Computer-Aided Design 43 (2011) 1422–1426

Contents lists available at SciVerse ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Positive Gordon–Wixom coordinates

Josiah Manson [a,*], Kuiyu Li [b], Scott Schaefer [a]

[a] Texas A&M University, 3112 Texas A&M University, College Station, TX 77843, United States

[b] Intel, United States

ARTICLE INFO

Keywords:
Barycentric coordinates
Transfinite
Interpolant

ABSTRACT

We introduce a new construction of transfinite barycentric coordinates for arbitrary closed sets in two dimensions. Our method extends weighted Gordon–Wixom interpolation to non-convex shapes and produces coordinates that are positive everywhere in the interior of the domain and that are smooth for shapes with smooth boundaries. We achieve these properties by using the distance to lines tangent to the boundary curve to define a weight function that is positive and smooth. We derive closed-form expressions for arbitrary polygons in two dimensions and compare the basis functions of our coordinates with several other types of barycentric coordinates.

1. Introduction

Barycentric coordinates provide a method of interpolating values from the boundary of a domain over its interior. They are useful in a variety of applications including finite element analysis [1–3], texture mapping [4], deformation [5], image compositing [6], volumetric texture synthesis [7], shading [8], and geometric modeling [9]. Barycentric coordinates can also be used to approximate solutions to Poisson problems [6]. In fact, the harmonic basis [10], which represents the solutions to a Poisson problem for different boundary values, is a type of barycentric basis. In this paper, we focus on transfinite barycentric coordinates, which are coordinates defined within domains that have smooth boundaries.

Suppose that we are given a domain Ω with a smooth boundary and boundary values sampled from a function g. We wish to find a transfinite barycentric basis $b(x, y)$ so that any function in the subspace defined by b has the form

$$f(x) = \int_{\partial\Omega} b(x, y)g(y)\mathrm{d}y, \tag{1}$$

where $x \in \Omega$ and $y \in \partial\Omega$. The basis should be positive,

$$b(x, y) \geq 0,$$

and should interpolate boundary values:

$$x \in \partial\Omega \Rightarrow b(x, y) = \begin{cases} 1, & \text{if } x = y \\ 0, & \text{if } x \neq y. \end{cases}$$

Furthermore, the basis should have linear precision, which means that the interpolant in Eq. (1) has the property that,

$$\forall g \in \mathbb{P}^1, \quad f(x) = g(x).$$

The interpolant should also be smooth within the domain:

$$x \in \Omega \setminus \partial\Omega \Rightarrow f(x, y) \in C^1.$$

We introduce an extension of transfinite Gordon–Wixom interpolation [11] that satisfies all of these properties for domains with smooth boundaries. Our implementation approximates smooth shapes by polygons, for which we present closed-form expressions.

1.1. Related work

Barycentric coordinates have existed since 1827 [12], and were originally defined to interpolate values over simplices. For each vertex, multiplying the basis function at each vertex by the value at that vertex defines an interpolant. For polygons other than triangles, there are many possible barycentric interpolants. Wachspress created an interpolant over two-dimensional convex polygons for solving finite-element problems [1]. Floater et al. [13] extended Wachspress coordinates into a family of related coordinates. However, this family of coordinates is only defined over convex polygons. Mean value coordinates (MVCs) [14,15] were the first barycentric coordinates to be defined over concave polygons, but they may have negative values. Lipman et al. [16] modified MVCs to ensure that the coordinates are always positive by only considering the boundary that is visible from the point of evaluation, but their coordinates are not smooth.

Joshi et al. [10] showed that solving a Poisson equation yields barycentric coordinates dubbed harmonic coordinates. This method partitions the domain into a dense set of triangles

* Corresponding author. Tel.: +1 9794026703.
E-mail address: josiahmanson@gmail.com (J. Manson).

0010-4485/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cad.2011.08.019

and solves a Laplacian partial differential equation (PDE) with values from a linear basis function as boundary conditions, which guarantees that the basis interpolates boundary values. Unlike other barycentric coordinates, harmonic coordinates depend on the geodesic distance rather than the Cartesian distance, which improves the locality of the basis functions but makes harmonic coordinates expensive to compute.

