Improved Specular Highlights with Adaptive Shading

Youngkwan Cho Ulrich Neumann Jongwook Woo
{ ykcho | uneumann | jongwook }@graphics.usc.edu
http://tracker.usc.edu/~ykcho/as.html
Computer Science Department,
Integrated Media Systems Center
University of Southern California

Abstract

Gouraud shading and Phong shading are widely used interpolation methods to render a polygon mesh of a curved surface. When an illumination equation has a specular reflection term, Phong shading produces more realistic results than does Gouraud shading. The specular highlights produced by Phong shading give visual information about surface geometry and properties. However, Phong shading is not implemented in most graphics workstations and software renderers due to its computational expense. This paper introduces an adaptive shading method that produces Phong-shaded quality images for a small increase in rendering time on Gouraud shading systems. It is possible to get higher quality images from existing graphics hardware or software within a resonable time by simple modifications of the application or library.

1. Introduction

Polygon meshes are often used to approximate curved surfaces in computer graphics. Other representations (e.g., NURBS and CSG) are usually converted to polygon meshes for rendering. These polygon meshes are shaded by interpolation methods to approximate the smooth appearance of the underlying geometry. Two shading methods are widely used: Gouraud shading [GOUR 71] and Phong shading [PHON 75a]. Gouraud shading computes an illumination equation only at polygon vertices and interpolates the interior pixel colors from the vertex colors. Phong shading interpolates vertex normals across a polygon computing an illumination equation at every pixel. When an illumination equation contains a specular reflection term, these two methods produce different results. It is well known that specular highlights are completely missed or distorted by Gouraud shading for polygons whose screen areas are greater than the highlight areas (Fig. 1 (a) and (b)).

Specular highlights provide useful visual cues about surface geometry and properties. In spite of this, most graphics workstations and software renderers do not perform Phong shading due to its computational expense. The cost comes from the renormalization of vectors and the evaluation of an illumination equation at every pixel. At this time only Pixel-Planes 5 implements full Phong shading in real-time and can do so because of its parallel pixel-processing architecture [FUCH 89].

There were several attempts to reduce processing time by implementing Phong shading efficiently [BISH 86] [FOLE 90] and by combining Phong shading and Gouraud shading [PHON 75b] [EINK 91]. As far as we know, these approaches have not been integrated into commercially available systems, and they can not make efficient use of the Gouraud shading hardware available in graphics workstations. This paper describes a simple method for obtaining Phong-shaded quality images on Gouraud shading systems for a small increase in processing time (Fig. 1 (c)). This method extends the set of adaptive and progressive

(a) Gouraud shading (b) Phong Shading (c) Adaptive Shading

Fig. 1 Specular-reflection illumination model approaches that are available for rendering [BERG 86], modeling [AIRE 90], and radiosity lighting [COHE 88]. An application using this *Adaptive Shading* (AS) method has control of the balance between shading quality and computation time so it also supports time-critical computing of images.

Adaptive shading advocates the use of different shading methods applied adaptively to the polygons in a scene. The instance of AS described in this paper is based on two observations:

1- Phong shading results are similar to Gouraud shading for polygons that are very small relative to the highlight size or do not contain any specular highlights.

2- A scene represented by many polygons usually contains only a very small number of polygons with highlights.

The second observation points out the inefficiency of non-adaptive shading. Many rendering systems do support multiple shading models but they normally assign models to objects in the scene statically [COOK 87]. The AS approach (Fig. 2) adaptively renders polygons based on highlights. Most polygons are rendered directly with Gouraud shading and highlighted polygons are handled specially. The set of polygons with highlights depends on the number of lights and the relative positions of objects, lights, and the view point. This set is determined on every frame as detailed in section two. Polygons with highlights are adaptively subdivided to reduce their screen size and to increase the sampling rate about the highlights. Subdivision options are described in section three.

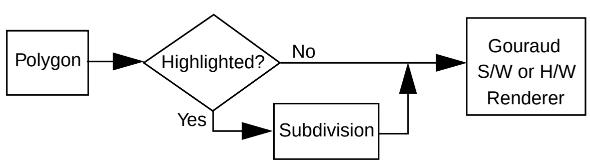


Fig. 2 Adaptive shading with Gouraud renderer

Polygons can be decomposed into triangles and, for simplicity, the remainder of this paper deals with triangles. We also assume perspective projections and point light sources.

2. Triangle classification

Adaptive shading requires a method for sorting triangles based on which shading method is appropriate for them. Our goal is to produce better highlights from Gouraud shading systems, so triangles with highlights must be identified. This test can be performed in any coordinate system before the perspective transformation, which does not preserve angle and length. In order to use existing rendering hardware or software, and to avoid transforming vertices, model space is chosen. The lights and the view point must be transformed into that space if necessary. The test must be simple and fast because it is applied to every triangle in the scene.

