CS554 Term Project – Mean value coordinates for closed triangular meshes

YU ZHENG*, Oregon State University, USA

This project re-implements a paper of SIGGRAPH 2005 in OpenGL[Ju et al. 2005].Specifically, I implenment the color interpolation and surface deformation by using the mean value coordinates. And the results align closely with the results presented in the paper.

ACM Reference Format:
Yu Zheng. 2023. CS554 Term Project – Mean value coordinates for closed triangular meshes. *ACM Trans. Graph.* 37, 4, Article 111 (August 2023), 3 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Constructing a function that interpolates values defined at mesh vertices is crucial for various computer graphics applications, including shading, surface deformation and volumetric textures. For surface deformation, Interpolation functions allow for smooth and seamless transitions between different shapes or poses in animations and modeling. This helps create realistic and visually appealing representations of objects and characters in 3D environments.For volumetric textures, interpolation functions are necessary to map 3D textures onto complex geometries without visible seams or distortions. This ensures that textures are distributed evenly across surfaces and contributes to the visual consistency and realism of the rendered objects.

In this paper, the paper presents a generalization of mean value coordinates from closed 2D polygons to closed triangular meshes, addressing the need for smooth and continuous interpolation in these applications. The new technique overcomes limitations of previous methods and provides a more robust and versatile solution for interpolation problems in computer graphics.

For my implementation, I did the boundary value interpolation and surface deformation by using the algorithm from this paper.

2 PREVIOUS WORK

Extending a function defined at the vertices of a closed mesh to its interior is a common challenge in computer graphics. Linear interpolation is often employed to compute intensities at the vertices of a triangle for applications such as Gouraud shading, and these intensities are then extended to the interior. The intensity at point v on the interior of the triangle can be expressed in the form:

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

Author's address: Yu Zheng, zhengy4@oregonstate.edu, Oregon State University, Corvallis, Oregon, USA.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2023 Association for Computing Machinery.
0730-0301/2023/8-ART111 $15.00
https://doi.org/XXXXXXX.XXXXXXX

In this equation, $\frac{\sum_j w_j f_j}{\sum_j w_j}$ is the barycentric coordinates and the interpolant $\hat{f}[v]$ is simply the sum of the f_j times their corresponding barycentric coordinates.

This type of interpolant is heavily used in the applications like Mesh parameterization[Hormann and Greiner 2000][Desbrun et al. 2002][Khodakovsky et al. 2003][Schreiner et al. 2004][Floater and Hormann 2005] and freeform deformation[Sederberg and Parry 1986][Coquillart 1990][MacCracken and Joy 1996][Khodakovsky et al. 2003].

An interpolant for convex polygons in 2D involving convex combinations of data values at the vertices of the polygons fails to extend to non-convex polygons and produces poles in the interpolant.[Gout 1985][Loop and DeRose 1989][Meyer et al. 2002] Recent work by Floater and Hormann focuses on creating well-behaved interpolants for non-convex polygons using the mean value theorem.[Hormann 2005][Malsch and Dasgupta 2003][Floater 1997][Floater 1998][Floater 2003] Hormann showed that the mean value interpolant is well-defined and reproduces linear functions everywhere[Hormann 2005]. This work generalizes the mean value construction for arbitrary closed surfaces, achieving linear precision and producing 2D and 3D mean value coordinates. The authors present an efficient method for evaluating the interpolant and explore practical applications, such as character animation.

3 BACKGROUND

The authors' method for calculate mean value coordinates for a arbitrary 3d closed triangular meshes constists of three steps:

(1) Project the surface onto a unit sphere centered at v
(2) Weight the point's associated value f[x] by $\frac{1}{|p|x]-v|}$ and integrate this weighted function over S_v.
(3) To ensure affine invariance of the resulting interpolant, divide the result by the integral of the weight function $\frac{1}{|p|x]-v|}$ taken over S_v.

So the the mean value interpolant has this form:

$$\hat{f}[v] = \frac{\int_x w[x,v] f[x] dS_v}{\int_x w[x,v] dS_v} \tag{2}$$

To get the closed discrete form of the interpolant equation, the authors follow these steps:

(1) Given spherical triangle, compute mean vector m (integral of unit normal).Note that the mean vector has the following relationship with each vertices, the interior points and weights for the vertices

$$m = \sum_{k=1}^{3} w_k \, (p_k - v) \tag{3}$$

(2) Build a wedge, (e.g. connect the spherical triangle with the sphere center) with face normal n_k,then apply stokes theorem.This equation can be derived.

$$\sum_{k=1}^{3} \frac{1}{2}\theta_k n_k + m = 0 \tag{4}$$

(3) Then cauculate weights for each vertex using matrix inversion of Equation3.This equation can be derived.

$$w_k = \frac{n_k \cdot m}{n_k \cdot (p_k - v)} \tag{5}$$

(4) Sum over all triangles.

$$f[v] = \frac{\sum_j \sum_{k=1}^{3} w_k^j f_k^j}{\sum_j \sum_{k=1}^{3} w_k^j} \tag{6}$$

Next, the authors consider two specific cases:

(1) zero denominators in Equation5 when v lies on a plane containing T
(2) numerical instability when computing weights for triangles with small projected area, which are common in meshes with large numbers of triangles.

