Cubic Mean Value Coordinates

Xian-Ying Li[1] Tao Ju[2] Shi-Min Hu[1]
[1]Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing
[2]Department of Computer Science and Engineering, Washington University in St. Louis

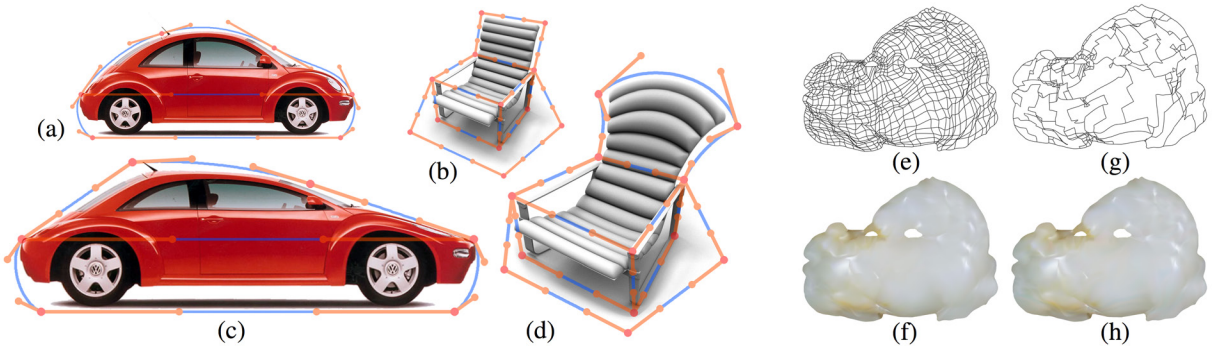


Figure 1: *Applications using cubic mean value coordinates. Left: shape deformation using curved cage networks, (a,b): input images, (c,d): deformed results. Right: an adaptive gradient mesh (g) created from a given gradient mesh (e), and (f,h) are the rasterized images of (e,g).*

Abstract

We present a new method for interpolating both boundary values and gradients over a 2D polygonal domain. Despite various previous efforts, it remains challenging to define a closed-form interpolant that produces natural-looking functions while allowing flexible control of boundary constraints. Our method builds on an existing transfinite interpolant over a continuous domain, which in turn extends the classical mean value interpolant. We re-derive the interpolant from the mean value property of biharmonic functions, and prove that the interpolant indeed matches the gradient constraints when the boundary is piece-wise linear. We then give closed-form formula (as generalized barycentric coordinates) for boundary constraints represented as polynomials up to degree 3 (for values) and 1 (for normal derivatives) over each polygon edge. We demonstrate the flexibility and efficiency of our coordinates in two novel applications, smooth image deformation using curved cage networks and adaptive simplification of gradient meshes.

CR Categories: G.1.1 [Interpolation]: Interpolation formulas; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems;

Keywords: interpolation, cubic, mean value, biharmonic, cage-based deformation, gradient mesh simplification

Links: DL PDF WEB

1 Introduction

Generalized barycentric coordinates are a simple yet powerful way to interpolate values on a polygonal domain. Interpolation at an arbitrary point $\mathbf{v}$ involves a weighted combination of values associated with the polygon vertices, where the weights are referred to as *coordinates* of $\mathbf{v}$. Among many possible choices of coordinates, *mean value coordinates* [Floater 2003] are particularly popular as they possess simple closed forms and produce natural-looking interpolations in arbitrary convex or non-convex domains. For this reason, mean value coordinates have been widely used for applications ranging from cage-based shape deformation [Ju et al. 2005] to approximation of PDEs [Farbman et al. 2009].

Oftentimes both values and gradients (i.e., normal derivatives) on the boundary need to be interpolated. For example, when deforming a shape using a network of cages, enforcing common deformation gradient along shared cage edges is necessary to achieve a globally smooth warp. In patch-based representation of a vector image, the color at a particular point is often evaluated based on both colors and gradients on a patch boundary [Sun et al. 2007]. Coordinates for these applications should have the following properties:

1. They should accommodate continuously varying values and gradients (i.e., at least linear on each edge).
2. They should have a closed-form that allows efficient computation.
3. The resulting interpolation should be smooth and intuitive.

Several coordinates have been proposed for Hermite interpolation, most notably the biharmonic coordinates [Weber et al. 2012] and the moving least square coordinates [Manson and Schaefer 2010]. However, neither coordinates possess all the properties above. While biharmonic coordinates yield shape-aware interpolation (as a biharmonic function), their closed forms only approximate the true solution and are restricted to discontinuous, piecewise constant gradients. Moving least square coordinates possess exact closed forms and accommodate continuous, high-order gradient constraints, but they tend to produce unnatural undulations in non-convex domains (see Figures 2 and 4).

We introduce a new type of coordinates for Hermite interpolation that possess these desirable properties in this paper. Our coordinates are based on the transfinite interpolant introduced by Floater and Schulz [2008] for Hermite constraints defined continuously over the boundary. The interpolant is an extension of mean value interpolation and enjoys similar theoretical properties, such as being well-defined and smooth over both convex and non-convex domains. Experiments demonstrate that the interpolant matches the boundary gradients while exhibiting a naturally varying shape. Since the interpolant reproduces cubic functions, we shall refer to it in this paper as the *cubic mean value (CMV) interpolant*.

We first reveal, in the transfinite setting, a new connection between mean value interpolant and the CMV interpolant: while the former is derived from the mean value property of harmonic functions, the latter can be similarly derived from the mean value property of biharmonic functions. We also prove that, for polygonal boundaries, the CMV interpolant exactly matches the boundary gradients. We next present the *cubic mean value coordinates*, a discrete closed-form coordinates for evaluating the interpolant over a polygonal domain. The CMV coordinates allow fast and exact interpolation of continuously varying values and gradients represented respectively as cubic and linear functions over each polygon edge.

We introduce two novel applications that showcase the flexibility and efficiency of CMV coordinates. In the first application, we perform smooth shape deformation with a network of cages made up of both straight and curved edges (Figure 1 Left). In the second application, we propose a variation of the gradient meshes [Lai et al. 2009] for representing vector images that supports irregular patches and adaptive coarsening (Figure 1 Right).

2 Related works

Coordinates for interpolating values Classical forms of coordinates, such as Wachspress coordinates [Wachspress 1975; Meyer et al. 2002] and discrete harmonic coordinates (or cotangent weights) [Pinkall and Polthier 1993], are well-defined within a convex polygon. However, these coordinates are not always defined outside the polygon or in a non-convex polygon. Floater introduced mean value cooridnates [2003], which is well defined everywhere in the plane given arbitrary polygonal domains [Hormann and Floater 2006]. These three types of coordinates all have simple closed forms, and in fact belong to a one-parameter family of coordinates [Floater et al. 2006]. More recently, several coordinates have been introduced to achieve stronger properties, such as positivity within a non-convex shape, including harmonic coordinates [Joshi et al. 2007], maximum entropy coordinates [Hormann and Sukumar 2008] and positive Gordon-Wixom coordinates [Manson et al. 2011]. However, they either lack analytical form or require numerical integrations.

Several coordinates are designed to minimize the angle distortion during deformation. Green coordinates [Lipman et al. 2008] and Cauchy coordinates [Weber et al. 2009] produce holomorphic mappings, while Hilbert coordinates [Weber and Gotsman 2010] produce holomorphic mappings with no zero derivatives (i.e., the mappings are conformal). Note that these methods lack exact boundary interpolation. This deficiency is addressed by the MAGIC complex barycentric coordinates proposed by Weber et al. [2011], which strictly interpolate boundary values.

Variational approaches seek smooth blendings or propagations by solving partial differential equations, e.g., higher order Laplacian weights [Botsch and Kobbelt 2004], heat diffusion weights [Baran and Popović 2007], bounded biharmonic weights [Jacobson et al. 2011], etc. These methods in general do not have close-form solutions. For an in-depth survey of these variational methods, readers may refer to [Botsch et al. 2010].

Coordinates for interpolating values and derivatives Langer and Seidel [2008] gave a method to convert any barycentric coordinates to higher order ones. However, these converted coordinates interpolate derivatives only at the vertices and lack derivative control over the rest of the boundary.

Floater and Schulz [2008] introduced a general method to interpolate value and derivative constraints defined continuously over closed boundaries in any dimensions. To interpolate derivatives up to order k, the method constructs univariate interpolants along straight rays from the evaluation location $\mathbf{v}$ to each boundary point, and seeks the value at $\mathbf{v}$ that minimizes the integral of squared $k+1$-st derivative over all rays. It turns out that the method yields mean value interpolation for $k = 0$. For $k = 1$, the authors show that the minimizer uniquely exists (a.k.a. the CMV interpolant), and can be expressed in an integral over the boundary.

Dyken and Floater introduced the transfinite Hermite mean value interpolant [2009], which extends the well-known 1D cubic Hermite interpolation to a continuous 2D boundary by replacing the linear Lagrange component in 1D with mean value interpolation in 2D. However, evaluating the interpolant requires double integration over the boundary, which makes derivation of closed-form coordinates much more challenging than the cubic mean value interpolant.

Moving least square coordinates [Manson and Schaefer 2010] provide a powerful framework for interpolating value and derivative (of any order) constraints on both open and closed polygons. The value is determined by the best-fitting polynomial in the least square sense, where the fitting error is weighted by 1 over distance to the boundary raised to the power of 2α for some user-specified value α. The coordinates possess exact closed forms for boundary values and gradients expressed as polynomials on each polygon edge. The authors observed experimentally that gradient interpolation is achieved by $\alpha \geq 2$, but no proof was given. Moreover, in our experiments we observed that interpolating gradient constraints using moving least square coordinates with $\alpha = 2$ (or greater) tend to exhibit unnatural "ripples" in a non-convex domain (see more discussion in Section 3.2).

