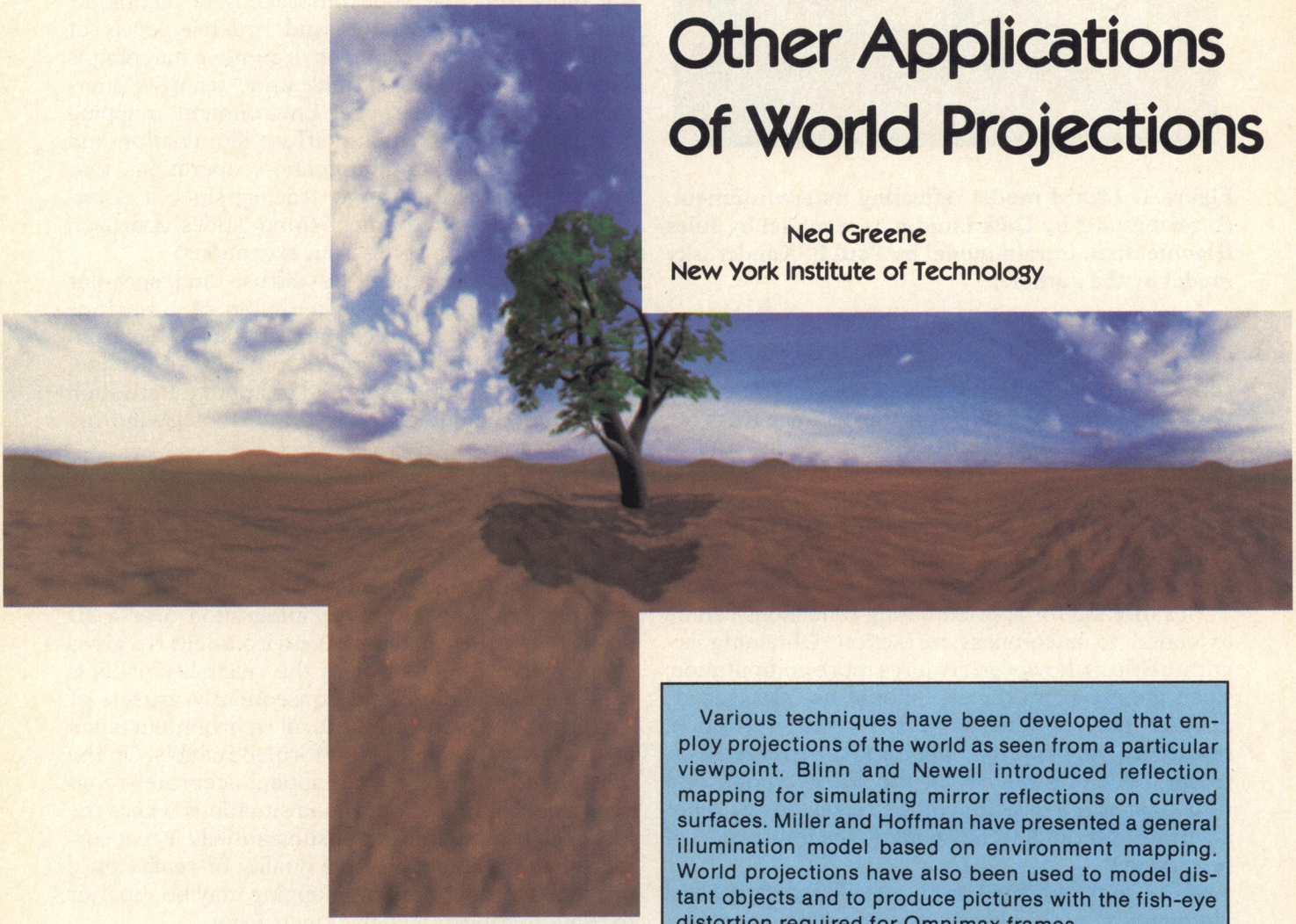
Environment Mapping and Other Applications of World Projections

Ned Greene
New York Institute of Technology



Various techniques have been developed that employ projections of the world as seen from a particular viewpoint. Blinn and Newell introduced reflection mapping for simulating mirror reflections on curved surfaces. Miller and Hoffman have presented a general illumination model based on environment mapping. World projections have also been used to model distant objects and to produce pictures with the fish-eye distortion required for Omnimax frames.

This article proposes a uniform framework for representing and using world projections and argues that the best general-purpose representation is the projection onto a cube. Surface shading and texture filtering are discussed in the context of environment mapping, and methods are presented for obtaining diffuse and specular surface illumination from prefiltered environment maps. Comparisons are made with ray tracing, noting that two problems with ray tracing—obtaining diffuse reflection and antialiasing specular reflection—can be handled effectively by environment mapping.