2.1. Highlight conditions

A highlight is a bright area on a surface where a light is reflected to the viewer. Though the boundary of a highlight is not well defined, the center (brightest point) is well defined. The center must satisfy the following condition: when a surface point P is a center of a highlight, the view point V should lie on the reflection ray of the light L from the point P (Fig. 3). That is, all three vectors should lie in a same plane in space, and θ should equal φ. The highlight conditions are summarized as:

H1: N_P is in the plane defined by L, V, and P.

H2: $\varphi = \theta$.

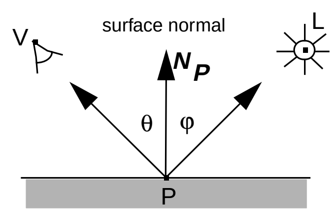


Fig. 3 Highlight conditions

2.2. Finding potentially highlighted triangles

First the H1 condition is tested. The plane LVP and the normal N_P move smoothly as the point P moves on the triangle ABC (Fig. 4). Assume that at the vertex A the normal N_A is on the left side of the plane LVA, and at the vertex B the normal N_B is on the right side of the plane LVB. As P moves on the edge AB, the plane LVP and the normal N_P continuously vary. On the edge AB there must be a point P where the normal N_P is in the plane LVP. More generally, if the normal vectors on the vertices A and B are on opposite sides of the planes LVA and LVB respectively, there is a point on the edge AB where the H1 condition is satisfied.

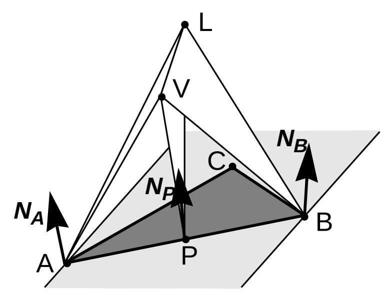


Fig. 4 The H1 condition test on A and B

If the normal N_C at the vertex C is on the right side of the plane LVC, the H1 condition is satisfied at a point on the edge AC. Otherwise, the H1 condition must be satisfied at a point on the edge AC.

The planes LVA, LVB, and LVC intersect with the plane of the triangle ABC and divide it into several strips. The center of a highlight should be located in the widest strip (shown in gray in Fig. 4) containing the triangle.

The H2 condition test requires the introduction of a halfway vector and a halfway plane. The halfway vector at the point B is the unit vector in the plane LVB that bisects the angle between the light and the view point (Fig. 5). A halfway plane is orthogonal to LVB and contains the halfway vector.

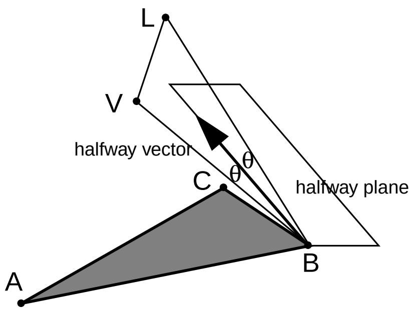


Fig. 5 Halfway vector and halfway plane on B

The H2 test is similar to the H1 test except that the halfway plane is used instead of the plane LVP. Any normal vector in the halfway plane satisfies the H2 condition. If a triangle has a highlight, there must be a pair of vertices for which the normals are on different sides of their corresponding halfway planes (Fig. 6). The halfway planes at

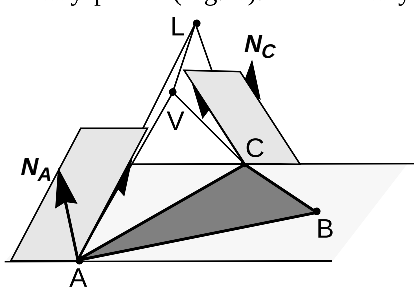


Fig. 6 The H2 condition test on A and C

the vertices also intersect the triangle's plane defining strips perpendicular to those created in the H1 test.

The strips of the H1 test and the H2 test create a bounding box of the triangle (Fig. 7). Any highlight centers

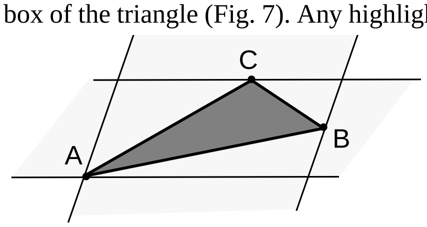


Fig. 7 Bounding box

in this box satisfy the H1 and H2 tests, even if they do not fall inside the triangle. For this reason we call triangles which pass these tests Potentially Highlighted Triangles (PHTs).