Recently, Weber et al. introduced biharmonic coordinates [Weber et al. 2012], an extension of harmonic coordinates to approximate the unique biharmonic function that interpolates Hermite boundary constraints. Using the boundary element method, the coordinates are presented as closed forms. However, the closed forms are derived assuming a constant gradient along each edge, which is not continuous across vertices. In terms of efficiency, evaluation of the biharmonic coordinates at a point requires $O(n^2)$ time for a polygon with n edges, which can be prohibitive for a finely tessellated curve. In contrast, evaluation of both moving least square coordinates and our cubic mean value coordinates take $O(n)$ time. Finally, biharmonic interpolation is only defined in the interior of the boundary, whereas Hermite mean value interpolation, moving least square interpolation, and our CMV interpolation are all defined both inside and outside the domain.

3 Transfinite interpolation

We start with a discussion on the transfinite CMV interpolant, introduced by Floater and Schulz [2008], upon which our coordinates are constructed. We first describe an alternative derivation of the CMV interpolant that is guided by the same "mean value" principle from which the original mean value interpolation was derived. We then discuss the properties of the interpolant and compare it with other Hermite interpolants.

3.1 Deriving CMV interpolant

Mean value interpolation Consider a closed curve P in $\mathbb{R}^2$ with a parameterization $\mathbf{p}[t]$, and a continuous scalar function $f[t]$ defined over P. The problem is to construct a function $\hat{f}[\mathbf{v}]$ for any $\mathbf{v} \in \mathbb{R}^2$ that interpolates $f[t]$ on P, that is, $\hat{f}[\mathbf{p}[t]] = f[t]$.

Ideally, this function should be as smooth as possible and avoids any unnatural undulations that are not implied by the boundary constraint f. A natural choice of $\hat{f}$ in the interior of P, ΩP, would be the harmonic function. The niceness of the harmonic function arises from its *mean value property*, which states that the value at a location $\mathbf{v}$ is the average of all those values on the unit circle $C_\mathbf{v}$ centered at $\mathbf{v}$ (assuming $C_\mathbf{v}$ is contained in ΩP).

Mean value interpolation is derived by enforcing the mean value property over an auxiliary function that linearly interpolates between $\hat{f}[\mathbf{v}]$ and $f[t]$ *along straight rays*. Specifically, consider the ray from $\mathbf{v}$ to $\mathbf{p}[t]$, we use the *linear* interpolant $F[r]$ (where r is distance from $\mathbf{v}$) with constraints

$$F[0] = \hat{f}[\mathbf{v}], \quad F[|\mathbf{u}|] = f[t],$$

where $\mathbf{u} = \mathbf{p}[t] - \mathbf{v}$. Note that $F[1]$, which is the value on the unit circle $C_\mathbf{v}$, is a linear function of the unknown $\hat{f}[\mathbf{v}]$. The mean value property of the union of F along all rays from $\mathbf{v}$ can be written as an integral over $C_\mathbf{v}$[1]:

$$\hat{f}[\mathbf{v}] = \frac{1}{2\pi} \int_t F[1] dC_\mathbf{v}. \tag{1}$$

Re-writing $F[1]$ as a linear function of $\hat{f}[\mathbf{v}]$, the above equality yields a simple linear equation for $\hat{f}[\mathbf{v}]$, whose unique solution is

$$\hat{f}[\mathbf{v}] = \int_t \frac{f[t]}{|\mathbf{u}|} dC_\mathbf{v} \Big/ \int_t \frac{1}{|\mathbf{u}|} dC_\mathbf{v}. \tag{2}$$

Cubic mean value interpolation Now suppose we have an additional continuous vector function $\mathbf{g}[t]$ on P that serves as the gradient constraints. The problem is now constructing a function $\hat{f}[\mathbf{v}]$ that interpolates both $f[t]$ and $\mathbf{g}[t]$, that is, $\hat{f}[\mathbf{p}[t]] = f[t]$ and $\nabla \hat{f}|_{\mathbf{p}[t]} = \mathbf{g}[t]$.

A natural choice of $\hat{f}$ would be the biharmonic function, which is an extension of harmonic functions with harmonic Laplacians everywhere. Since a biharmonic function minimizes the Hessian energy, it is in certain sense the least undulating function under the value and gradient constraints on the boundary. Like harmonic functions, biharmonic functions possess a similar mean value property that relates the value at a point to the weighted average of values in a circular neighborhood. Specifically, let $\mathcal{B}$ be a biharmonic function over a unit circle centered at the origin, and let $\mathcal{F}[\theta]$ and $\mathcal{G}[\theta]$ be respectively the value of $\mathcal{B}$ and derivative of $\mathcal{B}$ in the outward normal direction at point $\{\cos\theta, \sin\theta\}$. The value and gradient of $\mathcal{B}$ at the origin can be expressed as (see proof in Appendix A)

$$\begin{aligned} \mathcal{B}|_{\text{origin}} &= \frac{1}{4\pi} \int_0^{2\pi} (2\mathcal{F}[\theta] - \mathcal{G}[\theta]) d\theta, \\ \nabla\mathcal{B}|_{\text{origin}} &= \frac{1}{2\pi} \int_0^{2\pi} (3\mathcal{F}[\theta] - \mathcal{G}[\theta]) \{\cos\theta, \sin\theta\} d\theta. \end{aligned} \tag{3}$$

[1] $dC_\mathbf{v}$ is the projection of dt on $C_\mathbf{v}$. Their relation is given by [Ju et al. 2005]:

$$dC_\mathbf{v} = \frac{\mathbf{p}^\perp[t] \cdot (\mathbf{p}[t] - \mathbf{v})}{|\mathbf{p}[t] - \mathbf{v}|^2} dt,$$

where $\mathbf{p}^\perp[t]$ is tangent vector at $\mathbf{p}[t]$ rotated counterclockwise by $\pi/2$. The sign of $dC_\mathbf{v}$ captures whether P has folded back when projecting onto $C_\mathbf{v}$.

Following mean value interpolation, we seek an $\hat{f}$ by enforcing the mean value property on an auxiliary function that interpolates between $\hat{f}[\mathbf{v}]$ and the boundary constraints along straight rays. Specifically, consider the ray from $\mathbf{v}$ to $\mathbf{p}[t]$, we use the *cubic* interpolant $F[r]$ (where r is distance from $\mathbf{v}$) with Hermite constraints

$$\begin{aligned} F[0] &= \hat{f}[\mathbf{v}], & F[|\mathbf{u}|] &= f[t], \\ F'[0] &= \hat{\mathbf{g}} \cdot \frac{\mathbf{u}}{|\mathbf{u}|}, & F'[|\mathbf{u}|] &= \mathbf{g}[t] \cdot \frac{\mathbf{u}}{|\mathbf{u}|}. \end{aligned}$$

Here, $\hat{\mathbf{g}}$ is an unknown gradient constraint at $\mathbf{v}$. Note that Floater and Schulz used the same cubic interpolants in their radial function [2008]. Now, enforcing the mean value property in Equation 3 over the union of F along all rays yields

$$\begin{aligned} \hat{f}[\mathbf{v}] &= \frac{1}{4\pi} \int_t (2F[1] - F'[1]) dC_\mathbf{v}, \\ \hat{\mathbf{g}} &= \frac{1}{2\pi} \int_t (3F[1] - F'[1]) \frac{\mathbf{u}}{|\mathbf{u}|} dC_\mathbf{v}. \end{aligned} \tag{4}$$

where $F[1]$ and $F'[1]$ are the value and normal derivative on the unit circle $C_\mathbf{v}$. Re-writing $F[1], F'[1]$ as linear functions of $\hat{f}[\mathbf{v}]$, $\hat{\mathbf{g}}_X$ and $\hat{\mathbf{g}}_Y$ (the X,Y components of $\hat{\mathbf{g}}$), the above equalities yield a simple system of linear equations in these unknowns, whose unique solution has the form

$$\{\hat{f}[\mathbf{v}], \hat{\mathbf{g}}_X, \hat{\mathbf{g}}_Y\}^T = \mathbf{A}^{-1}\mathbf{b}, \tag{5}$$

where

$$\begin{aligned} \mathbf{A} &= \int_t \begin{pmatrix} 6 & 3\mathbf{u}_X & 3\mathbf{u}_Y \\ 3\mathbf{u}_X & 2\mathbf{u}_X^2 & 2\mathbf{u}_X\mathbf{u}_Y \\ 3\mathbf{u}_Y & 2\mathbf{u}_X\mathbf{u}_Y & 2\mathbf{u}_Y^2 \end{pmatrix} \frac{1}{|\mathbf{u}|^3} dC_\mathbf{v}, \\ \mathbf{b} &= \int_t \begin{pmatrix} 6f[t] - 3\mathbf{g}[t] \cdot \mathbf{u} \\ (3f[t] - \mathbf{g}[t] \cdot \mathbf{u})\mathbf{u}_X \\ (3f[t] - \mathbf{g}[t] \cdot \mathbf{u})\mathbf{u}_Y \end{pmatrix} \frac{1}{|\mathbf{u}|^3} dC_\mathbf{v}, \end{aligned} \tag{6}$$

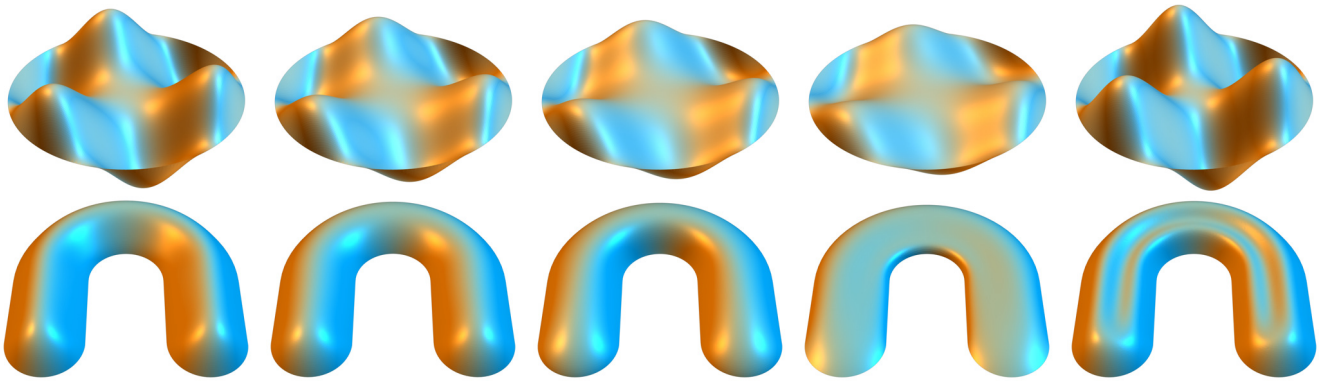
and $\mathbf{u}_X, \mathbf{u}_Y$ are the X, Y components of $\mathbf{u} = \mathbf{p}[t] - \mathbf{v}$ (detailed derivation is provided in Appendix B). The $\hat{f}[\mathbf{v}]$ in Equation 5 is exactly the CMV interpolant as derived in [Floater and Schulz 2008].