Reflection mapping, introduced by Blinn and Newell in 1976, is a shading technique that uses a projection of the world (a reflection map) as seen from a particular viewpoint (the "world center") to make rendered surfaces appear to reflect their environment.[1] The mirror reflection of the environment at a surface point is taken to be the point in the world projection corresponding to the direction of a ray from the eye as reflected by the surface. Consequently, reflections are geometrically accurate only if the surface point is at the world center, or if the

An earlier version of this article appeared in *Proc. Graphics Interface 86.*

0272-1716/86/1100-0021$01.00 © 1986 IEEE

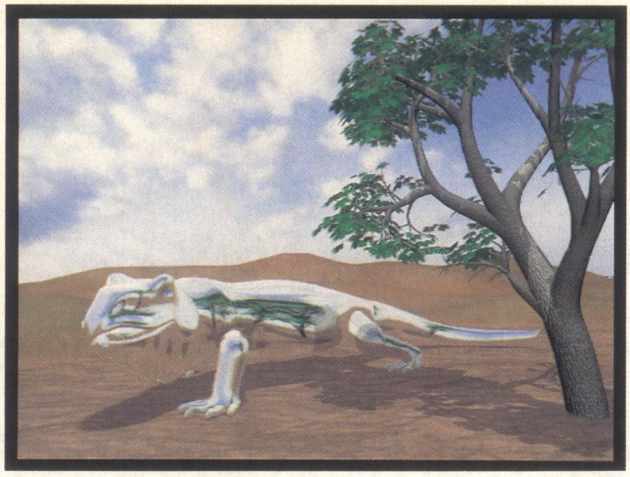


Figure 1. Lizard model reflecting its environment. (Lizard model by Dick Lundin, tree model by Jules Bloomenthal, terrain model by Paul P. Xander, sky model by the author.)

reflected object is greatly distant. Geometric distortion of reflections increases as the distance from the surface point to the world center increases and as the distance from the reflected object to the world center decreases. When applying reflection mapping to a particular object, the most satisfactory results are usually obtained by centering the world projection at the object's center.

This method for approximating reflections can be extended to encompass refraction. Obtaining accurate results, however, requires more computation, since the ray from the eye should be "ray traced" through the refractive object. As with reflection, results are only approximate for geometric reasons. (Simply bending the ray at the surface point and using this as the direction of the refracted ray is not accurate, although it may convey the impression of refraction.)

Miller and Hoffman have developed a general illumination model based on reflection mapping.[2] Essentially, they treat a world projection as an all-encompassing, luminous globe imprinted with a projection of the world that produces sharp reflections in smooth, glossy objects and diffuse reflections in matte objects. This is a good model of illumination in the real world, since it takes into account all illumination in the environment, although shadows are not explicitly handled, and, as with reflection mapping, results are only approximate for geometric reasons. To speak generically of this approach and conventional reflection mapping, the term "environment mapping" will be applied to techniques for shading and texturing surfaces that use a world projection. A world projection used for environment mapping is usually called an environment map.

Environment mapping vs. ray tracing

Environment mapping is often thought of as a poor man's ray tracing, a way of obtaining approximate reflection effects at a fraction of the computational expense. While ray tracing is unquestionably the more versatile and comprehensive technique, since it handles shadows and multiple levels of reflection and refraction,[3] environment mapping is superior in some ways, quite aside from its enormous advantage in speed. Environment mapping gracefully handles finding diffuse illumination and antialiasing specular illumination, operations that are problematical with ray tracing, since it point-samples the 3D environment. (Amanatides' approach of ray tracing with cones is an exception.[4])

Theoretically, finding the diffuse and specular illumination impinging on a region of a surface requires integration over part of the 3D environment, an inherently complex procedure.[5] Refinements to ray tracing proposed to approximate such integration include distributed ray tracing[5] and ray tracing with cones.[4]

Environment mapping simplifies the problem by treating the environment as a 2D projection, which allows integration to be performed by texture filtering. Specifically, diffuse and specular surface illumination can be found by filtering regions of an environment map. However, integration over a 2D projection rather than the 3D environment is a gross simplification that sacrifices the reliability of local shading information, and consequently aspects of shading that depend on the local environment (such as shadowing) cannot be performed reliably. On the other hand, environment mapping accurately conveys global illumination, and in situations where the local environment does not substantially affect surface shading, the subjective quality of reality cues produced by environment mapping may be superior to those produced by naive ray tracing.

An approach to shading that exploits the accuracy of ray tracing and the speed of environment mapping involves ray tracing the local 3D environment of an object and using an environment map representing the remainder of the environment as a backdrop.[6]

An example of environment mapping

The lizard of Figure 1 was rendered with reflection mapping (environment mapping with mirror reflections). Examination of this image reveals an inherent limitation with reflection mapping: Since the reflecting object is not normally in the environ-

ment map, that object cannot reflect parts of itself (for example, the legs are not reflected in the body). Actually, this limitation can be partially overcome by using different environment maps for different parts of an object.

But overall, reflection mapping performs well in this scene. The horizon and sky, which are the most prominent reflected features, are accurately reflected because of their distance from the reflecting object. The reflections of the tree and the foreground terrain are less accurate because of their proximity, but surface curvature makes this difficult to recognize. As this example shows, reflections need not be accurate to look right, although attention should be paid to scene composition. Planar reflecting surfaces, for example, may cause problems, since their distortion of reflections may be quite noticeable.

