η) for readability, the Jacobian $\nabla_\eta^T f(\eta)$ and the Hessian $H_\eta f(\eta)$ of the deformation function are given by:

$$\nabla_\eta^T f(\eta) = \sum_{c\in\partial\Omega}\sum_{k=0}^{\bar{N}_c}\bar{c}_k\,\nabla^T\phi_c^k + \bar{c}_k^{\perp}\,\nabla^T\psi_c^k \quad (25)$$

$$H_\eta f(\eta) = \sum_{c\in\partial\Omega}\sum_{k=0}^{\bar{N}_c}\bar{c}_k H\phi_c^k + \bar{c}_k^{\perp}H\psi_c^k \quad (26)$$

4.1.4 Harmonic-only coordinates. Note that *harmonic* deformations can be obtained using the following expression:

$$f(\eta) = \sum_{c\in\partial\Omega}\sum_{k=0}^{\bar{N}_c}\phi_k^c(\eta)\bar{c}_k + \psi_k^c(\eta)\bar{n}_k \quad (27)$$

i.e., by relaxing the normal alignment constraint (i.e., $\bar{n}_k \neq \bar{c}_k^{\perp}$). While it is common to use Eq. (24) for direct cage editing, Eq. (27) offers more DoFs and spans a larger harmonic deformation subspace commonly preferred in variational deformation scenarios (Sec. 7).

4.2 Biharmonic coordinates for polynomial curves

We demonstrate in this section that biharmonic coordinates, after a few transformations of the traditional expressions, can be expressed as combinations of $\{F_K^c\}$, thus allowing for their computation and the computation of their gradients and Hessians as well.

We follow in spirit the constructions of the biharmonic 3^{rd} and 4^{th}-order boundary conditions advocated in [Weber et al. 2012] and [Thiery et al. 2024], who introduced biharmonic coordinates for cages with *planar polygonal facets* in 2D and 3D respectively. Again, we defer mathematical details to Appendix A.

4.2.1 Third-order condition. By mimicking the construction presented in the previous section, we propose to model the third-order boundary condition using a polynomial curve $\bar{A}_c(t) = \sum_k t^k\bar{A}_c^k$, and define the contribution of polynomial curve c to the third order boundary condition's diffusion as:

$$\begin{aligned}f_b^c(\eta) &:= \int_{t=0}^{1}\bar{A}_c(t)\,\nabla_1 g(c(t),\eta)\cdot c'(t)^{\perp}dt \quad (28)\\ &= \sum_k\bar{A}_c^k\underbrace{\int_{t=0}^{1}\frac{t^k c_\eta(t)\cdot c'(t)^{\perp}}{8\pi}\left(2\log(\|c_\eta(t)\|)-1\right)dt}_{=:\,\Phi_k^c(\eta)}\end{aligned}$$

which thus introduces our first biharmonic coordinate $\Phi_k^c(\eta)$.

We note $P_k^c(t) := t^k c_\eta(t)\cdot c'(t)^{\perp}/(8\pi)$ and $w(t) := 1-2\log(\|c_\eta(t)\|)$. $P_k^c(t)$ is a simple polynomial ($P_k^c(t) := \sum_k\alpha_k t^k$) whose primitive $\tilde{P}_k^c(t)$ taking value 0 at $t=0$ can be computed analytically:

$$\tilde{P}_k^c(t) := \int_{u=0}^{t}P_k^c(u)du = \sum_k\alpha_k t^{k+1}/(k+1),$$

and w' is given by $w'(t) = -2c'(t)\cdot c_\eta(t)/\|c_\eta(t)\|^2$.

Using integration by parts ($\int_0^1 P_k^c w = [\tilde{P}_k^c w]_0^1 - \int_0^1\tilde{P}_k^c w'$), we obtain

$$\boxed{\Phi_k^c(\eta) = -w(1)\tilde{P}_k^c(1) - 2F_1^c\left[(c'\cdot c_\eta)\tilde{P}_k^c\right]} \quad (29)$$

Similarly, noting $Q_k^c(t) := t^k c'(t)^{\perp}/(8\pi)$ and $\tilde{Q}_k^c(t) = \int_{u=0}^{t}Q_k^c(u)du$ its primitive taking null value in $t=0$, we obtain

$$\boxed{\nabla_\eta\Phi_k^c(\eta) = w(1)\tilde{Q}_k^c(1) + 2F_1^c\left[(c'\cdot c_\eta)\tilde{Q}_k^c - (Q_k^c\cdot c_\eta)c_\eta\right]} \quad (30)$$

Finally, $H_\eta\Phi_k^c$ is given by

$$\boxed{\begin{aligned}H_\eta\Phi_k^c(\eta) = &\frac{1}{4\pi}F_1^c\left[X^k\left(c'^{\perp}c_\eta^T + c_\eta c'^{\perp T} + (c'^{\perp}\cdot c_\eta)I_2\right)\right]\\ &-\frac{1}{2\pi}F_2^c\left[X^k(c'^{\perp}\cdot c_\eta)c_\eta c_\eta^T\right]\end{aligned}} \quad (31)$$

4.2.2 Fourth-order condition. Setting the fourth-order boundary condition as a polynomial curve $\bar{B}_c(t) := \sum_k t^k\bar{B}_k^c$, the contribution of curve c to its diffusion can be expressed similarly:

$$\begin{aligned}f_B^c(\eta) &:= \int_{t=0}^{1}-\bar{B}_c(t)g(c(t),\eta)dt \quad (32)\\ &= \sum_k\bar{B}_c^k\underbrace{\int_{t=0}^{1}\frac{t^k\|c_\eta(t)\|^2}{8\pi}(1-\log(\|c_\eta(t)\|))dt}_{=:\,\Psi_k^c(\eta)}\end{aligned}$$

which leads to our second biharmonic coordinate $\Psi_k^c(\eta)$.

As in Sec. 4.2.1, we solve those integrals using integration by parts. We note $R_k^c(t) := t^k\|c_\eta(t)\|^2/(8\pi)$, $v(t) := (1-\log(\|c_\eta(t)\|))$, $S_k^c(t) := -t^k c_\eta(t)/(8\pi)$, and $w(t) := 1-2\log(\|c_\eta(t)\|)$. $R_k^c(t)$ and

$S_k^c(t)$ are simple polynomials whose primitives can be computed analytically, and $w'(t) = 2v'(t) = -2c'(t)\cdot c_\eta(t)/\|c_\eta(t)\|^2$.

Noting $\tilde{R}_k^c(t) := \int_{u=0}^t R_k^c(u)du$ and $\tilde{S}_k^c(t) = \int_{u=0}^t S_k^c(u)du$ the primitives of R_k^c and S_k^c taking null values in $t = 0$, we obtain closed-form expressions for $\Psi_k^c(\eta)$ and its derivatives as

$$\boxed{\Psi_k^c(\eta) = v(1)\tilde{R}_k^c(1) + F_1^c\left[\tilde{R}_k^c c'\cdot c_\eta\right]} \tag{33}$$

$$\boxed{\nabla_\eta \Psi_k^c(\eta) = w(1)\tilde{S}_k^c(1) + 2F_1^c\left[\tilde{S}_k^c c'\cdot c_\eta\right]} \tag{34}$$

$$\boxed{H_\eta \Psi_k^c(\eta) = \frac{w(1) + 2F_1^c\left[X^{k+1}c'\cdot c_\eta\right]}{8\pi(k+1)} I_2 - \frac{F_1^c\left[X^k c_\eta c_\eta^T\right]}{4\pi}} \tag{35}$$

These new coordinates $\{\Phi_k^c(\eta), \Psi_k^c(\eta)\}$ are biharmonic by construction (by definition of Eqs. (28) and (65)), and result in two biharmonic boundary conditions that are complementary to each other, which together create a rich space for biharmonic function design. While it is typically difficult to control deformations by setting those boundary conditions by hand explicitly, we experiment in Section 7 their utility in the context of variational shape deformations driven by polynomial biharmonic boundary conditions.

4.2.3 Final expression. Considering the whole cage $\partial\Omega$, we obtain:

$$f(\eta) = \sum_{c\in\partial\Omega}\sum_{k=0}^{\bar{N}_c} \phi_k^c(\eta)\bar{a}_c^k + \psi_k^c(\eta)\bar{b}_c^k + \Phi_k^c(\eta)\bar{A}_c^k + \Psi_k^c(\eta)\bar{B}_c^k \tag{36}$$

Note that those coordinates formally extend the ones we previously introduced, since Green coordinates for polynomial cages are obtained by setting $(\bar{a}_c^k = \bar{c}_k, \bar{b}_c^k = \bar{c}_k^\perp, \bar{A}_c^k = \bar{B}_c^k = 0)$.

Omitting (η) for readability, the Jacobian $\nabla_\eta^T f(\eta)$ and the Hessian $H_\eta f(\eta)$ of the deformation function are given by:

$$\nabla_\eta^T f(\eta) = \sum_{c\in\partial\Omega}\sum_{k=0}^{N_c} \bar{a}_c^k\, \nabla^T\phi_c^k + \bar{b}_c^k\, \nabla^T\psi_c^k + \bar{A}_c^k\, \nabla^T\Phi_c^k + \bar{B}_c^k\, \nabla^T\Psi_c^k \tag{37}$$

$$H_\eta f(\eta) = \sum_{c\in\partial\Omega}\sum_{k=0}^{N_c} \bar{a}_c^k H\phi_c^k + \bar{b}_c^k H\psi_c^k + \bar{A}_c^k H\Phi_c^k + \bar{B}_c^k H\Psi_c^k \tag{38}$$

5 A simple framework for the evaluation of $F_K^c[p]$ and the coordinates

5.1 Mathematical derivation

We recall that we need to evaluate $F_K^c[p]$ for any polynomial $p = \sum_n p_n X^n$, which, thanks to the linearity property of F (LIN), reduces to computing the following:

$$F_{K,n}^c := F_K^c\left[X^n\right] = \int_{t=0}^1 \frac{t^n dt}{P_c(t)^K} \tag{39}$$

We present in this section a simple mathematical derivation, which we separate from all necessary implementation details that we present in the next subsection.

We follow [Michel and Thiery 2023], and start by factorizing the polynomial $P_c(t) = \|c(t) - \eta\|^2$:

$$P_c = A\prod_j \left(X - \omega_j\right)^{m_j} \tag{40}$$

where $\omega_j \in \mathbb{C}$ are the complex roots of P_c with their multiplicity m_j (summing up to $2N_c$). These roots are found as the eigenvalues of the companion matrix of P_c [Edelman and Murakami 1995] using standard linear algebra tools (Eigen [2010] in our case) and A is the leading coefficient of P_c. This leads to

$$F_{K,n}^c = \frac{1}{A^K}\int_{t=0}^1 \frac{t^n}{\prod_j \left(t - \omega_j\right)^{m_j K}} dt \tag{41}$$

We next compute a partial fraction decomposition of $\frac{1}{P_c(t)^K}$ to express it as a sum of single-root terms (see Algo. 1 and Theorem 1):

$$\frac{1}{P_c^K} = \frac{1}{A^K \prod_j \left(X - \omega_j\right)^{m_j K}} = \frac{1}{A^K}\sum_i \frac{\alpha_i}{(X - \omega_{j_i})^{n_i}} \tag{42}$$

Exposing this decomposition is key to expressing $F_{K,n}^c$ using a simple and concise formula:

$$\boxed{F_{K,n}^c = \frac{1}{A^K}\sum_i \alpha_i G_{n,-n_i}(\omega_{j_i})} \tag{43}$$

where $G_{a,b}(\omega) := \int_{t=0}^1 t^a (t - \omega)^b$ can be evaluated recursively:

$$G_{a,b}(\omega) = \begin{cases} \omega^a D(\omega) + U_a(\omega) & b = -1 \\ \frac{1}{b+1}\left((1-\omega)^{b+1} - (0-\omega)^{b+1}\right) & a = 0 \\ \frac{1}{b+1}\left((1-\omega)^{b+1} - aG_{a-1,b+1}(\omega)\right) & a > 0 \text{ and } b \neq -1 \end{cases}$$

with

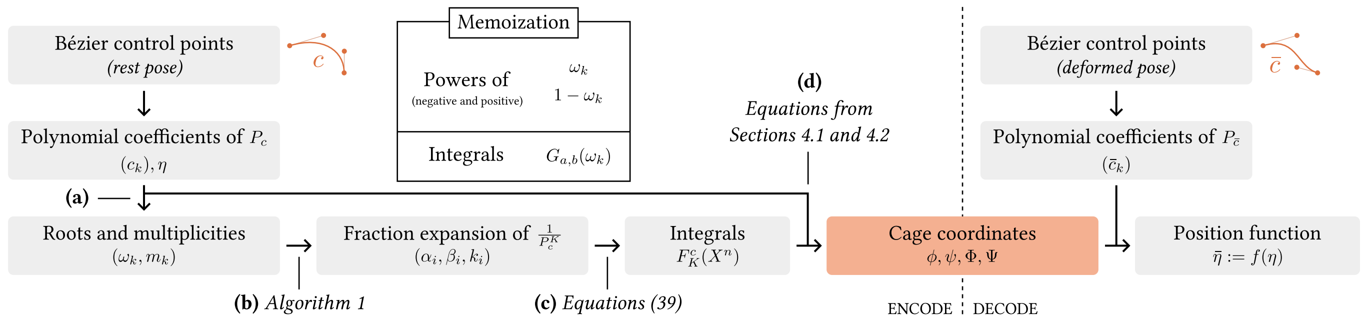
$$\begin{aligned} D(\omega) &:= \mathrm{Log}(1-\omega) - \mathrm{Log}(0-\omega) \\ &= \frac{1}{2}\log\left(1 + \frac{1 - 2\,\mathrm{Re}(\omega)}{\|\omega\|^2}\right) + i\,\mathrm{atan2}\left(\mathrm{Im}(\omega), \|\omega\|^2 - \mathrm{Re}(\omega)\right) \\ U_a(\omega) &:= \sum_{k=0}^{a-1}\frac{\omega^k}{a-k} \end{aligned}$$

The first case ($b = -1$) is a key result obtained by Michel and Thiery (see [Michel and Thiery 2023], Section 3.3, Lemma 2), the second case ($a = 0$) corresponds to direct integration of $(t-\omega)^b$, and the last case ($a > 0$ and $b \neq -1$) corresponds to a simple integration by parts of $G_{a,b}$ and leads to the recursive nature of $G_{a,b}$.