2.3. Efficient implementation

To find PHTs, all of the triangles should go through the PHT test (the H1 and H2 tests). A fast test procedure is critical for the success of adaptive shading. Normalized vectors (the surface normal, the light vector, and the view vector) are needed in illumination calculations, and normalization requires very expensive square root operations. Therefore this test procedure is designed to avoid using normalized vectors.

The H1 condition is tested by determining which side of the LVP plane contains the surface normal at each vertex (Fig 8). First compute the cross product of the light and the

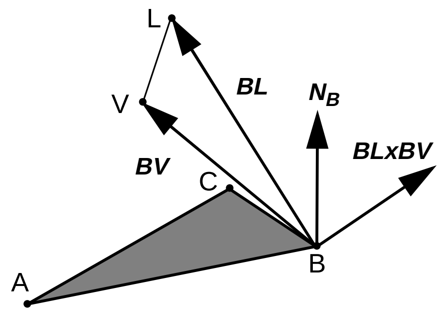


Fig. 8 The H1 test on B

view vectors on a test vertex. The sign of the dot product of the cross product and the normal vectors conveys which side of the plane the normal is on. The vectors need not be normalized because only the sign of the result is important. If the signs differ for any pair of vertices the triangle passes the H1 test.

The H2 test is performed by determining which side of the halfway plane the normal is on at each vertex. This can be determined efficiently by comparing $\cos\varphi$ and $\cos\theta$ (Fig. 9), and cosines are easily computed with the dot product of vectors. The vectors $\boldsymbol{N_P}$, $\boldsymbol{PL}$, and $\boldsymbol{PV}$ are not necessarily in the same plane in space.

If the dot products $\boldsymbol{PV} \cdot \boldsymbol{N_P}$ and $\boldsymbol{PL} \cdot \boldsymbol{N_P}$ have different signs it is obvious which cosine is bigger. If they are both negative, the triangle is back-facing and is not considered any further. In the case where they are both positive,

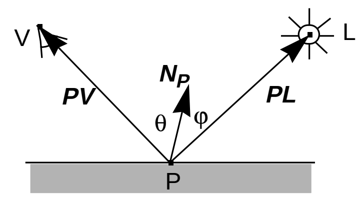


Fig. 9 The H2 test on P

the relative size of the angles must be determined. A dot product is a cosine value scaled by the magnitudes of vectors. Exact cosine values are not required for the H2 test, but only their relative magnitudes are needed for comparison. Squaring and scaling the positive cosines still allows their relative magnitudes to be compared. The squared and scaled $\cos\theta$ can be calculated as follows:

$$\boldsymbol{PV} \cdot \boldsymbol{N_P} = |\boldsymbol{PV}||\boldsymbol{N_P}|\cos\theta$$

$$|\boldsymbol{N_P}|\cos\theta = (\boldsymbol{PV} \cdot \boldsymbol{N_P})/|\boldsymbol{PV}|$$

$$(|\boldsymbol{N_P}|\cos\theta)^2 = (\boldsymbol{PV} \cdot \boldsymbol{N_P})^2/|\boldsymbol{PV}|^2$$

Similarly, the squared and scaled $\cos\varphi$ can be calculated

$$\boldsymbol{PL} \cdot \boldsymbol{N_P} = |\boldsymbol{PL}||\boldsymbol{N_P}|\cos\varphi$$

$$(|\boldsymbol{N_P}|\cos\varphi)^2 = (\boldsymbol{PL} \cdot \boldsymbol{N_P})^2/|\boldsymbol{PL}|^2$$

By comparing $\frac{(\boldsymbol{PV} \cdot \boldsymbol{N_p})^2}{|\boldsymbol{PV}|^2}$ and $\frac{(\boldsymbol{PL} \cdot \boldsymbol{N_P})^2}{|\boldsymbol{PL}|^2}$, we can decide which cosine value is larger without square root operations.

3. Subdivision methods

To improve the quality of the highlights produced by Gouraud shading an illumination equation must be evaluated at more points around the highlights. Since Gouraud shading calculates an illumination equation only on vertices, more vertices should be created by dividing triangles into smaller ones. A PHT is the triangle whose bounding box contains the center of a highlight. A highlight is not a point but an area, and it may extend into several triangles around a PHT. The PHT's neighbors should be subdivided to eliminate highlight artifacts at polygonal boundaries. Two alternatives for inserting new vertices are explored in sections 3.1 and 3.2.