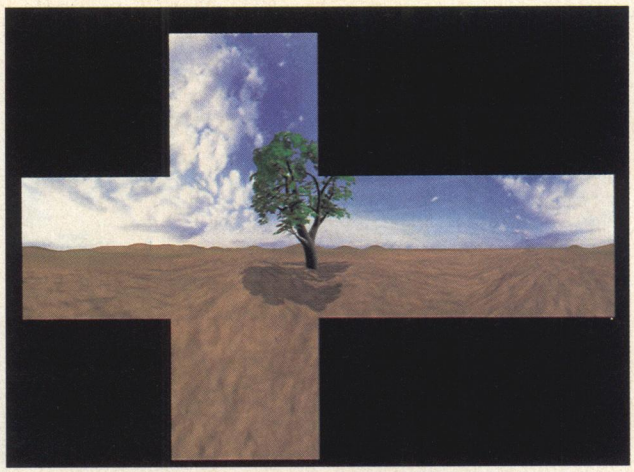


Figure 2. Unfolded cube projection of a 3D environment.

Rendering a cube projection

Environment mapping presupposes the ability to obtain a projection of the complete environment. Suppose we wish to obtain a world projection of a 3D synthetic environment as seen from a particular viewpoint. One method is to position the camera at the viewpoint and project the scene onto a cube by rendering six perspective views, each with a 90° view angle and each looking down an axis of a coordinate system with its origin at the viewpoint. Thus, a cube projection can be created with any rendering program that can produce perspective views.

The environment map used to shade the lizard in Figure 1 is a cube projection created in this manner; in Figure 2 it is unfolded.

Miller and Hoffman have also used this method of creating a cube projection as an intermediate step in making a latitude-longitude projection of the world, the format they prefer for environment mapping.[2] Blinn and Newell use a latitude-longitude projection also.[1] Hall prefers plotting longitude against sine of latitude so that pixels in the projection correspond to equal areas of "sky."[6]

Surface shading

Light reflected by a surface is assumed to have a diffuse component and a specular component. The diffuse component represents light scattered equally in all directions; the specular component represents light reflected at or near the mirror direction.

This discussion of surface shading will focus on obtaining diffuse and specular illumination from an environment map. The problem of combining this information with surface properties (color, glossiness, etc.) to determine the color reflected at a surface point is beyond the scope of this article, but the following formula is a rough rule for shading monochrome surfaces at a surface point:

$$\text{reflected color} = dc \times D + sc \times S,$$

where

dc is the coefficient of diffuse reflection
D is the diffuse illumination
sc is the coefficient of specular reflection
S is the specular illumination
(*dc* and *sc* depend mainly on surface glossiness)

No ambient term is required, since the diffuse illumination component accounts for all illumination from the environment.

Cook and Torrance[7] provide a general discussion of surface shading; Miller and Hoffman[2] discuss shading in the context of environment mapping.

Texture filtering

Environment mapping is a form of texture mapping wherein the texture applied to 3D surfaces is represented in an environment map. The diffuse and specular illumination impinging on a region of a surface can be found by texture filtering regions of the environment map with a space-variant filter.

Diffuse illumination at a surface point comes from the hemisphere of the world centered on the surface normal, and it can be found by filtering the region of the environment map corresponding to this hemisphere. Filtering should be done according to Lambert's law, which states that the illumination coming from a point on the hemisphere should be weighted by the cosine of the angle between the direction of that point and the surface normal. Since a hemisphere represents half of the environment

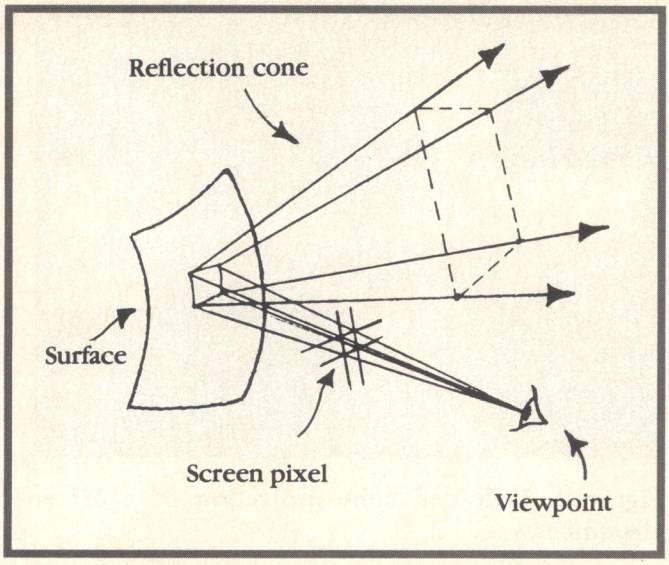


Figure 3. Rays from viewpoint through corners of a screen pixel are reflected by a surface. These four direction vectors define a "reflection cone" with a quadrilateral cross section.

