Pyramidal Parametrics

Lance Williams

Computer Graphics Laboratory
New York Institute of Technology
Old Westbury, New York

Abstract

The mapping of images onto surfaces may substantially increase the realism and information content of computer-generated imagery. The projection of a flat source image onto a curved surface may involve sampling difficulties, however, which are compounded as the view of the surface changes. As the projected scale of the surface increases, interpolation between the original samples of the source image is necessary; as the scale is reduced, approximation of multiple samples in the source is required. Thus a constantly changing sampling window of view-dependent shape must traverse the source image.

To reduce the computation implied by these requirements, a set of prefiltered source images may be created. This approach can be applied to particular advantage in animation, where a large number of frames using the same source image must be generated. This paper advances a "pyramidal parametric" prefiltering and sampling geometry which minimizes aliasing effects and assures continuity within and between target images.

Although the mapping of texture onto surfaces is an excellent example of the process and provided the original motivation for its development, pyramidal parametric data structures admit of wider application. The aliasing of not only surface texture, but also highlights and even the surface representations themselves, may be minimized by pyramidal parametric means.

General Terms: Algorithms.

Keywords and Phrases: Antialiasing, Illumination Models, Modeling, Pyramidal Data Structures, Reflectance Mapping, Texture Mapping, Visible Surface Algorithms.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation--display algorithms; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling--curve, surface, solid and object representations, geometric algorithms, languages and systems; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism--color, shading, shadowing, and texture.

1. Pyramidal Data Structures

Pyramidal data structures may be based on various subdivisions: binary trees, quad trees, oct trees, or n-dimensional hierarchies [17]. The common feature of these structures is a succession of levels which vary the resolution at which the data is represented.

The decomposition of an image by two-dimensional binary subdivision was a pioneering strategy in computer graphics for visible surface determination [15]. The approach was essentially a synthesis-by-analysis: the image plane was subdivided into quadrants recursively until analysis of a subsection showed that surface ordering was sufficiently simple to permit rendering. Such subdivision and analysis has been subsequently adopted to generate spatial data structures [5], which have been used to represent images [9] both for pattern recognition [13] and for transmission [10], [14]. In the field of computer graphics, such data structures have been adopted for texture mapping [4], [16], and generalized to represent objects in space [11].

The application of pyramidal data to image storage and transmission may permit significant compression of the data to be stored or transmitted. This is so because highly detailed features may be localized within an otherwise low-frequency image, permitting the sampling rate to be reduced for large sections of the image. Besides permitting bandwidth compression, the representation orders data in such a way that the general character of images may be recalled or transmitted before the specific details.

Pattern recognition and classification often require the comparison of a candidate image against a set of canonical patterns. This is an operation the expense of which increases as the square of the resolution at which it is performed. The use of pyramidal data structures in pattern recognition and classification permits the comparison of the gross features of two-dimensional functions preliminary to the minute particulars; a good general reference on this application is [12].

In computer graphics, pyramidal texture maps may be used to perform arbitrary mappings of a function with minimal aliasing artifacts and reduced computation. Once again, images may be represented at different spatial bandwidths. The concern is that inappropriate resolution misrepresents the data; that is, sampling high-resolution data at larger sample intervals invites aliasing.

2. Parametric Interpolation

By a pyramidal parametric data structure, we will mean simply a pyramidal structure with both intra- and inter-level interpolation. Consider the case of an image represented as a two-dimensional array of samples. Interpolation is necessary to produce a continuous function of two parameters, U and V. If, in addition, a third parameter (call it D) moves us up and down a hierarchy of corresponding two-dimensional functions, with interpolation between (or among) the levels of the pyramid providing continuity, the structure is pyramidal parametric.

The practical distinction between such a structure and an ordinary interpolant over an n-dimensional array of samples is that the number of samples representing each level of the pyramid may be different.

3. Mip Mapping

"Mip" mapping is a particular format for two-dimensional parametric functions, which, along with its associated addressing scheme, has been used successfully to bandlimit texture mapping at New York Institute of Technology since 1979. The acronym "mip" is from the Latin phrase "multum in parvo," meaning "many things in a small place." Mip mapping supplements bilinear interpolation of pixel values in the texture map (which may be used to smoothly translate and magnify the texture) with interpolation between prefiltered versions of the map (which may be used to compress many pixels into a small place). In this latter capacity, mip offers much greater speed than texturing algorithms which perform explicit convolution over an area in the texture map for each pixel rendered [1], [6].

