The Visual Computer

Horizon mapping: shadows for bump-mapped surfaces

Nelson L. Max

Lawrence Livermore National Laboratory,
P.O. Box 808, L-301 Livermore, CA 94550, USA

Bump mapping produces realistic shading by perturbing normal vectors to a surface, but does not show the shadows that the bumps cast on nearby parts of the same surface. In this paper, these shadows are found from precomputed tables of horizon angles, listing, for each position entry, the elevation of the horizon in a sampled collection of directions. These tables are made for bumps on a standard flat surface, and then a transformation is developed so that the same tables can be used for an arbitrary curved parametrized surface patch. This necessitates a new method for scaling the bump size to the patch size. Incremental calculations can be used in a scan line algorithm for polygonal surface approximations. The errors in the bump shadows are discussed, as well as their anti-aliasing. (An earlier version of this article appeared as Max [10].)

Key words: Bump mapping – Normal perturbation – Wrinkled surface – Horizon – Shadows – Parametrized surface – Shading – Illumination – Texture table

Current address: 3-28-8-412 Yaguchi, Ota-ku, Tokyo 146, Japan

James Blinn [1] showed how to simulate wrinkled surfaces by perturbing the normal vector, without moving the surface itself. This method, popularly known as "bump mapping," has since been widely applied to generate realistic images [2, 3].

The input to Blinn's algorithm is a bump function $F(u, v)$ of the surface parameters, stored as a two-dimensional table indexed by u and v. The wrinkled surface is formed by moving the parametrized surface $P(u, v)$ an amount $F(u, v)$ in the direction of the surface normal N. Assuming F is small compared with P, and using differential calculus and some vector identities, Blinn arrived at a simple formula for the new normal N' as perturbed by the effect of the wrinkles (see Sect. 3).

If this perturbed normal is used in a shading model, realistic highlights and reflections can be generated. A shading model which removes the contribution of a light source when the normal N' points away from it can also simulate self-shadowing of the bumps. However, no shading model which uses only N' can account for shadows cast by one bump on another. Near the edge between the illuminated and self-shadowed regions of a surface, where the normal is almost perpendicular to the light source direction, the bumps should cast long shadows which are important in making the rendering look realistic.

To simulate the cast shadows, a separate shadow map is needed for every light source direction. Since two angles are needed to specify a light source direction, it would seem that a two-parameter family of tables would be required. However, due to the special nature of the bump function, we will see in the following two sections that a one-parameter family of tables is sufficient. In the figures here, this parameter was sampled at only eight values, demonstrating that it is possible to generate good bump shadows with only eight extra numbers per texture entry, besides the bump height $F(u, v)$.

1 The sunset model

Imagine sunset at the equator on the spring equinox, when the sun appears to move in a vertical line towards the western horizon. Assume bumps on the earth, i.e., mountains, are defined by a single-valued bump function $F(u, v)$, so that there are no tunnels or overhanging cliffs. Then once a given point on the terrain becomes covered by the shadow of a mountain, it will remain in darkness until the next morning. Therefore, to specify the shadow

The Visual Computer (1988) 4:109–117

texture for every instant during the afternoon, it is only necessary to specify the time of sunset for each point on the terrain. This sunset time is determined by the angle of elevation of the western horizon, as viewed from a given terrain point. The horizon elevation angle is the maximum angle of elevation from the given point to any other terrain point located directly to its west. If the function $F(u, v)$ is tabulated with rows going west to east, only a single row of data is needed to find this maximum angle.

In computing the elevation angle, we assume a flat earth and do not take into account the extra lengthening of the shadows due to the curvature of the earth. In general, we will not know in advance the shape or curvature of the parametrized surface around which the bump texture will be wrapped, so we must precompute the shadows as if the surface were flat. Whenever $F(u, v)$ is small compared to the curvature of $P(u, v)$, this will give a good approximation to the shadows on the curved parametric patch $P(u, v)$ (see Sects. 5, 6).

2 Standard flat patches

Assume given a flat patch $P(u, v)=(u, v, 0)$ for $0 \leq u \leq 1$, $0 \leq v \leq 1$, with surface normal $N=(0, 0, 1)$, and a non-negative bump function $F(u, v)$. Then the wrinkled surface Q is $Q=\{(u, v, w) | 0 \leq u \leq 1, 0 \leq v \leq 1, w=F(u, v)\}$. Let L be a unit vector from a point on the patch in the direction towards a light source. In spherical coordinates with the pole along the Z axis, $L=(\sin \varphi \cos \theta, \sin \varphi \sin \theta, \cos \varphi)$. Here, φ is the angle L makes to the normal N, and θ is the angle of the projection of L on the (X, Y) plane, measured from the X axis. (Because of the convention for spherical coordinates, the horizon elevation angle discussed above is actually $90° - \varphi$.)