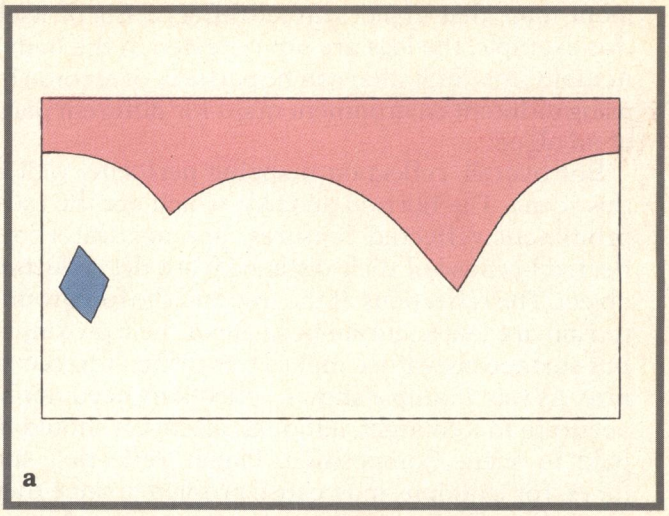


Figure 4. Regions subtended by two quadrilateral reflection cones in (a) latitude-longitude projection

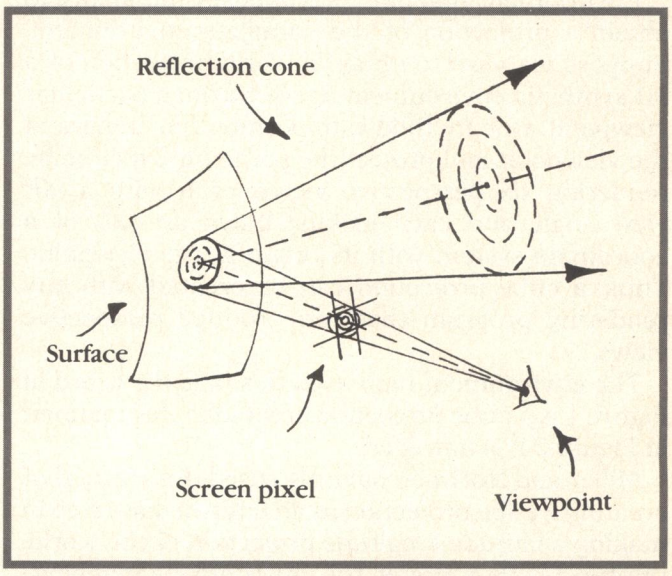


Figure 5. Reflection cone defined by viewpoint and elliptical screen pixel is approximated by an elliptical cone. Concentric ellipses within an elliptical screen pixel are approximated by concentric elliptical cones.

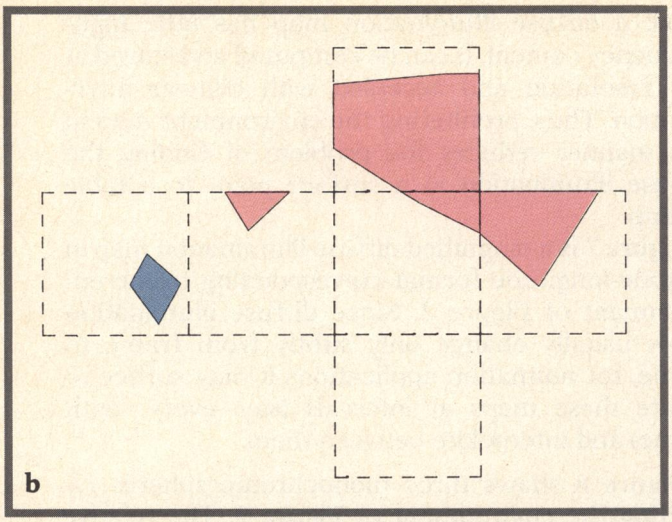
map, filtering methods that consider texture pixels individually are very slow. A method for obtaining diffuse illumination by table lookup is discussed later.

Specular illumination can also be found by filtering a region of the environment map. Figure 3 shows the quadrilateral cone of "sky" reflected by the region of a surface subtended by a screen pixel. To find the four direction vectors defining this "reflection cone," we assume that the surface is a perfect mirror, and we reflect each ray with respect to the surface normal where it intersects the surface. Actually, this simplifies the geometry somewhat, since we would not expect the beam reflected by a nonplanar surface to have a precisely quadrilateral cross section.

Given a reflection cone, an obvious approach to determining the specular illumination is to find the subtended region of the environment map and then average the pixel values within this region. (Using an unweighted average of the pixel values is equivalent to filtering the region with a box filter.)

Figures 4a and 4b show regions subtended by two quadrilateral reflection cones in environment maps represented in latitude-longitude format (Figure 4a) and cube format (Figure 4b). The red regions are subtended by a relatively wide reflection cone, and the blue regions are subtended by a much narrower reflection cone. As is apparent from Figure 3, the "width" of a reflection cone depends on surface curvature, so reflection cones vary widely in size. In practice, regions subtended by reflection cones are typically smaller than the regions illustrated in Figures 4a and 4b, since relatively flat surfaces generate narrow reflection cones. This is not necessarily the case, however; reflection cones may be very wide on average if bump mapping is performed.