Although harmonic coordinates are guaranteed to be positive and smooth, computing them requires partitioning the domain and solving a large linear system. Hormann and Sukumar avoid some of the computational problems of harmonic coordinates with their maximum entropy coordinates (MECs) [17], which are also positive. MECs are more efficient to calculate than harmonic coordinates, because an error function is minimized independently for each evaluation point rather than minimizing a global function. Even so, MECs require iterative minimization of a nonlinear function for each point.

Manson and Schaefer [18] showed that the affine construction of image deformation using moving least squares [19] forms barycentric coordinates that they called moving least squares coordinates (MLSCs). The authors generalized the MLSC construction to create higher-order interpolants, but do not guarantee positivity.

Some methods have extended barycentric coordinates to curved (transfinite) boundaries. Several methods [20,21] generalize barycentric coordinates to arbitrary convex sets that are bounded by a parameterized curve. In his theoretical analysis of barycentric coordinates [22], Belyaev proposed a generalization of Gordon–Wixom coordinates to concave domains. He showed that the transfinite extensions of MVCs and Wachspress coordinates are instances of generalized Gordon–Wixom coordinates. A disadvantage of these coordinates is that they can become negative in concave domains.

2. Gordon–Wixom coordinates

In 1974, Gordon and Wixom [11] proposed an elegant construction of transfinite barycentric coordinates for general convex domains in two dimensions. In convex domains, there are always exactly two intersection points. The point y_1 is the point that is intersected along the forward ray, and y_{-1} is along the backward ray. Our notation, shown in Fig. 1, is that the y_i are the points on $\partial\Omega$ intersected by the line passing through x at angle θ and $d_i = \|y_i - x\|$. Gordon and Wixom defined an interpolating function for any point $x \in \Omega$ as the average of values that are linearly interpolated from points on the boundary. If we define the linear interpolant between points y_i and y_j on the boundary as

$$L_{i,j}(x,\theta) = \frac{d_j}{d_i + d_j} g(y_i) + \frac{d_i}{d_i + d_j} g(y_j),$$

the average over all angles is expressed as an integral by

$$f(x) = \frac{1}{2\pi} \int_0^{2\pi} L_{-1,1}(x,\theta)\, \mathrm{d}\theta. \tag{2}$$

Belyaev [22] generalized the above Gordon–Wixom interpolation function to its weighted version as

$$f(x) = \frac{\int_0^{2\pi} L_{-1,1}(x,\theta) W_{-1,1}(x,\theta)\, \mathrm{d}\theta}{\int_0^{2\pi} W_{-1,1}(x,\theta)\, \mathrm{d}\theta}, \tag{3}$$

where $W_{i,j}(x,\theta)$ gives the weight in a weighted average. Any weighted Gordon–Wixom interpolant can be rewritten in the form of basis functions as shown in Eq. (1). Notice that the original Gordon–Wixom function in Eq. (2) is a special case of the weighted version, where $W_{-1,1}(x,\theta) = 1$. Also notice that Eq. (3) defines

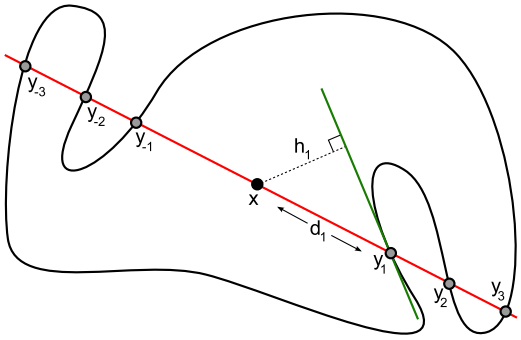

Fig. 1. Notation for positive Gordon–Wixom interpolation.

