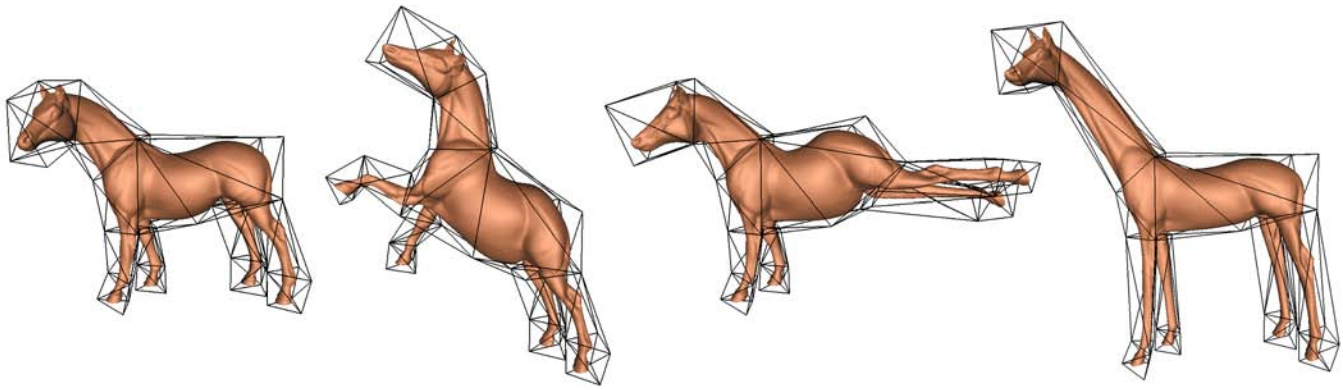
Mean Value Coordinates for Closed Triangular Meshes

Tao Ju, Scott Schaefer, Joe Warren
Rice University



(a) (b) (c) (d)

Figure 1: Original horse model with enclosing triangle control mesh shown in black (a). Several deformations generated using our 3D mean value coordinates applied to a modified control mesh (b,c,d).

Abstract

Constructing a function that interpolates a set of values defined at vertices of a mesh is a fundamental operation in computer graphics. Such an interpolant has many uses in applications such as shading, parameterization and deformation. For closed polygons, mean value coordinates have been proven to be an excellent method for constructing such an interpolant. In this paper, we generalize mean value coordinates from closed 2D polygons to closed triangular meshes. Given such a mesh P, we show that these coordinates are continuous everywhere and smooth on the interior of P. The coordinates are linear on the triangles of P and can reproduce linear functions on the interior of P. To illustrate their usefulness, we conclude by considering several interesting applications including constructing volumetric textures and surface deformation.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Boundary representations; Curve, surface, solid, and object representations; Geometric algorithms, languages, and systems

Keywords: barycentric coordinates, mean value coordinates, volumetric textures, surface deformation

1 Introduction

Given a closed mesh, a common problem in computer graphics is to extend a function defined at the vertices of the mesh to its interior. For example, Gouraud shading computes intensities at the vertices of a triangle and extends these intensities to the interior using linear interpolation. Given a triangle with vertices $\{p_1, p_2, p_3\}$ and associated intensities $\{f_1, f_2, f_3\}$, the intensity at point v on the interior of the triangle can be expressed in the form

$$\hat{f}[v] = \frac{\sum_j w_j f_j}{\sum_j w_j} \tag{1}$$

where w_j is the area of the triangle $\{v, p_{j-1}, p_{j+1}\}$. In this formula, note that each *weight* w_j is normalized by the sum of the weights, $\sum_j w_j$ to form an associated *coordinate* $\frac{w_j}{\sum_j w_j}$. The interpolant $\hat{f}[v]$ is then simply the sum of the f_j times their corresponding coordinate.

Mesh parameterization methods [Hormann and Greiner 2000; Desbrun et al. 2002; Khodakovsky et al. 2003; Schreiner et al. 2004; Floater and Hormann 2005] and freeform deformation methods [Sederberg and Parry 1986; Coquillart 1990; MacCracken and Joy 1996; Kobayashi and Ootsubo 2003] also make heavy use of interpolants of this type. Both applications require that a point v be represented as an affine combination of the vertices on an enclosing shape. To generate this combination, we simply set the data values f_j to be their associated vertex positions p_j. If the interpolant reproduces linear functions, i.e.;

$$v = \frac{\sum_j w_j p_j}{\sum_j w_j},$$

the coordinate functions $\frac{w_j}{\sum_j w_j}$ are the desired affine combination.