5.2 Algorithmic details

Fig. 2 summarizes our implementation. For a given curve c and a 2D evaluation point η, we evaluate all the $F_{K,n}^c$ we need together since they share the same roots $\{\omega_j\}$. In particular, we precompute the powers of ω_j and $1 - \omega_j$ for all j to speed-up the evaluation of $G_{a,b}$.

Partial fraction decomposition. Our progressive and efficient computation of the partial fraction decomposition is detailed in Algo. 1. We recall the (classical) partial fraction decomposition theorem, which we present in simplified form to accommodate our use case:



Bézier control points *(rest pose)*
c
Polynomial coefficients of P_c $(c_k), \eta$
Memoization
Powers of (negative and positive) ω_k $1-\omega_k$
Integrals $G_{a,b}(\omega_k)$
(d) *Equations from Sections 4.1 and 4.2*
Bézier control points *(deformed pose)*
$\bar{c}$
Polynomial coefficients of $P_{\bar{c}}$ $(\bar{c}_k)$
(a)
Roots and multiplicities (ω_k, m_k)
Fraction expansion of $\frac{1}{P_c^K}$ (α_i, β_i, k_i)
Integrals $F_K^c(X^n)$
Cage coordinates ϕ, ψ, Φ, Ψ
Position function $\bar{\eta} := f(\eta)$
(b) *Algorithm 1*
(c) *Equations (39)*
ENCODE
DECODE

Fig. 2. Overview of computations. **(a)** The roots of P_c are found using the eigenvalues of its companion matrix. **(b)** Algo. 1 provides the terms $\alpha_i(X-\omega_{k_i})^{-\beta_i}$ of the expansion of $1/P_c^K$. **(c)** We integrate these terms using Eq. (43). **(d)** With the $F_K^c(X^n)$ in hand, computing our various coordinates is straightforward.

THEOREM 1. *Let Q be a polynomial of the form $Q := \prod_{j=0}^{J}(X-\omega_j)^{m_j}$. $\frac{1}{Q}$ can be decomposed into a sum of single root fractions as:*

$$\frac{1}{Q} = \prod_{j=0}^{J} \frac{1}{(X-\omega_j)^{m_j}} = \sum_{j=0}^{J}\sum_{k=0}^{m_j-1} \frac{a_{jk}}{(X-\omega_j)^{m_j-k}} \quad \textit{with}$$

$$a_{jk} := \frac{1}{k!}\left(\frac{1}{Q_j}\right)^{(k)}(\omega_j)$$ [1]

$$Q_j := \prod_{l\neq j}(X-\omega_l)^{m_l} := \frac{Q}{(X-\omega_j)^{m_j}}$$

Note that, while the explicit formula for $\left(1/Q_j\right)^{(k)}$ is difficult to write explicitly, *computing those iteratively is straightforward.* Indeed, since $1/Q_j = \prod_{l\neq j}(X-\omega_l)^{-m_l}$, its derivative is given by

$$\left(1/Q_j\right)' = \sum_{l\neq j} -m_l(X-\omega_l)^{-m_l-1}\prod_{l'\neq l,j}(X-\omega_{l'})^{-m_{l'}} \ ,$$

and one can see by recurrence that derivatives of arbitrary orders of $1/Q_j$ are of the form of sum of polynomials:

$$\left(1/Q_j\right)^{(k)} = \sum_l \beta_l \prod_{l'}(X-\omega_{ll'})^{m_{ll'}}. \tag{44}$$

To compute efficiently and iteratively the derivatives of $1/Q_j$, we use an abstract representation of (rational) polynomials of the form of Eq. 44 as a list of lists $\{\beta_l \in \mathbb{R}; \{(\text{rootIndex}_{ll'}, m_{ll'}) \in \mathbb{N}\times\mathbb{Z}\}_{l'}\}_l$ and implement the associated `derivative` (see Algo. 2 for a simple pseudo code) and `evaluation` routines.

Link with [Liu et al. 2024]. Liu et al. obtain a different formula based on residuals computation. Our two different (yet equivalent) formulations are linked simply because residuals computations are one way to express the rational fraction decomposition (and/or their integral) we exhibit. The computational framework we use makes the exposition of our F functional much simpler, and allows for the "almost free" derivation of the derivatives of the coordinates (requiring the reformulation of the formulas to make our parametric integrals appear, and computing integrals of $t^n/\|c(t)-\eta\|^{2K}$ instead of $t^n/\|c(t)-\eta\|^2$ – i.e., our formulation allows for similar computations for the simple extension to the case $K>1$).

We also recall that, compared with [Liu et al. 2024], we introduce formulas for biharmonic coordinates w.r.t.. curved polynomial cages, and we also introduce formulas for gradients and Hessians of both sets of coordinates, allowing for variational shape deformation.

[1] $f^{(k)}$ denotes the kth derivative of f

ALGORITHM 1: Partial fraction decomposition of $1/Q$.

```
Data: The list of unique roots $(\omega_j)_{0\le j<J} \in \mathbb{C}$ and their multiplicity
      $m_j > 0$ for the polynomial $Q$ for which we want to compute
      the expansion of $\frac{1}{Q}$.
Result: A list of triplets $(\alpha_i, n_i, j_i) \in \mathbb{R}\times\mathbb{N}^*\times\mathbb{N}$ (i.e., $n_i > 0$)
      representing terms of the form $\alpha_i(X-\omega_{j_i})^{-n_i}$ which sum
      to $\frac{1}{Q}$. The number of terms is the degree $\sum_j m_j$ of $Q$.
Structures: $R$ is a sum of products of terms $(X-\omega_k)^m$ where
      $m \in \mathbb{Z}$; it is abstracted as a list of lists of $(k, m)$, which
      makes both evaluation and differentiation convenient.
$\mathcal{L} \leftarrow \emptyset$;
for $j \leftarrow 0$ to $J-1$ do
    // R is initialized to 1/Q without the $j^{th}$ root of Q:
    $R \leftarrow \prod_{l\neq j}(X-\omega_l)^{-m_l}$ // $R \leftarrow 1/Q_j$
    for $k = 0$ to $m_j - 1$ do
        $\alpha \leftarrow \frac{R(\omega_j)}{k!}$ // i.e., $\alpha$ is $a_{jk} = (1/Q_j)^{(k)}(\omega_j)/k!$
        $n \leftarrow m_j - k$;
        append $(\alpha, n, j)$ to $\mathcal{L}$;
        $R \leftarrow$ derivative($R$) // $R \leftarrow (1/Q_j)^{(k+1)}$, Algo 2
    end
end
return $\mathcal{L} = \{(\alpha_i, n_i, j_i)\}_i$ abstracting $1/Q = \sum_i \alpha_i(x-\omega_{j_i})^{-n_i}$
```

Numerical stability. When eigenvalues of the companion matrix of P_c are less than $\epsilon = 10^{-5}$ apart, we merge them into a single root with the sum of their multiplicities. This prevents the evaluation of $R(\omega)$ in Algorithm 1 from reaching numerical limits, and generalizes the stability test of Michel and Thiery [2023].

As they note, $D(\cdot)$ is well defined for $\eta \notin c$: since in such a case $P_c(t)$ is not null for $t=0$ or $t=1$, so $\omega \notin \{0,1\}$. Lastly, we found in practice that evaluation of $F_{K,n}^c$ may required increased floating point precision for robust evaluation. In our experiments, we observed that the computation of $F_{3,n}^c$, necessary for the evaluation of the Hessian of the ϕ harmonic coordinate, required sometimes

ALGORITHM 2: Derivative of a single fully-factorized polynomial $R = \beta \prod_{l'} (X - \omega_{l'})^{m_{l'}}$ (a single element of the sum in Eq. (44)).

```
Data: A structure FactorizedPoly containing the list of pairs (root
      index, exponent) $\{(\text{ind}_{l'}, m_{l'}) \in \mathbb{N} \times \mathbb{Z}\}_{l'}$ and a dominant
      coefficient $\beta$ abstracting $R$.
Result: A list abstracting $R'$ taking the form given in Eq. (44).
$\mathcal{L} \leftarrow$ [] // to fill
for $k = 0$ to R.terms.size() - 1 do
    FactorizedPoly newElem = $R$;
    $m$ = newElem.terms[$k$].exponent;
    if $m == 0$ then
        continue // constant terms disappear
    end
    newElem.dominantCoeff *= $m$;
    newElem.terms[$k$].exponent -= 1;
    if $m == 1$ then
        newElem.terms.erase(newElem.terms.begin() + $k$)
    end
    $\mathcal{L}$.push_back(newElem)
end
return $\mathcal{L}$ // abstraction of $R'$
```

Table 1. Timings **in μs** for roots $\{\omega_j\}$ extraction for a single evaluation point η, depending on the degree N_c of the input rest pose curvev c.

Degree N_c	1	2	3	4	5	6
Time (μs)	2.5	7.6	17.4	30.2	57.2	91

multiprecision for degree $n > 1$ polynomials. While it renders the computation much slower, we use multiprecision solely for the evaluation of $F^c_{3,n}$ in the computation of the Hessians (which we use only for sparse set of points distributed along the boundary in variational deformation methods – see Section 7).

Timings and complexity analysis. We provide here computation timings, which were recorded on a laptop with an Intel Processor (i7-10850H CPU @ 2.70GHz, 8 cores) and 32GB of RAM. All timings provided in this section were averaged over thousands of evaluation points. They exhibit a strong variance as they depend on many factors (input curve geometry, location of point η w.r.t. curves, etc.), and should therefore be understood as illustrating global tendencies and orders of magnitude only.

Table 1 gives timings for extracting the roots of the companion matrix for a single point w.r.t. to a single curve, those timings depending solely on the degree of the rest-pose curve. Table 2 gives timings for computing the Green harmonic coordinates (and gradients) w.r.t. a single curve, those timings depending both on the degree of the rest-pose curve and the degree of its deformable counterpart. Table 3 gives the same information for the biharmonic case.

Key points emerging from this analysis are that:

- root-finding and the rest are relatively on par overall,
- biharmonic coordinates require a bit less than twice the time needed for harmonic coordinates computation (similarly for their gradients and Hessians), and

Table 2. Timings T_{coord} **in μs** for the computation of Green/harmonic coordinates $\{\phi^k_c, \psi^k_c\}$ and timings T_{grad} for the computation of their gradients for a single evaluation point w.r.t. a single curve c, depending on its degree N_c in the input rest pose state and its degree $\bar{N}_c$ in the deformable state. We provide in each cell ($T_{\text{coord}}/T_{\text{coord}} + T_{\text{grad}}$).

N_c \ $\bar{N}_c$	1	2	3	4	5	6
1	8/15	9/20	11/25	12/26	12/27	13/28
2	20/47	22/55	24/59	25/68	27/68	28/69
3	38/168	42/108	45/118	50/125	51/135	51/140
4	69/174	74/191	79/215	79/212	82/225	101/235
5	111/293	113/305	117/323	124/340	135/363	137/371
6	171/461	173/477	175/481	182/510	195/532	202/555

Table 3. Timings T_{coord} **in μs** for the computation of biharmonic coordinates $\{\phi^k_c, \psi^k_c, \Phi^k_c, \Psi^k_c\}$ (supersetting the harmonic coordinates whose timings are reported in Table 2) and timings T_{grad} for the computation of their gradients for a single evaluation point w.r.t. a single curve c, depending on its degree N_c in the input rest pose state and its degree $\bar{N}_c$ in the deformable state. We provide in each cell ($T_{\text{coord}}/T_{\text{coord}} + T_{\text{grad}}$).