3.2 Properties and comparison

The CMV interpolant has a series of nice theoretical properties [Floater and Schulz 2008]. First, $\hat{f}$ is well-defined over the entire plane, regardless of the convexity of P. Second, $\hat{f}$ interpolates values $f[t]$ on P. Third, $\hat{f}$ is C^∞ smooth everywhere away from P. Finally, $\hat{f}$ reproduces any cubic function.

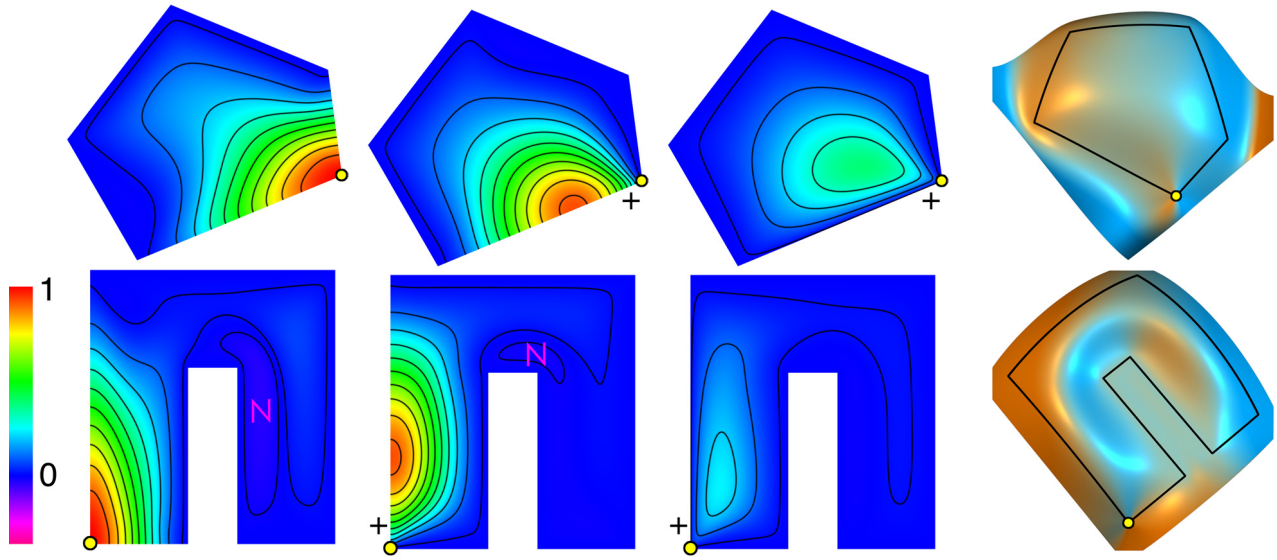
While experiments support that $\hat{f}$ matches the gradients $\mathbf{g}[t]$ on P, a proof was not given in [Floater and Schulz 2008]. As a first step in filling in this gap, we give (in Appendix C) a proof of this property for the case when P is a polygon.

Figure 2 compares the behavior of the CMV interpolant with several other Hermite interpolants, including the (exact) biharmonic function (BF), Hermite mean value (HMV) interpolant, and moving least squares (MLS) interpolant. For MLS, we use quadratic fitting function and show results with $\alpha = 1$ and $\alpha = 2$. We use a circular and a U-shaped domain, with a constant f and a varying $\mathbf{g}$ over P. Observe that CMV has a natural shape similar to that of BF and HMV in both curves. MLS with $\alpha = 1$ produces smooth functions but fails to interpolate the derivative constraints. MLS with $\alpha = 2$ interpolates the boundary derivatives but exhibits unnatural "ripples" in the middle of the non-convex U-curve.



(a) CMV (b) BF (c) HMV (d) MLS ($\alpha = 1$) (e) MLS ($\alpha = 2$)

Figure 2: *Comparing different transfinite Hermite interpolation methods in a convex (top) and non-convex (bottom) domain.*



(a) CMV coordinate a_i (b) CMV coordinate b_i^+ (c) CMV coordinate c_i^+ (d) interpolated surfaces

Figure 3: *(a,b,c): CMV coordinates at a marked vertex for a convex polygon (top) and a non-convex polygon, plotted using both color and iso-lines (at $j/10$ for $j = 1, 2, ..., 9$ and at $\pm 1/100$). Regions with negative values are noted by "N". (d): Surface rendering of interpolation using CMV coordinates for each polygon given mixed types of gradient constraints (constant on two incident edges of the marked vertex and linear on other edges).*

4 Coordinates for closed polygons

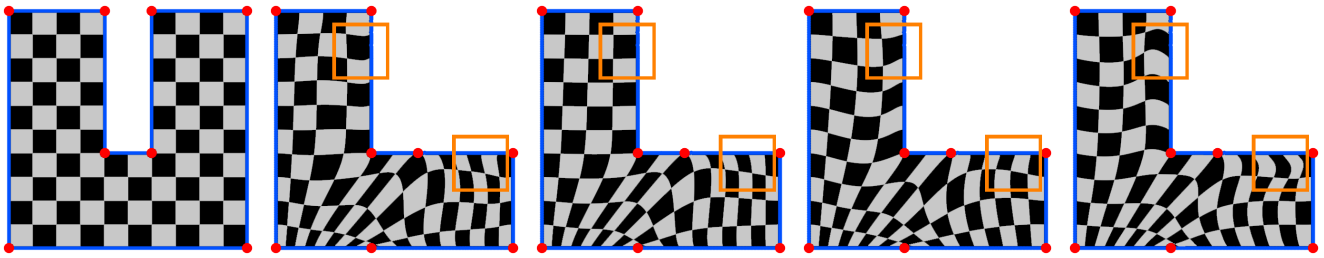
We next consider a polygonal P, where the CMV interpolant guarantees to match both values and gradients. We assume $f, \mathbf{g}$ are expressed as piece-wise polynomial functions over edges of P. For a useful class of polynomials, we will give closed-form formula for the integrals in $\mathbf{A}$ and $\mathbf{b}$, and hence the CMV interpolant, as weighted sums of $f, \mathbf{g}$ and their derivatives at the vertices.

To achieve C^1 interpolation, both f and $\mathbf{g}$ need to be continuously defined on the polygon. This requires the use of at least linear functions on each edge. We found that representing f as a cubic function and the outward normal component of $\mathbf{g}$ as a linear function on each edge offers sufficient flexibility for our applications. To define these functions, we need 5 scalars at each vertex $\mathbf{p}_i$ of P:

- f_i: the value of f at $\mathbf{p}_i$.
- f_i^-, f_i^+: the derivatives of f at $\mathbf{p}_i$ respectively along edges $\{\mathbf{p}_{i-1}, \mathbf{p}_i\}$ and $\{\mathbf{p}_i, \mathbf{p}_{i+1}\}$.
- h_i^-, h_i^+: the outward normal components of $\mathbf{g}$ at $\mathbf{p}_i$ respectively on edges $\{\mathbf{p}_{i-1}, \mathbf{p}_i\}$ and $\{\mathbf{p}_i, \mathbf{p}_{i+1}\}$.

Note that, when the two edges at $\mathbf{p}_i$ are not co-linear, the values of $h_i^\pm$ are completely determined by $f_i^\pm$ for the gradient to be continuous at $\mathbf{p}_i$. However, we still leave them as separate inputs in case a continuous gradient is not desired or impractical (i.e., at a joint in a cage network, see next section).

With this setting, the functions $f[t]$ and $\mathbf{g}[t]$ along each edge $\{\mathbf{p}_i, \mathbf{p}_{i+1}\}$ can be represented as a weighted sum of the constraints at $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$, where the weights are polynomials of t (we use arc



(a) Source image (b) CMV coordinates (c) Biharmonic coordinates (d) HMV interpolant (e) MLS coordinates

Figure 4: *Deforming a non-convex cage (a) using different Hermite interpolants and coordinates (b,c,d,e).*

length parameterization). Substituting these expressions into Equations 6 and 5 yields the following form of $\hat{f}[\mathbf{v}]$ as a weighted sum of all input constraints:

$$\hat{f}[\mathbf{v}] = \sum_i a_i[\mathbf{v}] f_i + \sum_{i,s} b_i^s[\mathbf{v}] f_i^s + \sum_{i,s} c_i^s[\mathbf{v}] h_i^s, \qquad (7)$$

where $s \in \{+, -\}$ and coefficients a, b, c, the CMV coordinates, are integrals that can be written in closed forms using the geometry of P and $\mathbf{v}$ (see Appendix D). Examples of these coordinate functions are plotted in Figure 3 for a convex polygon (top) and a non-convex polygon (bottom).