mean value coordinates [14] when setting $W_{-1,1}(x,\theta) = \frac{d_1 + d_{-1}}{d_1 d_{-1}}$, because

$$f(x) = \frac{\int_0^{2\pi} \left(\frac{g(y_1)}{d_1} + \frac{g(y_{-1})}{d_{-1}} \right) \mathrm{d}\theta}{\int_0^{2\pi} \left(\frac{1}{d_1} + \frac{1}{d_{-1}} \right) \mathrm{d}\theta} = \frac{\int_0^{2\pi} \frac{g(y_1)}{d_1}\, \mathrm{d}\theta}{\int_0^{2\pi} \frac{1}{d_1}\, \mathrm{d}\theta},$$

which is exactly the mean value interpolant.

Belyaev [22] also noticed that the same way that Hormann and Floater extended MVCs to concave polygons [15] could be used in transfinite domains. Namely, alternating intersections are given negative weights. Summing over all intersections of a ray at a given angle, the coordinates are found by

$$f(x) = \frac{\int_0^{2\pi} \sum_{i=1}^{m} g(y_i) \frac{(-1)^{i+1}}{d_i}\, \mathrm{d}\theta}{\int_0^{2\pi} \sum_{i=1}^{m} \frac{(-1)^{i+1}}{d_i}\, \mathrm{d}\theta},$$

where $\{y_1, y_2, \ldots, y_n\}$ is the sequence of points where the ray intersects the boundary. Like the polygonal form of MVCs, this construction can generate negative coordinates. Negative coordinates will cause non-intuitive deformations in cage-based deformation because $f(x)$ is not guaranteed to be in the convex hull of $g(y)$. Negative coordinates can also cause finite-element calculations to become unstable.

3. Positive Gordon–Wixom coordinates

We introduce a new form of Gordon–Wixom coordinates that are guaranteed to be positive and smooth for domains with smooth boundaries. As with MVCs, we sum the contribution of multiple intersections for a given parameter θ, but we cannot split our line integral into a ray integral. This means that our interpolant uses all intersection points, both ahead and behind of x, on lines passing through x. Our interpolant has the form

$$f(x) = \frac{\int_0^{2\pi} \sum_{i=1}^{m} \sum_{j=-n}^{-1} L_{i,j}(x,\theta) W_{i,j}(x,\theta)\, \mathrm{d}\theta}{\int_0^{2\pi} \sum_{i=1}^{m} \sum_{j=-n}^{-1} W_{i,j}(x,\theta)\, \mathrm{d}\theta}, \tag{4}$$

where $\{y_{-m}, y_{1-m}, \ldots, y_{-1}\}$ are the set of boundary intersections before x and $\{y_1, y_2, \ldots, y_n\}$ are the intersections after x on the line passing through x at angle θ, as shown in Fig. 1. We have found that, under this construction, it is possible to create an interpolant that is both positive and smooth by weighting pairs of points by

$$W_{i,j}(x,\theta) = \frac{(d_i + d_j) h_i h_j}{d_i^2 d_j^2}.$$

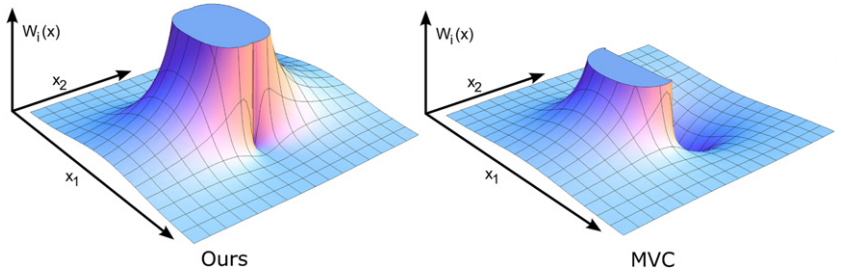

Fig. 2. Our weight function, $W_i(x)$, from Eq. (5), compared with the equivalent MVC weight function for an edge swept by y_i. These functions approach infinity along the edge, and we show the graphs clamped at a finite value.