N_c \ $\bar{N}_c$	1	2	3	4	5	6
1	14/35	17/35	22/42	24/50	27/52	29/57
2	35/78	39/86	43/88	47/100	49/112	53/120
3	62/160	70/173	80/177	82/202	86/215	94/216
4	119/286	129/300	138/315	143/325	150/359	157/358
5	193/454	207/470	206/478	216/500	222/524	234/549
6	272/656	297/688	299/725	312/740	322/780	349/793

Table 4. Timings T^H_{double} (resp. T^H_{256}) **in ms** for the computation of Green/harmonic coordinates' Hessians $\{H\phi^k_c, H\psi^k_c\}$ using double precision (resp. using 256 bits)for the evaluation of F^c_3 for a single evaluation point w.r.t. a single curve c, depending on its degree N_c in the input rest pose state and its degree $\bar{N}_c$ in the deformable state. We provide in each cell ($T^H_{\text{double}}/T^H_{256}$).

N_c \ $\bar{N}_c$	1	2	3	4	5	6
1	.03/1.1	.03/1.6	.04/1.6	.04/1.7	.04/2/0	.05/2.2
2	.1/7.4	.11/7.4	.12/9.3	.12/8.6	.14/9.2	.16/9.8
3	.3/29	.3/30	.3/30	.3/31	.3/32	.3/35
4	.6/89	.6/88	.6/89	.7/90	.7/92	.7/95
5	1.2/207	1.2/210	1.3/211	1.3/213	1.29/216	1.3/221
6	2.1/420	2.1/429	2.1/433	2.1/435	2.2/441	2.2/446

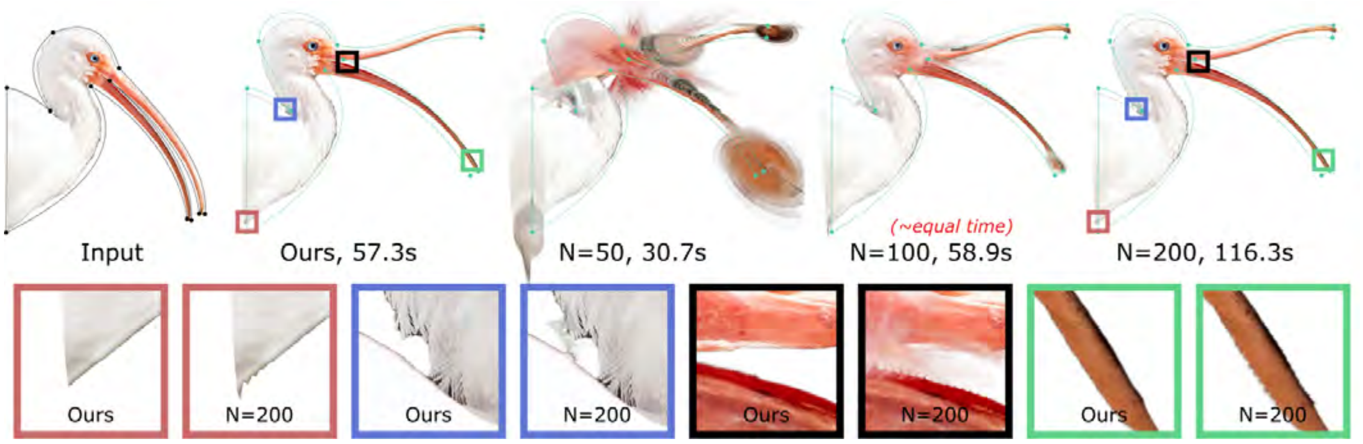


Fig. 3. Comparison with numerical integration. N denotes the number of discretization steps used per curve. Note that instabilities remain even in the highest discretization case (N=200), for which computations are more expensive than evaluating our formulas. The case N=100 provides an equal-time comparison.

- the degree N_c of the rest pose curve has a stronger impact on timings than the degree $\bar{N}_c$ of the deformation curve.

Table 4 gives timings for the computation of the Hessians of our Green coordinates, comparing the use of *standard double precision* or *256-bit precision* for the computation of F^c_3. As one can see, using multiprecision libraries (we use mpfr++[Fousse et al. 2007]) for the computation of F^c_3 is the main bottleneck in this setup. While this impacts the performances drastically, Hessians are typically only computed for very few points (points sparsely distributed around the boundary of the input cage) in common variational methods.

Simplest (and most frequent) case analysis. Though our expressions are meant to handle the general case and allow for computation of gradients and Hessians of all our coordinates, difficult cases are actually rare in practice. The polynomial $P_c(t) = \|c(t) - \eta\|^2$ has in general roots of multiplicity 1 (as in general, there is no t such that $c(t) = \eta$ since c is a $1D$ curve embedded in the $2D$ plane, i.e., of null Lebesgue measure) and the most frequent computations are required for coordinates only in our applications – needed for deforming the whole image inside the cage, while gradients and Hessians are typically required at very few points sampled inside the domain for variational methods. The most frequent case (by far) therefore requires relatively simple computations, since the decomposition of Eq. (42) leads to trivial terms only ($n_i = 1\ \forall i$ in Eq. (42)) in this case, and G then requires no recursion to compute $F^c_{1,n}$ (Eq. (43)).

Comparison with a numerical integration baseline. We illustrate in Fig. 3 the benefits of using our closed-form expressions compared with standard numerical integration. Not only does a numerical integration approach not allow reaching the quality offered by our closed-form expressions, numerical instabilities are also difficult to predict ahead of time as they may appear at different places depending on: the input curves geometry, the degree of the deformation curve, or the relative placement of the evaluation point to the cage. While resorting to numerical integration may seem a "simple alternative" to the actual implementation of our expressions, performing robust and accurate numerical integration of diffusion kernels such as the ones we consider is non trivial and is an active research domain on its own (see for example [Bang et al. 2023]).

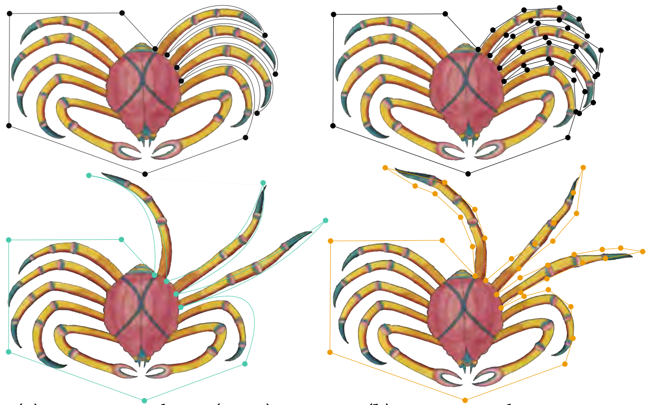

(a) From curved rest **(ours)**
(b) From straight rest

Fig. 4. Some rest shapes are not easily bounded by cages made of straight edges. Our method allows specifying rest cages made of polynomial curves (a). Previous methods require 4 times more segments in the rest cage (b), which makes the manipulation more prone to wiggling stretch factor.

6 Green Deformations Results

Our Green deformations strictly extend the ones of Michel and Thiery [2023] and share all their basic properties, namely they ensure angle-preserving deformations and are compatible with curved deformed cages. However, Michel and Thiery's coordinates can be used with straight rest cages only. We illustrate in Fig. 4 that overcoming this limitation facilitates the use of cage coordinates on shapes that are not provided in (sometimes artificially) straight poses. Straight cages can always end up fitting the shape under repeated refinement, but this can lead to a high number of coordinates, and a refined cage is not as easily controlled by an artist as a polynomial one. More examples can be found in Figs. 1, 3 and 12.

7 Variational Shape Deformation

In this section, we investigate the use of our coordinates for variational shape deformation.

In the following, we note $\bar{C}$ the set of control points for all deformed curves in order. All the variational methods presented in this section require minimizing deformation energies of the form:

$$\mathcal{E}^{\texttt{SUBSPACE}}_{\texttt{COST}}(\bar{C}) := \frac{1}{2}\left\|\mathcal{A}\bar{C} - \mathcal{B}\right\|^2 . \tag{45}$$

where COST refers to the type of penalty being considered (e.g., as-rigid-as-possible, ...) and SUBSPACE refers to the type of coordinates we use (Green, harmonic, biharmonic). The result of the minimization of $\mathcal{E}^{\texttt{SUBSPACE}}_{\texttt{COST}}$ is the set of cage parameters $\bar{C}$ which is then used to deform the entire space inside the rest pose cage.

Continuity enforcement. Our cage-based deformations functions (Eqs. (27) and (36)) require summing up the contributions of the various curves, which are each expressed as polynomials. This setup considers effectively that our cages are "curve-soups" (i.e., separate disconnected curves). While this is harmless when manipulating the cages directly (as the separate curves have their endpoints meeting at the same geometric location by design), we need to enforce exact continuity of the deformation function when "jumping" from the endpoint of a curve to the starting point of its adjacent curve in variational deformation scenarios. This is done by ensuring that endpoints of adjacent curves c and c^{next} coincide ($\bar{c}(1) = \bar{c}^{\text{next}}(0)$), i.e.:

$$\sum_k \bar{c}_k = \bar{c}_0^{\text{next}} \tag{46}$$

Noting $\bar{C}$ the set of control points for all curves in order, considering Eq. (46) for all pairs of successive curves (c, c^{next}) leads to

$$\Lambda\bar{C} = 0 \tag{CONT}$$

Minimizing Eq. (45) subject to Eq. (CONT) is done by solving for

$$\begin{pmatrix} \mathcal{A}^T\mathcal{A} & \Lambda^T \\ \Lambda & 0 \end{pmatrix}\begin{pmatrix} \bar{C} \\ \lambda \end{pmatrix} = \begin{pmatrix} \mathcal{A}^T\mathcal{B} \\ 0 \end{pmatrix} \tag{47}$$

which corresponds to the minimization of the general Lagrangian $\mathcal{L}(\bar{C}, \lambda) := \frac{1}{2}\left\|\mathcal{A}\bar{C} - \mathcal{B}\right\|^2 + \lambda^T\Lambda\bar{C}$.

For variational biharmonic coordinates, we enforce continuity of the third-order boundary condition at curve extremities as well, resulting in additional constraints of the form given in Eq. (46), doubling the number of lines in the system written in Eq. (CONT).

7.1 Variational Green deformations

We start by presenting a simple variational Green-based deformation solver. Given user prescribed point constraints $C_{\mathcal{P}} = \{p_i \mapsto \bar{p}_i\}_i$, we minimize the following quadratic energy

$$\mathcal{E}^{\texttt{GREEN}}_{\texttt{AAffAP}}(f) := \sum_{i \in C_{\mathcal{P}}} \|f(p_i) - \bar{p}_i\|^2 + \sigma_H \sum_{h \in C_{\mathcal{H}}} \|Hf(h)\|^2 \tag{48}$$

where $C_{\mathcal{H}}$ is a set of points distributed over the input domain, at which the Hessian of f is minimized (thus resulting in "as-affine-as-possible" – AAffAP – deformations). Following [Ben-Chen et al. 2009], we sample Hessian constraints $C_{\mathcal{H}}$ near $\partial\Omega$ (in practice, we sample 10 constraints regularly along each curve).

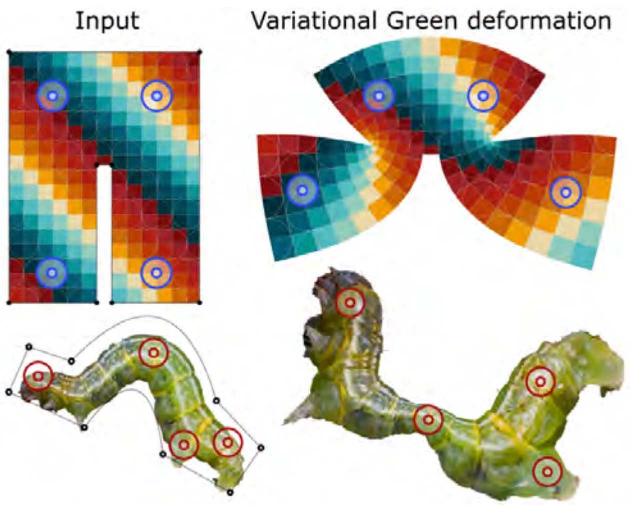


Fig. 5. We allow for variational Green-based deformations, by minimizing a simple smoothness energy such as an as-affine-as-possible one (minimizing the square norm of the Hessian) under positional constraints.

We show in Fig. 5 results of this approach when using a Green-based subspace (i.e., using f as in Eq. (27), subject to strict continuity constraints as described in Eq. (CONT)), for shapes with input curved cages. These results are similar to the ones presented in [Lin and Chen 2024], though our approach does not require any straightening step. While this allows for direct editing of the shape using simple positional constraints (as opposed to explicit cage manipulation), the resulting deformations exhibit possibly undesired local scaling since restricted to angle-preserving deformations (i.e., with arbitrarily large or small stretch), which is a typical pitfall that makes unconstrained Green-based variational methods difficult to control in practice. It may be possible to obtain guarantees on the bounds of the resulting deformation, by extending the optimization framework of Weber and Gotsman [2010] to the case of polynomial input/deformed curves. Now let us demonstrate the use of our harmonic and biharmonic coordinates for other energies.