For convex polygons in $2D$, a sequence of papers, [Wachspress 1975], [Loop and DeRose 1989] and [Meyer et al. 2002], have proposed and refined an interpolant that is linear on its boundaries and only involves convex combinations of data values at the vertices of the polygons. This interpolant has a simple, local definition as a rational function and reproduces linear functions. [Warren 1996; Warren et al. 2004] also generalized this interpolant to convex shapes in higher dimensions. Unfortunately, Wachspress's interpolant does not generalize to non-convex polygons. Applying

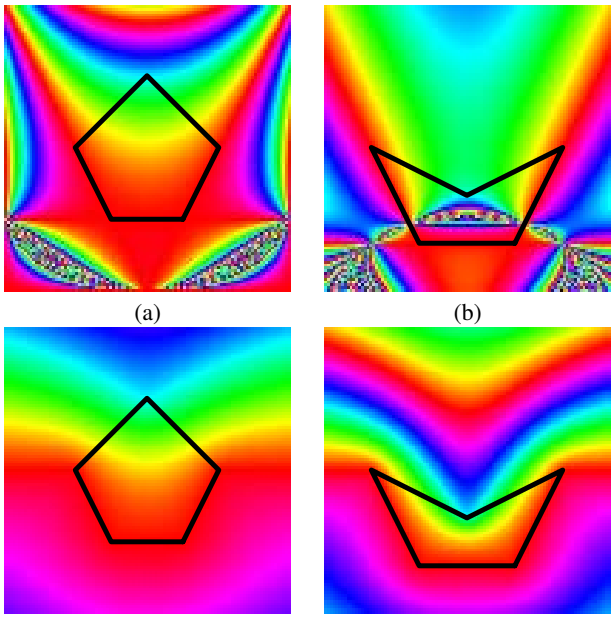


Figure 2: Interpolating hue values at polygon vertices using Wachspress coordinates (a, b) versus mean value coordinates (c, d) on a convex and a concave polygon.

the construction to such a polygon yields an interpolant that has poles (divisions by zero) on the interior of the polygon. The top portion of Figure 2 shows Wachspress's interpolant applied to two closed polygons. Note the poles on the outside of the convex polygon on the left as well as along the extensions of the two top edges of the non-convex polygon on the right.

More recently, several papers, [Floater 1997; Floater 1998; Floater 2003], [Malsch and Dasgupta 2003] and [Hormann 2004], have focused on building interpolants for non-convex 2D polygons. In particular, Floater proposed a new type of interpolant based on the mean value theorem [Floater 2003] that generates smooth coordinates for star-shaped polygons. Given a polygon with vertices p_j and associated values f_j, Floater's interpolant defines a set of weight functions w_j of the form

$$w_j = \frac{\tan\left[\frac{\alpha_{j-1}}{2}\right] + \tan\left[\frac{\alpha_j}{2}\right]}{|p_j - v|}. \tag{2}$$

where α_j is the angle formed by the vector $p_j - v$ and $p_{j+1} - v$. Normalizing each weight function w_j by the sum of all weight functions yields the *mean value coordinates* of v with respect to p_j.

In his original paper, Floater primarily intended this interpolant to be used for mesh parameterization and only explored the behavior of the interpolant on points in the kernel of a star-shaped polygon. In this region, mean value coordinates are always non-negative and reproduce linear functions. Subsequently, Hormann [Hormann 2004] showed that, for any simple polygon (or nested set of simple polygons), the interpolant $\hat{f}[v]$ generated by mean value coordinates is well-defined *everywhere* in the plane. By maintaining a consistent orientation for the polygon and treating the α_j as signed angles, Hormann also shows that mean value coordinates reproduce linear functions everywhere. The bottom portion of Figure 2 shows mean value coordinates applied to two closed polygons. Note that the interpolant generated by these coordinates possesses no poles anywhere even on non-convex polygons.

Contributions Horman's observation suggests that Floater's mean value construction could be used to generate a similar interpolant for a wider class of shapes. In this paper, we provide such a generalization for arbitrary closed surfaces and show that the resulting interpolants are well-behaved and have linear precision. Applied to closed polygons, our construction reproduces 2D mean value coordinates. We then apply our method to closed triangular meshes and construct 3D mean value coordinates. (In independent contemporaneous work, [Floater et al. 2005] have proposed an extension of mean value coordinates from 2D polygons to 3D triangular meshes identical to section 3.2.) Next, we derive an efficient, stable method for evaluating the resulting mean value interpolant in terms of the positions and associated values of vertices of the mesh. Finally, we consider several practical applications of such coordinates including a simple method for generating classes of deformations useful in character animation.

2 Mean value interpolation

Given a closed surface P in R^3, let $p[x]$ be a parameterization of P. (Here, the parameter x is two-dimensional.) Given an auxiliary function $f[x]$ defined over P, our problem is to construct a function $\hat{f}[v]$ where $v \in R^3$ that interpolates $f[x]$ on P, i.e.; $\hat{f}[p[x]] = f[x]$ for all x. Our basic construction extends an idea of Floater developed during the construction of 2D mean value coordinates.