This weight function introduces a new variable, h_i, for the unsigned distance of x to the line tangent to the boundary at y_i. Notice that we can decompose $W_{i,j}$ as

$$W_{i,j}(x,\theta) = (d_i + d_j)W_i(x,\theta)W_j(x,\theta),$$

where

$$W_i(x,\theta) = \frac{h_i}{d_i^2}$$

is the weight of a point on the boundary. If we can show that W_i is smooth for all boundary points, then $W_{i,j}$ must also be smooth, because the product of smooth functions is also smooth. To calculate the contribution of a line segment with weight $W_i(x,\theta)$ at point x, we integrate $W_i(x,\theta)$ over a circle:

$$W_i(x) = \int_0^{2\pi} W_i(x,\theta)\,\mathrm{d}\theta, \tag{5}$$

where $W_i(x,\theta) = 0$ if the ray from x with angle θ does not intersect the line segment. In MVCs, the weight given to a line segment is $(-1)^{i+1}/d_i$, which is equivalent to the inverse signed distance to an oriented boundary. It is because this weight function becomes negative in MVCs that MVCs have negative basis functions. We compare $W_i(x)$ with the equivalent function in MVCs in Fig. 2. Notice that $W_i(x)$ approaches infinity at the line segment, but is zero along the extension of the line segment. Our function is positive and smooth everywhere except for on the line segment itself, whereas the weight function of the MVCs approaches positive and negative infinity on opposite sides of the segment.

Our interpolant is expressed as a form of weighted Gordon–Wixom coordinates, and so it is possible to write the interpolant in terms of basis functions. Specifically, the basis function of a point y_i is given by

$$b(x,y_i) = \frac{\sum_{j=-n}^{-1} \frac{h_i h_j}{d_i^2 d_j}}{\int_0^{2\pi} \sum_{i=1}^{m} \sum_{j=-n}^{-1} W_{i,j}(x,\theta)\,\mathrm{d}\theta}. \tag{6}$$

We can now discuss the properties of our basis. First, our basis is interpolatory, because $W_{i,j}(x,\theta) = \infty$ only when $x = y_i$. Thus, as x approaches a point on the boundary, the weighted contribution from that point dominates. Furthermore, our basis has linear precision, because any weighted combination of linear functions $L_{i,j}(x,\theta)$ is also linear. Also notice that $b(x,y)$ is positive, because $W_{i,j}(x,\theta)$ is positive.

Showing that our basis is smooth requires a little more explanation. Looking at Eq. (6), one can see that we must consider smoothness both when the number of intersection points remains the same and whenever the number of intersection points in the summations changes. For a smooth boundary, the number of intersection points will change on lines tangent to the boundary curve. For any pair of points on the boundary, we must consider the four cases shown in Fig. 3.

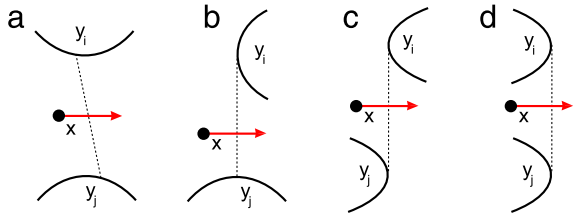

Fig. 3. The four cases for smoothness between line segment pairs. Red arrows show the movement of points. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

In case (a), there is no change in intersection points, so the smoothness of b depends only on the smoothness of $W_{i,j}$. If $\partial\Omega$ has smoothness of C^a, $W_{i,j}$ will have smoothness of C^{a-1}. Although d_i and d_j have smoothness of C^a, the variables h_i and h_j depend on the derivative of the boundary and have reduced smoothness of C^{a-1}. However, integrating over θ increases the smoothness of the interpolant to C^a. Therefore, in case (a), the smoothness of f is equal to the smoothness of $\partial\Omega$.