7.2 Variational as-rigid/similar-as-possible deformations

Among the various deformation energies that are ubiquitous in Computer Graphics, as-rigid-as-possible (ARAP) and as-similar-as-possible (ASAP) energies are very popular choices for variational shape deformation [Ben-Chen et al. 2009; Weber et al. 2012]. They measure how much a given spatial deformation function differs in the least-squares sense from locally-isometric and locally-conformal respectively. Inspired by the work of Ben-Chen and colleagues [Ben-Chen et al. 2009], we present results of variational deformations for the two following standard ARAP and ASAP energies:

$$\mathcal{E}^{\texttt{Har/Bihar}}_{\texttt{ARAP/ASAP}}(f) := \epsilon_f \sum_{p_i \in C_{\mathcal{P}}} \|f(p_i) - \tilde{p}_i\|^2 \tag{49}$$

$$+ \epsilon_g \sum_{p_j \in C_{\mathcal{J}}} \|\nabla^T f(p_j) - \tilde{J}_j\|^2 + \epsilon_h \sum_{p_h \in C_{\mathcal{H}}} \|Hf(p_j)\|^2$$

s.t. $\{\tilde{J}_j\}$ are rotations (ARAP) or similarities (ASAP),

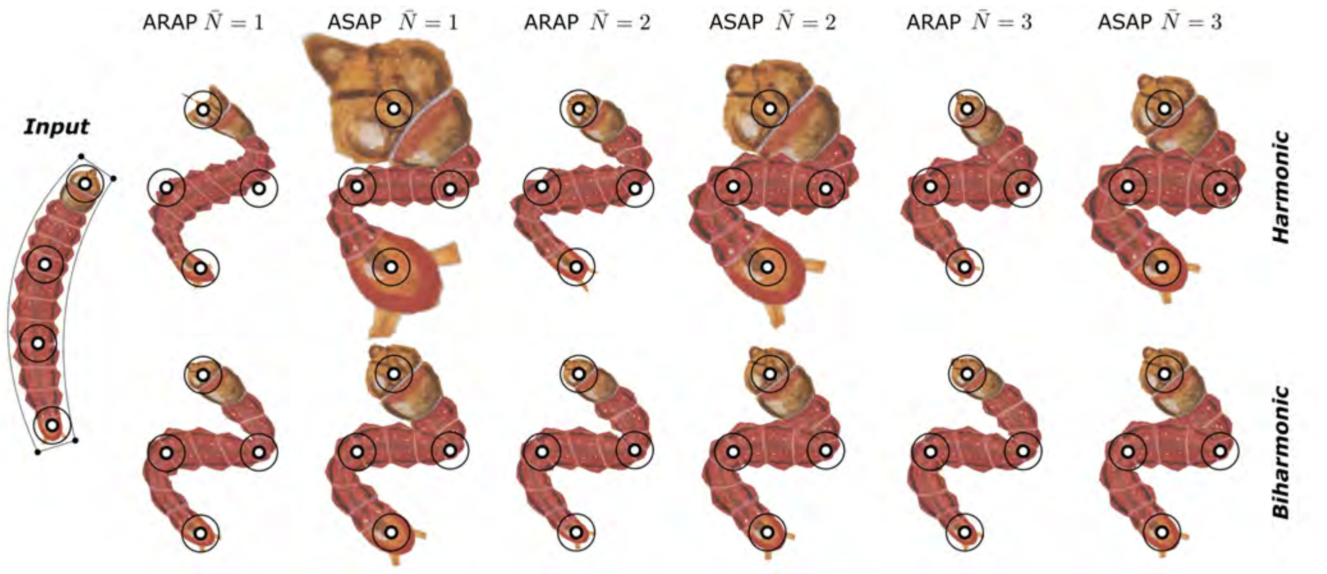


Fig. 6. Comparison of various settings of our variational deformation solver. Energy settings for ARAP: $(\epsilon_f, \epsilon_g, \epsilon_h) : (10^4, 10^2, 10^{-2})$. Energy settings for ASAP: $(\epsilon_f, \epsilon_g, \epsilon_h) : (10^4, 10^2, 10^{-4})$. $\bar{N}$ denotes the degree of the two long curves *in the deformation space* (the other two curves are kept as segments).

where $C_{\mathcal{P}}$ sample point constraints with target $\{\tilde{p}_i\}$, $C_{\mathcal{J}}$ sample Jacobian constraints with target $\{\tilde{J}_j\}$, and $C_{\mathcal{H}}$ sample Hessian penalties, using either an **harmonic** (Eq. (27)) or **biharmonic** (Eq. (36)) coordinate subspace.

Typically, the Jacobian targets $\{\tilde{J}_j\}$ are unknown rotation/similarity matrices, and the energy is minimized using a standard local/global iterative solver [Sorkine and Alexa 2007]:

- *In the local step*, the primary variables (here, the cage deformation parameters) are kept fixed, $\nabla^T f$ is evaluated at every point of $C_{\mathcal{J}}$, and the unknown $\{\tilde{J}_j\}$ are updated as closest rotations/similarities to $\{\nabla^T f(p_j)\}$;
- *In the global step*, the auxiliary variables $\{\tilde{J}_j\}$ are kept fixed, and the primary variables (here, the cage deformation parameters) are optimized by minimizing Eq. (49), which amounts to solving a linear system (Eq. (47)).

We compare the use of harmonic and biharmonic deformation subspaces for ARAP/ASAP deformations in Fig. 6. Using segments only ($N = 1$) does not provide a rich enough deformation space for the positional constraints to be met without sacrificing quality (though biharmonic deformations perform better than harmonic deformations, which result in the head being squashed). Increasing the degree of the deformation curves allows improving the quality quickly. Note that the energy settings for harmonic and biharmonic deformations differ slightly. We found that both types of deformation subspaces require different energy penalty settings to provide overall similar global behavior. Our understanding is that the baked-in regularity priors of the coordinates play an important role, as biharmonic diffused functions allow modeling stronger local deformation oscillations than harmonic diffused functions for example. While this prevents rigorous comparison between the two deformation subspaces, we set up in this example the energy parameters to allow for close-to-exact fitting of the positional constraints while best respecting the local Jacobian constraints each time.

We also present results of variational (harmonic ARAP) deformations in Fig. 7, in which we compare the use of curved cages with the use of straight ones. While it may not be always necessary to use polynomial curves of high degree to properly enclose a given shape to deform, we observe that the geometric structure of the cage impacts strongly the resulting deformations, and that much smoother deformations can be obtained by using smooth deformation curves as underlying subspace for variational deformations. We believe that both resulting deformations are acceptable, and that our technique ultimately offers alternative DoFs for the artist to control the deformation style. More examples can be found in Fig. 12.

7.3 Boundary-aligned deformations

We investigate in this section how to adapt the "Thickness-preserving" energies presented in [Weber et al. 2012] in 2D or [Thiery et al. 2024] in 3D. In those works on biharmonic cage-based deformations, the user prescribes cage vertex positions (i.e., the Dirichlet condition), and all other 3 boundary conditions for their cage-based biharmonic deformations are optimized to result in minimization of deformation energies sampled at points on the boundary of the cage.

We aim at obtaining the type of deformation behavior they offer in a purely variational context: the user merely prescribes positional constraints instead of setting up the deformed cage by hand (i.e., the Dirichlet boundary condition). To do so, we present novel deformation energies encouraging boundary-aligned transformations, and we adapt the previously-introduced local/global solver for minimizing those. Note that our goal is not as much to showcase "novel variational methods" as to provide an enriched context for a more detailed comparison of harmonic vs biharmonic subspaces.

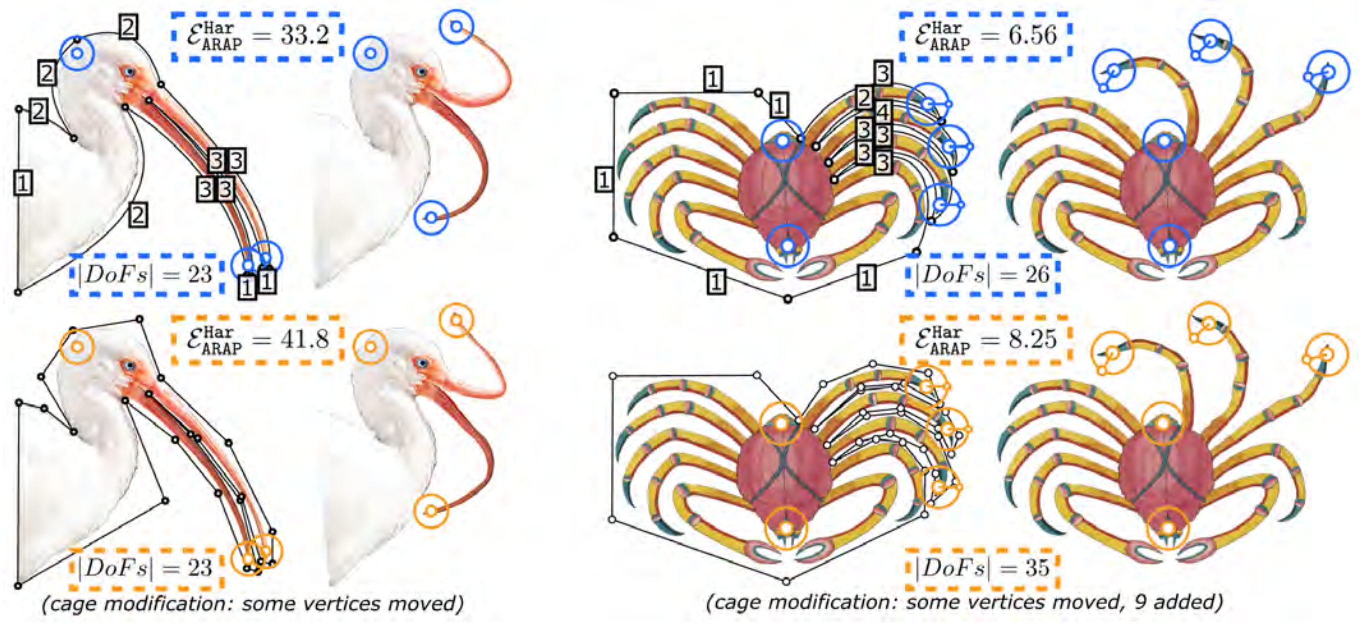

$\mathcal{E}^{\mathrm{Har}}_{\mathrm{ARAP}} = 33.2$
$|DoFs| = 23$
$\mathcal{E}^{\mathrm{Har}}_{\mathrm{ARAP}} = 41.8$
$|DoFs| = 23$
(cage modification: some vertices moved)
$\mathcal{E}^{\mathrm{Har}}_{\mathrm{ARAP}} = 6.56$
$|DoFs| = 26$
$\mathcal{E}^{\mathrm{Har}}_{\mathrm{ARAP}} = 8.25$
$|DoFs| = 35$
(cage modification: some vertices moved, 9 added)

Fig. 7. Our variational solver is compatible with curved rest cages. Using a curved cage (top row, we specify in the left figure the degrees of the various curves), the solver introduces less breaks in curvature than when using a straight cage, as displayed in the bottom row for comparison. We had to modify slightly the polygonal cages to make them valid: we moved a few vertices in the first example (ibis) to enclose the image, and we added a few vertices in the second example (crab) to enclose the image while avoiding self-intersections of the cage. To further quantify the expressiveness power of the subspace provided by curved cages, we provide inside the figure the ARAP energy ($\mathcal{E}^{\mathrm{Har}}_{\mathrm{ARAP}}$, Eq. (49)) reached by the two different solvers.

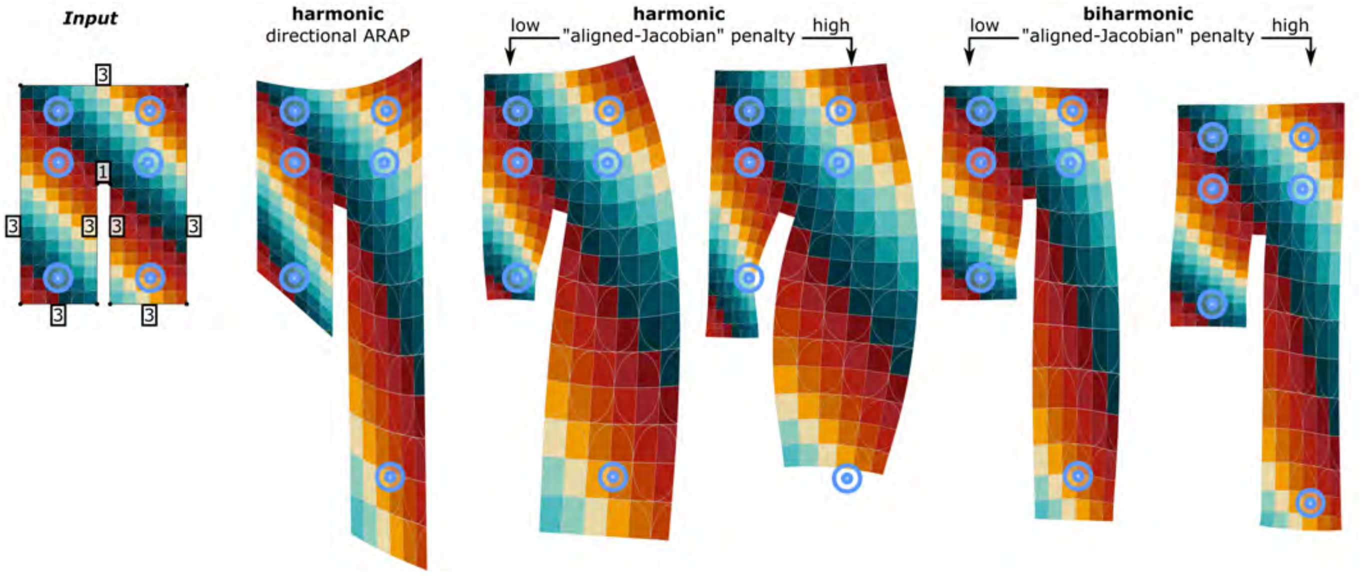


Fig. 8. We showcase our boundary-aligned deformations on a simple example. All curves in the rest pose cage are segments (degree 1), and their deformable counterparts are of degree 3 except for one small segment that remains of degree 1 in the deformation state (we show the degree near each curve in the input figure). Columns 2 to 6 showcase variational deformations obtained from 6 positional constraints. Column 2 shows a simple attempt at localizing rigid stretch along the boundary normal directions, and motivates the use of a more complex strategy. Columns 3 and 4 show our results for harmonic deformations (for 2 different energy weights settings) and columns 5 and 6 show our results for biharmonic deformations (same energy settings as in columns 3 and 4).

