A Comparison of GPU Tessellation Strategies for Multisided Patches

G.J. Hettinga[1], P.J. Barendrecht[1], J. Kosinka[1]

[1]Johann Bernoulli Institute, University of Groningen, Nijenborgh 9, 9747 AG Groningen, The Netherlands

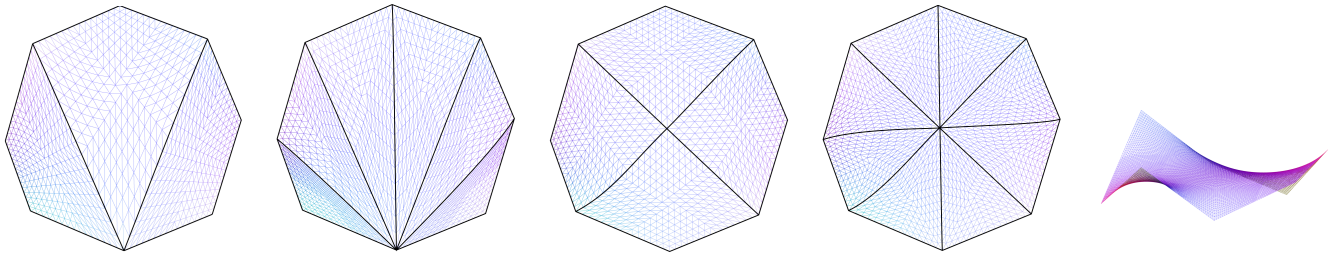

Figure 1: *Different tessellation strategies applied to a non-planar octagon. Left to right: fanning quadrangulation, fanning triangulation, symmetric quadrangulation, symmetric triangulation, and a side view of the curved octagon.*

Abstract

We propose an augmentation of the traditional tessellation pipeline with several different strategies that efficiently render multisided patches using generalised barycentric coordinates. The strategies involve different subdivision steps and the usage of textures. In addition, we show that adaptive tessellation techniques naturally extend to some of these strategies whereas others need a slight adjustment. The technique of Loop et al. [LSNC09], commonly known as ACC-2, is extended to multisided faces to illustrate the effectiveness of multisided techniques. A performance and quality comparison is made between the different strategies and remarks on the techniques and implementation details are provided.

CCS Concepts

•***Computing methodologies*** → ***Rendering;*** *Mesh geometry models; Parametric curve and surface models;*

1. Introduction

Hardware tessellation is a powerful tool for generating dense geometry from much coarser geometry in an efficient manner. The generated vertices, or tessellations, can be displaced to improve the quality of the geometry by applying parametric surface algorithms, such as Phong tessellation [BA08], or displacement mapping [SKU08].

The tessellation stage in modern pipelines is optimised with respect to a few abstract patch types. These are *triangles*, *quadrilaterals* and *iso-lines*. This is reflected in the types of coordinates which can be supplied in the evaluation stage, as these are either standard barycentric coordinates or (bi)linear coordinates. When considering higher-valency polygons there is generally no straightforward method available to render these.

Many interesting multisided parametric schemes have been developed in the past [LD89] and recent years [VSK16]. In computer graphics these types of patches have not been used frequently due to the inability to render such surfaces efficiently. However, due to recent advances in graphical computation power it becomes increasingly possible to use the capabilities of the GPU to render such surfaces in real-time.

In [MNP08] the use of *instancing* to render a special 5-sided patch, called a P_m patch, was already achieved, but it uses geometry shaders instead of tessellation shaders and divides the parametric domain in 5 different sectors. Similarly, geometry shaders and *transform feedback* were used in [LWZ11] to render a tessellation of multisided patches. In [HK17] multisided generalisations of Phong tessellation [BA08] and PN triangles [VPBM01] were rendered using a fanning triangulation of a regular domain and tessellation shaders.

This paper proposes several strategies (see Figure 1) to augment the existing graphics pipeline with the ability to efficiently render multisided polygons and parametric patches. In Section 2, a brief overview of generalised barycentric coordinates is given as well as a means to use the coordinates to parametrise arbitrary polygons. In Section 3 an in-depth explanation of the proposed strategies is given, including how they can be used efficiently in the rendering

DOI: 10.2312/egs.20181041

pipeline. In Section 4, adaptive tessellation for multisided polygons is discussed. After this, in Section 5, visual and performance comparisons of the strategies are made, and a multisided generalisation of ACC-2 [LSNC09] is given as an application of our methods. Finally, the paper is concluded in Section 7.