In cases (b), (c), and (d), the number of intersection points changes, but the number of intersection points changes only on lines tangent to $\partial\Omega$. Case (b) is the easiest to see as being smooth, because we already know that the contribution from W_j is smooth. The smoothness from the curve at point y_i, however, depends on W_i becoming zero continuously on the tangent line. One can see that W_i is continuous because, in the numerator, h_i is approximately linear and the denominator d_i is approximately constant. Again, our interpolant integrates over θ, so the interpolant is smooth. In case (c), the weight from one edge goes to zero while the other increases from zero, which means that case (c) is also smooth. In case (d), the weight function is the product of two functions h_i and h_j that approach zero, and is also smooth.

3.1. Basis functions of B-splines

While we discuss our barycentric coordinate construction in the context of B-splines, we can generate a finite set of coordinates for any parametric curve defined by basis functions, such as Catmull–Rom splines. B-splines are a common representation of smooth, closed curves that are piecewise, parametric polynomials and are defined by a set of control points. Every point on a B-spline is calculated as a weighted combination of control points C_i, where the weights are given by the B-spline basis function $B_i(t)$ associated with the control point. Thus, a point on the curve $y(t)$ parameterized by t is calculated as

$$y(t) = \sum_i B_i(t)C_i.$$

Similarly, the function values along the curve are specified by control values G_i as

$$g(t) = \sum_i B_i(t)G_i.$$

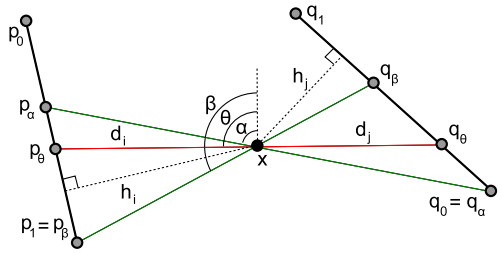

Fig. 4. The visibility cone $[\alpha, \beta]$ between two edges (p_0, p_1) and (q_0, q_1).

Because a *B*-spline is a weighted combination of control points, a curve has as many degrees of freedom as there are control points. Similarly, the transfinite basis function $b(x, y)$ of the curve has as many degrees of freedom as there are control points, and can be condensed to basis functions associated with the control points. This allows us to write an interpolant of boundary values as

$$f(x) = \sum_i b_i(x) G_i.$$

The basis function $b_i(x)$ for the ith control point is simply the weighted combination of basis functions around the boundary:

$$b_i(x) = \int_{\partial\Omega} B_i(t) b(x, y(t))\, \mathrm{d}t.$$

3.2. Numerical approximation

Due to the complexity of intersection calculations for smooth curves, we are unable to find a closed-form solution for *B*-splines. Instead, we build a discrete approximation of the boundary as a polygon $P = \{P_1, \ldots, P_\ell\}$ from the *B*-spline control points $C = \{C_1, \ldots, C_d\}$. Because the curve is a weighted combination of control points, we can write that $P = SC$, where S is a $d \times \ell$ matrix. We can then calculate the barycentric coordinates $\hat{b}(x) = \{\hat{b}_1(x), \ldots, \hat{b}_d(x)\}$ with respect to the vertices of P. We then map the polygon's basis functions $\hat{b}(x)$ to the basis functions $b(x)$ of the *B*-spline control points. Because S encodes the *B*-spline weights around the curve, we can write the mapping compactly as

$$b(x) = S^T \hat{b}(x). \tag{7}$$

The only step that remains is to show how basis functions are calculated for a polygon. Given a polygon, we can compute Eq. (4) by explicitly calculating whether each pair of edges contributes to the summations and, if the pair does contribute, we compute the exact integral. Given a pair of edges p and q, the pair will contribute to the integral if the projection of q onto the line formed by p through x overlaps with p. We use $[\alpha, \beta]$ to denote the range of angles over which the edges overlap.

To clarify our notation, we label the variables that we use in Fig. 4. The starting and ending points of edge p are p_0 and p_1, and we refer to the point where a line at angle θ intersects the edge p as $p_\theta = (1 - s_\theta)p_0 + s_\theta p_1$. We label the starting and ending points in the range $[\alpha, \beta]$ as p_α and p_β. We use a similar notation for the opposite edge q, which we parameterize by t_θ instead of s_θ.