Illustration of the complexity with a first simple attempt. As ARAP methods allow for efficient variational deformations, a natural idea is to restrain the rigidity constraint to the normal direction and let the tangent direction unconstrained. This can be done by sampling rigidity constraints on $\partial\Omega$ ("near $\partial\Omega$" to be precise, as our coordinates require otherwise special case computations on $\partial\Omega$, as pointed out in [Thiery et al. 2024] in 3D), and minimizing

$$\mathcal{E} := \epsilon_f \sum_{p_i \in C_{\mathcal{P}}} \|f(p_i) - \tilde{p}_i\|^2 + \epsilon_g \sum_{(p_j, n_j) \in C_{\mathcal{J}}} \|\partial_{n_j} f(p_j) - \tilde{n}_j\|^2 + \epsilon_h \sum_{(p_h, n) \in C_{\mathcal{H}}} \|\partial_n^2 f(p_j)\|^2 \text{ , s.t. } \{\tilde{n}_j\} \text{ are unit norm,}$$

therefore encouraging the transformation to be local rigid in the normal direction (constraining at boundary sample p_j with rest-pose normal n_j the normal derivative to remain unit-norm through the use of auxiliary variables $\tilde{n}_j$ describing the transformed normal). An illustration of this approach is given in Fig. 8, column 2. As one can see, while the constrained stretch along the normal direction results indeed in a local "preserved thickness", the resulting deformations appear unnatural, as transformed tangents and normals do not remain orthogonal along the cage boundary.

In order to obtain deformations "sliding along the boundary" of the cage such as the ones presented by Weber et al.[2012] (branded "thickness preserving"), we present a simple strategy.

7.3.1 Boundary-aligned constrained stretch. To model transformations allowing for arbitrary stretch but preserving the local angles between transformed normals and transformed tangents, we constrain the Jacobian to have boundary-aligned stretches. At a given boundary point p with rest-pose normal n, we aim at obtaining a deformation Jacobian exhibitting the following decomposition:

$$J = U \Sigma B_n^T \tag{50}$$

$B_n := (n, n^\perp) \in SO_2$, Σ being a diagonal matrix describing the local stretch along rest-pose boundary and tangent respectively, and $U \in SO_2$ describing the rotation post-stretch of the transformation. This constraint is equivalent to requiring the singular value decomposition of J to have normal/tangent stretch directions.

We denote the space of such matrices following Eq. (50) as S_n:

$$\mathsf{S}_n := \left\{ M = U\Sigma B_n^T, U \in SO_2, \Sigma = \left(\begin{array}{c|c} \alpha & 0 \\ 0 & \beta \end{array} \right) \right\} \subset \mathbb{R}^{2\times 2}. \tag{51}$$

We further denote the subspace of matrices in S_n verifying $\Sigma(0,0) = 1$ (i.e., the normal stretch is additionally constrained to 1) as S_n^1:

$$\mathsf{S}_n^1 := \left\{ M = U\Sigma B_n^T, U \in SO_2, \Sigma = \left(\begin{array}{c|c} 1 & 0 \\ 0 & \beta \end{array} \right) \right\} \subset \mathsf{S}_n. \tag{52}$$

Minimization using a global/local solver. While constraining the Jacobian to exhibit this structure is complex and requires non-linear optimization, we adapt the common local/global solver used for ARAP/ASAP transformations to obtain a simple iterative procedure.

To do so, given an input Jacobian matrix J, we **simply replace the local step of ARAP/ASAP optimizations** (see Sec. 7.2) with a novel projection procedure onto S_n (or S_n^1), and optimize for

$$\bar{J} = \underset{j \in \mathsf{S}_n / \mathsf{S}_n^1}{\operatorname{argmin}} \|J - j\|_F^2 \tag{53}$$

We detail our projection procedures in Appendix B and C for clarity, and focus in this section on showcasing their use for boundary-aligned stretchable cage-based deformations.

Equipped with this projection procedure, we minimize the energy

$$\mathcal{E}^{\texttt{Har/Bihar}}_{\texttt{Aligned}\partial}(f) := \epsilon_f \sum_{p_i \in C_{\mathcal{P}}} \|f(p_i) - \tilde{p}_i\|^2 + \epsilon_g \sum_{(p_j, n_j) \in C_{\mathcal{J}}} \| \nabla^T f(p_j) - \tilde{J}_j\|^2 + \epsilon_h \sum_{p_h \in C_{\mathcal{H}}} \|Hf(p_j)\|^2 \tag{54}$$
$$\text{s.t. } \{\tilde{J}_j \in \mathsf{S}_{n_j} / \mathsf{S}^1_{n_j}\} \text{ are transformations aligned onto } n_j,$$

which minimizers showcase deformed shapes that naturally slide along the cage boundary.

Fig. 8 (columns 3 to 6) show results of our approach when using a harmonic (columns 3 and 4) and a biharmonic (columns 5 and 6) subspace, using a free normal stretch (i.e., we use S_n as local projection space). While harmonic deformations provide already interesting results, we see that increasing the "aligned Jacobian" penalty weight results in positional constraints that become more difficult to meet. Biharmonic deformations allow for much more intuitive boundary-aligned deformations while fitting much better the positional constraints (the same penalty weights are used to compare the use of both subspaces). As noted in [Thiery et al. 2024], there is a fundamental reason behind this: it is not just that biharmonic subspaces "contain more degrees of freedom" than harmonic deformations; biharmonic deformations allow simply for more complex fitting of mixed boundary conditions than harmonic deformations. Harmonic deformations are mathematically uniquely defined by the Dirichlet condition alone (akin to positional constraints), while biharmonic deformations require mathematically the setting of both the Dirichlet (akin to positional constraints) and the Neumann (akin to derivative constraints that we set using our novel boundary-aligned Jacobian constraints) boundary conditions.

We compare the use of our "free normal stretch" (projecting onto S_n) and "rigid normal stretch" (projecting onto S_n^1) variants in Fig. 9. As one can see, in such a constrained setup, biharmonic deformations provide results that better fit the constraints (either alignment or positional constraints) than harmonic deformations.

ASAP vs Green-based deformations. One can compare conceptually ASAP (enforcing angle-preservation *in the least-squares sense only*) and Green-based (strictly angle-preserving, Sec. 7.1) deformations by noting that enforcing soft instead of hard constraints may provide extended flexibility to the user. For example, one can mix "local ARAP constraints", "local ASAP constraints" (with possible local bounds), and "local boundary-aligned Jacobian constraints", as all those are similarly implemented in a local/global solver. This would be impossible with strictly angle-preserving deformations that, by design, disallow stretch along preferred directions.

8 Discussion

Comparison to cage-straightening approaches. Other works, e.g. [Li et al. 2013; Lin and Chen 2024], allow using input cages that are curved. However, those do not diffuse the deformation function from the input curved cages, but instead *straighten* them into polygonal

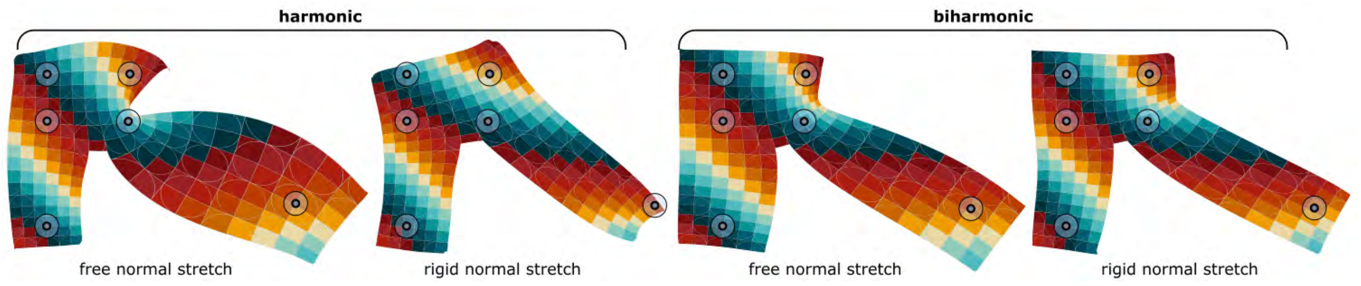


Fig. 9. Out of our two variants, biharmonic deformations provide results that fit better the constraints (either alignment or positional constraints) than harmonic deformations. The input configuration is the one given in Fig. 8.

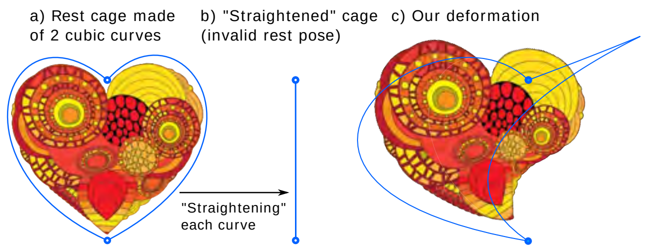

a) Rest cage made of 2 cubic curves
b) "Straightened" cage (invalid rest pose)
c) Our deformation
"Straightening" each curve

Fig. 10. Contrary to [Li et al. 2013; Lin and Chen 2024], our approach does not require *straightening* the cage before encoding. Our approach works therefore on any input curved cage that does not self-intersect.

cages, and consider this straightened state as intermediate structure for encoding. This strategy has several drawbacks:

(1) Some input cages simply cannot be straightened, demonstrating that this approach is not a general solution. Consider the simple case of a curved cage made of two curves: those cages degenerate to two segments with null interior (see Fig. 10).
(2) Straightening the curved cages has to be done without creating unwanted intersections, as Mean-Value (for [Li et al. 2013]) or Green ([Lin and Chen 2024]) and biharmonic coordinates (based on Green's identities) require cages to be the boundary of a proper Euclidean volume: no self intersection can occur, or the coordinates cannot be defined. This poses specific challenges for complex input configurations.

On the contrary, as long as our input curved cages are free of self-intersection, our coordinates and their derivatives can be defined everywhere inside the curved cage. To the best of our knowledge, our approach is the only one allowing for variational deformations stemming from the use of (non-intersecting) polynomial 2D cages.