3.1. Highlight subdivision

This method attempts to minimize the number of additional vertices by finding the center of a highlight, placing a new vertex on it, and subdividing the triangle into three smaller triangles. Additional subdivision may be performed by adding new vertices on the edges as described in section 3.1.2.

3.1.1. Finding a highlight. A triangle representing a planar surface has a constant normal over its surface, and the exact highlight position can be computed easily. For triangles representing a curved surface, a 2D iterative root finding procedure is needed. A 2D bisection method has computational complexity O($\log_2$ N), where N is the resolution of possible positions in the longer side of the bounding box. In the absence of any method for making a direct approximation to the highlight position, bisection is a reasonable approach.

The approximated center of the highlight can be directly computed by solving three 1D problems. First find the point Q where the H1 condition is satisfied on the edge AB (Fig. 10). Then on the edge AC or CB find the point R

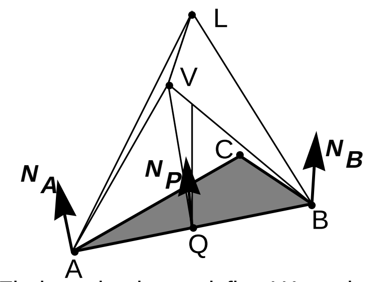


Fig. 10 Find a point that satisfies H1 on the edge AB

where the H1 condition is satisfied. Finally find the point which satisfies the H2 condition on the line QR (Fig. 11).

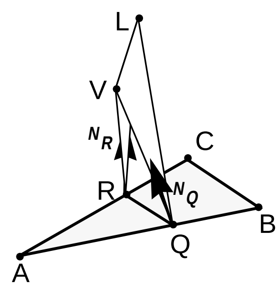


Fig. 11 Find a point which satisfies the H2 condition

Because the locus of points where the H1 condition is satisfied may be a curve, the resulting point on the line QR does not necessarily satisfy the H1 condition. More details are described in the appendix.

If the actual highlight is far from RQ or if a more accurate position is required, the H1 and the H2 tests can be repeated in an interleaved manner (Fig 12). After estimating P_1 we choose a point P_2 which satisfies the H1 condition on the line AP_1. If the point P_2 is not sufficiently accurate, we make a new estimate P_3 on the line RP_2 or P_2Q, depending on the H2 condition on the P_2. The next

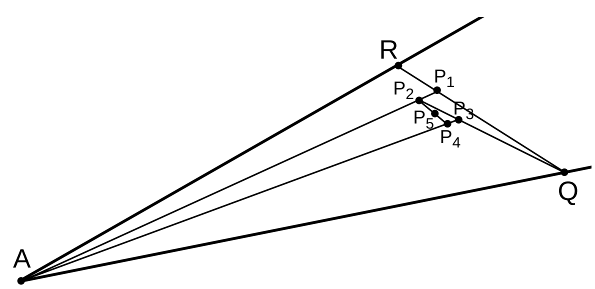


Fig. 12 Find a point which satisfies the H2 condition

estimate is the point on the line AP_3 which satisfies the H1 condition. This process continues until the highlight is determined to sufficient accuracy.

3.1.2. Additional vertex insertion. When the center of a highlight falls in a triangle, the triangle is subdivided into three small triangles by inserting a new vertex on the center of the highlight and putting edges between the new vertex and the old vertices. This new vertex guarantees that Gouraud shading finds the highlight inside the triangle. When the center of highlight falls outside the triangle, the triangle is divided into two triangles by inserting a vertex on the point Q or R depending on which one is nearer to the center of the highlight.

The plot of actual intensities between two points is not linear as Gouraud shading interpolates intensities (Fig. 13).

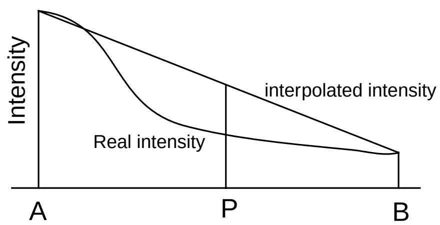


Fig. 13 Real and interpolated intensities along AB

Therefore a highlight spreads over the whole polygon if it falls on a vertex. Some extra vertices on edges which meet at the highlighted vertex limit the highlight to a smaller area (Fig. 14). There are many factors which influence the number of extra vertices and their best positions. We simply choose the middle point of an edge as a candidate position for vertex insertion. The difference between the interpolated color and real color is computed at the candidate point. If the difference is larger than a threshold, the candidate point becomes a vertex and the triangle is divided into two small triangles. The subdivided triangle which has a highlight on its vertex is recursively subdivided until the intensity error is below a threshold (Fig. 14). This is time consuming because the extra vertex insertion test repeatedly calculates an illumination equation on two vertices and a candidate point. Controlling the threshold allows the application to control the balance between shading quality and processing time.