To construct $\hat{f}[v]$, we project a point $p[x]$ of P onto the unit sphere S_v centered at v. Next, we weight the point's associated value $f[x]$ by $\frac{1}{|p[x]-v|}$ and integrate this weighted function over S_v. To ensure affine invariance of the resulting interpolant, we divide the result by the integral of the weight function $\frac{1}{|p[x]-v|}$ taken over S_v. Putting the pieces together, the *mean value interpolant* has the form

$$\hat{f}[v] = \frac{\int_x w[x,v] f[x] dS_v}{\int_x w[x,v] dS_v} \tag{3}$$

where the weight function $w[x,v]$ is exactly $\frac{1}{|p[x]-v|}$. Observe that this formula is essentially an integral version of the discrete formula of Equation 1. Likewise, the continuous weight function $w[x,v]$ and the discrete weights w_j of Equation 2 differ only in their numerators. As we shall see, the $\tan\left[\frac{\alpha}{2}\right]$ terms in the numerators of the w_j are the result of taking the integrals in Equation 3 with respect to dS_v.

The resulting mean value interpolant satisfies three important properties.

Interpolation: As v converges to the point $p[x]$ on P, $\hat{f}[v]$ converges to $f[x]$.

Smoothness: The function $\hat{f}[v]$ is well-defined and smooth for all v not on P.

Linear precision: If $f[x] = p[x]$ for all x, the interpolant $\hat{f}[v]$ is identically v for all v.

Interpolation follows from the fact that the weight function $w[x,v]$ approaches infinity as $p[x] \to v$. Smoothness follows because the projection of $f[x]$ onto S_v is continuous in the position of v and taking the integral of this continuous process yields a smooth function. The proof of linear precision relies on the fact that the integral of the unit normal over a sphere is exactly zero (due to symmetry). Specifically,

$$\int_x \frac{p[x] - v}{|p[x] - v|} dS_v = 0$$

since $\frac{p[x]-v}{|p[x]-v|}$ is the unit normal to S_v at parameter value x. Rewriting this equation yields the theorem.

$$v = \int_x \frac{p[x]}{|p[x] - v|} dS_v \Big/ \int_x \frac{1}{|p[x] - v|} dS_v$$

Notice that if the projection of P onto S_v is one-to-one (i.e.; v is in the kernel of P), then the orientation of dS_v is non-negative, which guarantees that the resulting coordinate functions are positive. Therefore, if P is a convex shape, then the coordinate functions are positive for all v inside P. However, if v is not in the kernel of P, then the orientation of dS_v is negative and the coordinates functions may be negative as well.

3 Coordinates for piecewise linear shapes

In practice, the integral form of Equation 3 can be complicated to evaluate symbolically [1]. However, in this section, we derive a simple, closed form solution for piecewise linear shapes in terms of the vertex positions and their associated function values. As a simple example to illustrate our approach, we first re-derive mean value coordinates for closed polygons via mean value interpolation. Next, we apply the same derivation to construct mean value coordinates for closed triangular meshes.

3.1 Mean value coordinates for closed polygons

Consider an edge E of a closed polygon P with vertices $\{p_1, p_2\}$ and associated values $\{f_1, f_2\}$. Our first task is to convert this discrete data into a continuous form suitable for use in Equation 3. We can linearly parameterize the edge E via

$$p[x] = \sum_i \phi_i[x] p_i$$

where $\phi_1[x] = (1-x)$ and $\phi_2[x] = x$. We then use this same parameterization to extend the data values f_1 and f_2 linearly along E. Specifically, we let $f[x]$ have the form

$$f[x] = \sum_i \phi_i[x] f_i.$$

Now, our task is to evaluate the integrals in Equation 3 for $0 \le x \le 1$. Let $\overline{E}$ be the circular arc formed by projecting the edge E onto the unit circle S_v, we can rewrite the integrals of Equation 3 restricted to $\overline{E}$ as

$$\frac{\int_x w[x,v] f[x] d\overline{E}}{\int_x w[x,v] d\overline{E}} = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{4}$$

where weights $w_i = \int_x \frac{\phi_i[x]}{|p[x]-v|} d\overline{E}$.