Determining the specular illumination impinging on a surface by constructing quadrilateral reflection cones and then averaging subtended pixels in the environment map gives results that are reasonable but not optimal. For theoretical reasons, filtering should not be restricted to the region of texture



and (b) cube projection. In cube format, regions on cube faces are always polygonal.

Figures 6a and 6b show regions subtended by concentric elliptical cones which correspond to the quadrilateral cones represented in Figures 4a and 4b. Regions subtended by elliptical cones in cube-format environment maps are always bounded by second-degree curves: ellipses, hyperbolas, and parabolas (for example, the concentric curves in Figure 6b).

Prefiltered environment maps

Since the width of a reflection cone depends on surface curvature, the area of "sky" reflected at a pixel can be arbitrarily large. A sphere covering a single pixel, for example, reflects nearly the entire world. So determining specular illumination normally requires filtering large areas of the environ-

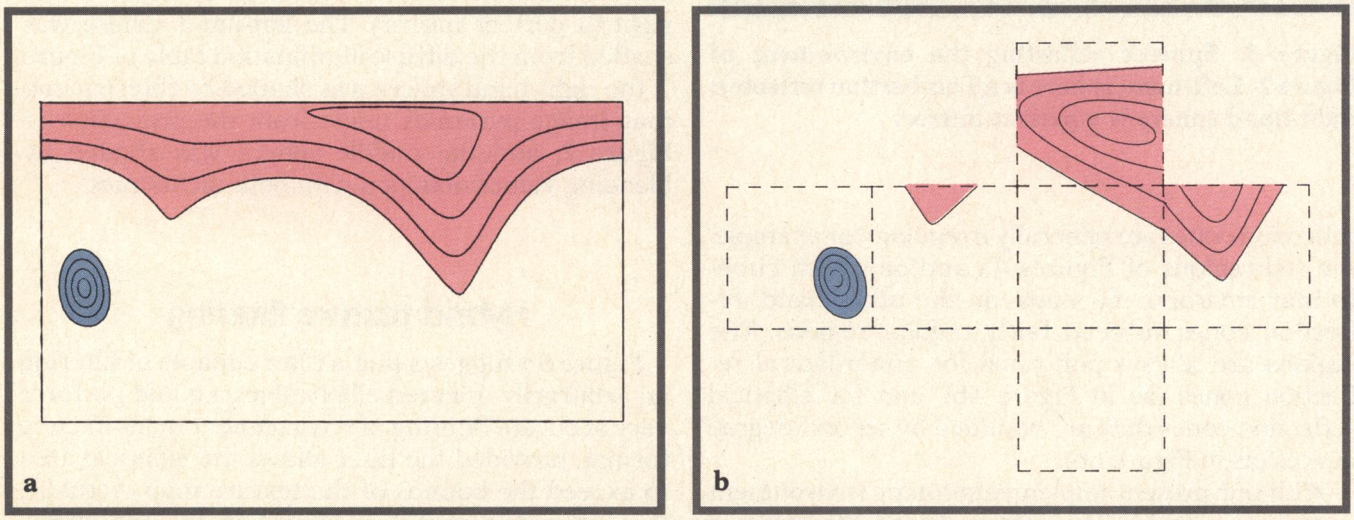


Figure 6. Regions subtended by two sets of concentric elliptical reflection cones in (a) latitude-longitude projection and (b) cube projection. In cube format, intersection lines are always second-degree curves: ellipses, hyperbolas, or parabolas.

corresponding to screen pixel bounds, and averaging of pixel values should be weighted by proximity to the center of the region being filtered.[8]

More accurate filtering is achieved by treating pixels as ellipses that are somewhat larger than the rectangular pixels defined by the raster grid and weighting pixel values according to displacement from pixel center (for example, weighting according to a Gaussian hump with an elliptical base[8]). As shown in Figure 5, elliptical pixels produce approximately elliptical reflection cones, and concentric elliptical cones corresponding to concentric ellipses within an elliptical screen pixel subtend regions of equivalent proximity to pixel center.

ment map for some pixels, and filtering methods that evaluate texture pixels individually are very slow.

Approximate filtering techniques that use prefiltered texture are much faster. Several methods have been developed that have low "constant cost" per output pixel; that is, filtering time is independent of the size of the region being filtered. However, these techniques are presently limited to filtering rectangular or elliptical areas oriented parallel to the coordinate axes of the environment map.[9]

Constant-cost filtering is problematical when environment maps are represented in latitude-longitude format, because regions subtended by wide

Figure 7. Magnified diffuse illumination map of the environment of Figure 2 in latitude-longitude format. Table resolution is 40 × 20.