For each fixed θ, we may imagine the sun setting in a vertical plane with φ increasing from 0° to 90°. As above, the horizon angle in the direction corresponding to θ completely determines the shadow pattern for any sun vector L in this plane. Let $\beta(u, v, \theta)$ be a table of horizon angles, indexed by the surface parameters u, and v, and the angle θ.

This table is the one-parameter family of texture tables promised in the introduction. It is customary to use from 64 to 512 sampling values for u and v, since the bump function $F(u, v)$ is handled as a raster image. However, it is often sufficient to sample θ at only eight different values: 0°, 45°, 90°, 135°, 180°, 225°, 270° and 315°. These correspond to E, NE, N, NW, W, SW, S and SE for a sunset direction, and the horizon angle β can easily be found using rows, columns, or diagonals of the bump function table. This coarse sampling in θ still permits the shadows to move smoothly because we interpolate the sunset angle β between two adjacent tabulated values, and then compare the result with the actual sun angle to determine whether a point is in shadow. If we instead computed two shading intensities, including shadows, and then interpolated between them, we would see one shadow fade out and the other fade in, which would look less convincing. If the bump map consists of long ridges, as in the figures here, then at the worst case angle of $22^1/_2°$, the shadows are shortened by $\cos 22^1/_2 = 0.93$, so the error is only 7%. With 16 samples, the worst error would be 2%. Of course, if the texture consists of isolated narrow peaks, the errors will be more obvious. A film was produced showing craters on a rotating moon, with bump shadows rendered using only 8 samples for θ. The expected variation in the shadow length of the craters as the moon rotates is barely visible.

3 Scaling for curved patches

Let $P(u, v)$ be a parametrized surface, and $F(u, v)$ be a bump function, both defined for $0 \leq u \leq 1$ and $0 \leq v \leq 1$. Following Blinn [1] we can take P_u and P_v to be the tangent vectors (partial derivative vectors) in the u and v directions, $N=P_u \times P_v$ to be the surface normal, and $P'=P+F\,N/|N|$ to be the perturbed surface. After dropping terms of first order in F, Blinn finds

$$P'_u = P_u + F_u\,N/|N|,$$

$$P'_v = P_u + F_v\,N/|N|,$$

and then

$$N' = P'_u \times P'_v = N + D,$$

where

$$D = (F_u\,N \times P_v - F_v\,N \times P_u)/|N|$$

is the perturbation of the normal due to the bumps.

Blinn next interprets this perturbation as a rotation of the normal N by an angle $\tan^{-1}(|D|/|N|)$ about the axis A perpendicular to N and N'. He remarks

that if a patch is made twice as large, N will be multiplied by 4 but D only by 2, so the normal will be rotated less, causing an undesired flattening of the bumps.

His solution is to find the rotation angle from a standard flat unit-sized patch, as in Sect. 2 above, and then determine only the axis of rotation from the geometry of the actual patch. However, the resulting shading may not correspond to any physical surface.

For example, suppose the bump function is defined by the pyramid ABCDE of height 1/2 as shown in Fig. 1. Each of the slanted faces of the pyramid slopes by 45°.

The patch $P(u, v)=(2u, v, 0)$ stretches the u direction by 2 but leaves v unchanged. Blinn's scale adjustment would keep the slopes of 45°, resulting in inconsistent heights for the center vertex. As shown in Fig. 2, E' has height 1/2, but F' has height 1.

It should theoretically be possible to determine a 3-D shape from its shading (especially if there are two light sources of different colors), but nobody has ever noticed any visually unreasonable effects due to Blinn's scaling algorithm. However, the situation in Fig. 2 is clearly inadequate if shadows are to be determined, so we use an alternate scaling method. We define a perturbed surface P'' by

$$P''=P+FN/|N|^{1/2}.$$

Now, if the surface is scaled up by a factor of 2, N will be scaled up by 4, but $N/|N|^{1/2}$ will be scaled up only by 2 and the apparent shapes of the bumps will be unchanged. If only u is stretched by a factor of 2, some regions will become more steeply sloped, and some less. As shown in Fig. 3, E'' has a compromise height of $(1/2)\sqrt{2}$.