Our next goal is to compute the corresponding weights w_i for edge E in Equation 4 without resorting to symbolic integration (since this will be difficult to generalize to 3D). Observe that the following identity relates w_i to a vector,

$$\sum_i w_i (p_i - v) = m. \tag{5}$$

where $m = \int_x \frac{p[x]-v}{|p[x]-v|} d\overline{E}$ is simply the integral of the outward unit normal over the circular arc $\overline{E}$. We call m the **mean vector** of $\overline{E}$, as scaling m by the length of the arc yields the centroid of the circular arc $\overline{E}$. Based on 2D trigonometry, m has a simple expression in terms of p_1 and p_2. Specifically,

$$m = \tan[\alpha/2]\left(\frac{(p_1 - v)}{|p_1 - v|} + \frac{(p_2 - v)}{|p_2 - v|}\right)$$

where α denotes the angle between $p_1 - v$ and $p_2 - v$. Hence we obtain $w_i = \tan[\alpha/2]/|p_i - v|$ which agrees with the Floater's weighting function defined in Equation 2 for 2D mean value coordinates when restricted to a single edge of a polygon.

Equation 4 allows us to formulate a closed form expression for the interpolant $\hat{f}[v]$ in Equation 3 by summing the integrals for all edges E_k in P (note that we add the index k for enumeration of edges):

$$\hat{f}[v] = \frac{\sum_k \sum_i w_i^k f_i^k}{\sum_k \sum_i w_i^k} \tag{6}$$

where w_i^k and f_i^k are weights and values associated with edge E_k.

3.2 Mean value coordinates for closed meshes

We now consider our primary application of mean value interpolation for this paper; the derivation of mean value coordinates for triangular meshes. These coordinates are the natural generalization of 2D mean value coordinates.

Given triangle T with vertices $\{p_1, p_2, p_3\}$ and associated values $\{f_1, f_2, f_3\}$, our first task is to define the functions $p[x]$ and $f[x]$ used in Equation 3 over T. To this end, we simply use the linear interpolation formula of Equation 1. The resulting function $f[x]$ is a linear combination of the values f_i times basis functions $\phi_i[x]$.

As in 2D, the integral of Equation 3 reduces to the sum in Equation 6. In this case, the weights w_i have the form

$$w_i = \int_x \frac{\phi_i[x]}{|p[x]-v|} d\overline{T}$$

where $\overline{T}$ is the projection of triangle T onto S_v. To avoid computing this integral directly, we instead relate the weights w_i to the mean vector m for the spherical triangle $\overline{T}$ by inverting Equation 5. In matrix form,

$$\{w_1, w_2, w_3\} = m \{p_1 - v, p_2 - v, p_3 - v\}^{-1} \tag{7}$$

All that remains is to derive an explicit expression for the mean vector m for a spherical triangle $\overline{T}$. The following theorem solves this problem.

Theorem 3.1 *Given a spherical triangle $\overline{T}$, let θ_i be the length of its i^{th} edge (a circular arc) and n_i be the inward unit normal to its i^{th} edge (see Figure 3 (b)). Then,*

$$m = \sum_i \frac{1}{2} \theta_i n_i \tag{8}$$

where m, the mean vector, is the integral of the outward unit normals over $\overline{T}$.

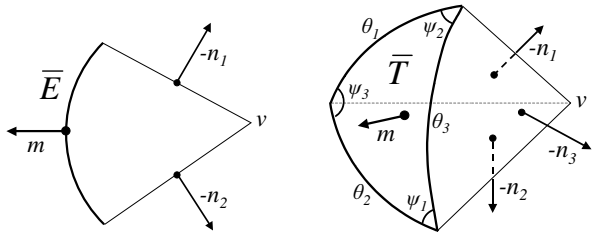
Proof: Consider the solid triangular wedge of the unit sphere with cap $\overline{T}$. The integral of outward unit normals over a closed surface is always exactly zero [Fleming 1977, p.342]. Thus, we can partition the integral into three triangular faces whose outward normals are $-n_i$ with associated areas $\frac{1}{2}\theta_i$. The theorem follows since $m - \sum_i \frac{1}{2} \theta_i n_i$ is then zero. $\perp$

Note that a similar result holds in 2D, where the mean vector m defined by Equation 3.1 for a circular arc $\overline{E}$ on the unit circle can be interpreted as the sum of the two inward unit normals of the vectors $p_i - v$ (see Figure 3 (a)). In 3D, the lengths θ_i of the edges of the spherical triangle $\overline{T}$ are the angles between the vectors $p_{i-1} - v$ and $p_{i+1} - v$ while the unit normals n_i are formed by taking the cross

[1] To evaluate the integral of Equation 3, we can relate the differential dS_v to dx via

$$dS_v = \frac{p^{\perp}[x].(p[x]-v)}{|p[x]-v|^2} dx$$

where $p^{\perp}[x]$ is the cross product of the $n-1$ tangent vectors $\frac{\partial p[x]}{\partial x_i}$ to P at $p[x]$. Note that the sign of this expression correctly captures whether P has folded back during its projection onto S_v.