Figure 8. Spheres reflecting the environment of Figure 2. Left-hand sphere is a Lambertian reflector; right-hand sphere is a perfect mirror.

reflection cones are generally irregular (for example, the red regions of Figures 4a and 6a). With cube-format environment maps, on the other hand, reflection cones subtend fairly regular regions. The regions are always polygonal for quadrilateral reflection cones (as in Figure 4b), and for elliptical reflection cones they are bounded by second-degree curves (as in Figure 6b).

With our present implementation of environment mapping, specular illumination is obtained from a prefiltered cube projection in the form of six "mip map" image pyramids,[10] one for each cube face. Texture filtering with mip maps offers fast, constant-cost performance, but it is limited to filtering square regions of texture with a box filter, so results are only approximate. In the case of Figure 4b, for example, each of the polygonal areas on the cube faces would need to be approximated by a square. Crow[11] and Perlin[12] have proposed similar and somewhat more general approximate filtering techniques with constant-cost performance.[13]

We turn now to applying prefiltered texture to find diffuse illumination. Miller and Hoffman have shown how a diffuse illumination map can be created by convolving an environment map with a Lambert's law cosine function, a kernel covering one hemisphere of the world.[2] This map, which is indexed by surface normal, can be thought of as indicating colors reflected by a monochrome spherical Lambertian reflector placed at the world center. Since a diffuse illumination map has little high-frequency content, it can be computed and stored at low resolution and accessed with bilinear interpolation. Thus, prefiltering the environment map in this manner reduces the problem of finding the diffuse illumination at a surface point to a table lookup.

Figure 7 is a magnified diffuse illumination map in latitude-longitude format corresponding to the environment of Figure 2. Since diffuse illumination maps usually change only subtly from frame to frame, for animation applications it may suffice to create these maps at intervals (say, every tenth frame) and interpolate between them.

Figure 8 shows three monochrome spheres reflecting the environment of Figure 2. The relative weighting of the diffuse and specular reflection coefficients (*dc* and *sc* in the shading formula given earlier) varies from completely diffuse on the left (a Lambertian reflector) to completely specular on the right (a perfect mirror). The left-hand sphere was shaded from the diffuse illumination table of Figure 7; the right-hand sphere was shaded by filtering mip map image pyramids made from the projection of Figure 2; and the middle sphere was shaded by blending values obtained with both techniques.

Hybrid texture filtering

Figure 6b suggests that a filter capable of filtering an arbitrarily oriented elliptical area could perform very accurate filtering of environment maps in cube format, provided the filter allows the elliptical area to exceed the bounds of the texture map. Actually, the concentric curves in Figure 6b include hyperbolas in addition to ellipses, but these patterns can be closely approximated by concentric ellipses.

According to Heckbert's survey of texture mapping techniques (included in this issue of *IEEE CG&A*),[9] no existing constant-cost filters are capable of filtering arbitrarily oriented elliptical areas, but a method having nearly constant-cost performance has been described.[8,9] Filtering any elliptical area can be closely approximated by filtering a small elliptical area at the appropriate level in a prefiltered image pyramid. Actually, filtering at two adjacent levels in the image pyramid and interpolating the results as described by Williams[10] is preferable. Within the image pyramid, high-quality filtering is performed on arbitrarily oriented elliptical areas, and texture samples are weighted by proximity to ellipse center. The *elliptical weighted average filter*[8] provides such capabilities. Although this hybrid filtering method is more costly than constant-cost methods, it offers quality approaching that of direct convolution.

In sum, the combination of environment maps in cube format, elliptical reflection cones, and the above hybrid filtering method is proposed as an efficient and highly accurate method of obtaining specular illumination for environment mapping.

Cube projections vs. latitude-longitude projections

The fast filtering methods described above produce more accurate results when environment maps are represented in cube format rather than latitude-longitude format, because the regions to be filtered are more regular. Cube projections have other advantages as well.

As previously described, a cube projection of a 3D environment can be rendered directly as six perspective views. Producing the corresponding latitude-longitude projection usually involves rendering a cube projection and then converting to a latitude-longitude projection by filtering the cube faces. (This is not always required; for example, ray tracers can produce a latitude-longitude projection directly.) This conversion requires additional computation, and the added generation of filtering can only degrade image clarity.

Moreover, latitude-longitude projections are nonlinear in the sense that plotting 3D direction vectors requires trigonometric operations. Projecting direction vectors onto a unit cube is much faster because it requires only division. This is a significant computational advantage, given the frequency with which this operation must be performed.

Cheap chrome effects

Environment mapping also has wide application where the objective is to produce a striking visual effect without particular regard for realism. Often the intent is to give objects a chrome-plated look, and the content of reflections is unimportant. In Robert Abel and Associates' "Sexy Robot" animation, for example, the "reflection map" was a smooth color gradation from earth colors at low elevations to sky colors at high elevations, and at a given elevation color was constant.[14] (Of course, this is not a true reflection map.)

For applications of this sort, a color map or other one-dimensional color table suffices to specify the color gradation, and rendering computation can be greatly reduced by filtering this table instead of performing texture filtering. Since an environment map is not used, memory requirements are reduced accordingly, and no setup time is required to make and prefilter a world projection. Highlights can be produced by adding point light sources independently of environment mapping.