The derivations of the basis functions are symmetric, so we show the derivation of only one equation and then give the replacement rules to find the other three equations. The points p_α and p_β can be written as linear combinations of the vertices of edge p in the form

$$p_\alpha = (1 - s_\alpha)p_0 + s_\alpha p_1$$
$$p_\beta = (1 - s_\beta)p_0 + s_\beta p_1.$$

This relates the weights calculated on the edge (p_α, p_β) to the weights on the edge (p_0, p_1). Using the above equations, and solving for the weights in $\hat{b}_{p_0} p_0 + \hat{b}_{p_1} p_1 = \hat{b}_{p_\alpha} p_\alpha + \hat{b}_{p_\beta} p_\beta$, we find that

$$\hat{b}_{p_0} = (1 - t_\alpha)\hat{b}_{p_\alpha} + (1 - t_\beta)\hat{b}_{p_\beta}$$
$$\hat{b}_{p_1} = t_\alpha \hat{b}_{p_\alpha} + t_\beta \hat{b}_{p_\beta}. \tag{8}$$

The weights along the edge (p_α, p_β) are calculated by integrating the boundary values associated with each vertex:

$$\hat{b}_{p_\alpha} = \int_\alpha^\beta \frac{(1 - s_\theta) d_i h_i h_j}{d_i^2 d_j^2}\, \mathrm{d}\theta$$
$$\hat{b}_{p_\beta} = \int_\alpha^\beta \frac{s_\theta d_j h_i h_j}{d_i^2 d_j^2}\, \mathrm{d}\theta.$$

Notice that h_i and h_j are constant, but that d_i, d_j, and s_θ are functions of θ. By writing the coordinates of p_α and p_β in polar form relative to x, we find that

$$s_\theta = \frac{d_{p_\alpha} \sin(\theta - \alpha)}{d_{p_\alpha} \sin(\theta - \alpha) + d_{p_\beta} \sin(\beta - \theta)}$$
$$d_i = \frac{d_{p_\alpha} d_{p_\beta} \sin(\beta - \alpha)}{d_{p_\alpha} \sin(\theta - \alpha) + d_{p_\beta} \sin(\beta - \theta)},$$

where d_{p_α} and d_{p_β} are the distances from x to p_α and p_β. Evaluating this integral, we find that

$$\hat{b}_{p_\alpha} = \frac{d_{p_\alpha}(d_{q_\alpha} + d_{q_\beta}) + d_{p_\beta}(d_{q_\alpha} + 2d_{q_\beta}) + d_{p_\beta} d_{q_\beta} \cos(\beta - \alpha)}{h_i^{-1} h_j^{-1} 6 d_{p_\alpha}^2 d_{p_\beta} d_{q_\alpha} d_{q_\beta} \cot\left(\frac{\beta - \alpha}{2}\right)}. \tag{9}$$

The previous formulas are similar for opposite points and edges, because the equations are symmetric. The formulas for weight of p_β rather than p_α are found by switching the α's and β's in Eq. (9). Similarly, formulas for edge q rather than p are found by switching p's with q's and i's with j's. With the basis function of the edges in the range $[\alpha, \beta]$ computed, we calculate the contribution of spans to polygon edges by substituting the span weights into Eq. (8). Note that we do not compute edge intersections, which may be numerically unstable, but find mutual visibility between line segments using angles. We have not noticed any numerical stability problems with this approach.

After summing the contribution of all pairs of edges that are mutually visible through x, we must still account for the normalizing factor in Eq. (4). Fortunately, normalization is easy. The sum of basis functions is unity because f has linear precision, so the denominator of Eq. (4) is simply the sum of the basis functions at x. With the basis functions $\hat{b}$ of the polygon vertices now computed, we calculate the basis functions of the control points by evaluating Eq. (7).