(a) (b)

Figure 3: Mean vector m on a circular arc $\overline{E}$ with edge normals n_i (a) and on a spherical triangle $\overline{T}$ with arc lengths θ_i and face normals n_i.

product of $p_{i-1} - v$ and $p_{i+1} - v$. Given the mean vector m, we now compute the weights w_i using Equation 7 (but without doing the matrix inversion) via

$$w_i = \frac{n_i \cdot m}{n_i \cdot (p_i - v)} \tag{9}$$

At this point, we should note that projecting a triangle T onto S_v may reverse its orientation. To guarantee linear precision, these folded-back triangles should produce negative weights w_i. If we maintain a positive orientation for the vertices of every triangle T, the mean vector computed using Equation 8 points towards the projected spherical triangle $\overline{T}$ when $\overline{T}$ has a positive orientation and away from $\overline{T}$ when $\overline{T}$ has a negative orientation. Thus, the resulting weights have the appropriate sign.

3.3 Robust mean value interpolation

The discussion in the previous section yields a simple evaluation method for mean value interpolation on triangular meshes. Given point v and a closed mesh, for each triangle T in the mesh with vertices $\{p_1, p_2, p_3\}$ and associated values $\{f_1, f_2, f_3\}$,

1. Compute the mean vector m via Equation 8
2. Compute the weights w_i using Equation 9
3. Update the denominator and numerator of $\hat{f}[v]$ defined in Equation 6 respectively by adding $\sum_i w_i$ and $\sum_i w_i f_i$

To correctly compute $\hat{f}[v]$ using the above procedure, however, we must overcome two obstacles. First, the weights w_i computed by Equation 9 may have a zero denominator when the point v lies on plane containing the face T. Our method must handle this degenerate case gracefully. Second, we must be careful to avoid numerical instability when computing w_i for triangle T with a small projected area. Such triangles are the dominant type when evaluating mean value coordinates on meshes with large number of triangles. Next we discuss our solutions to these two problems and present the complete evaluation algorithm as pseudo-code in Figure 4.

- **Stability:**

  When the triangle T has small projected area on the unit sphere centered at v, computing weights using Equation 8 and 9 becomes numerically unstable due to cancelling of unit normals n_i that are almost co-planar. To this end, we next derive a stable formula for computing weights w_i. First, we substitute Equation 8 into Equation 9, using trigonometry we obtain

$$w_i = \frac{\theta_i - \cos[\psi_{i+1}]\theta_{i-1} - \cos[\psi_{i-1}]\theta_{i+1}}{2\sin[\psi_{i+1}]\sin[\theta_{i-1}]|p_i^k - v|}, \tag{10}$$

```
// Robust evaluation on a triangular mesh
for each vertex p_j with values f_j
    d_j ← ||p_j − x||
    if d_j < ε return f_j
    u_j ← (p_j − x)/d_j
totalF ← 0
totalW ← 0
for each triangle with vertices p_1, p_2, p_3 and values f_1, f_2, f_3
    l_i ← ||u_{i+1} − u_{i−1}||      // for i = 1,2,3
    θ_i ← 2 arcsin[l_i/2]
    h ← (Σ θ_i)/2
    if π − h < ε
        // x lies on t, use 2D barycentric coordinates
        w_i ← sin[θ_i] l_{i−1} l_{i+1}
        return (Σ w_i f_i)/(Σ w_i)
    c_i ← (2 sin[h] sin[h − θ_i])/(sin[θ_{i+1}] sin[θ_{i−1}]) − 1
    s_i ← sign[det[u_1, u_2, u_3]] √(1 − c_i^2)
    if ∃i, |s_i| ≤ ε
        // x lies outside t on the same plane, ignore t
        continue
    w_i ← (θ_i − c_{i+1} θ_{i−1} − c_{i−1} θ_{i+1})/(d_i sin[θ_{i+1}] s_{i−1})
    totalF += Σ w_i f_i
    totalW += Σ w_i
f_x ← totalF/totalW
```

Figure 4: Mean value coordinates on a triangular mesh

where $\psi_i (i = 1,2,3)$ denotes the angles in the spherical triangle $\overline{T}$. Note that the ψ_i are the dihedral angles between the faces with normals n_{i-1} and n_{i+1}. We illustrate the angles ψ_i and θ_i in Figure 3 (b).

To calculate the cos of the ψ_i without computing unit normals, we apply the half-angle formula for spherical triangles [Beyer 1987],

$$\cos[\psi_i] = \frac{2\ \sin[h]\sin[h-\theta_i]}{\sin[\theta_{i+1}]\sin[\theta_{i-1}]} - 1, \tag{11}$$

where $h = (\theta_1 + \theta_2 + \theta_3)/2$. Substituting Equation 11 into 10, we obtain a formula for computing w_i that only involves lengths $|p_i - v|$ and angles θ_i. In the pseudo-code from Figure 4, angles θ_i are computed using *arcsin*, which is stable for small angles.