Modeling background objects with world projections

While the discussion thus far has been confined to using world projections for surface shading, they can also be used to model background objects.[15] The sky component in frames of moving-camera animation can be modeled as a half-world projection, for example, the projection onto the upper half of a cube. As the camera moves through the scene, the appropriate region of the projection comes into view.

Of course, the advantage of this technique is speed: A scene element (in this case the sky) can be rendered from the world projection with texture mapping which, for complex scenes, is much faster than rendering the corresponding 3D models. This method assumes that objects in the projection are a great distance from the camera, and results are only approximate when this is not the case.

Figure 9 is a frame from animation in which the sky component was rendered from a half-world projection made from painted panels.

Since half-world models cover the whole sky, they are particularly useful for creating world projections for environment mapping. The sky component of Figure 2 was modeled by projecting a 180° fish-eye photograph of sky onto a half cube, shown in isolation in Figure 10. This model served two purposes in producing the picture of Figure 1: modeling the sky in the background, and making the environment map used to shade the lizard.

Making sky models from photographs is an attractive option because of the difficulty of synthesizing complex sky scenes with realistic cloud forms and lighting. A 180° fish-eye lens (round type) allows the whole sky to be photographed at once, thus avoiding problems associated with making photo mosaics.

In animated scenes where the camera rotates but doesn't change location, this approach to modeling sky can be extended to modeling the whole background. In this case a single world projection centered at the camera is generated, and the background in the frames of animation is rendered directly from this projection. The geometry of the scene is faithfully reproduced regardless of how far objects in the scene are from the camera; results are not just approximate.

This technique can also be applied to moving-camera animation in some situations. A world projection of the distant environment centered at a typical camera position can be used to render distant objects in the scene while the near environment is

Figure 9. Frame from animation using a half-world sky model made from painted panels. (Animation: "Three Worlds" by the author, 1983.)

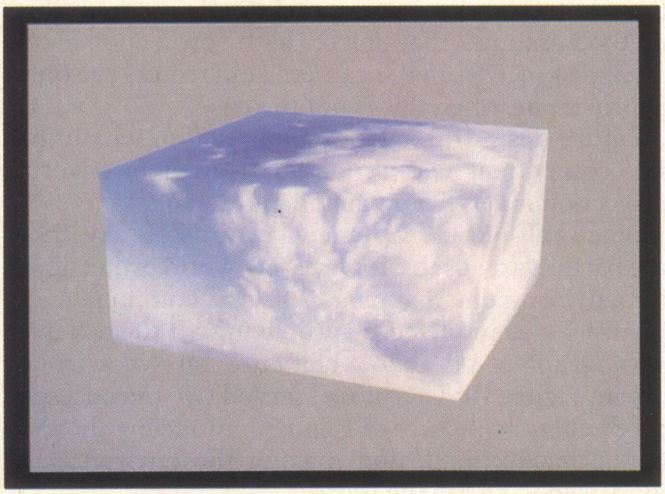


Figure 10. Half-world sky model made from a 180° fish-eye photograph.

Figure 11. Omnimax projection of the environment of Figure 2.

rendered from 3D models at each frame and then composited with the distant environment. This approach is analogous to using foreground and background levels in cel animation. As before, it is convenient to represent the world as a cube projection, since it can be rendered directly from a 3D model of the scene.

Assuming that the world is represented as a cube projection, the cube faces should have substantially higher resolution than the output frames, since only a small part of the world is subtended by the viewing pyramid for typical view angles. For example, a viewing pyramid with a 3:4 aspect ratio and a 45° vertical view angle subtends only about 6 percent of the world.

Rendering background objects from a world projection is a form of texture mapping, since each pixel in the output image is rendered by determining the corresponding region in the world projection and then filtering a neighborhood of this region. Since each output pixel corresponds to relatively few pixels in the world projection, high-quality filtering methods that consider texture pixels individually are appropriate (as opposed to approximate methods using prefiltered texture).

This approach to rendering background objects suggests a method for performing motion blur. The region of texture traversed by the projection of an output pixel in the time interval between frames is determined, and then this region is filtered. The simplicity of performing accurate motion blur in this situation is due to the two-dimensional nature of the model.

For this application, a space-variant filter should be employed because different output pixels may traverse differently shaped paths. (This occurs, for example, if the camera rolls.) Approximate results can be obtained by filtering elliptical regions in the world projection with a filter capable of filtering arbitrarily oriented elliptical areas such as the elliptical weighted average filter.[8]

Nonlinear projections

In addition to their use in generating perspective views of an environment, world projections can be used to create nonlinear projections such as the fish-eye projection required for Omnimax frames.

An Omnimax theater screen is hemispherical, and film frames are projected through a fish-eye lens.[16] Omnimax projections of 3D scenes have been obtained by projecting the scene onto a cube centered at the camera and then filtering regions of the cube faces to obtain pixels in the output image.[8] This technique is similar to the method described above for rendering background objects from world projections. Figure 11 is an Omnimax projection made from the cube projection of Figure 2.