2. Parametrisation of Multisided Polygons

Rendering arbitrary polygonal patches can be problematic as higher-valency polygons admit various parametrisations. Further, the vertices of a polygon need not be co-planar and, even worse, the polygon could be folded over or self-intersecting. It is also not uniquely defined what the exact surface that can be constructed from a polygonal boundary is.

Generalised barycentric coordinates are a powerful tool to parametrise multisided polygons. However, these methods are typically only valid for planar polygons. Therefore, instead of trying to parametrise an arbitrary polygon directly, we seek a planar equivalent. This planar domain is then defined to be the parametrisation domain of the actual arbitrary polygon or polygonal patch.

Generalised barycentric coordinates can be used to express any point on a planar polygon as a linear combination of the polygon's vertices. Let ω be a planar polygon, considered as a closed set, with vertices $\mathbf{w}_i$, $i = 1 \dots n$, and let $\mathbf{p} \in \omega$. Then generalised barycentric coordinate functions ϕ_i, $i = 1 \dots n$ on ω are defined by the following three properties:

Partition of unity:	$\sum_{i=1}^{n} \phi_i(\mathbf{p}) = 1$;
Linear reproduction:	$\sum_{i=1}^{n} \phi_i(\mathbf{p})\mathbf{w}_i = \mathbf{p}$;
Non-negativity:	$\phi_i(\mathbf{p}) \geq 0, \forall i$.

The coordinate functions, when continous in $\mathrm{Int}(\omega)$, satisfy the Lagrange property: $\phi_i(\mathbf{w}_j) = \delta_{ij}$, where δ_{ij} is the Kronecker delta, and linearly interpolate along the edges of ω.

There exist various forms of generalised barycentric coordinates [Flo15] such as Wachspress coordinates [Wac75] and harmonic coordinates [JMD*07]. The latter have been used frequently in computer graphics for their attractive properties, especially in the case of mesh deformations, but they are intensive to compute.

We consider ω as the planar parametrisation domain of an arbitrary (possibly non-planar) polygon Ω with vertices $\mathbf{v}_i, i = 1 \dots n$. Then for any point $\mathbf{p} \in \omega$ its generalised barycentric coordinates ϕ can be found with respect to ω. Using these coordinates we can also construct a point $\mathbf{q}$ on Ω by taking a linear combination of $\mathbf{v}_i$ with these same coordinates, i.e., $\mathbf{q} = \sum_{i=1}^{n} \phi_i(\mathbf{p})\mathbf{v}_i$.

A regular polygon, inscribed in the unit circle, is one of the simplest choices of parametrisation domain. Due to its convexity, a regular parametrisation domain has the added advantage of a trivial subdivision (see Section 3). Naturally, it supports the use of Wachspress coordinates; for $n = 4$ the regular polygon is a square producing bilinear coordinates. We have used Wachspress coordinates in all examples and evaluations presented in this paper, due to the aforementioned reasons and their efficiency.

3. Rendering Multisided Polygonal Patches

The following steps are required to render multisided polygons and multisided patches. First, the parametrisation domain should be tri-

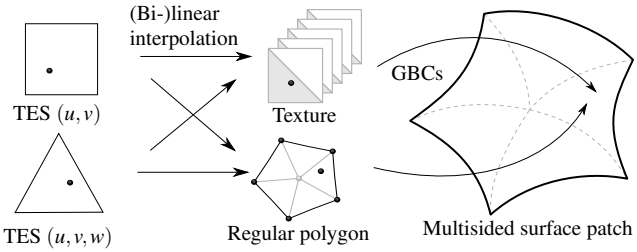

Figure 2: *Our tessellation process. Bilinear or barycentric coordinates are used to find a position on a texture or on a regular polygon. From these positions generalised barycentric coordinates (GBCs) can be retrieved or calculated.*

angulated or quadrangulated in a subdivision step. Each polygonal patch is then rendered as a collection of either triangular or quadrilateral patches. Regular barycentric coordinates or bilinear coordinates, as supplied by the Tessellation Evaluation or Domain Shader stage, can be used to find a position on this sub-polygon. For this position on the parametrisation domain generalised barycentric coordinates can be calculated using the whole parametrisation domain, see Figure 2. For this purpose, we propose two pairs of different, but straight-forward subdivision strategies of the regular domain, each pair consisting of a triangulation and a quadrangulation. The subdivision step is only achieved implicitly through the use of *instanced rendering*. This allows for defining the data of the polygons only once, but rendering several instances of that same polygon, based on the subdivision strategy.