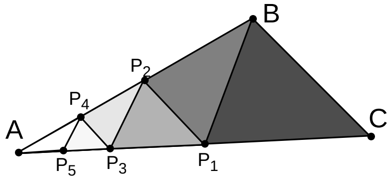
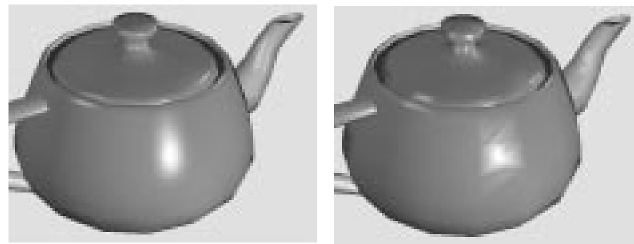


Fig. 14 Extra vertex insertion and subdivision sequence



(a) Phong (b) Highlight subdivision

Fig. 15 Comparison of Phong and Highlight subdivision

3.2. Regular subdivision

The highlight subdivision method consumes time to find the highlighted point in a triangle and calculate an illumination equation for extra vertex insertions. Regular subdivision is an alternative that minimizes the subdivision time by recursively splitting triangles into four smaller triangles regardless of where the highlight falls within a PHT (Fig. 16). The mid points on the three edges are connected

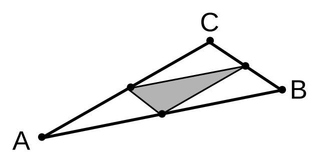
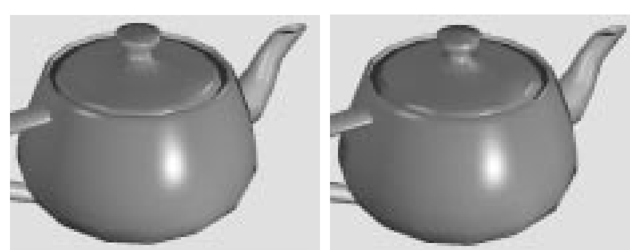


Fig.16 Regular subdivision

to each other [SHU 91]. The subdivision is applied recursively based on the screen area measure of the original triangle. The vertices of a PHT are transformed into screen space where their area is calculated by taking half of the magnitude of the cross product of any two edges ($|\boldsymbol{AB} \times \boldsymbol{AC}|/2$). Each subdivided triangle has 1/4 of



(a) Phong (b) Regular subdivision

Fig. 17 Comparison of Phong and Regular subdivision

the area of the parent triangle, so we calculate the area only once for each PHT.

3.3. Phong shading

PHTs may simply be rendered with Phong shading, while the remaining triangles are Gouraud shaded [PHON

75b]. If the rendering system supports both methods, this is a simple approach. But since most graphics hardware only supports Gouraud shading, Phong shading may only be possible through software. This option may not be attractive for hardware-assisted rendering systems.

4. Experimental results

These algorithms were implemented on a Sun SparcStation20SX with a software renderer and a hardware renderer. The software renderer uses a software z-buffer and scanline rasterizing algorithm, and the hardware renderer uses a hardware z-buffer and a Gouraud interpolation engine with software lighting calculation.

The Utah teapot was rendered 60 times with four finite distance lights and different viewpoints in each image size (256x256 and 512x512) (Fig. 18). Table 1 shows the average processing times in milliseconds The threshold of the highlight subdivision is 0.3, which means up to 30% of relative intensity difference is allowed. The threshold of the regular subdivision is 32 pixels, meaning a PHT is subdivided until the screen areas of subdivided triangles are less than 32 pixels

Table 1. Processing times (unit: milliseconds)

256x256 image size	S/W	H/W
Gouraud	480	110
Highlight subdivision	881	366
Regular subdivision	664	177
Gouraud + Phong	1104	---
Phong	3763	---

512x512 image size	S/W	H/W
Gouraud	957	183
Highlight subdivision	1408	450
Regular subdivision	1632	387
Gouraud + Phong	3436	---
Phong	14630	---

The original data is consisted of 896 triangles. The highlight subdivision method divides 74.5 triangles into 716.5 triangles and generates 1538 triangles in both image sizes. The regular subdivision method divides 288.4 triangles and generates a total of 1075.5 triangles in the 256x256 image and, 1955.6 triangles in the 512x512 image. Table 2 summarizes the number of triangles subdivided and rendered in each AS approach.