4. Results

Any form of barycentric coordinates can be calculated over a smooth boundary curve when the curve is approximated as a polygon, using Eq. (7). We compare our new coordinates against several well-known types of barycentric coordinates in two categories. Harmonic coordinates and maximum entropy coordinates (MECs) are positive, but are difficult to calculate. Moving least squares coordinates (MLSCs) and mean value coordinates (MVCs) have analytic solutions, but can become negative in concave domains.

In our examples, we show iso-contours for basis functions at increments of 0.1. The domain is bounded by a quadratic *B*-spline shown in black, with a control polygon shown in green. We draw zero contours of basis functions in blue, and we draw contours between 0 and 1 in red; the vertex of the control polygon

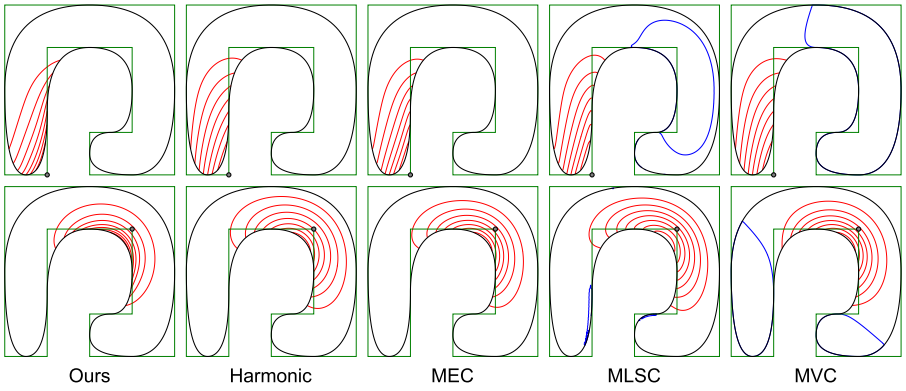

Fig. 5. Comparison between several methods of two different basis functions (top and bottom rows) for a non-convex shape. Contour lines are drawn at multiples of $\frac{1}{10}$, where red lines are positive, and blue lines are at zero. We assemble the basis functions of the control points from basis functions of the subdivided polygon using Eq. (7). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

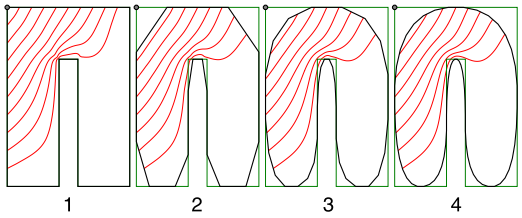

Fig. 6. As the resolution of the boundary increases, our numerical solution converges. The original control polygon is shown first, followed by successive levels of quadratic *B*-spline subdivision. The number of subdivisions is shown below each polygon. Contours are drawn at multiples of $\frac{1}{10}$.

that is associated with the basis function is drawn as a black and gray dot. We used five subdivisions of the control polygon in Fig. 5, where we compare the basis functions calculated using different methods. Note that our method generates positive basis functions with low curvature and smooth contours, like harmonic coordinates and MECs. However, our method does not rely on any optimization (global or local) and generates a geometric construction for positive barycentric coordinates. On the other hand, MLSCs and MVCs produce basis functions with large negative regions. Negative values are undesirable, because interpolated values may extend beyond the range of the boundary values in regions where the basis functions are negative.

We show the convergence of our numerical approximation in Fig. 6. Even in the undivided control polygon, the basis functions appear to be almost smooth. Once the boundary polygon has converged to its limit shape, the basis functions have also converged. In our implementation of coordinate evaluation, the times to evaluate all basis functions for the shape shown in Fig. 5 after four subdivisions using our coordinates, harmonic coordinates, MECs, MLSCs, and MVCs, respectively, were 11.6, 102, 5.7, 3.0, and 0.42 s, to calculate the coordinates of 16 136 points. One idea that may speed up computation for complex boundaries is to subdivide edges far from the evaluation point fewer times than nearby boundary edges. The shape of the basis functions should be preserved, because far edges have only a small influence on the local shape of the basis functions.