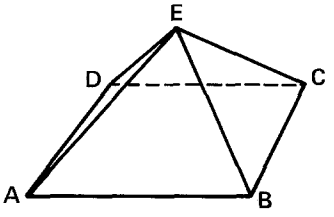

Fig. 1. A piecewise-linear bump function defined for $0\leq x\leq 1$ and $0\leq y\leq 1$. It takes the value 0 at the corners of the squares ABCD, and the value 1/2 at the center. The sides slope at 45°.

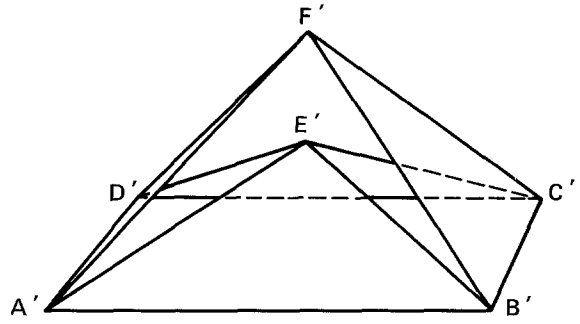

Fig. 2. The bump function of Fig. 1, stretched over the rectangle $A'B'C'D'$, with $0\leq x\leq 2$, and $0\leq y\leq 1$, and taking the value 0 on the corners. If the sides must slope by 45°, there are two inconsistant positions E' and F' for the center.

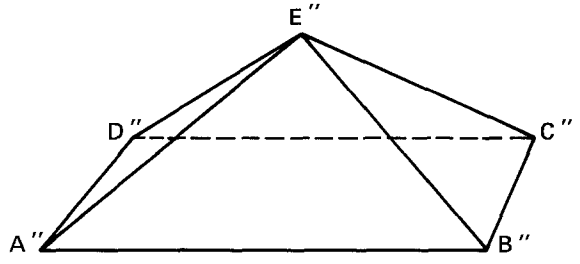

Fig. 3. The bump function of Fig. 1, stretched over the same rectangle as Fig. 2, but with center vertex E'' of height $(1/2)\sqrt{2}$. The face $A''E''D''$ has slope less than 45°, and the face $A''E''B''$ has slope greater than 45°.

With this adjustment, the approximations for P_u'' and P_v'' become

$$P_u''=P_u+F_u N/|N|^{1/2}$$

and

$$P_v''=P_v+F_v N/|N|^{1/2}$$

and the perturbation D'' becomes

$$D''=(F_u N\times P_v-F_v N\times P_u)/|N|^{1/2}.$$

Now consider the skew transformation $P(u, v)=(u+v, v, 0)$. In this case $P_u=(1, 0, 0)$ and $P_v=(1, 1, 0)$, so $N=P_u\times P_v=(0, 0, 1)$. The bumps will still keep their original height, which is appropriate because the skewing transformation does not change the surface area.

4 Efficient calculation for polygonal approximations

When rendering smooth surfaces such as spheres, cylinders, or polynomial patches, the partial derivative vectors P_u and P_v may be available analytically. But this is not the case when rendering polygonal approximations to smooth surfaces.

The input for a polygonal approximation usually specifies the coordinates, the normal, and the u and v parameters at each vertex, and the u, v, and normal are then interpolated across each polygon. But in the formula for D'' above, what we really need are the vectors $N \times P_u$ and $N \times P_v$. For determining the final perturbed normal, we also need the vector N itself.

If the input polygons are all triangles, then the u and v coordinates at the three vertices uniquely determine a linear (u, v) parametrization of each triangle, and thereby, the partial derivative vectors P'_u and P'_v in the plane of each triangle. For a smooth surface, these vectors should be projected onto the tangent plane, which is perpendicular to the normal. Since this projection is accomplished by adding a multiple of the normal N, and such an addition does not affect the value of $N \times P'_u$ and $N \times P'_v$, it is sufficient to use the constant vectors P'_u and P'_v for the whole triangle. If a polygon is not a triangle, some average of the P'_u and P'_v for its subtriangles must be used.

When a polygon is scan converted in a scan-line or Z-buffer algorithm, the normal N is linearly interpolated down the edges of the polygon, to get the values at the ends of a scan line segment, and then interpolated across the segment. The vectors $N \times P'_u$ and $N \times P'_v$ must similarly be interpolated across the scan segment, which can be done incrementally with one addition per pixel per vector component. This is much more efficient than the two multiplications and a subtraction needed to find each component of a cross product.

The trees in Fig. 9 were built from triangles, textured and shadowed by this technique. There are potential discontinuities in P'_u and P'_v across triangle boundaries, but these do not appear evident in this picture. These discontinuities could be eliminating by defining P_u and P_v at each vertex as the average of the values at triangles touching the vertex, or including P_u and P_v in the input data at each vertex. The vectors $N \times P_u$ and $N \times P_v$ could then be interpolated across the triangles in the same manner as N.