The use of cubic f and linear $\mathbf{g}$ allows flexible control over the behavior of the interpolant. As an example, the interpolations (using CMV coordinates) in the last column of Figure 3 are computed under the following setting: f is cubic along each edge, the normal component of $\mathbf{g}$ is constant on the two edges incident to the marked vertex and linear on all other edges. Let the marked vertex be $\mathbf{p}_j$, this setting is achieved by computing h_i^-, h_i^+ from f_i^-, f_i^+ for all vertices $i \neq j$ and assigning $h_j^+ = h_{j+1}^-$ and $h_j^- = h_{j-1}^+$. Observe that each interpolation, defined over the entire plane, is C^1 everywhere on the polygon except at the marked vertex where it is C^0 (due to the discontinuity of gradient constraints there).

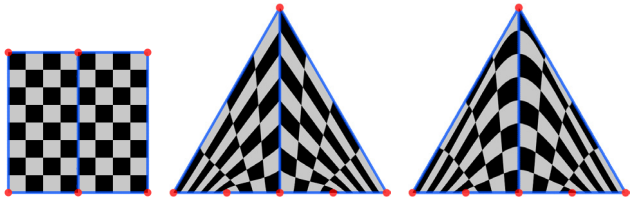
5 Applications

We show two applications, shape deformation and vector image representation, that harvest the flexible control over boundary constraints allowed by CMV coordinates.

5.1 Shape deformation using curved cage networks

Interactive shape deformation using cages has gained popularity in recent years. Cage-based deformation can be formulated as a boundary value interpolation problem: given cage vertices $\mathbf{p}_i$ and their deformed locations $f[\mathbf{p}_i]$, compute the deformed location $\hat{f}[\mathbf{v}]$ for any point $\mathbf{v}$. The problem can be efficiently solved using coordinates that interpolate boundary values, such as mean value coordinates. However, without constraining the gradient of $\hat{f}$, the deformation is only C^0 at the cage boundary. As a result, when using multiple abutting cages (or a cage network) to deform different portions of the shape, "seams" will become noticeable along shared cage boundaries (see Figure 5 (b)).

By specifying gradient constraints, we can ensure C^1 transition across shared edges between abutting cages. To do so, for each cage in the network, we set f_i at a cage vertex $\mathbf{p}_i$ to be the deformed location $f[\mathbf{p}_i]$, and compute $f_i^\pm$ so that f is linear along every cage edge (because the deformed cage edges remain straight). If $\mathbf{p}_i$ is



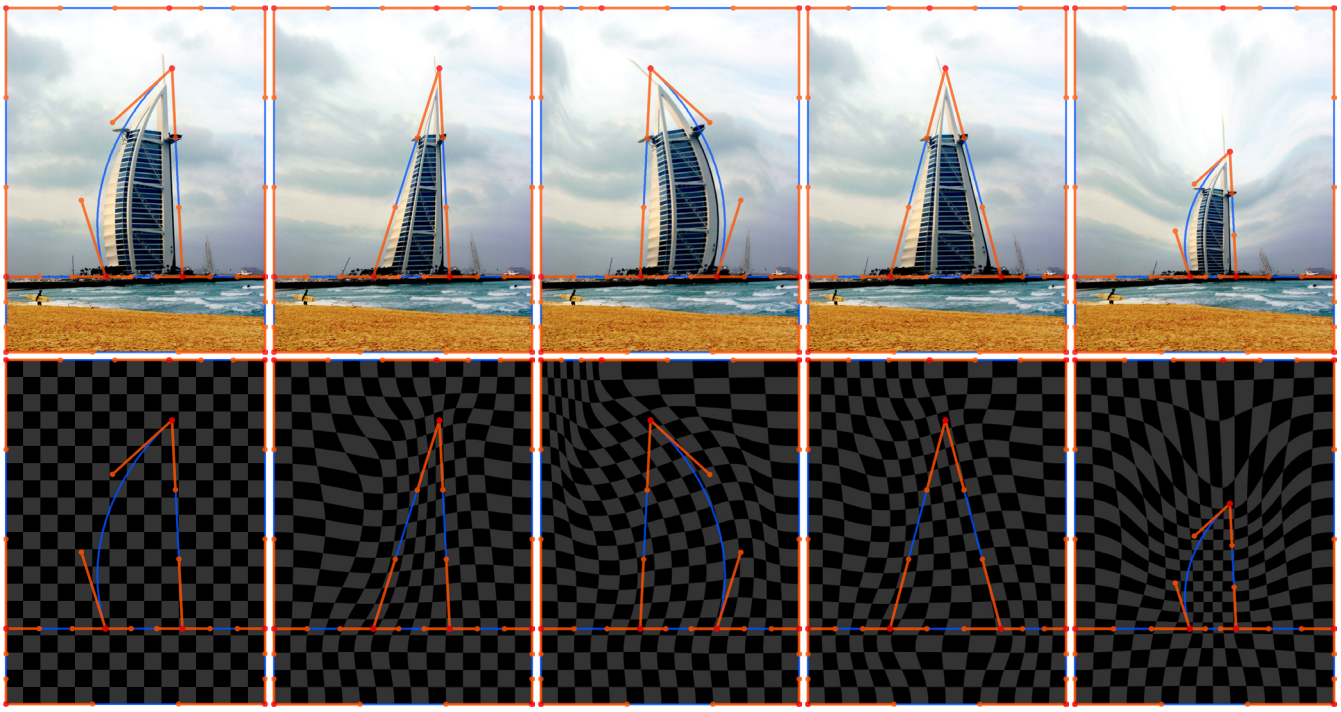
(a) input (b) MV (c) CMV

Figure 5: *Deforming a cage network using mean value (MV) coordinates (b) and CMV coordinates (c).*

incident to only two edges in the cage network (called a degree-2 vertex), $h_i^\pm$ are fixed by $f_i^\pm$. Otherwise, $h_i^\pm$ are obtained by first computing the best fitting gradient to the derivative constraints of f along all incident edges at $\mathbf{p}_i$ and projecting the gradient to the outward normal direction of the two edges incident to $\mathbf{p}_i$ in the current cage. The above setting ensures abutting cages use the same location and gradient constraints along their shared edges and vertices. A deformation that matches these constraints is not only C^1 across cage edges, but also C^1 at any degree-2 cage vertex or a vertex whose 1-ring neighbors undergo an affine transformation. The deformation using CMV coordinates is shown in Figure 5 (c).

We now compare the deformation produced by CMV coordinates with those created by other Hermite interpolants under the same gradient constraints over the cage boundary. As illustrated in Figure 4, CMV coordinates achieve qualitatively similar results as biharmonic coordinates and Hermite mean value (HMV) interpolant. However, the latter two are much more costly to compute. HMV requires numerical integration over the boundary. Since biharmonic coordinates only work for constant gradients per edge, the cage has to be finely tessellated to approximate the linearly varying gradient, effectively turning each evaluation into an integration over the boundary. Finally, although the moving least squares (MLS) coordinates have comparable efficiency as CMV coordinates, the deformation exhibits unnatural wiggles (e.g., near the top and right ends of "L"). This echoes our earlier observation on the MLS interpolation function in a non-convex domain (Figure 2).

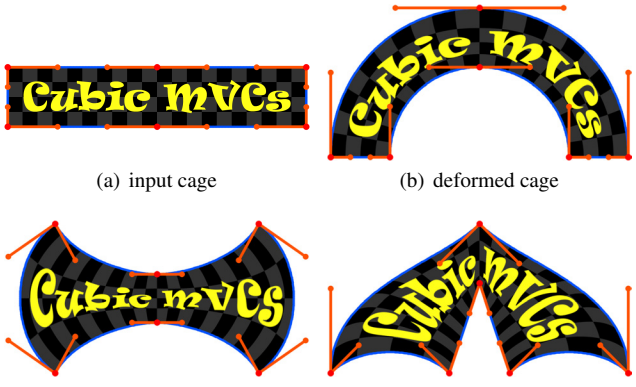
To further exploit the space of constraints allowed by CMV coordinates, we introduce *curved* cages. In this scenario, the user has control not only over the locations of cage vertices but also two tangent vectors on the ends of each cage edge - similar to free-form curve design in popular tools like Adobe Illustrator. Since the two tangent vectors define a cubic Bezier curve, it can be captured exactly by our cubic model of boundary values ($f_i^\pm$ are now determined from the tangent vectors). Curved cages allow creation of smooth deformations that precisely conform to user-defined curved



(a) input image (b) intermediate image (c) output image (d) output image (e) output image

Figure 6: *Image deformation using a curved cage network: (a) Input image and cage, (b) an intermediate image and a straightened cage, both computed by the system from the input, (c,d,e) deformed images for three new cages, computed from the intermediate image and straightened cage. The bottom row replaces the input image with a checkerboard pattern for visualization.*

boundaries (see examples in Figure 7), which are not possible with a linear model of boundary values.