5. Conclusion

We have introduced a new form of Gordon–Wixom coordinates that are positive and smooth for domains with smooth boundaries. A nice feature of our coordinates is that they have a clear geometric interpretation as a weighted sum of linear interpolants. Analytic solutions for smooth boundaries are difficult to find, but we have provided closed-form solutions for polygons, which can approximate smooth boundaries. Ideally, barycentric coordinates should be smooth even for polygonal boundaries, but the coordinates described in our paper are only as smooth as the boundary. In the future, we would like to investigate coordinates with higher orders of smoothness.

Acknowledgments

Funding is provided in part by NSF grant CCF-07024099. Josiah Manson is funded by the NSF GRFP.

References

[1] Wachspress EL. A rational finite element basis. Mathematics in science and engineering, vol. 114. Academic Press; 1975.
[2] Sukumar N, Tabarraei A. Conforming polygonal finite elements. International Journal for Numerical Methods in Engineering 2004;61:2045–66.
[3] Sukumar N, Malsch EA. Recent advances in the construction of polygonal finite element interpolants. Archives of Computational Methods in Engineering 2006;13:129–63.
[4] Desbrun M, Meyer M, Alliez P. Intrinsic parameterizations of surface meshes. Computer Graphics Forum 2002;21:209–18.
[5] Ju T, Schaefer S, Warren J. Mean value coordinates for closed triangular meshes. SIGGRAPH 2005;561–6.
[6] Farbman Z, Hoffer G, Lipman Y, Cohen-Or D, Lischinski D. Coordinates for instant image cloning. In: SIGGRAPH. 2009. p. 67:1–67:9.
[7] Takayama K, Sorkine O, Nealen A, Igarashi T. Volumetric modeling with diffusion surfaces. In: SIGGRAPH Asia. 2010. p. 180:1–180:8.
[8] Hormann K, Tarini M. A quadrilateral rendering primitive. In: SIGGRAPH/EUROGRAPHICS conference on graphics hardware. 2004. p. 7–14.
[9] Loop CT, DeRose TD. A multisided generalization of Bézier surfaces. ACM Transactions on Graphics 1989;8(3):204–34.
[10] Joshi P, Meyer M, DeRose T, Green B, Sanocki T. Harmonic coordinates for character articulation. ACM Transactions on Graphics 2007;26(3):71:1–71:10.
[11] Gordon WJ, Wixom JA. Pseudo-harmonic interpolation on convex domains. SIAM Journal on Numerical Analysis 1974;11(5):909–33.
[12] Möbius AF. Der barycentriche calcul. Journal of Barth, Leipzig 1827.
[13] Floater M, Hormann K, Kós G. A general construction of barycentric coordinates over convex polygons. Advances in Computational Mathematics 2006;24(1):311–31.
[14] Floater MS. Mean value coordinates. Computer Aided Geometric Design 2003; 20:19–27.
[15] Hormann K, Floater MS. Mean value coordinates for arbitrary planar polygons. ACM Transactions on Graphics 2006;25(4):1424–41.
[16] Lipman Y, Kopf J, Cohen-Or D, Levin D. GPU-assisted positive mean value coordinates for mesh deformations. In: Symposium on geometry processing. 2007. p. 117–24.
[17] Hormann K, Sukumar N. Maximum entropy coordinates for arbitrary polytopes. Computer Graphics Forum 2008;27(5):1513–20.
[18] Manson J, Schaefer S. Moving least squares coordinates. Computer Graphics Forum 2010;29(5):1517–24.
[19] Schaefer S, McPhail T, Warren J. Image deformation using moving least squares. ACM Transactions on Graphics 2006;25(3):533–40.
[20] Warren JD, Schaefer S, Hirani AN, Desbrun M. Barycentric coordinates for convex sets. Advances in Computational Mathematics 2007;27(3):319–38.
[21] Schaefer S, Ju T, Warren J. A unified, integral construction for coordinates over closed curves. Computer Aided Geometric Design 2007;24(8–9):481–93.
[22] Belyaev A. On transfinite barycentric coordinates. In: Symposium on geometry processing. 2006. p. 89–99.