- **Co-planar cases:** Observe that Equation 9 involves division by $n_i \cdot (p_i - v)$, which becomes zero when the point v lies on plane containing the face T. Here we need to consider two different cases. If v lies on the plane **inside** T, the continuity of mean value interpolation implies that $\hat{f}[v]$ converges to the value $f[x]$ defined by linear interpolation of the f_i on T. On the other hand, if v lies on the plane **outside** T, the weights w_i become zero as their integral definition $\int \frac{\phi_i[x]}{|p[x]-v|} d\overline{T}$ becomes zero. We can easily test for the first case because the sum $\Sigma_i \theta_i = 2\pi$ for points inside of T. To test for the second case, we use Equation 11 to generate a stable computation for $\sin[\psi_i]$. Using this definition, v lies on the plane outside T if any of the dihedral angles ψ_i (or $\sin[\psi_i]$) are zero.

4 Applications and results

While mean value coordinates find their main use in boundary value interpolation, these coordinates can be applied to a variety of applications. In this section, we briefly discuss several of these applications including constructing volumetric textures and surface deformation. We conclude with a section on our implementation of these coordinates and provide evaluation times for various shapes.

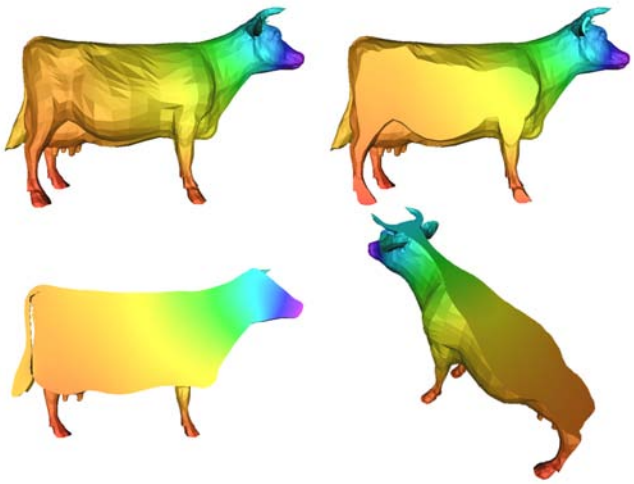


Figure 5: Original model of a cow (top-left) with hue values specified at the vertices. The planar cuts illustrate the interior of the function generated by 3D mean value coordinates.

4.1 Boundary value interpolation

As mentioned in Section 1, these coordinate functions may be used to perform boundary value interpolation for triangular meshes. In this case, function values are associated with the vertices of the mesh. The function constructed by our method is smooth, interpolates those vertex values and is a linear function on the faces of the triangles. Figure 5 shows an example of interpolating hue specified on the surface of a cow. In the top-left is the original model that serves as input into our algorithm. The rest of the figure shows several slices of the cow model, which reveal the volumetric function produced by our coordinates. Notice that the function is smooth on the interior and interpolates the colors on the surface of the cow.

4.2 Volumetric textures

These coordinate functions also have applications to volumetric texturing as well. Figure 6 (top-left) illustrates a model of a bunny with a 2D texture applied to the surface. Using the texture coordinates (u_i, v_i) as the f_i for each vertex, we apply our coordinates and build a function that interpolates the texture coordinates specified at the vertices and along the polygons of the mesh. Our function extrapolates these surface values to the interior of the shape to construct a volumetric texture. Figure 6 shows several slices revealing the volumetric texture within.

4.3 Surface Deformation

Surface deformation is one application of mean value coordinates that depends on the linear precision property outlined in Section 2. In this application, we are given two shapes: a model and a control mesh. For each vertex v in the model, we first compute its mean value weight functions w_j with respect to each vertex p_j in the undeformed control mesh. To perform the deformation, we move the vertices of the control mesh to induce the deformation on the original surface. Let $\hat{p}_j$ be the positions of the vertices from the deformed control mesh, then the new vertex position $\hat{v}$ in the deformed model is computed as

$$\hat{v} = \frac{\sum_j w_j \hat{p}_j}{\sum_j w_j}.$$

Notice that, due to linear precision, if $\hat{p}_j = p_j$, then $\hat{v} = v$. Figures 1 and 7 show several examples of deformations generated with this process. Figure 1 (a) depicts a horse before deformation and the surrounding control mesh shown in black. Moving the vertices of the control mesh generates the smooth deformations of the horse shown in (b,c,d).