The details for these two cases are complicated. I will provide more implementation details in the following sections.

4 TECHNIQUE

In this section, I describe the key techniques and steps involved in implementing the mean value coordinates for closed triangular meshes. I provide a detailed explanation of each step, as well as any modifications or optimizations I made in my implementation. And I will outline the challenges I encountered and my analytical thoughts on the method presented in the paper.

4.1 Overview of the Algorithm

To compute the mean value coordinates on a triangular mesh, I follow the pseudocode provided by the authors

```
// 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_i²)
    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
```

The algorithm can be divided into three cases.

(1) If the evaluated vertex v is too close to vertex P, then the value of v is simply set to the value of P.
(2) If the evaluated vertex lies on the spherical triangle, 2D barycentric coordinates are used.
(3) The normal case is the 3D mean value coordinates, which can be computed using the algorithm presented in the pseudocode.

4.2 Implementation Details

I applied mean value coordinates to two specific applications: color interpolation and surface deformation. Although the paper didn't specify what v represents, I drew inspiration from the surface deformation technique and used a control mesh to perform color interpolation. I treated each vertex of the model mesh as a v and assigned a color to each vertex of the control mesh. This way, every v was placed in one of the triangles of the control mesh. To assign colors to the vertices, I used the following method:

(1) Compute the bounding box of the control mesh and obtain two vectors maxV and minV.
(2) Calculate the difference between maxV and minV and obtain dx, dy, and dz.
(3) Iterate through all vertices and set r to (vertex->pos.x - minV.x) / dx, g to (vertex->pos.y - minV.y) / dy, and b to (vertex->pos.z - minV.z) / dz.

This approach yields a rainbow-like color scheme, which facilitates testing of the color interpolation by making any artifacts more prominent.

For the parameters setting, I set epsilon in the pseudo code to 0.01.

4.3 Challenges

I met several challenges as below.

(1) I was only able to locate two models that had both mesh and control mesh available online, and these models were well interpolated by using the algorithm and are presented in the results section. To address models that did not have a control mesh available, I attempted to use the convex hull to generate one. However, this method resulted in artifacts in the interpolated model as shown in the Fig3, even after subdivision. The poor quality of the convex hull mesh, which contained many long and narrow triangles, may have contributed to this issue. The algorithm may have difficulty handling such special cases.
(2) The two models I found on the internet were in .obj format, which I converted to .ply using Meshlab software. However, the resulting .ply files were in byte form instead of ASCII, making them unreadable by our learnply program. To solve this issue, I wrote a Python script that utilized the trimesh library to convert the .ply files to the correct format.
(3) Building an interface for a user to deform a model is a challenging task. Therefore, to address this issue, I resort to using Mashlab software for the deformation process. Specifically, I apply an oversmoothing on the model mesh to achieve the deformation.

5 RESULTS

The results for the dog mesh is in the Fig.1. The results in the thors' paper is in the Fig.2. As we can see in Fig.1,it is evident that the color interpolation of the model mesh is seamless internally and

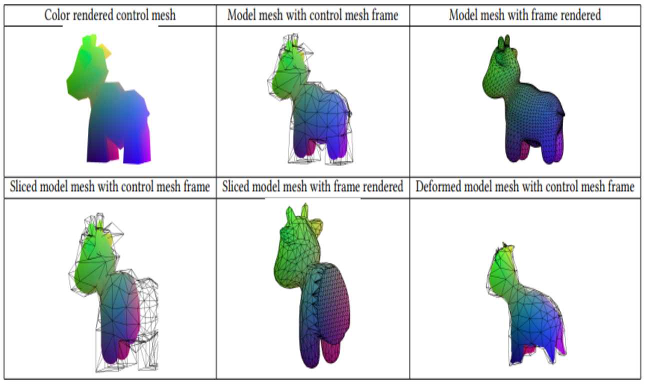

Fig. 1. Dog mesh result

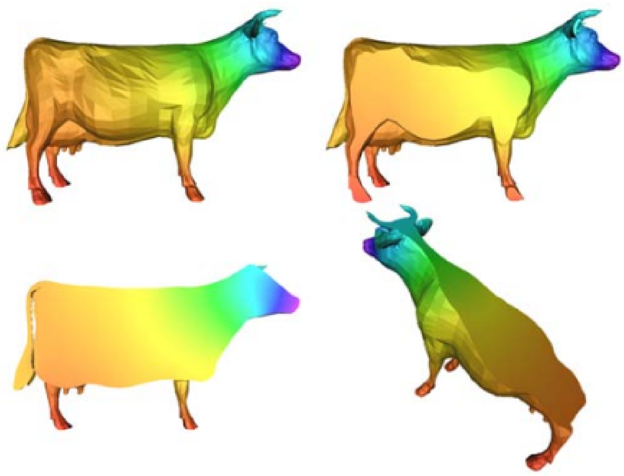

Fig. 2. boundary value interpolation result in the paper

smoothly blends colors on the surface of the dog. Moreover, the deformation of the model aligns with our expectations as it deforms when we manipulate the control mesh.