5 Light vector transformation

We saw in Sect. 2 how to compute shadows for a wrinkled surface Q defined on a standard flat patch, and in Sect. 3 how to define a wrinkled surface P'' corresponding to a curved patch P. Now we will look for an affine transformation B which brings Q to a position which approximates P'' at the point $P''(u_0, v_0)$. The affine transformation B is made up of a linear transformation C and a translation. We need to have

$$\begin{aligned} B(u_0, v_0, 0) &= P(u_0, v_0) \\ B(u_0, v_0, F(u_0, v_0)) &= P''(u_0, v_0) \\ C(1, 0, 0) &= P_u(u_0, v_0) \\ C(0, 1, 0) &= P_v(u_0, v_0) \\ C(0, 0, 1) &= N/|N|^{1/2}. \end{aligned}$$

Therefore,

$$\begin{aligned} B(u, v, w) = (u-u_0)\, P_u(u_0, v_0) + (v-v_0)\, P_v(u_0, v_0) \\ + w\, N/|N|^{1/2} + P(u_0, v_0). \end{aligned}$$

If the curvature of $P(u, v)$ is small, in a sense to be discussed in Sect. 6, then the transformed standard bump surface $B(Q)$ is close enough to the wrinkled patch P'' for the purpose of shadow computation. Since an affine transformation takes straight lines to straight lines, transforming directions by its linear part, the point $(u_0, v_0, F(u_0, v_0))$ on surface Q will be in shadow from a light in direction K if and only if the point $P''(u_0, v_0)$ on surface $B(Q)$ is in shadow from a light in direction $C(K)$.

Therefore, if L is the direction from point $P''_0 = P''(u_0, v_0)$ to the light source, we can take $C^{-1}(L)$ to be the light direction on our standard bump surface Q, and compute the shadow as in Sect. 3. The matrix for the linear part C of B is made up from the three vectors $P_u(u_0, v_0)$, $P_v(u_0, v_0)$ and $N/|N|^{1/2}$.

The angles φ and θ are computed from the components of $C^{-1}(L)/|C^{-1}(L)|$, a horizon angle β is computed by interpolation in the tables $\beta(u, v, \theta)$, and the point P''_0 is in shadow if $\varphi > \beta$. Note that this method will also work for light sources at a finite distance or even inside the field of view. In these cases, the light source direction L should be recomputed at each surface point, as it should with any other finite-light-source shading model. If multiple light sources are used, the inverse matrix C^{-1} need be computed only once per surface point.

It is also possible to use the angles φ and β to create penumbras for a circular area light source like the sun. Since $\varphi-\beta$ represents the angle of the sun's center below the horizon, the visible fraction of the sun is a function of $\varphi-\beta$, which can be used to multiply the contribution of the shading from the sun illumination. If δ is the apparent angular diameter of the light source (1/2° for the sun), let $\alpha=(\beta-\varphi)/(\delta/2)$.

Then by integration or plane geometry area formulas, the visible fraction of the sun is

$$f(\alpha)=\begin{cases}0.0 & \alpha\le -1\\ 0.5+(\sin^{-1}\alpha+\alpha\sqrt{1-\alpha^2})/\pi & -1\le\alpha\le 1\\ 1.0 & 1\le\alpha.\end{cases}$$

This function can be tabulated, and used to determine the shading of the penumbra. It will give an accurate answer for the case of a shadow cast by a horizontal ridge in the bump map, but will be slightly inaccurate for sloping bump profiles, because $\varphi-\beta$ measures an angle in a plane perpendicular to the original surface, not perpendicular to the bump profile. Also, the transformation C^{-1} can take a round sun into an elliptical one, so δ should be taken to be the angular extent of the ellipse in the β direction.

The method of bump mapping as described by Blinn [1] does not effect the profile of a surface, only its shading. Therefore, when a polygonal or smooth- surface based hidden-surface/shadow-casting algorithm is applied for inter-surface occlusion and shadows, the shadows cast by bumpy surfaces will not appear bumpy. Rob Cook [4, 8] describes a generalization of bump maps called displacement maps, which can change the profile of a bumpy surface. Such displacement maps are also capable of casting bumpy shadows from one surface to another, if the shadow algorithm of Williams [5] is used. (See [11] for a more recent implementation of this shadow algorithm.)

6 Effects of surface curvature

Shadows from bumpy profiles are difficult, and not treated here. However, it is easier to cast the shadows of a smooth profile on the nearby bumps belonging to the same surface, taking into account its curvature. This should add to the realism of textures with scattered small bumps on an otherwise smooth surface.