(a) input cage (b) deformed cage

(c) deformed cage (d) deformed cage

Figure 7: *Deforming an input straight cage (a) to various curved cages (b,c,d) whose boundaries are piecewise cubic Bezier curves (in blue, and handles in orange) using CMV coordinates.*

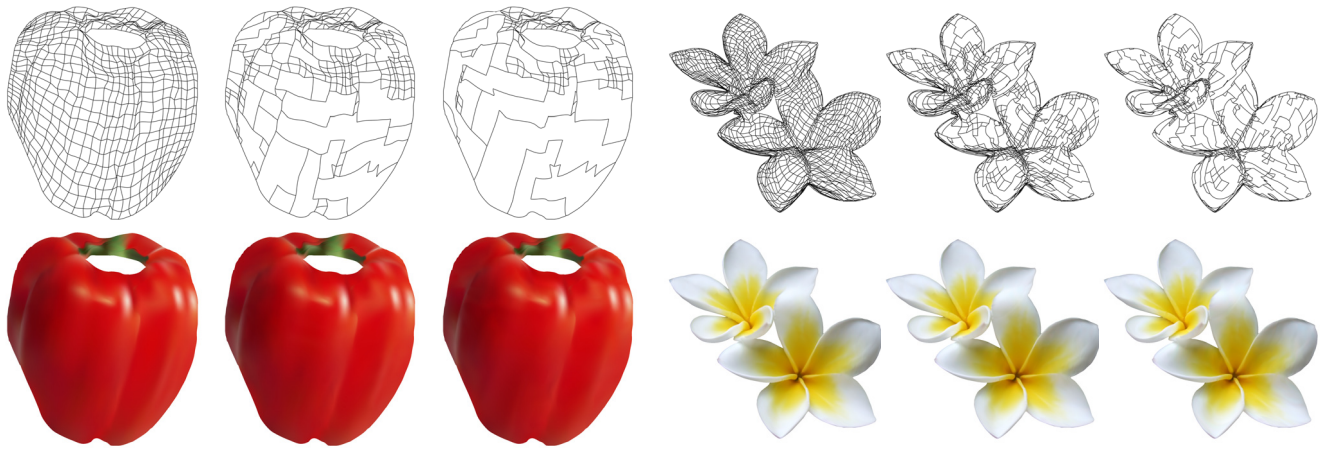
For image editing, we can further allow the use of curved initial cages, in addition to curved deformed cages. Given an initial image I and cage (or cage network) C (e.g., Figure 6 (a)), a "straightened" cage C^* is first constructed that replaces curved edges in C by straight segments. Then an intermediate image I^* is computed so that the color at each pixel $\mathbf{v}$ is obtained by $I^*[\mathbf{v}] = I[\hat{f}[\mathbf{v}]]$ where $\hat{f}$ is the deformation function from C^* to C computed using CMV coordinates, as described before (Figure 6 (b)). The deformation to any newly deformed cage is then computed from the intermediate image I^* and the straightened cage C^* (Figure 6 (c,d,e)). Note that the construction of I^*, C^* are hidden to the user, which allows fluid manipulation with free-form curved cages (see further examples in Figure 1 left and the accompanied video).

5.2 Adaptive gradient mesh

Gradient mesh is a powerful and widely used vector representation for smoothly changing images. A gradient mesh is a regular grid of Ferguson patches [Ferguson 1964; Sun et al. 2007] that interpolate both color and gradients stored at the nodes of the mesh. However, this representation does not support adaptive representation; the grid cannot be locally refined or coarsened to adapt to the density of image features. Although alternative representations with more flexible structure have been proposed, such as diffusion curves [Orzan et al. 2008] and curvilinear patches [Xia et al. 2009], they do not offer direct control of color gradients.

With the ability to interpolate both values and gradients, CMV coordinates offer a natural extension of gradient meshes to support irregular patch shapes. Given color and gradient at a vertex of a multi-sided patch, we set f_i to be the color and $f_i^{\pm}, h_i^{\pm}$ the components of the gradient along corresponding edge directions and their outward normal directions. Note that the interpolation interior to the patch using CMV coordinates reproduces only cubic functions (while Ferguson patches are bicubic). However, with a flexible patch structure that can adapt to image contents, such difference is usually unnoticeable.

We use a simple greedy algorithm to create an adaptive gradient



(a) initial mesh (b) simplified mesh (c) simplified mesh (d) initial mesh (e) simplified mesh (f) simplified mesh

Figure 8: *Two examples of creation of adaptive gradient meshes. For each example, we show the initial (regular) gradient mesh, two simplified meshes at different thresholds, and the images rendered from each mesh.*

mesh from an initial, fine mesh (e.g., created automatically from a raster image using [Lai et al. 2009]) by iterative node removal. At each step, our algorithm removes a node in the mesh and merges all incident patches into a large patch. The "cost" of removal is measured as the difference between the CMV interpolant in the merged patch and the original raster image (evaluated at sampled locations within the patch). The algorithm picks the removal operation with the least cost, and iterates until that cost exceeds a user-defined threshold. We then apply similar principle to simplify the boundaries of each patch by removing nodes and merging incident edges. Two examples are shown in Figure 8 (see the accompanying video for the simplification process). Note that image regions with smoother color variations are represented with patches of larger sizes. Also, the patch boundaries are aligned with major curvlinear image features.

5.3 Implementation and performance

We have implemented CMV coordinates in C++ on a Windows 7 platform. Using their closed forms, the time complexity for computing the coordinates at each point is $O(n)$ for a polygon with n edges. In our experiments, computing the CMV coordinates for $1,000,000$ points within a quadrangle took less than 3.5 seconds using an Intel Xeon E5620 2.40GHz CPU, which is about $12\times$ the time taken for computing mean value coordinates (using code provided by authors of [Ju et al. 2005]).

Like other types of coordinates, computing CMV coordinates for a large set of points and can be fully parallelized. We used OpenMP on a 16-core machine for the two application tasks. For cage-based deformation, computing the coordinates for all 750×1000 pixels of the intermediate image in Figure 6 (b) took less than 0.5 second, after which the deformation given new cages takes place in real time. For gradient mesh simplification, rasterizing a simplified mesh on a 1000×1000 resolution took less than 1 second, and the simplification process from an initial mesh took less than 0.5 second, for all examples shown.

6 Conclusion and discussion

We introduced the cubic mean value coordinates for interpolating Hermite constraints over a 2D polygonal boundary. The coordinates have closed-forms and yield natural interpolations that exactly match values and gradients expressed as cubic and linear functions over each edge. The coordinates are built upon a known transfinite interpolant [Floater and Schulz 2008], for which we give an alternative derivation. We demonstrate the utility of the coordinates in two applications, shape deformation and vector image representation.

As in mean value coordinates (and many other coordinates that rely on Euclidean distances), a boundary constraint may have (often negative) impact on distant locations when interpolating with CMV coordinates in a non-convex polygon (see Figure 3 lower-left). On the other hand, biharmonic coordinates have the advantage of producing more "shape-aware" functions, since the impact of a boundary constraint is propagated only through the interior of the domain.

While this paper focuses on 2D domains, the techniques can be extended to higher dimensions and possibly to matching higher order derivatives. In particular, [Floater and Schulz 2008] gives a general recipe for building transfinite interpolants in any dimensions matching derivatives of any order. While the cubic mean value interpolant has the same form in higher dimensions, the exact form of interpolants for higher order derivatives remain unknown. Similarly, our mean-value-based derivation of cubic mean value interpolant can be conceptually extended to higher order derivatives, using mean value properties for poly-harmonic functions [Goyal and Goyal 2012]. Whether the extensions of these two different derivations would yield equivalent interpolants, and how to construct coordinates of these interpolants over discrete domains in both 2D and 3D, are interesting topics for future investigation.

7 Acknowledgement

We thank all the reviewers for valuable comments. We thank previous researchers for sharing their codes. We thank Song-Pei Du and Yu-Kun Lai for helpful discussion. This work was supported by National Basic Research Project of China (2011CB302202), Natural Science Foundation of China (61120106007), National High Technology Research and Development Program of China (2013AA013903), Tsinghua University Initiative Scientific Research Program, and National Science Foundation (IIS-0846072). Xian-Ying Li was partially supported by the New PhD Researcher Award from the Chinese Ministry of Education.

References

BARAN, I., AND POPOVIĆ, J. 2007. Automatic rigging and animation of 3d characters. *ACM Transactions on Graphics 26*, 3, 72:1–8.

BOTSCH, M., AND KOBBELT, L. 2004. An intuitive framework for real-time freeform modeling. *ACM Transactions on Graphics 23*, 3, 630–634.

BOTSCH, M., KOBBELT, L., PAULY, M., ALLIEZ, P., AND LÉVY, B. 2010. *Polygon Mesh Processing*. A. K. Peters, Natick.

DYKEN, C., AND FLOATER, M. S. 2009. Transfinite mean value interpolation. *Computer Aided Geometric Design (CAGD) 26*, 1, 117–134.

FARBMAN, Z., HOFFER, G., LIPMAN, Y., COHEN-OR, D., AND LISCHINSKI, D. 2009. Coordinates for instant image cloning. *ACM Transactions on Graphics 28*, 3, 67:1–9.

FERGUSON, J. 1964. Multivariable curve interpolation. *Journal of the ACM 11*, 2, 221–228.

FLOATER, M. S., AND SCHULZ, C. 2008. Pointwise radial minimization: Hermite interpolation on arbitrary domains. *Computer Graphics Forum 27*, 5, 1505–1512.

FLOATER, M. S., HORMANN, K., AND KÓS, G. 2006. A general construction of barycentric coordinates over convex polygons. *Advances in Computational Mathematics 24*, 3, 311–331.

FLOATER, M. S. 2003. Mean value coordinates. *Computer Aided Geometric Design (CAGD) 20*, 1, 19–27.

GOYAL, S., AND GOYAL, V. B. 2012. Mean value results for second and higher order partial differential equations. *Applied Mathematical Sciences 6*, 77-80, 3941–3957.

HORMANN, K., AND FLOATER, M. S. 2006. Mean value coordinates for arbitrary planar polygons. *ACM Transactions on Graphics 25*, 4, 1424–1441.

HORMANN, K., AND SUKUMAR, N. 2008. Maximum entropy coordinates for arbitrary polytopes. *Computer Graphics Forum 27*, 5, 1513–1520.