Table 2. Number of triangles

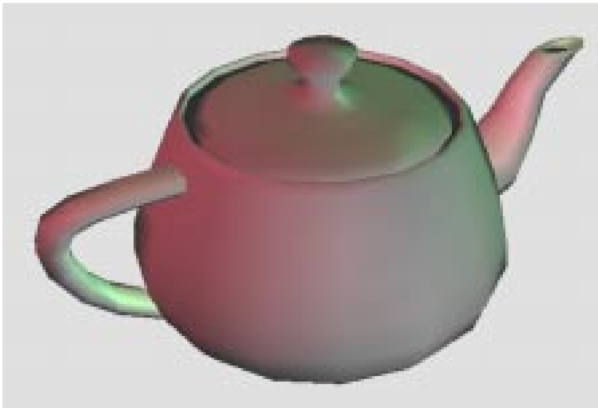
# of final triangles (# of subdivided ones)	256x256	512x512
Gouraud	896.0	896.0
Highlight subdivision	1538.0 (74.5)	1538.0 (74.5)
Regular subdivision	1075.5 (288.4)	1955.6 (288.4)

With the software renderer both adaptive shading methods require less than twice the time of Gouraud shading, and Phong shading requires 7.8 times as long as Gouraud shading in the 256x256 image and 15.3 times as long in the 512x512 image. With the hardware renderer the regular subdivision requires twice the time of Gouraud shading and the highlight subdivision requires three times as long.

The regular subdivision method produces better images because it usually generates more and regular shape triangles than the highlight subdivision method given roughly equivalent processing time. Since it takes a triangle's screen area into account for subdivision, it generates more triangles in the larger image, automatically adjusting itself to the image size. In contrast, the highlight subdivision method generates the same number of triangles regardless of the image size, since it only depends on intensities at vertices. It often subdivides a triangle into irregular ones, depending on the center of a highlight, generating an odd-looking polygonal shape highlight.

With the software renderer, the regular subdivision takes more time than the highlight subdivision when it renders more triangles. By leveraging the efficiency of the hardware renderer, the regular subdivision takes less time than the highlight subdivision even though it renders more triangles.

A broad highlight that is centered in one triangle may extend into neighbors, which should also be specially treated. We subdivide neighbors which share vertices with a PHT. It is possible for a highlight to be very broad, in which case subdivision of the immediate neighbors is not enough. This is another control factor between image quality and processing time.



(a) Gouraud

(b) Phong

(c) Highlight subdivision

(d) Regular subdivision

Fig. 18 Utah teapots shaded by various methods

5. Summary and conclusions

We described a fast PHT test that determines which triangles have highlights on them. Triangles that fail the test are rendered with Gouraud shading. Triangles that pass the test are specially handled at greater expense to produce more accurate highlights. Two subdivision methods were described that have different advantages. According to our experiment, the regular shading method adaptively produces better images in various image sizes. It is also faster when it works with a hardware renderer. The PHT test works with general N-sided polygons as well but the subdivision process becomes more complex.

There were other attempts to combine Phong shading and Gouraud shading [PHON 75b] [EINK 91]. [PHON 75b] finds triangles with highlights and renders them with Phong shading, while it renders other triangles with Gouraud shading. This method is simple but it requires both renderers. It is not attractive to graphic systems with Gouraud hardware renderer. [EINK 91] uses a very similar method to ours but it generates small irregular triangles. Since all processes are done in screen coordinates, the algorithm modifies the middle of a rendering pipeline and does not work with existing renderers.

Adaptive shading is presented as a method for efficiently allocating processing resources where they have an impact on the image. As an example of adaptive shading, two methods are proposed for efficiently obtaining high-quality highlights from existing Gouraud software and hardware rendering systems. Adaptive shading also permits the application to control the quality of the image and processing time requirements by adjusting thresholds.

6. Appendix

It is not simple to find the center of highlight in a triangle representing a curved surface. This appendix explains the related mathematics in detail.

The approximated highlight position can be directly computed by solving three 1D problems. First find the point Q on the edge AB where the H1 condition is satisfied (Fig. 19). At a point P the normal $\boldsymbol{N_P}$ is linearly interpolated from the vertex A and vertex B values. The point P and the normal $\boldsymbol{N_P}$ can be represented parametrically,

$$\begin{aligned} \boldsymbol{P} &= \mathrm{A} + t(\mathrm{B} - \mathrm{A}) \\ \boldsymbol{N_P} &= \boldsymbol{N_A} + t(\boldsymbol{N_B} - \boldsymbol{N_A}) \end{aligned}$$

If the normal $\boldsymbol{N_P}$ is on the plane LVP, the dot product of $\boldsymbol{N_P}$ and the cross product between the light vector $\boldsymbol{PL}$ and the view vector $\boldsymbol{PV}$ must be zero.