The first minimal triangulation has been previously described in [HK17]. This triangulation is a fanning triangulation from one vertex to the other vertices of the polygon such that $n-2$ triangles are defined. In the same vein, a fanning quadrangulation can be constructed on polygons with even valencies, defining $n/2-2$ quadrilaterals.

The second strategy is to define a symmetric subdivision of the parametrisation domain using the barycentre of the parametric domain. In this fashion the domain is partitioned into n triangular sectors or $n/2$ quadrilateral sectors, respectively. Again, this is only possible for polygons of even valency in the latter case.

These symmetric subdivision strategies lend themselves well for an alternative approach that uses textures. As the parametrisation domain is defined to be a regular polygon, the coordinate functions are n-fold symmetric. In effect, by only saving the coordinates for one triangular sector of the polygon, the other coordinate functions can be determined by symmetry. The same applies for the quadrilateral case, but then for a quadrilateral sector, taking three consecutive vertices and connecting them to the centre position. By using 3D textures a whole sector of coordinate functions can be compactly stored in video memory and efficiently retrieved. Moreover, by storing the coordinates in textures the possibility of using other (more computationally expensive) coordinate types, like harmonic coordinates, becomes available. The sectors are stored in the (u, v)-domain, using the depth component to store the n coordinate components, for triangular sectors only one half of the domain is used such that $u + v \leq 1$.

The tessellation of each separate sub-polygon can be done by taking into account which instance is currently being rendered. For the texture-based method this entails *swizzling* the retrieved coordinate vectors around such that the coordinate functions correspond to the correct vertices. For coordinate calculation this requires changing the vertices being interpolated based on the current instance k. For the triangular strategies one can use three vertices of the parametrisation domain $\mathbf{w}_{(k+i) \bmod n}$, $i = 1,2$ and $\mathbf{w}_1$ as the origin of the fan, or the barycentre in the symmetric strategy. Similarly, in the quadrilateral case we take $\mathbf{w}_{(2k+i) \bmod n}$, $i = 1 \ldots 3$ and either $\mathbf{w}_1$ or the barycentre, for the fanning or symmetric strategies, respectively.

For every different valency n a separate shader is made and stored in a shader array. The quadrangulation strategies create shader arrays that are interleaved with shaders that handle triangulations for odd valencies. For coordinate calculations the regular parametrisation domain can be hardcoded in the shaders. By grouping the polygons of a polygonal model by valency, the overhead of switching shaders can be minimised as for each valency only one draw call is needed. A polygonal model with polygons of l different valencies requires only l separate draw calls.

4. Adaptive Tessellation

We apply the commonly-used adaptive tessellation strategy as described in [Can11]. In this approach, the tessellation levels are computed in the Tessellation Control shader by calculating the length (in pixels) of the polygon's edges projected into screen space. Setting the desired number of pixels per triangle edge then determines the level of the tessellation.

Naturally, cracks in the surface should be prevented, and therefore shared edges should be assigned identical tessellation levels. This is automatically ensured for the boundary edges of the polygons (the black edges in Figure 2), but not for the interior edges (spokes) defining the sectors of a (symmetric) subdivision (grey edges of the polygon in Figure 2, bottom middle). Our approach is to approximate the centre of the multisided patch. By averaging the n vertices of the polygon or averaging the inner-most ring of control points of a parametric patch, a good approximation can be found. Note that each spoke can still have its own (fractional) tessellation level. Alternatively, the exact centre position of the parametric patch can be calculated, but this requires that all patch data should be made available.

5. Results & Performance

It can be observed in Figures 1 and 3 that the two symmetric subdivisions provide a more uniform tessellation of the polygon. The fanning subdivisions clearly show discrete mean curvature artefacts in the regions where the density of the tessellation changes (Figure 3), although all patches represent the same parametric surface. The uniformity of the symmetric strategies is typically favourable over the variable density of the fanning approach.

Performance In Figure 4 a performance comparison is shown of the rendering of a model mainly consisting of multisided polygons. The tests were performed at the maximum tessellation level of 64×64 on a PC with an Intel Core i7-4770K CPU and an NVIDIA GeForce GTX 770 with 4GB of memory running NVIDIA drivers for Ubuntu Linux using OpenGL 4.5.