Automatic curved cages. The automatic or semi-automatic construction of curved cages allowing for the deformation of arbitrary 2D graphics is a challenging task. In this paragraph, we describe a simple and practical procedure to produce such cages, by constructing a very loose envelope approximated with a closed Bezier spline. Given an input image, we convert it to a binary raster mask, where foreground pixels are labeled 1 (white) and background pixels are labeled 0 (black). Next, the foreground region is morphologically dilated by n_d pixels (in our experiments, n_d is typically 20-50 for images of about 2000×2000 pixels). The polygonal outer boundary of the dilated region, which we assume is a single connected component by this point, is then extracted by tracing pixels. This loop is broken at an arbitrary point to create an open polyline. Finally, a piecewise cubic Bezier spline is fitted to this polyline using a variant of the algorithm of Schneider [1990], with modifications

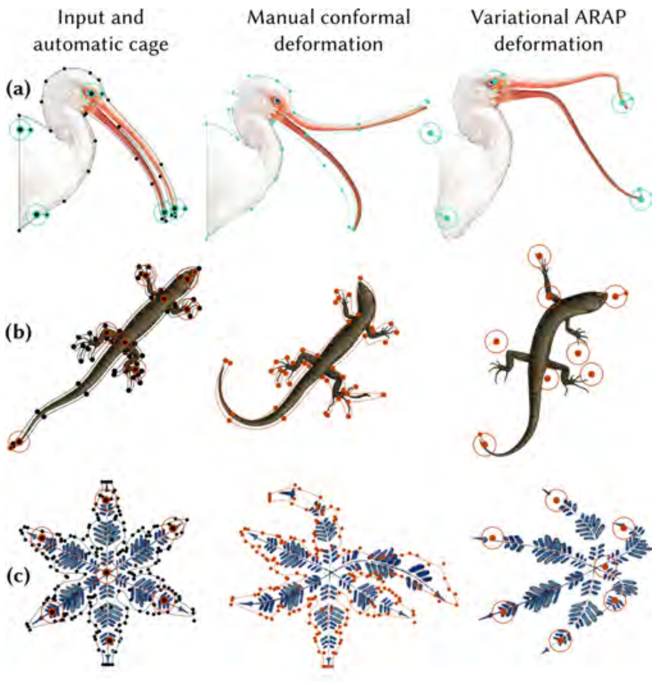


Fig. 11. Our "autocage" approach generates rest cages made of curved arcs that accurately fit the input image while generating a number of control points small enough to be manually handled. It can also be used for variational deformation, where the low number of control points enables interactive manipulation. Example **(c)** shows that we can handle input shapes with challenging geometry and topology like this snowflake, without leading to an excessive number of control points – though we are not able to obtain an artist-made cage such as in Fig. 12 with our method. A future direction to improve our "autocage" approach would be to try preserving symmetries of the original input, which is not currently the case.

to preserve very sharp corners. Since sparse, smooth cages are desired and not tight envelopes, the Bezier fitting tolerance is set to a very high value (typically 0.3-0.8n_d) in order to smoothly approximate the polygon with a very small number of control points. This process certainly does not guarantee that the result will be free of self-intersections, and providing a real-time cage design method with guaranteed validity is an interesting future work direction.

Fig. 11 shows results of our approach on three different examples. While defining cages for shape deformation is fundamentally ill-posed and depends on the type of deformations intended by the artist, one can use those cages as a starting point for their creation. Note that, while manipulating cages with many control points may be tedious (compare our autocage on the Snowflake example, vs our artists-made cage for the same input in Fig. 12), our variational approach allows deforming the cage's content by specifying few positional/orientation constraints, circumventing this issue.

Limitations and Future work. Our work has several limitations. The first obvious limitation is that we require the artist to provide the rest cage. While we mitigate this issue with our "autocage" approach, it remains an open challenge to provide efficient controls for the cage creation while obtaining guarantees on, e.g., the topology of the resulting interior. This opens up new exciting research avenues to Bezier contour optimization and Vector Graphics approximation in general. Based on our experiments in this area, we envision that an intuitive method providing efficient control over (i) the local level of detail of the cage, (ii) the local range of the curve degrees, (iii) the local distance to the vector graphics to appximate/deform and (iv) the geometric regularity of the control structure in terms of parameterization stretch and range of curve curvature would find several interesting applications beyound cage-based deformations.

While our method allows computing in theory arbitrary integrals of the form $F^c_{K,n}$ (Eq. 39) and coordinates of any order, we observed in practice that double floating point precision was necessary to compute those for large values of the parameters, limiting in practice their computation beyond a certain point before having to rely on multi-precision computation libraries. One could argue the limited usefulness of curves of very large degree (above 5) in everyday-life practical scenarios, but we believe that improving the computation of our coordinates is an interesting theoretical work.

In addition, we believe that speeding-up our computation process can be achieved by exploiting the spatial coherency and smoothness of the problem at hand. For example, while we compute roots of $\|c(t) - \eta\|^2$ by using an iterative decomposition of the companion matrix from scratch for every different point η, we know that the roots exhibit a smooth and controlled geometric structure for smoothly-varying η positions. A possible strategy could be to warm start the iterative solvers using solutions found for neighboring pixels/positions, paving the road for an efficient bottom-up multiresolution computation strategy. Possible accelerations could be obtained similarly by exploring different root-finding strategies, exploiting again the fact that the roots for neighboring pixels are a controllable approximation of the solution, since we can relate search regions for the roots to simple geometric measures on the polynomials, such as leading coefficients [Hirst and Macey 1997]. Implementing our coordinates on the GPU would allow speeding up their computations. While we believe that the strategy we exposed for roots-finding would allow for coarse-to-fine GPU implementation, *conservative predictive memory bounding* for our iterative rational fraction derivation (Algo. 2) would be necessary as well, as dynamic memory allocations are not permitted on the GPU.

Finally, extending our technique to computing other types of coordinates is of interest. For example, computing close form expressions for k-harmonic functions ($k > 2$) using polynomial cages seems feasible, since one can always fall back on the computation of our $F^c_{K,n}$ using integration by parts. Deriving expressions for each case is still a tedious mathematical problem, and finding an elegant formulation for arbitrary k-harmonic functions would maybe find applications beyond Computer Graphics. Interestingly, finding Mean-Value Coordinates for arbitrary polynomial curves cannot however be done simply by extending our approach though, as the MVC kernel is an odd-degree kernel of the form $1/P_c(t)^{3/2}$, breaking our assumptions that it can be decomposed into a rational fraction with simple elements. We believe that extending Mean-Value Coordinates for arbitrary polynomial curves both in the rest and the deformed state, by extending either the formulation of Li and colleagues [2013] or the more standard formulation (not involving gradient interpolation) of Floater [2003], is an interesting and challenging future work.

Conclusion. We provide in this work closed-form expressions and a practical computation method for Green (harmonic) and biharmonic coordinates along with their gradients and Hessians, for input rest cages made of polynomial curves of arbitrary order that can be curved. The usage of closed-form expressions for the derivatives permits the implementation of various variational methods, allowing artists to deform shapes either by editing the cage curves directly or by editing few control points of the deformed shape directly, which facilitates the use of cage-based deformations on complex inputs.

References

Seungbae Bang, Kirill Serkh, Oded Stein, and Alec Jacobson. 2023. An Adaptive Fast-Multipole-Accelerated Hybrid Boundary Integral Equation Method for Accurate Diffusion Curves. *ACM Trans. Graph.* 42, 6, Article 215 (Dec. 2023), 28 pages. doi:10.1145/3618374

Rick Beatson, Michael S. Floater, and Carl Emil Kåshagen. 2018. Hermite mean value interpolation on polygons. *Comput. Aided Geom. Des.* 60 (2018), 18–27. doi:10.1016/j.cagd.2018.01.002

Mirela Ben-Chen, Ofir Weber, and Craig Gotsman. 2009. Variational harmonic maps for space deformation. *ACM Transactions on Graphics (TOG)* 28, 3 (2009), 1–11.

Renjie Chen and Ofir Weber. 2015. Bounded distortion harmonic mappings in the plane. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–12.

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

Fernando De Goes and Doug L James. 2017. Regularized kelvinlets: sculpting brushes based on fundamental solutions of elasticity. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–11.

Alan Edelman and H Murakami. 1995. Polynomial roots from companion matrix eigenvalues. *Math. Comp.* 64, 210 (1995), 763–776.

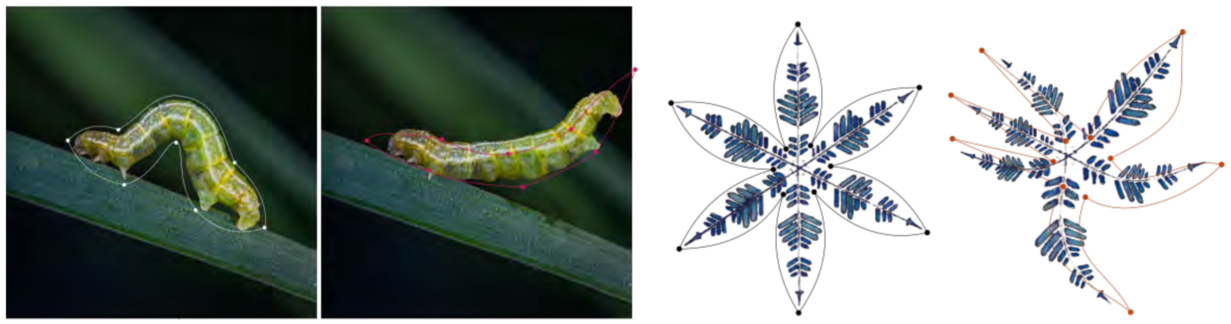
Michael S Floater. 2003. Mean value coordinates. *Computer aided geometric design* 20, 1 (2003), 19–27.

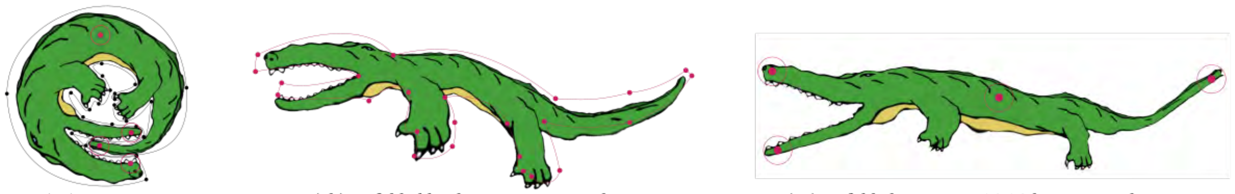
Laurent Fousse, Guillaume Hanrot, Vincent Lefèvre, Patrick Pélissier, and Paul Zimmermann. 2007. MPFR: A multiple-precision binary floating-point library with correct rounding. *ACM Trans. Math. Softw.* 33, 2 (June 2007), 13–es. doi:10.1145/1236463.1236468

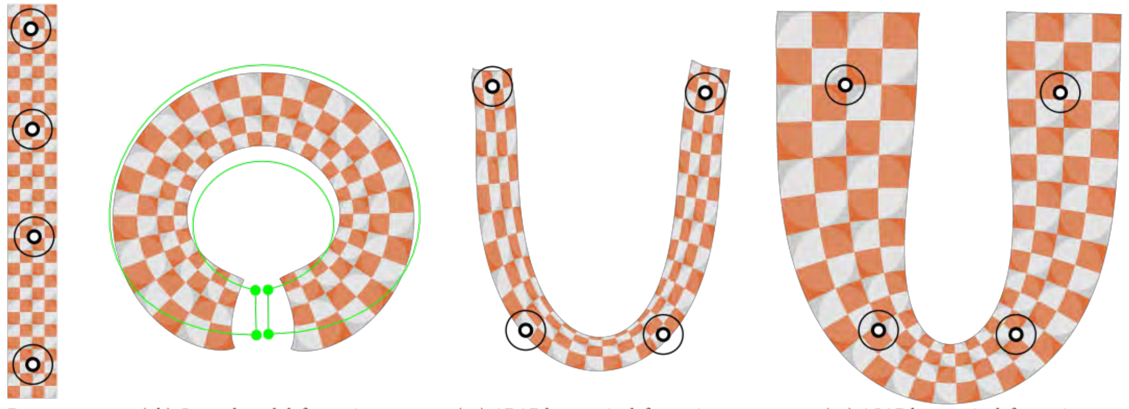
Gaël Guennebaud, Benoît Jacob, et al. 2010. Eigen v3. http://eigen.tuxfamily.org.

Holly P Hirst and Wade T Macey. 1997. Bounding the roots of polynomials. *The College Mathematics Journal* 28, 4 (1997), 292–295.

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


(1a) Rest pose. (1b) Deformed pose. (2a) Rest pose. (2b) Deformed pose.


(3a) Rest pose. (3b) Unfolded by direct cage manipulation. (3c) Unfolded using our ARAP harmonic solver.


(4a) Rest pose. (4b) Green-based deformation. Degrees 2, 4, 2, 4. (4c) ARAP harmonic deformation. (4c) ASAP harmonic deformation.

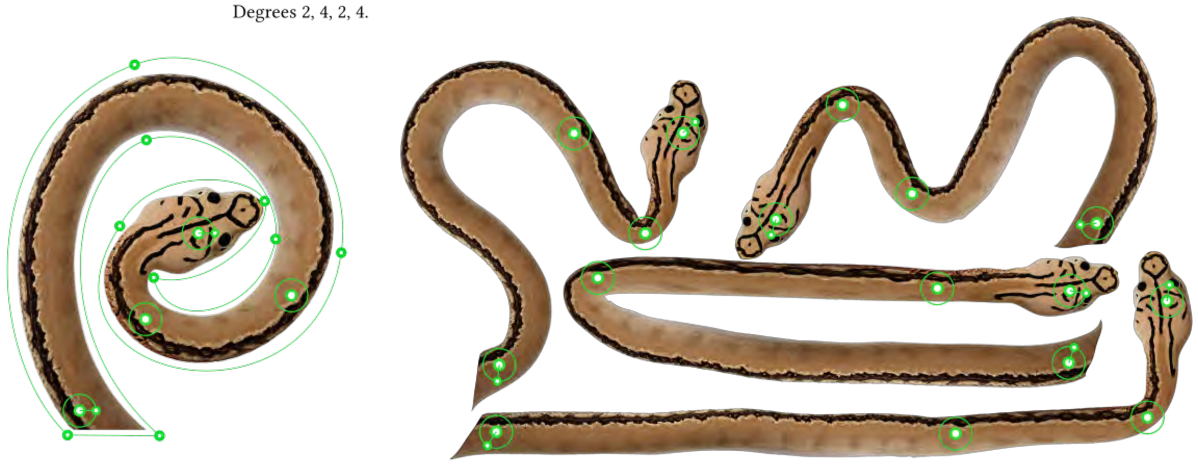

(5a) Rest pose. (5b) Biharmonic ARAP deformations.