$$(\boldsymbol{PL} \times \boldsymbol{PV}) \cdot \boldsymbol{N_P} = 0$$

The light vector $\boldsymbol{PL}$ can be expressed in terms of t,

$$\begin{aligned} \boldsymbol{PL} &= \mathrm{L} - \mathrm{P} \\ &= \mathrm{L} - (\mathrm{A} + t(\mathrm{B} - \mathrm{A})) \\ &= \mathrm{L} - \mathrm{A} - t(\mathrm{B} - \mathrm{A}) \\ &= \boldsymbol{AL} - t\boldsymbol{AB} \end{aligned}$$

The view vector $\boldsymbol{PV}$ is represented similarly,

$$\begin{aligned} \boldsymbol{PV} &= \mathrm{V} - \mathrm{P} \\ &= \boldsymbol{AV} - t\boldsymbol{AB} \end{aligned}$$

The cross product of $\boldsymbol{PL}$ and $\boldsymbol{PV}$ becomes

$$\begin{aligned} \boldsymbol{PL} \times \boldsymbol{PV} &= (\boldsymbol{AL} - t\boldsymbol{AB}) \times (\boldsymbol{AV} - t\boldsymbol{AB}) \\ &= \boldsymbol{AL} \times \boldsymbol{AV} - t\boldsymbol{AB} \times \boldsymbol{AV} - \boldsymbol{AL} \times t\boldsymbol{AB} \\ &= \boldsymbol{AL} \times \boldsymbol{AV} - t(\boldsymbol{AB} \times \boldsymbol{AV} + \boldsymbol{AL} \times \boldsymbol{AB}) \end{aligned}$$

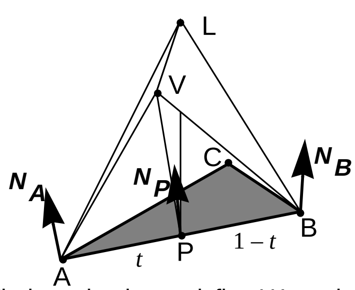


Fig. 19 Find a point that satisfies H1 on the edge AB

The condition to solve for is:

$$\begin{aligned} (\boldsymbol{PL} \times \boldsymbol{PV}) \cdot \boldsymbol{N_P} &= 0 \\ &= (\boldsymbol{AL} \times \boldsymbol{AV} - t(\boldsymbol{AB} \times \boldsymbol{AV} + \boldsymbol{AL} \times \boldsymbol{AB})) \\ &\quad \cdot (\boldsymbol{N_A} + t(\boldsymbol{N_B} - \boldsymbol{N_A})) \\ &= (\boldsymbol{AL} \times \boldsymbol{AV}) \cdot \boldsymbol{N_A} \\ &\quad + (\boldsymbol{AL} \times \boldsymbol{AV}) \cdot t(\boldsymbol{N_B} - \boldsymbol{N_A}) \\ &\quad - t(\boldsymbol{AB} \times \boldsymbol{AV} + \boldsymbol{AL} \times \boldsymbol{AB}) \cdot \boldsymbol{N_A} \\ &\quad - t(\boldsymbol{AB} \times \boldsymbol{AV} + \boldsymbol{AL} \times \boldsymbol{AB}) \cdot t(\boldsymbol{N_B} - \boldsymbol{N_A}) \\ &= (\boldsymbol{AL} \times \boldsymbol{AV}) \cdot \boldsymbol{N_A} \\ &\quad + t\begin{pmatrix} (\boldsymbol{AL} \times \boldsymbol{AV}) \cdot (\boldsymbol{N_B} - \boldsymbol{N_A}) \\ -(\boldsymbol{AB} \times \boldsymbol{AV} + \boldsymbol{AL} \times \boldsymbol{AB}) \cdot \boldsymbol{N_A} \end{pmatrix} \\ &\quad - t^2(\boldsymbol{AB} \times \boldsymbol{AV} + \boldsymbol{AL} \times \boldsymbol{AB}) \cdot (\boldsymbol{N_B} - \boldsymbol{N_A}) \end{aligned}$$

The condition is a quadratic in t and its solution is well known.

$$a\boldsymbol{t}^2 + b\boldsymbol{t} + c = 0$$

$$\begin{aligned} \boldsymbol{t} &= \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \\ a &= (\boldsymbol{AB} \times \boldsymbol{AV} + \boldsymbol{AL} \times \boldsymbol{AB}) \cdot (\boldsymbol{N_B} - \boldsymbol{N_A}) \\ b &= (\boldsymbol{AB} \times \boldsymbol{AV} + \boldsymbol{AL} \times \boldsymbol{AB}) \cdot \boldsymbol{N_A} \\ &\quad - (\boldsymbol{AL} \times \boldsymbol{AV}) \cdot (\boldsymbol{N_B} - \boldsymbol{N_A}) \\ c &= -(\boldsymbol{AL} \times \boldsymbol{AV}) \cdot \boldsymbol{N_A} \end{aligned}$$