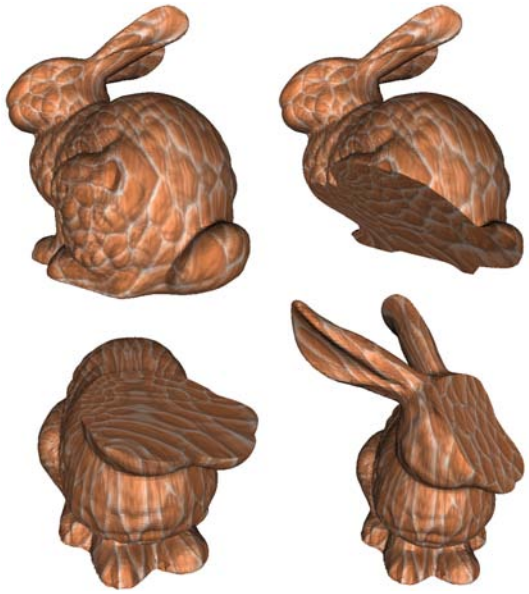


Figure 6: Textured bunny (top-left). Cuts of the bunny to expose the volumetric texture constructed from the surface texture.

Previous deformation techniques such as freeform deformations [Sederberg and Parry 1986; MacCracken and Joy 1996] require volumetric cells to be specified on the interior of the control mesh. The deformations produced by these methods are dependent on how the control mesh is decomposed into volumetric cells. Furthermore, many of these techniques restrict the user to creating control meshes with quadrilateral faces.

In contrast, our deformation technique allows the artist to specify an arbitrary closed triangular surface as the control mesh and does not require volumetric cells to span the interior. Our technique also generates smooth, realistic looking deformations even with a small number of control points and is quite fast. Generating the mean value coordinates for figure 1 took 3.3s and 1.9s for figure 7. However, each of the deformations only took 0.09s and 0.03s respectively, which is fast enough to apply these deformations in real-time.

4.4 Implementation

Our implementation follows the pseudo-code from Figure 4 very closely. However, to speed up computations, it is helpful to precompute as much information as possible.

Figure 8 contains the number of evaluations per second for various models sampled on a 3GHz Intel Pentium 4 computer. Previously, practical applications involving barycentric coordinates have been restricted to 2D polygons containing a very small number of line segments. In this paper, for the first time, barycentric coordinates have been applied to truly large shapes (on the order of 100,000 polygons). The coordinate computation is a global computation and all vertices of the surface must be used to evaluate the function at a single point. However, much of the time spent is determining whether or not a point lies on the plane of one of the triangles in the mesh and, if so, whether or not that point is inside that triangle. Though we have not done so, using various spatial partitioning data structures to reduce the number of triangles that

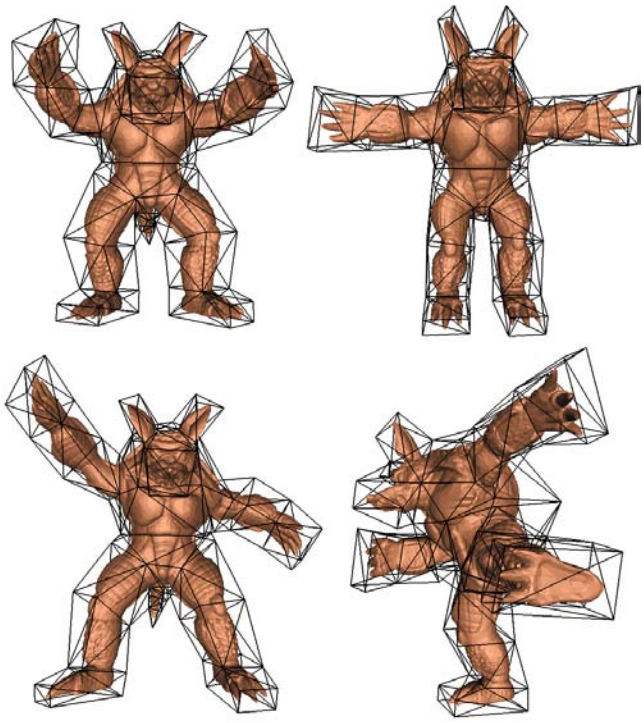


Figure 7: Original model and surrounding control mesh shown in black (top-left). Deforming the control mesh generates smooth deformations of the underlying model.

Model	Tris	Verts	Eval/s
Horse control mesh (fig 1)	98	51	16281
Armadillo control mesh (fig 7)	216	111	7644
Cow (fig 5)	5804	2903	328
Bunny (fig 6)	69630	34817	20

Figure 8: Number of evaluations per second for various models.

must be checked for coplanarity could greatly enhance the speed of the evaluation.

5 Conclusions and Future Work

Mean value coordinates are a simple, but powerful method for creating functions that interpolate values assigned to the vertices of a closed mesh. Perhaps the most intriguing feature of mean value coordinates is that fact that they are well-defined on both the interior and the exterior of the mesh. In particular, mean value coordinates do a reasonable job of extrapolating value outside of the mesh. We intend to explore applications of this feature in future work.