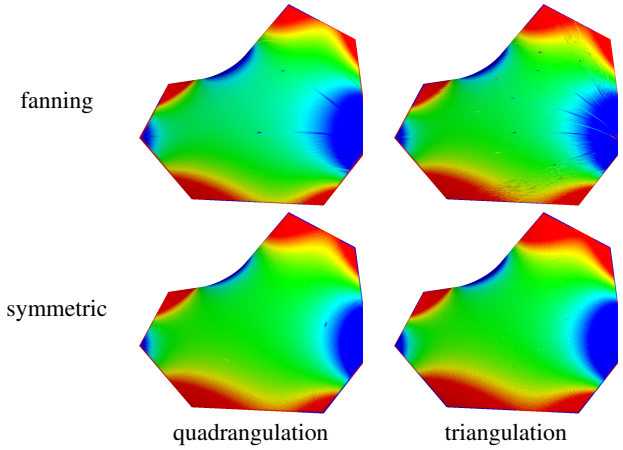

Figure 3: *Discrete mean curvature [MDSB03] visualised on a non-planar octagon with respect to the different subdivision methods, cf. Figure 1. Note that the fanning subdivisions lead to artefacts when estimating discrete mean curvature due to uneven tessellation densities.*

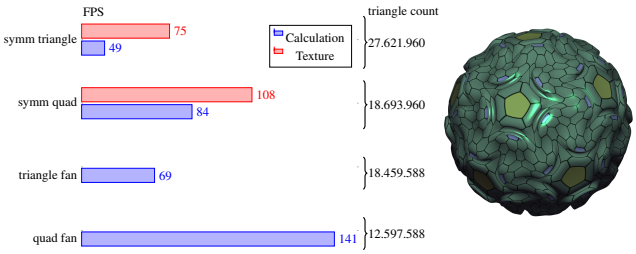

Figure 4: *Performance comparison in FPS and number of generated triangles: Rendering a model (right) containing 60 quads, 12 pentagons and 750 hexagons using different tessellation strategies at the maximum tessellation level of* 64×64.

The results show that there is a big performance increase when using the texture-based approach for retrieval of generalised barycentric coordinates. The two symmetric subdivisions perform significantly better than the fanning triangulation. The most efficient strategy is to use the fanning quadrangulation, which outperforms all strategies by far. The fanning and symmetric triangulation have the worst performance, due to the number of sub-polygons. Effectively, the performance of the strategies can be attributed to the number of generated triangles. Taking this into account we can see that symmetric quadrangulation gives a very balanced performance with respect to the number of generated triangles, especially when using the texture-based approach.

ACC-2 We extended the approximate Catmull-Clark scheme by Loop et al. [LSNC09] to arbitrary polygonal meshes. The calculation of the vertex, edge and face points remains entirely the same, but the data is used to construct multisided Gregory patches using

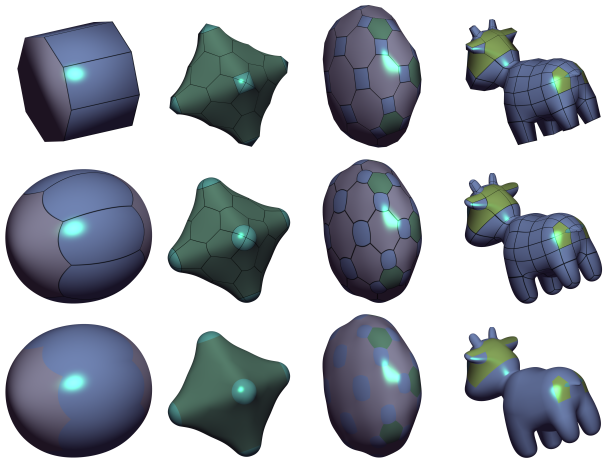

Figure 5: *Simple shapes rendered using (generalised) ACC-2 patches where patches have been grouped by colour based on valency. From top to bottom: input mesh, ACC-2 with patch structure, and ACC-2 with Phong shading.*

Gregory generalised Bézier (GGB) patches [HK18]. In the triangular case, the hybrid cubic-quartic patch is used and for quads the standard form of the Gregory patch is employed. The rest of the polygons use GGB patches with each side constructed as a side of a quadrilateral Gregory patch. The resulting surface properties remain the same, creating G^1 surfaces in irregular regions and reproducing C^2 bi-cubic B-splines in regular regions. The results can be seen for a number of shapes containing different polygon types in Figure 5.

6. Discussion

Due to present-day limitations in the structure of the programmable shaders it seems most efficient to create separate shaders for the different valencies of polygons. In a polygonal mesh containing faces of many different valencies, this requires switching of shaders between draw calls. By restricting the subdivision step to only one type of polygon we also reduce the number of shader switches. Naturally, a pentagon could be subdivided into a quad and a triangle, but this would mean yet another shader switch just to render a single pentagon.