The left side of Fig. 4 shows a section curve S of a surface P'', made by the plane T containing the normal N and the light source direction L at a point $P_0=P(u_0, v_0)$. There is an isolated bump which moves P_0 to P_0'' by an amount $h=|N|^{1/2}F(u_0, v_0)$. We want to find the condition for the line from P_0'' in the direction L to be tangent to the surface P at the profile point Q.

To second order, the section curve S is approximated by a circle with center C, whose radius R is the radius of curvature of S. The curvature of S is $k=1/R$. It can be computed from the first and second partial derivative vectors P_u, P_v, P_{uu}, P_{uv}, and P_{vv}, and the angle θ found at the end of Sect. 5, which defines the direction of the section plane T. To second order, $h=k\,x^2/2=x^2/(2R)$, where x is the distance from P_0'' to the tangency point Q, and $\gamma\cong x/R$, where γ is the central angle QCP_0 on the approximating circle. Eliminating x, we find $x\cong\sqrt{2Rh}$ and $\gamma\cong\sqrt{2h/R}$.

Recall that φ is the angle between N and L. Angle $QP_0''C$ is the supplement of angle φ and the complement of angle γ, so $\varphi=90°+\gamma$. Thus γ measures the tilt of the horizon below the 90° angle which would be expected if the surface P were flat. So if we replace the estimated horizon angle β from our table by $\beta'=\beta+\gamma$, the same test as before will apply: the point P_0'' is in shadow whenever $\varphi>\beta'$. If the surface is concave γ should be subtracted from β instead. These adjustments of β have the effect of lengthening shadows on convex surfaces and shortening them on concave surfaces, but not necessarily by the correct amount.

We can also use the curvature of the section S to determine the relative error in the length of the shadows, caused by the curvature of the surface. Suppose a light source at an angle α with N casts a shadow from D'' to P''. The right side of Fig. 4 shows the worst case, where the points P_0 and P_0'' are equal, because $F(u_0, v_0)=0$, but at D, F takes on its maximum value G, and $DD''=G|N|^{1/2}$.

If the angle γ is small, then angle $D''DP_0$ is approximately a right angle, and angle $DD''P_0$ is approximately equal to α. Thus x is approximately $DD''\tan\alpha=G|N|^{1/2}\tan\alpha$. As above, the change in the height of the bump due to the curvature is approximately $0.5\,k\,x^2$. So the relative change in the bump height is $0.5\,k\,x^2/(G|N|^{1/2})=(0.5/k)\,G|N|^{1/2}\tan^2\alpha$. This is also equal to the relative change in the shadow length. Thus, the affinely transformed surface $B(Q)$ discussed in Sect. 5 is