Mip owes its speed in compressing texture to two factors. First, a fair amount of filtering of the original texture takes place when the mip map is first created. Second, subsequent filtering is approximated by blending different levels of the mip map. This means that all filters are approximated by linearly interpolating a set of square box filters, the sides of which are powers-of-two pixels in length. Thus, mapping entails a fixed overhead, which is independent of the area filtered to compute a sample.

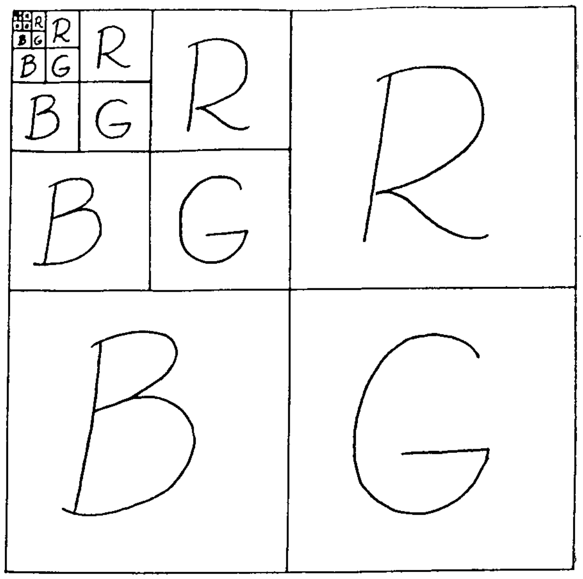


Figure (1)
Structure of a Color Mip Map
Smaller and smaller images diminish into the upper left corner of the map. Each of the images is averaged down from its larger predecessor.

(Below:)
Mip maps are indexed by three coordinates: U, V, and D. U and V are spatial coordinates of the map; D is the variable used to index, and interpolate between, the different levels of the pyramid.

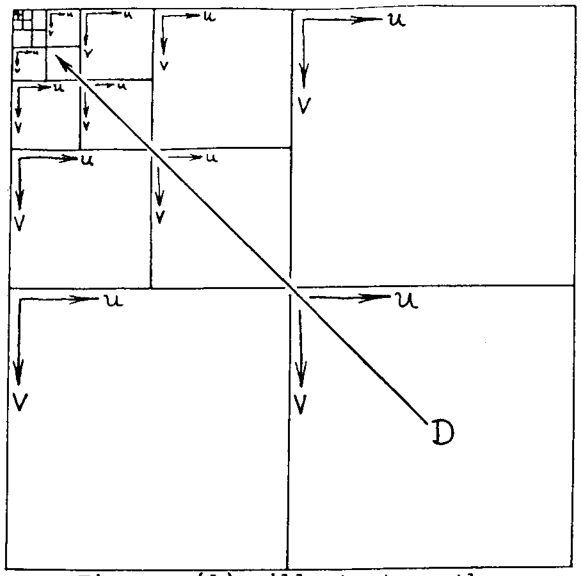


Figure (1) illustrates the memory organization of a color mip map. The image is separated into its red, green, and blue components (R, G, and B in the diagram). Successively filtered and down-sampled versions of each component are instanced above and to the left of the originals, in a series of smaller and smaller images, each half the linear dimension (and a quarter the number of samples) of its parent. Successive divisions by four partition the frame buffer equally among the three components, with a single unused pixel remaining in the upper left-hand corner.

The concept behind this memory organization is that corresponding points in different prefiltered maps can be addressed simply by a binary shift of an input U, V coordinate pair. Since the filtering and sampling are performed at scales which are powers of two, indexing the maps is possible with inexpensive binary scaling. In a hardware implementation, the addresses in all the corresponding maps (now separate memories) would be instantly and simultaneously available from the U, V input.

The routines for creating and accessing mip maps at NYIT are based on simple box (Fourier) window prefiltering, bilinear interpolation of pixels within each map instance, and linear interpolation between two maps for each value of D (the pyramid's vertical coordinate). For each of the three components of a color mip map, this requires 8 pixel reads and 7 multiplications. This choice of filters is strictly for the sake of speed. Note that the bilinear interpolation of pixel values at the extreme edges of each map instance must be performed with pixels from the opposite edge(s) of that map, for texture which is periodic. For non-periodic texture, scaling or clipping of the U, V coordinates prevents the intrusion of an inappropriate map or color component into the interpolation.