JACOBSON, A., BARAN, I., POPOVIĆ, J., AND SORKINE, O. 2011. Bounded biharmonic weights for real-time deformation. *ACM Transactions on Graphics 30*, 4, 78:1–8.

JOSHI, P., MEYER, M., DEROSE, T., GREEN, B., AND SANOCKI, T. 2007. Harmonic coordinates for character articulation. *ACM Transactions on Graphics 26*, 3, 71:1–9.

JU, T., SCHAEFER, S., AND WARREN, J. 2005. Mean value coordinates for closed triangular meshes. *ACM Transactions on Graphics 24*, 3, 561–566.

LAI, Y.-K., HU, S.-M., AND MARTIN, R. R. 2009. Automatic and topology-preserving gradient mesh generation for image vectorization. *ACM Transactions on Graphics 28*, 3, 85:1–8.

LANGER, T., AND SEIDEL, H.-P. 2008. Higher order barycentric coordinates. *Computer Graphics Forum 27*, 2, 459–466.

LIPMAN, Y., LEVIN, D., AND COHEN-OR, D. 2008. Green coordinates. *ACM Transactions on Graphics 27*, 3, 78:1–10.

MANSON, J., AND SCHAEFER, S. 2010. Moving least squares coordinates. *Computer Graphics Forum 29*, 5, 1517–1524.

MANSON, J., LI, K., AND SCHAEFER, S. 2011. Positive Gordon-Wixom coordinates. *Computer Aided Design 43*, 11, 1422–1426.

MEYER, M., LEE, H., BARR, A., AND DESBRUN, M. 2002. Generalized barycentric coordinates on irregular polygons. *Journal of Graphics Tools 7*, 1, 13–22.

ORZAN, A., BOUSSEAU, A., WINNEMÖLLER, H., BARLA, P., THOLLOT, J., AND SALESIN, D. 2008. Diffusion curves: a vector representation for smooth-shaded images. *ACM Transactions on Graphics 27*, 3, 92:1–8.

PINKALL, U., AND POLTHIER, K. 1993. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics 2*, 1, 15–36.

POLYANIN, A. D. 2002. *Handbook of linear partial differential equations for engineers and scientists*. Chapman & Hall / CRC, London.

SUN, J., LIANG, L., WEN, F., AND SHUM, H.-Y. 2007. Image vectorization using optimized gradient meshes. *ACM Transactions on Graphics 26*, 3, 11:1–7.

WACHSPRESS, E. 1975. *A Rational Finite Element Basis*. London: Academic Press.

WEBER, O., AND GOTSMAN, C. 2010. Controllable conformal maps for shape deformation and interpolation. *ACM Transactions on Graphics 29*, 4, 78:1–11.

WEBER, O., BEN-CHEN, M., AND GOTSMAN, C. 2009. Complex barycentric coordinates with applications to planar shape deformation. *Computer Graphics Forum 28*, 2, 587–597.

WEBER, O., BEN-CHEN, M., GOTSMAN, C., AND HORMANN, K. 2011. A complex view of barycentric mappings. *Computer Graphics Forum 30*, 5, 1533–1542.

WEBER, O., PORANNE, R., AND GOTSMAN, C. 2012. Biharmonic coordinates. *Computer Graphics Forum 31*, 8, 2409–2422.

XIA, T., LIAO, B., AND YU, Y. 2009. Patch-based image vectorization with automatic curvilinear feature alignment. *ACM Transactions on Graphics 28*, 5, 115:1–10.

A Proof of Equation 3

Proof: Given the boundary constraints $\mathcal{F}[\theta]$ and $\mathcal{G}[\theta]$ on the unit circle, the unique biharmonic solution of this Diriclet problem can be expressed analytically as follows (refer to [Polyanin 2002]):

$$\begin{aligned}\mathcal{B}[\mathbf{v}] = &\frac{(1-r^2)^2}{2\pi}\int_0^{2\pi}\frac{1-r\cos(\theta-\phi)}{(1+r^2-2r\cos(\theta-\phi))^2}\mathcal{F}[\theta]d\theta \\ &-\frac{(1-r^2)^2}{4\pi}\int_0^{2\pi}\frac{1}{1+r^2-2r\cos(\theta-\phi)}\mathcal{G}[\theta]d\theta,\end{aligned} \tag{8}$$

where $0 \le r \le 1$ and $0 \le \phi < 2\pi$ are the polar coordinates of $\mathbf{v}$.

The first equation in Equation 3 can be obtained by substituting the variable r in Equation 8 with 0.

On the other hand, defferentiating Equation 8, we can get

$$\begin{aligned}
&\frac{\partial}{\partial r}\mathcal{B}[\mathbf{v}] = \frac{-4r}{1-r^2}\mathcal{B}[\mathbf{v}] \\
&+\frac{(1-r^2)^2}{2\pi}\int_0^{2\pi}\frac{4(1-r\cos(\theta-\phi))(\cos(\theta-\phi)-r)}{(1+r^2-2r\cos(\theta-\phi))^3}\mathcal{F}[\theta]d\theta \\
&-\frac{(1-r^2)^2}{2\pi}\int_0^{2\pi}\frac{\cos(\theta-\phi)}{(1+r^2-2r\cos(\theta-\phi))^2}\mathcal{F}[\theta]d\theta \\
&-\frac{(1-r^2)^2}{2\pi}\int_0^{2\pi}\frac{\cos(\theta-\phi)-r}{(1+r^2-2r\cos(\theta-\phi))^2}\mathcal{G}[\theta]d\theta.
\end{aligned}$$

Substituting $r=0$ and $\phi = 0, \pi/2$ in the above equation, we get

$$\begin{aligned}
\frac{\partial}{\partial x}\mathcal{B}|_{\text{origin}} &= \frac{1}{2\pi}\int_0^{2\pi}(3\mathcal{F}[\theta]-\mathcal{G}[\theta])\cos\theta d\theta \\
\frac{\partial}{\partial y}\mathcal{B}|_{\text{origin}} &= \frac{1}{2\pi}\int_0^{2\pi}(3\mathcal{F}[\theta]-\mathcal{G}[\theta])\sin\theta d\theta,
\end{aligned}$$

forming the second equation in Equation 3. □

B Derivation of Equation 5

Using 1D cubic interpolation for $F[r]$, we get

$$\begin{aligned}
F[1] &= \hat{f}[\mathbf{v}] + \hat{\mathbf{g}}\cdot\frac{u}{|u|} + \beta + \gamma \\
F'[1] &= \hat{\mathbf{g}}\cdot\frac{u}{|u|} + 2\beta + 3\gamma,
\end{aligned}$$

where

$$\begin{aligned}
\beta &= \frac{1}{|\mathbf{u}|^2}(3f[t] - g[t]\cdot u - 3\hat{f}[\mathbf{v}] - 2\hat{\mathbf{g}}\cdot u) \\
\gamma &= \frac{1}{|\mathbf{u}|^3}(-2f[t] + g[t]\cdot u + 2\hat{f}[\mathbf{v}] + \hat{\mathbf{g}}\cdot u).
\end{aligned}$$

Substituting the above $F[1]$, $F'[1]$ into Equation 4, and note that $\int_t dC_v = 2\pi$, $\int_t \frac{u}{|u|}dC_v = 0$, $\int_t \frac{u^T u}{|u|^2}dC_v = \pi I_2$, we can get that $\int_t \gamma dC_v = 0$, and $\int_t \beta\frac{u}{|u|}dC_v = 0$, yielding Equation 5.

C Interpolation of derivatives

Let P be a planar polygon, and suppose f and $\mathbf{g}$ are continuous functions defined on P. For each boundary point $\mathbf{p}[t_0]$, we write Equation 5 as

$$\begin{aligned}
&\{\frac{\hat{f}[\mathbf{v}]-f[t_0]}{|\mathbf{d}|}, \hat{\mathbf{g}}_X, \hat{\mathbf{g}}_Y\}^T = \\
&\{-\frac{\mathbf{d}}{|\mathbf{d}|}\cdot\mathbf{g}[t_0], \mathbf{g}[t_0]_X, \mathbf{g}[t_0]_Y\}^T + \mathbf{A}_0^{-1}\mathbf{b}_0,
\end{aligned} \tag{9}$$

where $\mathbf{d} = \mathbf{p}[t_0] - \mathbf{v}$, and

$$\begin{aligned}
\mathbf{A}_0 &= \int_t \begin{pmatrix} 6|\mathbf{d}|^2 & 3|\mathbf{d}|\mathbf{u}_X & 3|\mathbf{d}|\mathbf{u}_Y \\ 3|\mathbf{d}|\mathbf{u}_X & 2\mathbf{u}_X^2 & 2\mathbf{u}_X\mathbf{u}_Y \\ 3|\mathbf{d}|\mathbf{u}_Y & 2\mathbf{u}_X\mathbf{u}_Y & 2\mathbf{u}_Y^2 \end{pmatrix}\frac{1}{|\mathbf{u}|^3}dC_{\mathbf{v}}, \\
\mathbf{b}_0 &= \int_t \begin{pmatrix} (6f_r[t] - 3\mathbf{g}_r[t]\cdot\mathbf{u})|\mathbf{d}| \\ (3f_r[t] - \mathbf{g}_r[t]\cdot\mathbf{u})\mathbf{u}_X \\ (3f_r[t] - \mathbf{g}_r[t]\cdot\mathbf{u})\mathbf{u}_Y \end{pmatrix}\frac{1}{|\mathbf{u}|^3}dC_{\mathbf{v}}.
\end{aligned}$$

where

$$\begin{aligned}
f_r[t] &= f[t] - f[t_0] - (\mathbf{u}-\mathbf{d})\cdot\mathbf{g}[t_0], \\
\mathbf{g}_r[t] &= \mathbf{g}[t] - \mathbf{g}[t_0]
\end{aligned}$$

are the residuals of Taylor expansions of f and $\mathbf{g}$ at $\mathbf{p}[t_0]$.