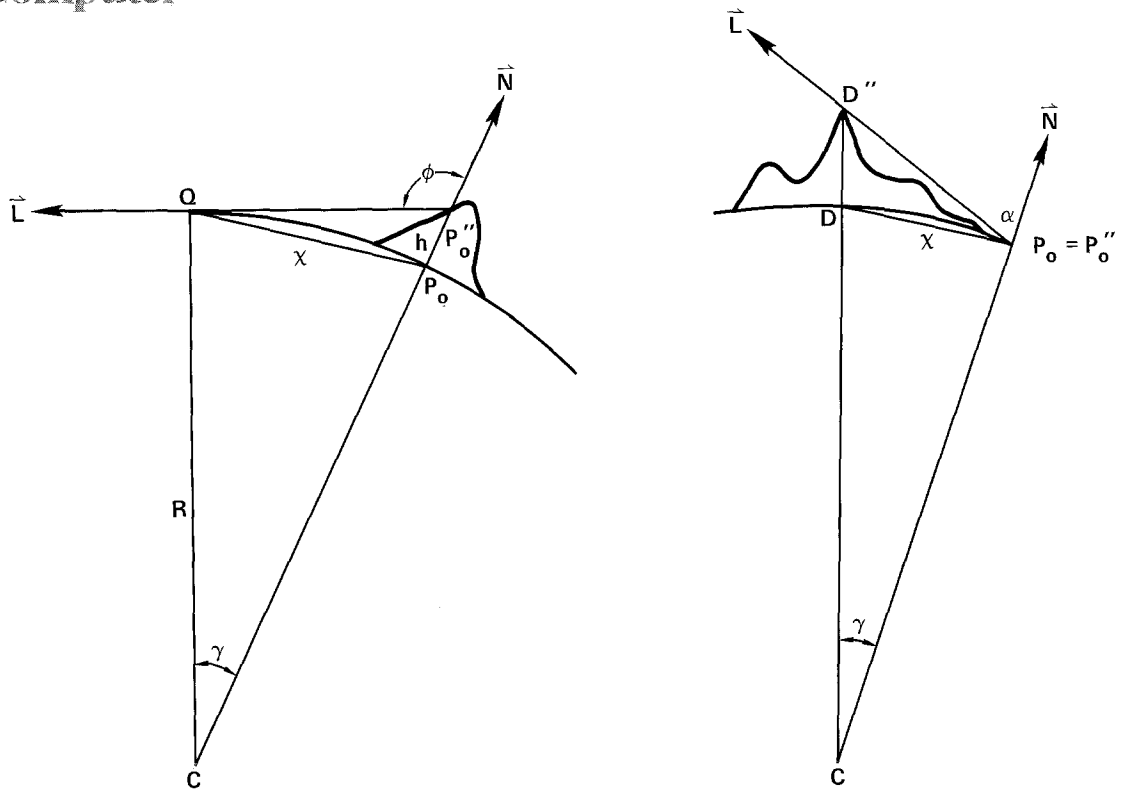

Fig. 4. *Left:* a section of the surface $P''(u, v)$ in a plane containing the normal and light-source directions at P_0. The illuminating ray is just tangent to the surface at Q. *Right:* A similar section with the illuminating ray just passing the top D'' of a bump of maximum height, and making an angle of α with the surface normal

sufficiently close to the actual surface P'' if $k\,G|N|^{1/2}\tan^2\alpha$ is small. The errors due to surface curvature will be greatest for grazing illumination, when $\tan^2\alpha$ is large.

7 Anti-aliasing

The comparison of the angle φ with the horizon angle β is an all-or-nothing test, so the shadows computed by this method will have jagged edges. To eliminate these jaggies, we can compute by interpolation the values of φ and β at the four corners of each pixel and use the method of Duff [6], Fig. 1, to determine the fraction of the area where $\varphi > \beta$. This will only work if all four corners of the pixel belong to the same patch, in which case the four values of φ and β can be reused for adjacent pixels.

The above method is appropriate if the projected size of the pixel on the bump map is small compared to the spatial frequency of the bumps, so that most pixels will contain only one shadow edge, or none at all. If the pixel covers several squares of the texture, it may be better to assume φ is constant, interpolate β only in the θ direction for each (u, v) map entry in the pixel's projection, and count the number of entries where $\varphi > \beta$. A superior method, which would also cure aliasing in the bump shading, would use several tables for the bump function, of successively coarser resolution, as suggested by Williams [7]. In addition, the penumbra calculation in Sect. 4 above will tend to smooth the shadow edges even if the penumbra itself is too narrow to be visible.

If the viewpoint is far enough from the surface, there will be so much of the texture mapped to each pixel that neither modest sized bumps nor their shadows should be visible in a correctly anti-aliased image. Instead, the bumps and their shadows should contribute to the surface roughness embodied in the bidirectional reflection function used for shading. Cabral [9] shows how to use horizon mapping to efficiently account for shadow effects when computing bidirectional reflection functions from surface bump maps.

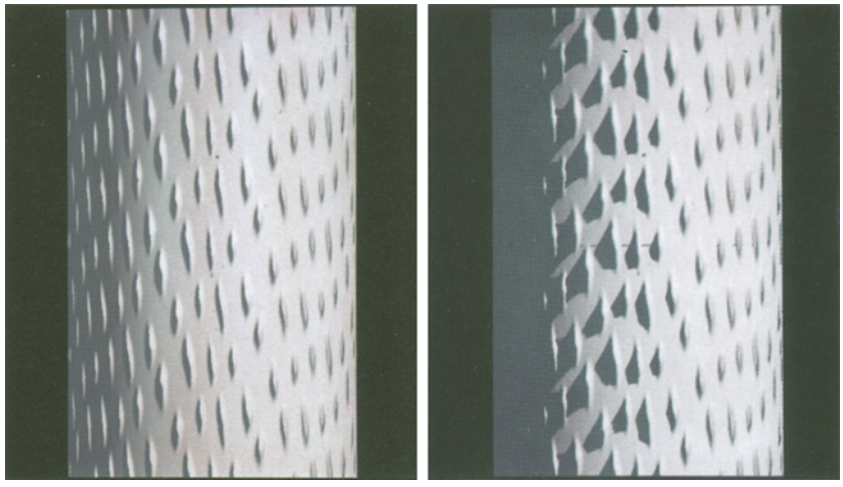

Fig. 5. A bump-mapped image of a tree trunk with a blister pattern on its bark. *Left*, with only normal perturbation, and *right*, with shadows