The box (Fourier) window used to create the mip maps illustrated here, and the tent (Bartlett) window used to interpolate them, are far from ideal; yet probably the most severe compromise made by mip filtering is that it is symmetrical. Each of the prefiltered levels of the map is filtered equally in X and Y. Choosing a value of D trades off aliasing against blurring, which becomes a tricky proposition as a pixel's projection in the texture map deviates from symmetry. Heckbert [8] suggests:

$$d = \max\left(\sqrt{\left(\frac{\partial u}{\partial x}\right)^2+\left(\frac{\partial v}{\partial x}\right)^2},\ \sqrt{\left(\frac{\partial u}{\partial y}\right)^2+\left(\frac{\partial v}{\partial y}\right)^2}\right)$$

where D is proportional to the "diameter" of the area in the texture to be filtered, and the partials of U and V (the texture-map coordinates) with respect to X and Y (the screen coordinates) can be calculated from the surface projection.

Illustrations of mapping performed by the mip technique are the subject of Figures (2) through (10). The NYIT Test Frog in Figure (2) is magnified by simple point sampling in (3), and by interpolation in (4). The hapless amphibian is similarly compressed by point sampling in (5) and by mipping in (6).

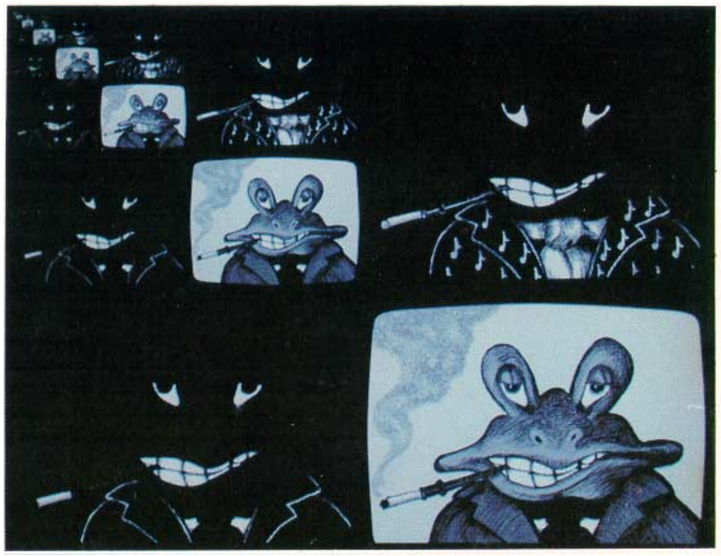


Figure (2)
Mip map of the flexible NYIT Test Frog.

The more general and interesting case -- continuously variable upsampling and downsampling of the original texture -- is illustrated in (7) on a variety of surfaces. Since the symmetry of mip filtering would be expected to show up badly when texture is compressed in only one dimension, figures (8) through (10) are of especial interest. These pictures, created by Ed Emshwiller at NYIT for his videotape, "Sunstone," were mapped using Alvy Ray Smith's TEXAS animation program, which in turn used MIP to antialias texture. As the panels rotate edge-on, the texture collapses to a line smoothly and without apparent artifacts.

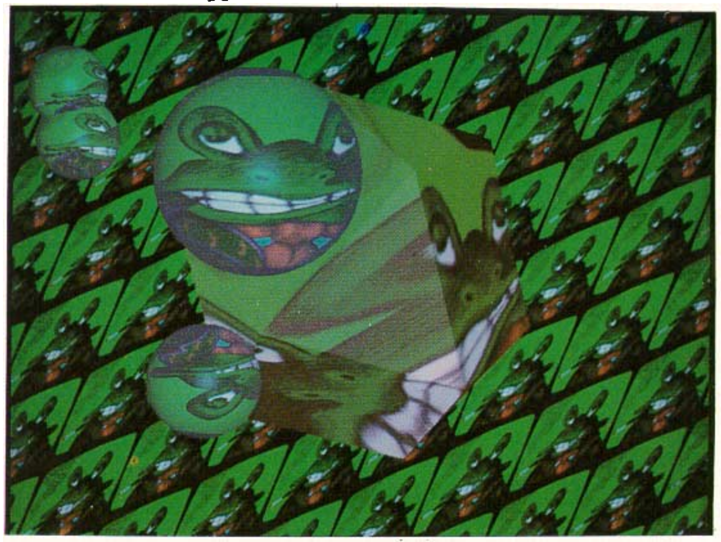


Figure (7)
General mapping: interpolation and pyramidal compression.