Conclusion

The projection of the environment onto a cube is a convenient and useful representation of the world for the applications cited in this article. In the case of environment mapping, cube projections allow relatively accurate texture filtering to be performed with fast, constant-cost (or nearly constant-cost) methods.

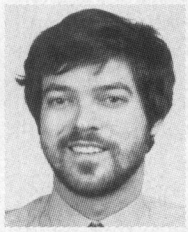
Generally, the methods described are ways of approximating a three-dimensional problem with a two-dimensional problem to reduce computational expense. Environment mapping approximates ray tracing, and rendering background objects from world projections with texture mapping approximates image rendering from 3D models. The subjective quality of reality cues in images produced with these approximate methods often compares favorably with results obtained by more expensive image generation techniques. For complex environments, approximate techniques may be the only practical way of producing animation having the desired features with moderate computing resources. ■

Acknowledgments

My thanks to Paul Heckbert, who collaborated to incorporate environment mapping in his polygon renderer, devised the "elliptical weighted average filter," and provided a critical reading of this article. Thanks also to Nelson Max and John Lewis for their comments, and to Dick Lundin, who assembled and rendered the scene of Figure 1. Ariel Shaw and Rich Carter provided photographic assistance.

References

1. James F. Blinn and Martin E. Newell, "Texture and Reflection in Computer Generated Images," *Comm. ACM,* Vol. 19, No. 10, Oct. 1976, pp.542-547.
2. Gene S. Miller and C. Robert Hoffman, "Illumination and Reflection Maps: Simulated Objects in Simulated and Real Environments," *SIGGRAPH 84: Advanced Computer Graphics Animation Seminar Notes,* July 1984.
3. Turner Whitted, "An Improved Illumination Model for Shaded Display," *Comm. ACM,* Vol. 23, No. 6, June 1980, pp.343-349.
4. John Amanatides, "Ray Tracing with Cones," *Computer Graphics* (Proc. SIGGRAPH 84), Vol. 18, No. 3, July 1984, pp.129-135.
5. Robert L. Cook, Thomas Porter, and Loren Carpenter, "Distributed Ray Tracing," *Computer Graphics* (Proc. SIGGRAPH 84), Vol. 18, No. 3, July 1984, pp.137-145.
6. Roy Hall, "Hybrid Techniques for Rapid Image Synthesis," course notes, *SIGGRAPH 86: Image Rendering Tricks,* Aug. 1986.
7. Robert L. Cook and Kenneth E. Torrance, "A Reflectance Model for Computer Graphics," *Computer Graphics* (Proc. SIGGRAPH 81), Vol. 15, No. 3, Aug. 1981, pp.307-316.
8. Ned Greene and Paul S. Heckbert, "Creating Raster Omnimax Images from Multiple Perspective Views Using the Elliptical Weighted Average Filter," *IEEE CG&A,* Vol. 6, No. 6, June 1986, pp.21-27.
9. Paul S. Heckbert, "Survey of Texture Mapping," *IEEE CG&A,* Vol. 6, No. 11, Nov. 1986, pp.56-67. An earlier version appeared in *Proc. Graphics Interface 86,* May 1986, pp.207-212.
10. Lance Williams, "Pyramidal Parametrics," *Computer Graphics* (Proc. SIGGRAPH 83), Vol. 17, No. 3, July 1983, pp.1-11.
11. Franklin C. Crow, "Summed-Area Tables for Texture Mapping," *Computer Graphics* (Proc. SIGGRAPH 84), Vol. 18, No. 3, July 1984, pp.207-212.
12. Kenneth Perlin, "Course Notes," *SIGGRAPH 85: State of the Art in Image Synthesis Seminar Notes,* July 1985, pp.297-300.
13. Paul S. Heckbert, "Filtering by Repeated Integration," *Computer Graphics* (Proc. SIGGRAPH 86), Vol. 20, No. 4, Aug. 1986.
14. Torrey Byles, "Displays on Display: Commercial Demonstrates State-of-the-Art Computer Animation," *IEEE CG&A,* Vol. 5, No. 4, Apr. 1985, pp.9-12.
15. Ned Greene, "A Method of Modeling Sky for Computer Animation," *Proc. First Int'l Conf. Engineering and Computer Graphics,* Beijing, Aug. 1984, pp.297-300.
16. Nelson Max, "Computer Graphics Distortion for Imax and Omnimax Projection," *Proc. Nicograph 83,* Dec. 1983, pp. 137-159.



Ned Greene has been a senior member of the research staff at the New York Institute of Technology Computer Graphics Lab since 1980. From 1978 to 1980 he was a programmer in computer-aided design for McDonnell Douglas Automation.

His research interests include graphics software for image synthesis and animation, and computer animation as an artistic medium. His computer images have been widely published, and excerpts from his animation have been televised in the US, Europe, and Japan. Greene holds a BA in mathematics from the University of Kansas. He is a member of ACM SIGGRAPH.

The author can be contacted at the NYIT Computer Graphics Lab, PO Box 170, Old Westbury, NY 11568.