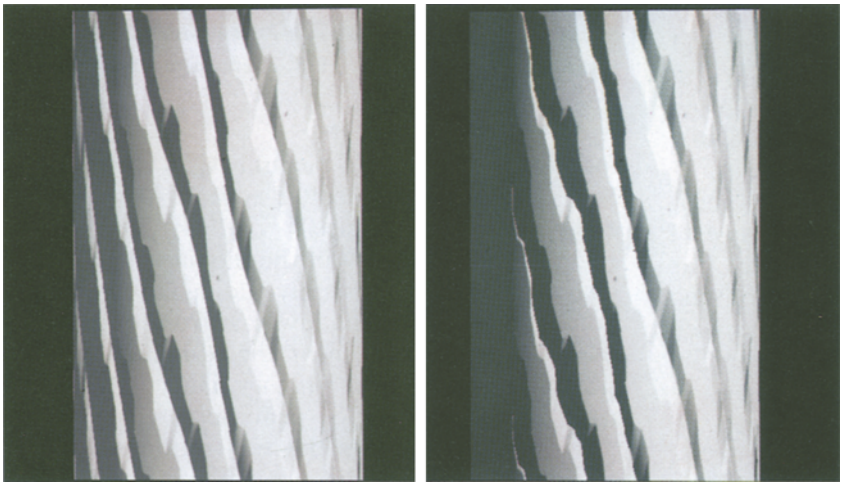

Fig. 6. A bump mapped image of a tree trunk, using a mathematically defined bump height function for the bark. *Left*, with only normal perturbation, and *right*, with shadows

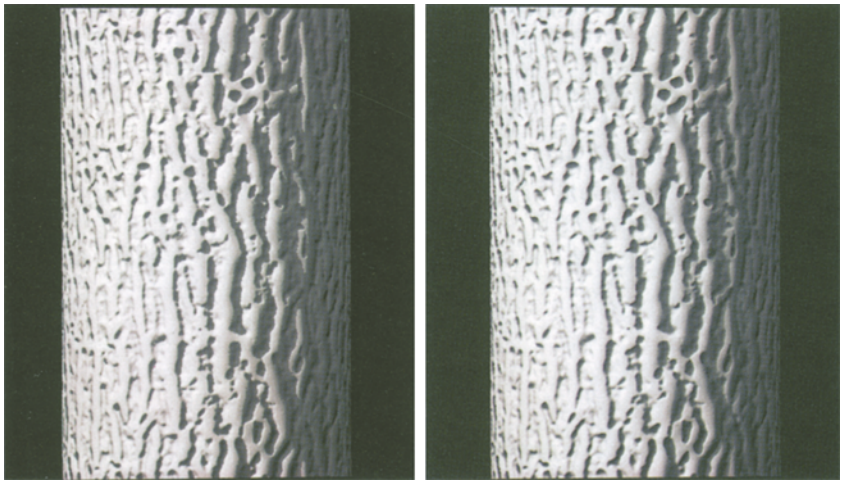

Fig. 7. An image of a tree trunk, with an actual bark bump map taken from Bloomenthal [3]. The ambient light in the shadowed region was taken as coming from the same direction as the sun, but with only 7% intensity, so the normal perturbation still gives texture to the shadow areas, unlike Figs. 5 and 6. *Left*, with point source illumination, and *right*, with penumbras from the sun's disk

8 Results

Figures 5, 6, and 7 show shadows on tree bark, formed by the methods above. In Figs. 5 and 6 the bump textures for tree bark were mathematically defined periodic functions on a 256×256 grid. In the horizon tables, the angle β was scaled to fit into 8 bits, and the values for β in the eight directions at each grid point were packed into a single 64 bit word on the Cray-1. In Figs. 5 and 6, the left side shows the results of Blinn's algorithm, and the right side shows the algorithm presented here. Notice that with Blinn's algorithm, parts of the bark whose perturbed normal points towards the light source are brightly illuminated, even though they are hidden from the light source by the rest of the cylinder. In Fig. 5, the shapes of the shadows give visual cues which are important in understanding the shapes of the bumps.

It took the Cray-1 18 s to compute the bump and horizon maps for the bark of Fig. 6, and 11 s to render Fig. 6, right, at 512×512 resolution, using these tables. For contrast, Fig. 6, left, took 6 s. These images are not anti-aliased.