Fig. 12. Extra results. In examples (1) and (2) the user directly manipulates the cage control points to deform the bound shape. In example (3), we compare this direct approach (3b) with a variational solver (3c). In example (4), we show Green-based deformations obtained using direct manipulation of curves of degree 2 to 4, and show ARAP and ASAP deformations for positional constraints outside the reach of isometric deformations. In example (5), we provide additional results of biharmonic ARAP deformations, using curves of degree 2 (the rest pose contains curves of degree ranging from 1 to 4).

Alec Jacobson, Ilya Baran, Jovan Popovic, and Olga Sorkine. 2011. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.* 30, 4 (2011), 78. doi:10.1145/2010324.1964973

Alec Jacobson, Zhigang Deng, Ladislav Kavan, and John P Lewis. 2014. Skinning: Real-time shape deformation (full text not available). In *ACM SIGGRAPH 2014 Courses.* 1–1.

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. In *ACM Siggraph 2005 Papers.* 561–566.

Torsten Langer, Alexander Belyaev, and Hans-Peter Seidel. 2006. Spherical barycentric coordinates. In *Symposium on Geometry Processing.* 81–88.

Torsten Langer and Hans-Peter Seidel. 2008. Higher Order Barycentric Coordinates. *Comput. Graph. Forum* 27, 2 (2008), 459–466. doi:10.1111/j.1467-8659.2008.01143.x

Xian-Ying Li, Tao Ju, and Shi-Min Hu. 2013. Cubic Mean Value Coordinates. *ACM Transactions on Graphics* 32, 4 (2013), 126:1–10.

Zhehui Lin and Renjie Chen. 2024. Polynomial Cauchy Coordinates for Curved Cages. In *SIGGRAPH Asia 2024 Conference Papers.* 1–8.

Yaron Lipman, Johannes Kopf, Daniel Cohen-Or, and David Levin. 2007. GPU-assisted positive mean value coordinates for mesh deformations. In *Symposium on geometry processing.*

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

Shibo Liu, Ligang Liu, and Xiao-Ming Fu. 2024. Polynomial 2D Green Coordinates for High-order Cages. *arXiv preprint arXiv:2408.06831* (2024).

Josiah Manson and Scott Schaefer. 2010. Moving Least Squares Coordinates. *Comput. Graph. Forum* 29, 5 (2010), 1517–1524. doi:10.1111/j.1467-8659.2010.01760.x

Élie Michel and Jean-Marc Thiery. 2023. Polynomial 2D Green Coordinates for Polygonal Cages. In *SIGGRAPH '23 Conference Proceedings.* ACM, 16–1–9. doi:10.1145/3588432.3591499 preprint: https://portfolio.exppad.com/documents/2023__Michel__PolynomialGreenCoords.pdf.

Philip J. Schneider. 1990. An Algorithm for Automatically Fitting Digitized Curves. In *Graphics Gems*, Andrew Glassner (Ed.). 612–626.

Thomas W Sederberg and Scott R Parry. 1986. Free-form deformation of solid geometric models. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques.* 151–160.

J. Smith and Scott Schaefer. 2015. Selective Degree Elevation for Multi-Sided Bézier Patches. *Comput. Graph. Forum* 34, 2 (2015), 609–615. doi:10.1111/cgf.12588

Olga Sorkine and Marc Alexa. 2007. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, Vol. 4. 109–116.

Daniel Ströter, Jean-Marc Thiery, Kai Hormann, Jiong Chen, Qingjun Chang, Sebastian Besler, Johannes Sebastian Mueller-Roemer, Tamy Boubekeur, André Stork, and Dieter W Fellner. 2024. A Survey on Cage-based Deformation of 3D Models. In *Computer Graphics Forum*, Vol. 43. Wiley Online Library, e15060.

Jean-Marc Thiery and Tamy Boubekeur. 2022. Green Coordinates for Triquad Cages in 3D. In *SIGGRAPH Asia 2022 Conference Papers.* Article 38, 8 pages. doi:10.1145/3550469.3555400

Jean-Marc Thiery, Pooran Memari, and Tamy Boubekeur. 2018. Mean value coordinates for quad cages in 3D. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–14.

Jean-Marc Thiery, Élie Michel, and Jiong Chen. 2024. Biharmonic Coordinates and their Derivatives for Triangular 3D Cages. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–17.

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 and Craig Gotsman. 2010. Controllable conformal maps for shape deformation and interpolation. In *ACM SIGGRAPH 2010 papers.* 1–11.

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

A Detailed derivation of our coordinates and their derivatives

We provide here a fully detailed derivation of our coordinates and their gradients and Hessians.

A.1 Green coordinates for polynomial curves

We build upon the formalism of Michel and Thiery [2023], and first derive Green coordinates for polynomial curves, effectively extending their formulation by allowing rest-pose cages to be made of non-straight polynomial curves.

A.1.1 Dirichlet term. Eq. (10), $\phi_k^c(\eta)$ can be directly rewritten as

$$\boxed{\phi_k^c(\eta) = \frac{1}{2\pi} F_1^c \left[X^k c_\eta \cdot c'^\perp \right].} \tag{55}$$

Derivatives. We easily differentiate $\phi_k^c(\eta)$ using (DIF):

$$\boxed{\nabla_\eta(\phi_k^c)(\eta) = -\frac{1}{2\pi} F_1^c \left[X^k c'^\perp \right] + \frac{1}{\pi} F_2^c \left[X^k \left(c_\eta \cdot c'^\perp \right) c_\eta \right].} \tag{56}$$

We repeat this process once again to express its Hessian:

$$\boxed{\begin{aligned} H_\eta(\phi_k^c)(\eta) = & -\frac{1}{\pi} F_2^c \left[X^k \left(c'^\perp c_\eta^T + c_\eta c'^{\perp T} + c_\eta \cdot c'^\perp I_2 \right) \right] \\ & + \frac{4}{\pi} F_3^c \left[X^k \left(c_\eta \cdot c'^\perp \right) c_\eta c_\eta^T \right] \end{aligned}} \tag{57}$$

A.1.2 Neumann term. We recall that $c_\eta := c(t) - \eta$, leading to $\nabla_\eta^T c_\eta = -I_2$, $\nabla_\eta(\|c_\eta\|^2) = -2c_\eta$ and $H_\eta(\|c_\eta\|^2) = 2I_2$. Using integration by parts ($\int_a^b u'v = [uv]_a^b - \int_a^b uv'$, with $u' = \bar{c}', v = \log(\|c_\eta\|)$ and $u = \bar{c}, v' = c'_\eta / \|c_\eta\|^2$), the Neumann term can be rewritten as

$$\begin{aligned} f_N^c(\eta) &= \frac{-1}{2\pi} \int_{t=0}^{1} \log(\|c_\eta(t)\|) \bar{c}'(t)^\perp dt \\ &= \frac{-1}{2\pi} \Big[\log\left(\|c_\eta(t)\|\right) \bar{c}(t)^\perp \Big]_{t=0}^{1} + \int_{t=0}^{1} \frac{c_\eta(t) \cdot c'(t)}{2\pi \|c_\eta(t)\|^2} \bar{c}(t)^\perp dt \\ &= \frac{-1}{2\pi} \log(\|c_\eta(1)\|) \sum_k \bar{c}_k^\perp + \frac{1}{2\pi} \log(\|c_\eta(0)\|) \bar{c}_0^\perp \\ &\quad + \sum_k \int_{t=0}^{1} \frac{t^k c_\eta(t) \cdot c'(t)}{2\pi \|c_\eta(t)\|^2} dt \bar{c}_k^\perp \\ &= \sum_{k>0} \underbrace{\frac{1}{2\pi} \left(-\log(\|c_\eta(1)\|) + F_1^c \left[X^k c_\eta \cdot c' \right] \right)}_{:= \psi_k^c(\eta)} \bar{c}_k^\perp \end{aligned}$$

leading to

$$\boxed{\psi_k^c(\eta) = \frac{1}{2\pi} \left(-\log(\|c_\eta(1)\|) + F_1^c \left[X^k c_\eta \cdot c' \right] \right).} \tag{58}$$

As noted by Michel and Thiery [2023], $\psi_0^c(\eta) = 0$, which is trivial to see beforehand since $\bar{c}_0$ does not appear in the expression of $\bar{c}'(t)$.

Derivatives. The gradient of $\psi_k^c(\eta)$ is more easily obtained by differentiating directly $f_N^c(\eta)$ with respect to η:

$$\nabla_\eta^T f_N^c(\eta) = \int_{t=0}^{1} \frac{\bar{c}'(t)^\perp c_\eta(t)^T}{2\pi \|c_\eta(t)\|^2} dt = \sum_{k>0} \bar{c}_k^\perp \Bigg(\underbrace{\frac{k}{2\pi} F_1^c \left[X^{k-1} c_\eta \right]}_{=: \nabla_\eta \psi_k^c(\eta)} \Bigg)^T$$

leading to

$$\boxed{\nabla_\eta \psi_k^c(\eta) = \frac{k}{2\pi} F_1^c \left[X^{k-1} c_\eta \right].} \tag{59}$$

Differentiating $\nabla_\eta \psi_k^c(\eta)$ using (DIF), we obtain

$$\boxed{H_\eta \psi_k^c(\eta) = \frac{-k}{2\pi} F_1^c \left[X^{k-1} \right] I_2 + \frac{k}{\pi} F_2^c \left[X^{k-1} c_\eta c_\eta^T \right].} \tag{60}$$

A.2 Biharmonic coordinates for polynomial curves

A.2.1 Third-order condition. We recall that we model the third-order boundary condition using a polynomial curve $\bar{A}_c(t) = \sum_k t^k \bar{A}_c^k$, and define the contribution of polynomial curve c to the third order boundary condition's diffusion as:

$$\begin{aligned} f_b^c(\eta) &:= \int_{t=0}^{1} \bar{A}_c(t)\, \nabla_1 g(c(t),\eta)\cdot c'(t)^{\perp} dt \qquad (61)\\ &= \int_{t=0}^{1} \bar{A}_c(t) \frac{c_\eta(t)\cdot c'(t)^{\perp}}{8\pi}\left(2\log(\|c_\eta(t)\|)-1\right)dt \\ &= \sum_k \bar{A}_c^k \underbrace{\int_{t=0}^{1} \frac{t^k c_\eta(t)\cdot c'(t)^{\perp}}{8\pi}\left(2\log(\|c_\eta(t)\|)-1\right)dt}_{=:\,\Phi_k^c(\eta)} \end{aligned}$$

which thus introduces our first biharmonic coordinate $\Phi_k^c(\eta)$. Differentiating it w.r.t. η, we obtain

$$\nabla_\eta\, \Phi_k^c(\eta) = \int_{t=0}^{1} \frac{t^k c'(t)^{\perp}}{8\pi}\left(1-2\log(\|c_\eta(t)\|)\right)dt$$
$$-\int_{t=0}^{1} \frac{t^k c_\eta(t)\cdot c'(t)^{\perp} c_\eta(t)}{4\pi\|c_\eta(t)\|^2}dt$$
$$H_\eta \Phi_k^c(\eta) = \int_{t=0}^{1} \frac{t^k\left(c'(t)^{\perp}c_\eta(t)^T + c_\eta(t)c'(t)^{\perp T}\right)}{4\pi\|c_\eta(t)\|^2}dt$$
$$+\int_{t=0}^{1} \frac{t^k\left(c_\eta(t)\cdot c'(t)^{\perp} I_2\right)}{4\pi\|c_\eta(t)\|^2}dt - \int_{t=0}^{1} \frac{t^k\left(c_\eta(t)\cdot c'(t)^{\perp}\right)c_\eta(t)c_\eta(t)^T}{2\pi\|c_\eta(t)\|^4}dt$$

Those expressions include, in this form, terms for which we do not have available formulas (in particular, the ones containing the logarithm). In the following, we rework these expressions and demonstrate that they can equivalently be expressed as linear combinations of our parametric F_K^c functional.

We first focus on the derivation of Φ_k^c.

We note $P_k^c(t) := t^k c_\eta(t)\cdot c'(t)^{\perp}/(8\pi)$ and $w(t) := 1-2\log(\|c_\eta(t)\|)$. $P_k^c(t)$ is a simple polynomial ($P_k^c(t) := \sum_k \alpha_k t^k$) whose primitive $\tilde{P}_k^c(t)$ taking value 0 at $t=0$ can be computed analytically:

$$\tilde{P}_k^c(t) := \int_{u=0}^{t} P_k^c(u)du = \sum_k \alpha_k t^{k+1}/(k+1),$$

and w' is given by $w'(t) = -2c'(t)\cdot c_\eta(t)/\|c_\eta(t)\|^2$.