Because f and $\mathbf{g}$ are continuous, hence for arbitrary $\delta > 0$, there exists some $\epsilon > 0$, s.t.

$$\begin{aligned}
|f_r[t]| < |\mathbf{u}-\mathbf{d}|\delta, \\
|\mathbf{g}_r[t]_X| < \delta,\ \ |\mathbf{g}_r[t]_Y| < \delta
\end{aligned} \tag{10}$$

for all $|\mathbf{p}[t] - \mathbf{p}[t_0]| = |\mathbf{u}-\mathbf{d}| < \epsilon$.

Now assume $\mathbf{v}$ converges towards $\mathbf{p}[t_0]$ from a fixed direction, i.e., $\frac{\mathbf{d}}{|\mathbf{d}|}$ is fixed while $|\mathbf{d}| \to 0$. Note that P is a polygon, so the matrix $\mathbf{A}_0$ in Equation 9 can be written in closed-form in terms of P and $\mathbf{v}$ (see Appendix D), which has the following form when $|\mathbf{d}| \to 0$:

$$\mathbf{A}_0 = \frac{1}{|\mathbf{d}|}\mathbf{A}_1 + \mathbf{A}_r,$$

where $\mathbf{A}_1$ is an invertible matrix that depends on the direction $\frac{\mathbf{d}}{|\mathbf{d}|}$ but not related to $|\mathbf{d}|$, and each element of the residual matrix $\mathbf{A}_r$ is $o(\frac{1}{|\mathbf{d}|})$. Denote the maximum element of $\mathbf{A}_1^{-1}$ by M (a constant depends on the direction $\frac{\mathbf{d}}{|\mathbf{d}|}$), then each element of $\mathbf{A}_0^{-1}$ is bounded by $2M|\mathbf{d}|$.

On the other hand, we can separate the matrix $\mathbf{b}_0$ in Equation 9 into two parts: the first part is the integral over $|\mathbf{p}[t] - \mathbf{p}[t_0]| > \epsilon$, denoted by $\mathbf{b}_{0,P\setminus\epsilon}$, while the second part is the integral over $|\mathbf{p}[t] - \mathbf{p}[t_0]| < \epsilon$, denoted by $\mathbf{b}_{0,\epsilon}$.

For the first part, since $|\mathbf{u}| > |\mathbf{u}-\mathbf{d}| - |\mathbf{d}| > \epsilon - \epsilon/2 = \epsilon/2$, and note that both $f, \mathbf{g}, \mathbf{u}$ are bounded, hence each element of $\mathbf{b}_{0,P\setminus\epsilon}$ is bounded by ϵ^{-3} multiplied by some constant (the maximum constant is denoted by C_1).

For the second part, using Equation 10, we know that each element of $\mathbf{b}_{0,\epsilon}$ is bounded by $\frac{\delta}{|\mathbf{d}|}$ multiplied by some constant (the maximum constant is denoted by C_2).

Therefore, each element of $\mathbf{A}_0^{-1}\mathbf{b}_0 = \mathbf{A}_0^{-1}(\mathbf{b}_{0,P\setminus\epsilon} + \mathbf{b}_{0,\epsilon})$ is bounded by $6M|\mathbf{d}|(\frac{C_1}{\epsilon^3} + \frac{C_2}{|\mathbf{d}|}\delta)$, which is bounded by $6M(C_1 + C_2)\delta$ when $|\mathbf{d}| < \epsilon^3\delta$. This implies that, when $\mathbf{v}$ converges towards $\mathbf{p}[t_0]$ from an arbitrary fixed direction, the elements of $\mathbf{A}_0^{-1}\mathbf{b}_0$ converge to 0. Thus, from Equation 9, we know that

$$\frac{\hat{f}[\mathbf{v}] - f[t_0]}{|\mathbf{d}|} \to -\frac{\mathbf{d}}{|\mathbf{d}|}\cdot\mathbf{g}[t_0],$$

which means $\hat{f}$ interpolates boundary derivatives.

D Deriving closed-form expressions of coordinates

Using the cubic boundary value model and the linear boundary normal derivative model on each edge $\{\mathbf{p}_i, \mathbf{p}_{i+1}\}$, $f[t]$ and $\mathbf{g}[t]$ can be written as

$$\begin{aligned}
f[t] &= \begin{bmatrix} f_i \\ f_i^+ \\ f_{i+1} \\ f_{i+1}^- \end{bmatrix}^T \begin{bmatrix} 1 & 0 & -3 & 2 \\ 0 & L_i & -2 & L_i \\ 0 & 0 & 3 & -2 \\ 0 & 0 & 1 & -L_i \end{bmatrix} \begin{bmatrix} 1 \\ t_i \\ t_i^2 \\ t_i^3 \end{bmatrix}, \\
\mathbf{g}[t] &= \begin{bmatrix} f_i \\ f_i^+ \\ f_{i+1} \\ f_{i+1}^- \end{bmatrix}^T \begin{bmatrix} 0 & -6/L_i & 6/L_i \\ 1 & -4 & 3 \\ 0 & 6/L_i & -6/L_i \\ 0 & 2 & -3 \end{bmatrix} \begin{bmatrix} 1 \\ t_i \\ t_i^2 \end{bmatrix} \mathbf{t}_i \\
&\quad + \begin{bmatrix} h_i^+ \\ h_{i+1}^- \end{bmatrix}^T \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ t_i \end{bmatrix} \mathbf{n}_i,
\end{aligned}$$

where $L_i = |\mathbf{p}_i - \mathbf{p}_{i+1}|$, $t_i = |\mathbf{p}[t] - \mathbf{p}_i|/|\mathbf{p}_{i+1} - \mathbf{p}_i|$, $\mathbf{t}_i = (\mathbf{t}_{i_X}, \mathbf{t}_{i_Y}) = (\mathbf{p}_{i+1} - \mathbf{p}_i)/L_i$, and $\mathbf{n}_i = (\mathbf{n}_{i_X}, \mathbf{n}_{i_Y})$ is the outward unit normal vector of $\{\mathbf{p}_i, \mathbf{p}_{i+1}\}$.

For notational simplicity, we introduce the following integrals:

$$\mathcal{Z}^i_{k,m,n} = \int_{t_i=0}^{1} t_i^k \frac{\mathbf{u}_X^m \mathbf{u}_Y^n}{|\mathbf{u}|^3} dC_v,$$

where subscripts $k \in \{0,1,2,3\}$, $m \in \{0,1,2\}$, $n \in \{0,1,2\}$, $m+n \leq 2$, $k+m+n \leq 4$. Note that $\mathcal{Z}^i_{k,m,n}$ are constants depending on the geometry of P and $\mathbf{v}$.

The elements of matrix $\mathbf{A}$ can be now written as

$$\mathbf{A} = \sum_i \begin{pmatrix} 6\mathcal{Z}^i_{0,0,0} & 3\mathcal{Z}^i_{0,1,0} & 3\mathcal{Z}^i_{0,0,1} \\ 3\mathcal{Z}^i_{0,1,0} & 2\mathcal{Z}^i_{0,2,0} & 2\mathcal{Z}^i_{0,1,1} \\ 3\mathcal{Z}^i_{0,0,1} & 2\mathcal{Z}^i_{0,1,1} & 2\mathcal{Z}^i_{0,0,2} \end{pmatrix},$$

and the elements of the vector $\mathbf{b}$ can be written as the following weighted sums of the constraints $f_i, f_i^{\pm}, h_i^{\pm}$,

$$\mathbf{b} = \sum_i \begin{pmatrix} a_{i,0}[\mathbf{v}] f_i \\ a_{i,1}[\mathbf{v}] f_i \\ a_{i,2}[\mathbf{v}] f_i \end{pmatrix} + \sum_{i,s} \begin{pmatrix} b^s_{i,0}[\mathbf{v}] f^s_i + c^s_{i,0}[\mathbf{v}] h^s_i \\ b^s_{i,1}[\mathbf{v}] f^s_i + c^s_{i,1}[\mathbf{v}] h^s_i \\ b^s_{i,2}[\mathbf{v}] f^s_i + c^s_{i,2}[\mathbf{v}] h^s_i \end{pmatrix},$$

where the coefficients $a_{i,j}, b^s_{i,j}, c^s_{i,j}$ are:

$$\begin{aligned}
a_{i,j} &= U_j(\mathcal{Z}^i_{0,m_j,n_j} - 3\mathcal{Z}^i_{2,m_j,n_j} + 2\mathcal{Z}^i_{3,m_j,n_j}) \\
&\quad + 6V_j \mathbf{t}_{iX}/L_i(\mathcal{Z}^i_{2,m_j+1,n_j} - \mathcal{Z}^i_{3,m_j+1,n_j}) \\
&\quad + 6V_j \mathbf{t}_{iY}/L_i(\mathcal{Z}^i_{2,m_j,n_j+1} - \mathcal{Z}^i_{3,m_j,n_j+1}) \\
&\quad + U_j(3\mathcal{Z}^{i-1}_{2,m_j,n_j} - 2\mathcal{Z}^{i-1}_{3,m_j,n_j}) \\
&\quad - 6V_j \mathbf{t}_{i-1X}/L_{i-1}(\mathcal{Z}^{i-1}_{2,m_j+1,n_j} - \mathcal{Z}^{i-1}_{3,m_j+1,n_j}) \\
&\quad - 6V_j \mathbf{t}_{i-1Y}/L_{i-1}(\mathcal{Z}^{i-1}_{2,m_j,n_j+1} - \mathcal{Z}^{i-1}_{3,m_j,n_j+1}), \\
b^+_{i,j} &= U_j(L_i\mathcal{Z}^i_{1,m_j,n_j} - 2\mathcal{Z}^i_{2,m_j,n_j} + L_i\mathcal{Z}^i_{3,m_j,n_j}) \\
&\quad - V_j \mathbf{t}_{iX}(\mathcal{Z}^i_{1,m_j+1,n_j} - 4\mathcal{Z}^i_{2,m_j+1,n_j} + 3\mathcal{Z}^i_{3,m_j+1,n_j}) \\
&\quad - V_j \mathbf{t}_{iY}(\mathcal{Z}^i_{1,m_j,n_j+1} - 4\mathcal{Z}^i_{2,m_j,n_j+1} + 3\mathcal{Z}^i_{3,m_j,n_j+1}), \\
b^-_{i,j} &= U_j(\mathcal{Z}^{i-1}_{2,m_j,n_j} - L_{i-1}\mathcal{Z}^{i-1}_{3,m_j,n_j}) \\
&\quad - V_j \mathbf{t}_{i-1X}(2\mathcal{Z}^{i-1}_{2,m_j+1,n_j} - 3\mathcal{Z}^{i-1}_{3,m_j+1,n_j}) \\
&\quad - V_j \mathbf{t}_{i-1Y}(2\mathcal{Z}^{i-1}_{2,m_j,n_j+1} - 3\mathcal{Z}^{i-1}_{3,m_j,n_j+1}), \\
c^+_{i,j} &= V_j \mathbf{n}_{iX}(\mathcal{Z}^i_{1,m_j+1,n_j} - \mathcal{Z}^i_{0,m_j+1,n_j}) \\
&\quad + V_j \mathbf{n}_{iY}(\mathcal{Z}^i_{1,m_j,n_j+1} - \mathcal{Z}^i_{0,m_j,n_j+1}), \\
c^-_{i,j} &= -V_j \mathbf{n}_{i-1X}\mathcal{Z}^{i-1}_{1,m_j+1,n_j} - V_j \mathbf{n}_{i-1Y}\mathcal{Z}^{i-1}_{1,m_j,n_j+1},
\end{aligned}$$

where $(m_0, n_0) = (0,0), (m_1, n_1) = (1,0), (m_2, n_2) = (0,1)$, and $U_0 = 6, U_1 = U_2 = 3, V_0 = 3, V_1 = V_2 = 1$.

Finally, the CMV coordinates a_i, b^s_i, c^s_i in Equation 7 have the form

$$\begin{aligned}
a_i &= \{1,0,0\}\mathbf{A}^{-1}\{a_{i,0}, a_{i,1}, a_{i,2}\}^T \\
b^s_i &= \{1,0,0\}\mathbf{A}^{-1}\{b^s_{i,0}, b^s_{i,1}, b^s_{i,2}\}^T \\
c^s_i &= \{1,0,0\}\mathbf{A}^{-1}\{b^s_{i,0}, c^s_{i,1}, c^s_{i,2}\}^T.
\end{aligned}$$

We next give closed-form expressions for integrals $\mathcal{Z}^i_{k,m,n}$. To do so, we move to the complex plane and treat each vector as a complex number. Let $\mathbf{u}_i = \mathbf{p}_i - \mathbf{v}$, j be the imaginary unit, $z = \frac{\mathbf{u}}{|\mathbf{u}|}$, $\mu = \frac{j}{2}\frac{\overline{\mathbf{u}_i}}{\mathrm{Im}(\mathbf{u}_i\overline{\mathbf{u}_{i+1}})}$, and $\kappa = \frac{j}{2}\frac{\overline{\mathbf{u}_i} - \overline{\mathbf{u}_{i+1}}}{\mathrm{Im}(\mathbf{u}_i\overline{\mathbf{u}_{i+1}})}$. We have $\frac{1}{|\mathbf{u}|} = \kappa z + \overline{\kappa}\frac{1}{z}$, $t_i = (\mu z + \overline{\mu}\frac{1}{z})/(\kappa z + \overline{\kappa}\frac{1}{z})$.

Consider the following complex integrals:

$$\mathcal{C}_{k,m,n} = \int_{z_i}^{z_{i+1}} z^m \frac{1}{jz}(\mu z + \overline{\mu}\frac{1}{z})^n(\kappa z + \overline{\kappa}\frac{1}{z})^{k-n} dz,$$

where $z_i = \frac{\mathbf{u}_i}{|\mathbf{u}_i|}$, $z_{i+1} = \frac{\mathbf{u}_{i+1}}{|\mathbf{u}_{i+1}|}$. The integrals $\mathcal{Z}^i_{k,m,n}$ can be reduced to $\mathcal{C}_{k,m,n}$ by

$$\begin{aligned}
\mathcal{Z}^i_{k,0,0} &= \mathcal{C}_{3,0,k} \\
\mathcal{Z}^i_{k,1,0} &= \mathrm{Re}[\mathcal{C}_{2,1,k}] \\
\mathcal{Z}^i_{k,0,1} &= \mathrm{Im}[\mathcal{C}_{2,1,k}] \\
\mathcal{Z}^i_{k,1,1} &= \frac{1}{2}\mathrm{Im}[\mathcal{C}_{1,2,k}] \\
\mathcal{Z}^i_{k,2,0} &= \frac{1}{2}(\mathcal{C}_{1,0,k} + \mathrm{Re}[\mathcal{C}_{1,2,k}]) \\
\mathcal{Z}^i_{k,0,2} &= \frac{1}{2}(\mathcal{C}_{1,0,k} - \mathrm{Re}[\mathcal{C}_{1,2,k}]),
\end{aligned}$$

and $\mathcal{C}_{k,m,n}$ have the following closed-form expressions:

$$\begin{aligned}
\mathcal{C}_{3,0,0} &= 2\mathrm{Re}[-j(\kappa^3\mathcal{I}_0 + 3\kappa^2\overline{\kappa}\mathcal{I}_1)] \\
\mathcal{C}_{3,0,1} &= 2\mathrm{Re}[-j(\kappa^2\mu\mathcal{I}_0 + (\kappa^2\overline{\mu} + 2\kappa\overline{\kappa}\mu)\mathcal{I}_1)] \\
\mathcal{C}_{3,0,2} &= 2\mathrm{Re}[-j(\mu^2\kappa\mathcal{I}_0 + (\mu^2\overline{\kappa} + 2\mu\overline{\mu}\kappa)\mathcal{I}_1)] \\
\mathcal{C}_{3,0,3} &= 2\mathrm{Re}[-j(\mu^3\mathcal{I}_0 + 3\mu^2\overline{\mu}\mathcal{I}_1)] \\
\mathcal{C}_{2,1,0} &= -j(\kappa^2\mathcal{I}_0 + 2\kappa\overline{\kappa}\mathcal{I}_1 + \overline{\kappa}^2\mathcal{I}_2) \\
\mathcal{C}_{2,1,1} &= -j(\kappa\mu\mathcal{I}_0 + (\kappa\overline{\mu} + \mu\overline{\kappa})\mathcal{I}_1 + \overline{\kappa\mu}\mathcal{I}_2) \\
\mathcal{C}_{2,1,2} &= -j(\mu^2\mathcal{I}_0 + 2\mu\overline{\mu}\mathcal{I}_1 + \overline{\mu}^2\mathcal{I}_2) \\
\mathcal{C}_{2,1,3} &= -j(\frac{\mu^3}{\kappa}\mathcal{I}_0 + \frac{\overline{\mu}^3}{\overline{\kappa}}\mathcal{I}_2) + (3\mu^2\overline{\mu} - \frac{\mu^3\overline{\kappa}}{\kappa})\mathcal{H}_1 \\
&\quad + (3\mu\overline{\mu}^2 - \frac{\overline{\mu}^3\kappa}{\overline{\kappa}})\mathcal{H}_0 \\
\mathcal{C}_{1,0,0} &= 2\mathrm{Re}[-j\kappa\mathcal{I}_1] \\
\mathcal{C}_{1,0,1} &= 2\mathrm{Re}[-j\mu\mathcal{I}_1] \\
\mathcal{C}_{1,0,2} &= 2\mathrm{Re}[-j\frac{\mu^2}{\kappa}\mathcal{I}_1] + 2(\mu\overline{\mu} - \mathrm{Re}[\frac{\mu^2\overline{\kappa}}{\kappa}])\mathcal{H}_0 \\
\mathcal{C}_{1,2,0} &= -j(\kappa\mathcal{I}_0 + \overline{\kappa}\mathcal{I}_1) \\
\mathcal{C}_{1,2,1} &= -j(\mu\mathcal{I}_0 + \overline{\mu}\mathcal{I}_1) \\
\mathcal{C}_{1,2,2} &= -j(\frac{\mu^2}{\kappa}\mathcal{I}_0 + \frac{\overline{\mu}^2}{\overline{\kappa}}\mathcal{I}_1) + 2(\mu\overline{\mu} - \mathrm{Re}[\frac{\mu^2\overline{\kappa}}{\kappa}])\mathcal{H}_1,
\end{aligned}$$

where $\mathcal{I}_0 = (z_{i+1}^3 - z_i^3)/3$, $\mathcal{I}_1 = z_{i+1} - z_i$, $\mathcal{I}_2 = -\overline{\mathcal{I}_1}$, and $\mathcal{H}_0 = \frac{1}{|\kappa|}(tan^{-1}(\frac{\kappa}{|\kappa|}z_{i+1}) - tan^{-1}(\frac{\kappa}{|\kappa|}z_i))$, $\mathcal{H}_1 = \frac{1}{\kappa}\mathcal{I}_1 - \frac{\overline{\kappa}}{\kappa}\mathcal{H}_0$.