In Fig. 7, the bump texture for the bark was measured by Bloomenthal [3] from a plaster cast of maple bark. On the left, a point light source was simulated causing aliasing at the high contrast shadow edges. On the right, a large disk was used for the sun, creating a penumbra, as discussed in Sect. 4. This gives a softer edge to the shadow, and removes the "jaggies."

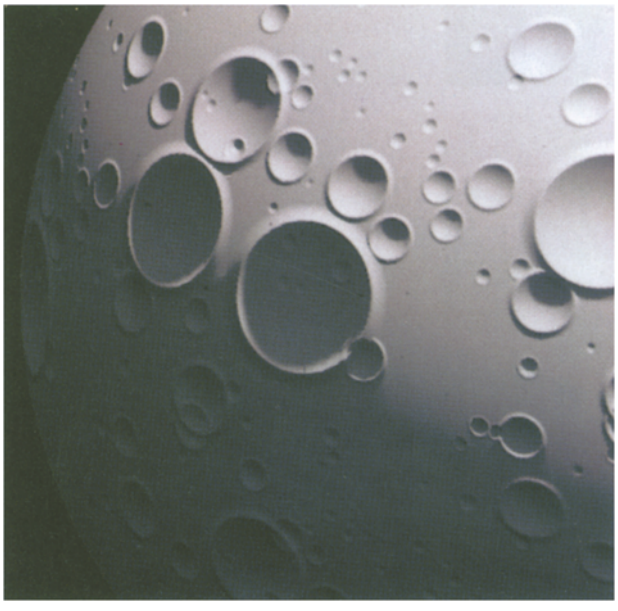

Fig. 8. A crater texture pattern on the moon, with the shadow illuminated by 7% earthshine

Fig. 9. The bark texture of Fig. 7, applied to polygon data for the tree trunks

Figure 8 shows a view of the moon, with a bump texture formed by randomly adding craters of random size. Note the shadow cast from one rim to the other, on craters near the terminator of the sunlit region. A sun of angular diameter $1^1/_2°$ was used, giving realistic penumbras. A movie, "Moon" of similar images was produced by the algorithm, and took $2^1/_2$ s per frame on the Cray XMP at 512×384 pixel resolution, using vectorized loops with a single processor.

Figure 9 shows the bark of Fig. 7, applied to a polygonal approximation of tree trunks, also designed by Jules Bloomenthal. The shadows of the bumps were computed as in Sect. 4 above, while the shadows of the polygons (including shadows in the haze) were computed by the algorithm of Max [12]. The image was sampled along 1633 slanted scan lines, as required by the polygon shadow algorithm, and then resampled into a 1024×1024 pixel raster.

Acknowledgements. I would like to thank Maria Lopez for typing this manuscript and Charles Grant for carefully reading it and suggesting the penumbra computation. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract number W-7405-Eng-48.

References

1. Blinn F (1978) Simulation of wrinkled surfaces. Computer Graphics 12(3):286–292
2. Whitted T (1980) An improved illumination model for shaded display. Commun ACM 23:343–349
3. Bloomenthal J (1985) Modelling the mighty maple. Computer Graphics 19(3):305–311
4. Cook RL (1984) Shade trees. Computer Graphics 18(3):223–231
5. Williams L (1978) Casting curved shadows on curved surfaces. Computer Graphics 12(3):270–274
6. Duff T (1985) Compositing 3-*D* rendered images. Computer Graphics 19(3):41–44
7. Williams L (1983) Pyramidal parametrics. Computer Graphics 17(3):1–11
8. Cook R, Carpenter L, Catmull E (1987) The reyes image rendering architecture. Computer Graphics 21(4):95–102
9. Cabral B, Max N, Springmeyer R (1987) Bidirectional reflection functions from surface bump maps. Computer Graphics 21(4):273–281
10. Max N (1986) Shadows for Bump-Mapped surfaces. In: Kunii TL (ed) Advanced Computer Graphics. Springer, Tokyo Berlin Heidelberg New York, pp 145–156
11. Reeves W, Salesin D, Cook R (1987) Rendering Antialiased Shadows with Depth Maps. Computer Graphics 21(4):283–291
12. Max N (1986) Atmospheric illumination and shadows. Computer Graphics 20(4):117–124

NELSON MAX is currently computing director at Hikari Kinema, and is working in Japan on an Omnimax stereo computer animated film to be exhibited at the Fujitsu Pavillion at Expo '90 in Osaka, Japan. He received his BA in mathematics at John Hopkins University in 1963 and his Ph.D. in mathematics at Harvard University in 1967. His research interests include computer representation of molecular surfaces and interactions and realistic representation of natural phenomena in computer graphics. Max has several awardwinning films to his credit.