Using the same process we find point R on the edge AC or CB where the H1 condition is satisfied. It remains to find the point which satisfies the H2 condition on the line QR (Fig. 20). Since the trace of the points where the H1 condition is satisfied is a curve between QR, points on the line QR do not necessarily satisfy the H1 condition. If high curvature surfaces are approximated with many triangles, the difference between $\boldsymbol{N_Q}$ and $\boldsymbol{N_R}$ is very small and it is reasonable to use the line as an approximation of the curve.

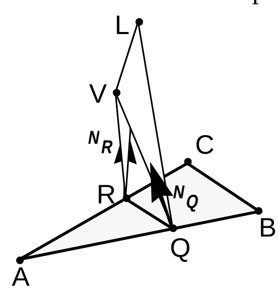


Fig. 20 Find a point which satisfies the H2 condition

Estimating where the highlight falls along QR is simplified if we assume that the length of the line QR is short relative to the light and view point distances from QR (Fig. 21). In that case we can approximate $|\boldsymbol{RL}| = |\boldsymbol{QL}|$ and $|\boldsymbol{RV}| = |\boldsymbol{QV}|$. Cosine values for the angles between the reflected light $\mathrm{L_R}$ and the view point (Fig. 21) at points Q

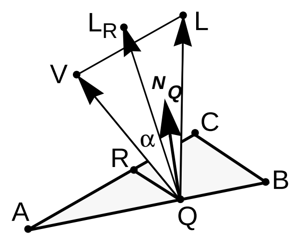


Fig. 21 Calculate a measure on Q

and R convey a relative measure of distance from the H2 condition at points Q and R.

$$V \cdot \boldsymbol{L_R} = |\boldsymbol{V}||\boldsymbol{L_R}|\cos\alpha$$

The relative difference between the cosine terms at Q and R determines the estimated highlight position along QR.

References

AIRE 90

Airey, John M. Increasing Update Rates in the Building Walkthrough System with Automatic Model-Space Subdivision and Potentially Visible Set Calculations. Ph.D. thesis, UNC Chapel Hill, 1990

BERG 86

Bergman, L., Fuchs, E. Grant, and S. Spach, “Image Rendering by Adaptive Refinement,” SIGGRAPH 86, pp. 29-37

BISH 86

Bishop, Gary, Weimer, D. M., Fast Phong Shading, pp. 103 - 106, Vol. 20, No. 4, Siggraph 1986

COHE 88

Cohen M.F., S.E. Chen, J.R. Wallace, and D.P. Greenberg, “A Progressive Refinement Approach to Fast Radiosity Image Generation,” SIGGRAPH 88, pp.75-84

COOK 87

Cook, R.L., L. Carpenter, and E. Catmull, “The Reyes Image Rendering Architecture,” SIGGRAPH 87, pp. 95-102

EINK 91

Einkauf, M. A., Klock, M. M., and L. N. Thi, “Shading of 3-D objects in graphics system - applying gouraud shading adaptively to triangular sizes heuristically derived using surface normal and incident light vectors”, European Patent # 0 456 408 A2, 1991

FOLE 90

Foley, J. D. and A. van Dam,. Computer Graphics: Principles and Practice, 1990

FUCH 89

Fuchs, H., J. Poulton, J. Eyles, T. Greer, J. Gold feather, D. Ellsworth, S. Molnar, G. Turk, B. Tebbs, and L. Israel, “Pixel-Planes 5: A Heterogeneous Multiprocessor Graphics System Using Processor-Enhanced Memories,” SIGGRAPH 89, pp.79-88.

GOUR 71

Gouraud, H. “Continuous Shading of Curved Surfaces”, IEEE Transactions on Computers, vol. 20, no. 6, pp. 623-629, 1971

PHON 75a

Phong, B. T. “Illumination for Computer Generated Pictures,” CACM, 18(6), June 1975, pp. 311-317

PHON 75b

Phong, B. T. and Crow, Franklin C. “Improved Rendition of Polygonal Models of Curved Surfaces,” Second USA-JAPAN Computer Conference, 1975, pp. 475-480

SHU 91

Shu, Renben, Liu Alan, “A Fast Ray Casting Algorithm Using Isotriangular Subdivision,” IEEE Visualization’91 pp. 232-237, October 1991, Conference Proceeding.