We have tested the strategies for polygons with valencies up to 8. However, in the event that tessellation of higher order polygons is needed we see no reason why these strategies would not work for these polygons. The efficiency of the polygonal methods with respect to rendering traditional dense triangular or quadrilateral meshes shows without a doubt that multisided methods are slower. However, multisided polygons are able to model meshes consisting of large polygons. Designing such models requires awareness of the abilities of multisided patches. This will facilitate the creation of complex polygonal meshes composed of large multisided facets, which would otherwise have to be finer when considering traditional triangle and quad meshes.

7. Conclusion

For the usage of GPU tessellation of multisided polygons there are different tessellation strategies that subdivide a multisided polygon into sub-polygons in order to be able to tessellate the whole domain. The symmetric strategies create more sub-polygons than the fanning strategies, but create a more uniform tesselation of the polygonal surface. Conversely, the fanning quadrangulation is the most efficient strategy. The usage of textures is a viable and compact way to store parametrisations of multisided polygons, provides a way to use other coordinate types like harmonic coordinates, and is very efficient.

Methods previously only defined for triangles and quadrilaterals, like ACC-2, can easily be extended to multisided patches and rendered in real-time in the modern graphics pipeline using the newly proposed strategies. With these strategies the inclusion of multisided facets in polygonal models becomes more attractive for use in computer graphics and geometric modelling.

References

[BA08] BOUBEKEUR T., ALEXA M.: Phong tessellation. In *ACM Transactions on Graphics (TOG)* (2008), vol. 27, ACM, p. 141. 1

[Can11] CANTLAY I.: Directx 11 terrain tessellation. *Nvidia whitepaper 8*, 11 (2011). 3

[Flo15] FLOATER M. S.: Generalized barycentric coordinates and applications. *Acta Numerica 24* (2015), 161–214. 2

[HK17] HETTINGA G. J., KOSINKA J.: Phong Tessellation and PN Polygons for Polygonal Models. In *EG 2017 - Short Papers* (2017), The Eurographics Association, pp. 49–52. 1, 2

[HK18] HETTINGA G. J., KOSINKA J.: Multisided Generalisations of Gregory Patches. *To appear in Computer Aided Geometric Design (Special issue of GMP'18), 10.1016/j.cagd.2018.03.005* (2018). 4

[JMD*07] JOSHI P., MEYER M., DEROSE T., GREEN B., SANOCKI T.: Harmonic coordinates for character articulation. In *ACM Transactions on Graphics (TOG)* (2007), vol. 26, ACM, p. 71. 2

[LD89] LOOP C. T., DEROSE T. D.: A multisided generalization of Bézier surfaces. *ACM Trans.on Graphics (TOG) 8*, 3 (1989), 204–234. 1

[LSNC09] LOOP C., SCHAEFER S., NI T., CASTAÑO I.: Approximating subdivision surfaces with Gregory patches for hardware tessellation. In *ACM Transactions on Graphics (TOG)* (2009), vol. 28, ACM, pp. 151:1–151:9. 1, 2, 3

[LWZ11] LEUNG Y.-S., WANG C. C., ZHANG Y.: Localized construction of curved surfaces from polygon meshes: A simple and practical approach on GPU. *Computer-Aided Design 43*, 6 (2011), 573–585. 1

[MDSB03] MEYER M., DESBRUN M., SCHRÖDER P., BARR A. H.: Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and mathematics III*. Springer, 2003, pp. 35–57. 3

[MNP08] MYLES A., NI T., PETERS J.: Fast Parallel Construction of Smooth Surfaces from Meshes with Tri/Quad/Pent Facets. *Computer Graphics Forum 27*, 5 (2008), 1365–1372. 1

[SKU08] SZIRMAY-KALOS L., UMENHOFFER T.: Displacement mapping on the GPU - State of the art. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 1567–1592. 1

[VPBM01] VLACHOS A., PETERS J., BOYD C., MITCHELL J. L.: Curved PN triangles. In *Proceedings of the 2001 symposium on Interactive 3D graphics* (2001), ACM, pp. 159–166. 1

[VSK16] VÁRADY T., SALVI P., KARIKÓ G.: A multi-sided Bézier patch with a simple control structure. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 307–317. 1

[Wac75] WACHSPRESS E.: *A Rational Finite Element Basis*. Academic Press, 1975. 2