6 CONCLUSION

My work involves the re-implementation of two mean value coordinate applications - boundary value interpolation and surface deformation. While my results are satisfactory, there are some limitations that need to be addressed. As per the paper's authors, the control model is provided by artists and is not generated automatically. Although I computed the convex hull, it produced artifacts, possibly due to poor quality of the control mesh. Going forward, it would be interesting to investigate the root cause of the artifacts and devise a way to automatically generate the control mesh. Furthermore, to enhance the deformation process, it would be beneficial to build a user interface that allows real-time model mesh deformation via cursor manipulation.

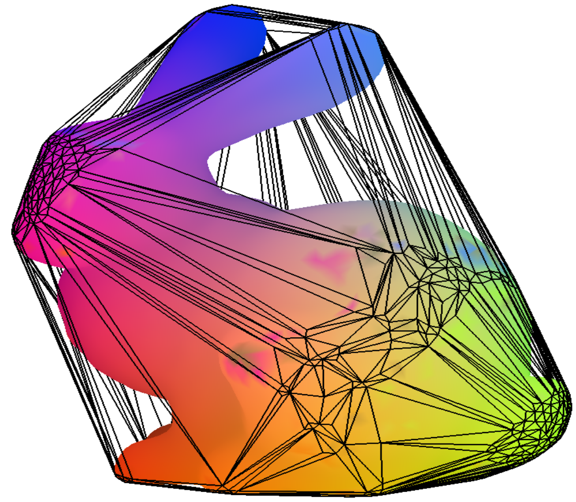

Fig. 3. Bunny mesh result using convex hull as the control mesh

7 BIBLIOGRAPHY

REFERENCES

Sabine Coquillart. 1990. Extended free-form deformation: A sculpturing tool for 3D geometric modeling. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*. 187–196.

Mathieu Desbrun, Mark Meyer, and Pierre Alliez. 2002. Intrinsic parameterizations of surface meshes. In *Computer graphics forum*, Vol. 21. Wiley Online Library, 209–218.

Michael S Floater. 1997. Parametrization and smooth approximation of surface triangulations. *Computer aided geometric design* 14, 3 (1997), 231–250.

Michael S Floater. 1998. Parametric tilings and scattered data approximation. *International Journal of Shape Modeling* 4, 03n04 (1998), 165–182.

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

Michael S Floater and Kai Hormann. 2005. Surface parameterization: a tutorial and survey. *Advances in multiresolution for geometric modelling* (2005), 157–186.

JL Gout. 1985. Rational Wachspress-type finite elements on regular hexagons. *IMA J. Numer. Anal.* 5, 1 (1985), 59–77.

Kai Hormann. 2005. *Barycentric coordinates for arbitrary polygons in the plane.* Technische Universität Clausthal.

Kai Hormann and Günther Greiner. 2000. *MIPS: An efficient global parametrization method.* Technical Report. Erlangen-Nuernberg Univ (Germany) Computer Graphics Group.

Tao Ju, Scott Schaefer, and Joe Warren. 2005. Mean value coordinates for closed triangular meshes. In *ACM Siggraph 2005 Papers*. 561–566.

Andrei Khodakovsky, Nathan Litke, and Peter Schröder. 2003. Globally smooth parameterizations with low distortion. *ACM transactions on graphics (TOG)* 22, 3 (2003), 350–357.

Charles T Loop and Tony D DeRose. 1989. A multisided generalization of Bézier surfaces. *ACM Transactions on Graphics (TOG)* 8, 3 (1989), 204–234.

Ron MacCracken and Kenneth I Joy. 1996. Free-form deformations with lattices of arbitrary topology. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 181–188.

Elisabeth Anna Malsch and Gautam Dasgupta. 2003. Algebraic construction of smooth interpolants on polygonal domains. In *Challenging The Boundaries Of Symbolic Computation: (With CD-ROM)*. World Scientific, 81–88.

Mark Meyer, Alan Barr, Haeyoung Lee, and Mathieu Desbrun. 2002. Generalized barycentric coordinates on irregular polygons. *Journal of graphics tools* 7, 1 (2002), 13–22.

John Schreiner, Arul Asirvatham, Emil Praun, and Hugues Hoppe. 2004. Inter-surface mapping. In *ACM SIGGRAPH 2004 Papers*. 870–877.

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