Another interesting point is the relationship between mean value coordinates and Wachspress coordinates. In 2D, both coordinate functions are identical for convex polygons inscribed in the unit circle. As a result, one method for computing mean value coordinates is to project the vertices of the closed polygon onto a circle and compute Wachspress coordinates for the inscribed polygon. However, in 3D, this approach fails. In particular, inscribing the vertices of a triangular mesh onto a sphere does not necessarily yield a convex polyhedron. Even if the inscribed polyhedron happens to be convex, the resulting Wachspress coordinates are rational functions of the vertex position v while the mean value coordinates are transcendental functions of v.

Finally, we only consider meshes that have triangular faces. One important generalization would be to derive mean value coordinates for piecewise linear mesh with arbitrary closed polygons as faces. On these faces, the coordinates would degenerate to standard 2D mean value coordinates. We plan to address this topic in a future paper.

Acknowledgements
We'd like to thank John Morris for his help with designing the control meshes for the deformations. This work was supported by NSF grant ITR-0205671.

References

BEYER, W. H. 1987. *CRC Standard Mathematical Tables (28th Edition)*. CRC Press.

COQUILLART, S. 1990. Extended free-form deformation: a sculpturing tool for 3d geometric modeling. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, ACM Press, 187–196.

DESBRUN, M., MEYER, M., AND ALLIEZ, P. 2002. Intrinsic Parameterizations of Surface Meshes. *Computer Graphics Forum 21*, 3, 209–218.

FLEMING, W., Ed. 1977. *Functions of Several Variables*. Second edition. Springer-Verlag.

FLOATER, M. S., AND HORMANN, K. 2005. Surface parameterization: a tutorial and survey. In *Advances in Multiresolution for Geometric Modelling*, N. A. Dodgson, M. S. Floater, and M. A. Sabin, Eds., Mathematics and Visualization. Springer, Berlin, Heidelberg, 157–186.

FLOATER, M. S., KOS, G., AND REIMERS, M. 2005. Mean value coordinates in 3d. *To appear in CAGD*.

FLOATER, M. 1997. Parametrization and smooth approximation of surface triangulations. *CAGD 14*, 3, 231–250.

FLOATER, M. 1998. Parametric Tilings and Scattered Data Approximation. *International Journal of Shape Modeling 4*, 165–182.

FLOATER, M. S. 2003. Mean value coordinates. *Comput. Aided Geom. Des. 20*, 1, 19–27.

HORMANN, K., AND GREINER, G. 2000. MIPS - An Efficient Global Parametrization Method. In *Curves and Surfaces Proceedings (Saint Malo, France)*, 152–163.

HORMANN, K. 2004. Barycentric coordinates for arbitrary polygons in the plane. Tech. rep., Clausthal University of Technology, September. http://www.in.tu-clausthal.de/ hormann/papers/barycentric.pdf.

KHODAKOVSKY, A., LITKE, N., AND SCHROEDER, P. 2003. Globally smooth parameterizations with low distortion. *ACM Trans. Graph. 22*, 3, 350–357.

KOBAYASHI, K. G., AND OOTSUBO, K. 2003. t-ffd: free-form deformation by using triangular mesh. In *SM '03: Proceedings of the eighth ACM symposium on Solid modeling and applications*, ACM Press, 226–234.

LOOP, C., AND DEROSE, T. 1989. A multisided generalization of Bézier surfaces. *ACM Transactions on Graphics 8*, 204–234.

MACCRACKEN, R., AND JOY, K. I. 1996. Free-form deformations with lattices of arbitrary topology. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM Press, 181–188.

MALSCH, E., AND DASGUPTA, G. 2003. Algebraic construction of smooth interpolants on polygonal domains. In *Proceedings of the 5th International Mathematica Symposium*.

MEYER, M., LEE, H., BARR, A., AND DESBRUN, M. 2002. Generalized Barycentric Coordinates for Irregular Polygons. *Journal of Graphics Tools 7*, 1, 13–22.

SCHREINER, J., ASIRVATHAM, A., PRAUN, E., AND HOPPE, H. 2004. Inter-surface mapping. *ACM Trans. Graph. 23*, 3, 870–877.

SEDERBERG, T. W., AND PARRY, S. R. 1986. Free-form deformation of solid geometric models. In *SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, ACM Press, 151–160.

WACHSPRESS, E. 1975. *A Rational Finite Element Basis*. Academic Press, New York.

WARREN, J., SCHAEFER, S., HIRANI, A., AND DESBRUN, M. 2004. Barycentric coordinates for convex sets. Tech. rep., Rice University.

WARREN, J. 1996. Barycentric Coordinates for Convex Polytopes. *Advances in Computational Mathematics 6*, 97–108.