Using integration by parts ($\int_0^1 P_k^c w = [\tilde{P}_k^c w]_0^1 - \int_0^1 \tilde{P}_k^c w'$), we obtain

$$\boxed{\Phi_k^c(\eta) = -w(1)\tilde{P}_k^c(1) - 2F_1^c\left[(c'\cdot c_\eta)\tilde{P}_k^c\right]} \qquad (62)$$

Similarly, noting $Q_k^c(t) := t^k c'(t)^{\perp}/(8\pi)$ and $\tilde{Q}_k^c(t) = \int_{u=0}^{t} Q_k^c(u)du$ its primitive taking null value in $t=0$, we obtain

$$\boxed{\nabla_\eta \Phi_k^c(\eta) = w(1)\tilde{Q}_k^c(1) + 2F_1^c\left[(c'\cdot c_\eta)\tilde{Q}_k^c - (Q_k^c\cdot c_\eta)c_\eta\right]} \qquad (63)$$

Finally, $H_\eta \Phi_k^c$ does not require any transformation and can be rewritten as is in terms of F_k^c as

$$\boxed{\begin{aligned} H_\eta \Phi_k^c(\eta) =& \frac{1}{4\pi}F_1^c\left[X^k\left(c'^{\perp}c_\eta^T + c_\eta c'^{\perp T} + (c'^{\perp}\cdot c_\eta)I_2\right)\right] \\ &- \frac{1}{2\pi}F_2^c\left[X^k(c'^{\perp}\cdot c_\eta)c_\eta c_\eta^T\right] \end{aligned}} \qquad (64)$$

A.2.2 Fourth-order condition. Setting the fourth-order boundary condition as a polynomial curve $\bar{B}_c(t) := \sum_k t^k \bar{B}_k^c$, the contribution of curve c to its diffusion can be expressed similarly:

$$\begin{aligned} f_B^c(\eta) &:= \int_{t=0}^{1} -\bar{B}_c(t) g(c(t),\eta)dt \qquad (65)\\ &= \int_{t=0}^{1} \bar{B}_c(t)\frac{\|c_\eta(t)\|^2}{8\pi}(1-\log(\|c_\eta(t)\|))dt \\ &= \sum_k \bar{B}_c^k \underbrace{\int_{t=0}^{1} \frac{t^k\|c_\eta(t)\|^2}{8\pi}(1-\log(\|c_\eta(t)\|))dt}_{=:\,\Psi_k^c(\eta)} \end{aligned}$$

which leads to our second biharmonic coordinate $\Psi_k^c(\eta)$. Its derivatives are given by:

$$\nabla_\eta \Psi_k^c(\eta) = \int_{t=0}^{1} \frac{-t^k c_\eta(t)}{8\pi}(1-2\log(\|c_\eta(t)\|))dt$$
$$H_\eta \Psi_k^c(\eta) = \int_{t=0}^{1} \frac{t^k(1-2\log(\|c_\eta(t)\|))}{8\pi}dt I_2 - \int_{t=0}^{1} \frac{t^k c_\eta(t)c_\eta(t)^T}{4\pi\|c_\eta(t)\|^2}dt.$$

As in Sec. A.2.1, we solve those integrals using integration by parts. We note $R_k^c(t) := t^k\|c_\eta(t)\|^2/(8\pi)$, $v(t) := (1-\log(\|c_\eta(t)\|))$, $S_k^c(t) := -t^k c_\eta(t)/(8\pi)$, and $w(t) := 1-2\log(\|c_\eta(t)\|)$. $R_k^c(t)$ and $S_k^c(t)$ are simple polynomials whose primitives can be computed analytically, and $w'(t) = 2v'(t) = -2c'(t)\cdot c_\eta(t)/\|c_\eta(t)\|^2$.

Noting $\tilde{R}_k^c(t) := \int_{u=0}^{t} R_k^c(u)du$ and $\tilde{S}_k^c(t) = \int_{u=0}^{t} S_k^c(u)du$ the primitives of R_k^c and S_k^c taking null values in $t=0$, we obtain closed-form expressions for $\Psi_k^c(\eta)$ and its derivatives as

$$\boxed{\Psi_k^c(\eta) = v(1)\tilde{R}_k^c(1) + F_1^c\left[\tilde{R}_k^c c'\cdot c_\eta\right]} \qquad (66)$$

$$\boxed{\nabla_\eta \Psi_k^c(\eta) = w(1)\tilde{S}_k^c(1) + 2F_1^c\left[\tilde{S}_k^c c'\cdot c_\eta\right]} \qquad (67)$$

$$\boxed{H_\eta \Psi_k^c(\eta) = \frac{w(1) + 2F_1^c\left[X^{k+1}c'\cdot c_\eta\right]}{8\pi(k+1)}I_2 - \frac{F_1^c\left[X^k c_\eta c_\eta^T\right]}{4\pi}} \qquad (68)$$

B Boundary-aligned stretches optimization

We provide in this section the mathematical derivation of the boundary-aligned stretch local optimization of Section 7.3.

Given an input Jacobian matrix J, we optimize for

$$\bar{J} = \underset{j\in \mathsf{S}_n}{\operatorname{argmin}}\, \mathsf{P}_n(J,j) := \|J-j\|_F^2 \qquad (69)$$

Thanks to the trivial parameterization of SO_2, we can derive a closed-form expression (instead of an iterative procedure such as singular value decomposition in the general case) for this minimization procedure.

Since $B_n \in SO_2$, we note that

$$\begin{aligned}\mathsf{P}_n(J,j) &= \|J - j\|_F^2 \\ &= \|J - U\Sigma B_n^T\|_F^2 \\ &= \|B_n^T J B_n - B_n^T U \Sigma\|_F^2 \end{aligned} \tag{70}$$

We note $B_n^T J B_n = \left(\begin{array}{c|c} a & b \\ c & d \end{array}\right)$, $B_n^T U = \left(\begin{array}{c|c} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{array}\right)$ and $\Sigma = \left(\begin{array}{c|c} \alpha & 0 \\ 0 & \beta \end{array}\right)$, θ parameterizing the rotation $R_\theta = B_n^T U$ and (α, β) being the two boundary/tangent stretches.

Noting $(c_\theta, s_\theta) = (\cos(\theta), \sin(\theta))$ for simplicity, our penalty functional (Eq. (70)) reads

$$\mathsf{P}_n(J,j) = (a - \alpha c_\theta)^2 + (b + \beta s_\theta)^2 + (c - \alpha s_\theta)^2 + (d - \beta c_\theta)^2 \tag{71}$$

Setting its partial derivatives w.r.t. $\{\theta, \alpha, \beta\}$ to 0 leads to

$$0 = c_\theta(-c\alpha + b\beta) + s_\theta(a\alpha + d\beta) \tag{72}$$

$$\alpha = a c_\theta + c s_\theta \tag{73}$$

$$\beta = d c_\theta - b s_\theta \tag{74}$$

Plugging the second and third equations in the first one leads to an equation with θ as sole variable:

$$\delta(s_\theta^2 - c_\theta^2) + \mu c_\theta s_\theta = 0 \tag{75}$$

with

$$\delta := ac - bd \tag{76}$$

$$\mu := a^2 + d^2 - b^2 - c^2. \tag{77}$$

Setting $X = c_\theta$, leading to $s_\theta^2 = 1 - X^2$, we obtain a fourth-degree polynomial in X:

$$X^2 w(1 - X^2) - \delta^2 = 0 \tag{78}$$

with $w := 4\delta^2 + \mu^2$.

We set $Y = X^2$ and find $Y = X^2 = c_\theta^2$ verifying

$$-wY^2 + wY - \delta^2 = 0 \tag{79}$$

The discriminant of this second degree polynomial is $\Delta = w^2 - 4w\delta^2 = \mu^2 w \geq 0$. This equation admits therefore always solutions, given by:

$$Y = c_\theta^2 = \frac{w \pm \sqrt{\Delta}}{2w} = \frac{1}{2} \pm \frac{\mu}{2\sqrt{w}} \tag{80}$$

We note that $w \geq \mu^2$ leading to $\frac{\mu}{2\sqrt{w}} \leq \frac{1}{2}$, meaning that both solutions are acceptable for a squared cosine, as both solutions are contained in the range $[0, 1]$.

At this point, while it may seem that many solutions may appear (in particular, the sign of the cosine seems undefined from Eq. (80)), we note that θ has to verify Eq. (75), meaning that any choice of sign for the cosine leads to a fixed sign of the sine of θ:

$$\mathrm{sign}(s_\theta) = -\mathrm{sign}(\mu c_\theta)\mathrm{sign}(\delta(s_\theta^2 - c_\theta^2)) \tag{81}$$

resulting in two equivalent solutions $(c_\theta, s_\theta) \equiv (-c_\theta, -s_\theta)$ corresponding to the arbitrary orientation for the direction corresponding to θ ($\mathrm{dir}(\theta) \equiv \mathrm{dir}(\theta + \pi)$).

There are therefore two extrema values for θ, corresponding to the local maximum and local minimum of P_n (Eq. (71)).

Noting $\gamma_0 = \frac{1}{2} + \frac{\mu}{2\sqrt{w}}$ and $\gamma_1 = \frac{1}{2} - \frac{\mu}{2\sqrt{w}}$, we compute the corresponding candidate values for the decomposition:

$$\begin{cases} \cos(\theta_i) &= \sqrt{\gamma_i} \\ \mathrm{sign}_i &= -\mathrm{sign}(\mu\cos(\theta_i))\mathrm{sign}(\delta(1 - 2\cos(\theta_i)^2)) \\ \sin(\theta_i) &= \mathrm{sign}_i\sqrt{1 - \cos(\theta_i)^2} \\ \alpha_i &= a\cos(\theta_i) + c\sin(\theta_i) \\ \beta_i &= d\cos(\theta_i) - b\sin(\theta_i) \end{cases}$$

and pick the solution giving the lowest penalty value (in Eq. (71)).

Finally, the solution to Eq. (69) is given by

$$\underset{j \in \mathsf{S}_n}{\mathrm{argmin}}\, \mathsf{P}_n(J,j) = B_n R_\theta \Sigma B_n^T \tag{82}$$

C Boundary-aligned stretches optimization under normal rigidity constraints

We present in this appendix the solution to the projection procedure

$$\bar{J} = \underset{j \in \mathsf{S}_n^1}{\mathrm{argmin}}\, \mathsf{P}_n^1(J,j) := \|J - j\|_F^2 \tag{83}$$

that echoes the procedure presented in the previous appendix.

We recall that S_n^1 denotes the set of matrices whose stretch directions are aligned onto n and $n^\perp$, and have a unit stretch in the direction of n, i.e., $j \in \mathsf{S}_n^1$ if $j = B_n R_\theta \Sigma B_n^T$ with $\Sigma = diag(1, \beta)$ (using the notations of the previous appendix section).

Constraining $\Sigma(0,0) = 1$ (i.e., $\alpha = 1$ compared with the previous derivation) leads to a simplified functional to minimize:

$$\mathsf{P}_n^1(J,j) = (a - c_\theta)^2 + (b + \beta s_\theta)^2 + (c - s_\theta)^2 + (d - \beta c_\theta)^2 \tag{84}$$

Setting its derivatives w.r.t. $\{\theta, \beta\}$ to 0 leads to

$$0 = c_\theta(-c + b\beta) + s_\theta(a + d\beta) \tag{85}$$

$$\beta = d c_\theta - b s_\theta \tag{86}$$

Plugging the second equation into the first one leads to

$$bd(1 - 2c_\theta^2) + c c_\theta = s_\theta(a + (d^2 - b^2)c_\theta) \tag{87}$$

Setting $X = c_\theta$ ($s_\theta^2 = 1 - X^2$) and squaring this last equation leads to a quartic polynomial in X:

$$\begin{aligned} 0 = &X^4 \left[4b^2d^2 + (d^2 - b^2)^2\right] \\ &+X^3 \left[-4bcd + 2a(d^2 - b^2)\right] \\ &+X^2 \left[-4b^2d^2 + a^2 + c^2 - (d^2 - b^2)^2\right] \\ &+X \left[2bcd - 2a(d^2 - b^2)\right] \\ &+1 \left[b^2d^2 - a^2\right] \end{aligned} \tag{88}$$

Given its roots $\{r_i\}$ corresponding to extrema of Eq. (84),we compute the corresponding candidate values for the decomposition:

$$\begin{cases} \cos(\theta_i) = & \min(1, \max(-1, r_i)) \\ \mathrm{sign}_i = & \mathrm{sign}(a + (d^2 - b^2)\cos(\theta_i)) \\ & \mathrm{sign}(bd(1 - 2\cos(\theta_i)^2) + c\cos(\theta_i)) \\ \sin(\theta_i) = & \mathrm{sign}_i\sqrt{1 - \cos(\theta_i)^2} \\ \beta_i = & d\cos(\theta_i) - b\sin(\theta_i) \end{cases}$$

and pick the solution giving the lowest penalty value (in Eq. (84)).

Finally, the solution to Eq. (83) is given by

$$\operatorname*{argmin}_{j \in \mathsf{S}^1_{\boldsymbol{n}}} \mathsf{P}^1_{\boldsymbol{n}}(J, j) = B_{\boldsymbol{n}} R_\theta \Sigma B^T_{\boldsymbol{n}} \tag{89}$$