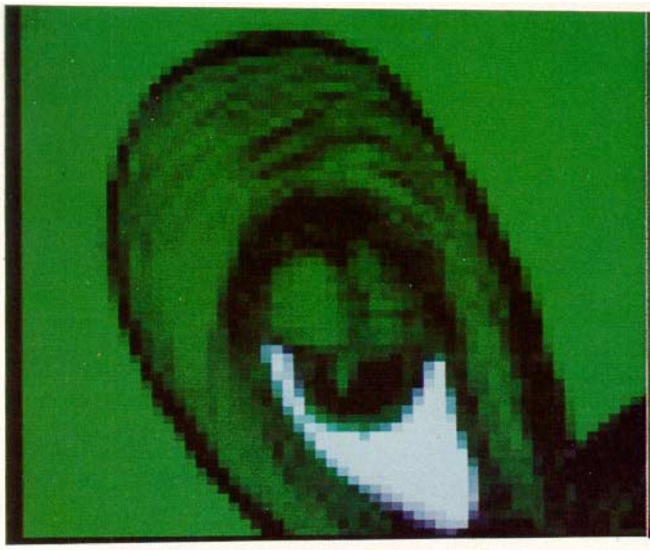


Figure (3)
Upsampling the frog: magnification by point sampling.

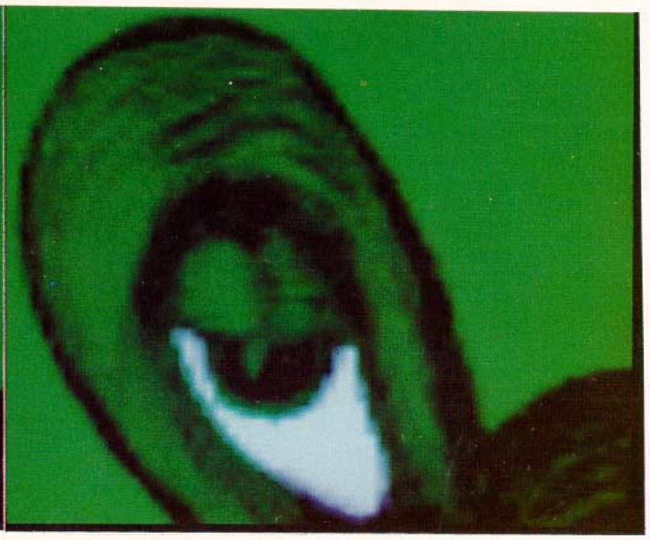


Figure (4)
Upsampling the frog: magnification by bilinear interpolation.

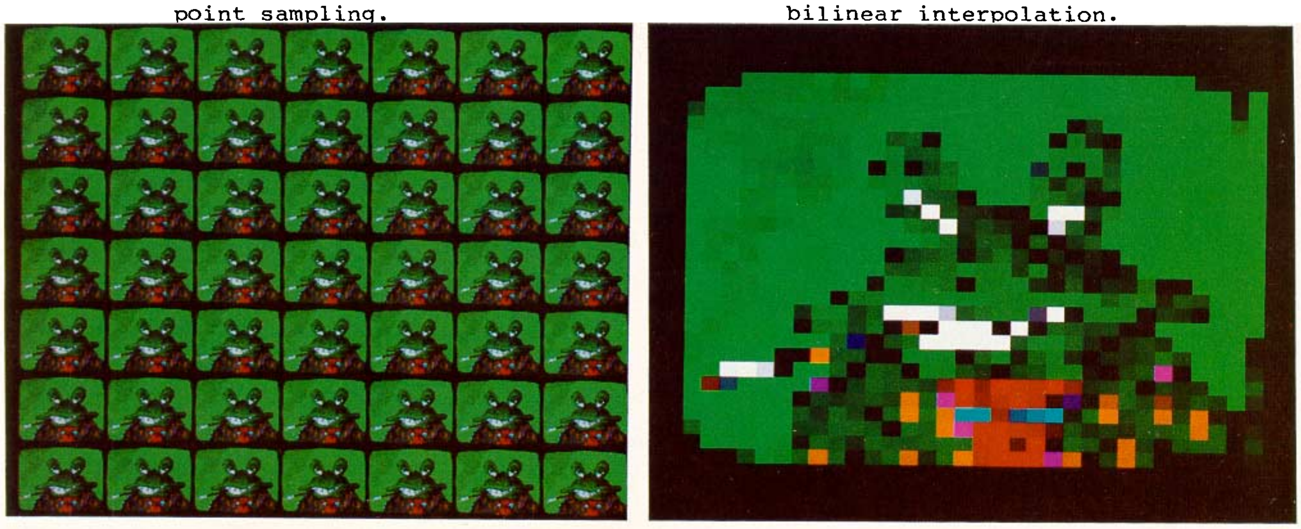


Figure (5)
Downsampling the frog: compression by point sampling (detail, right).

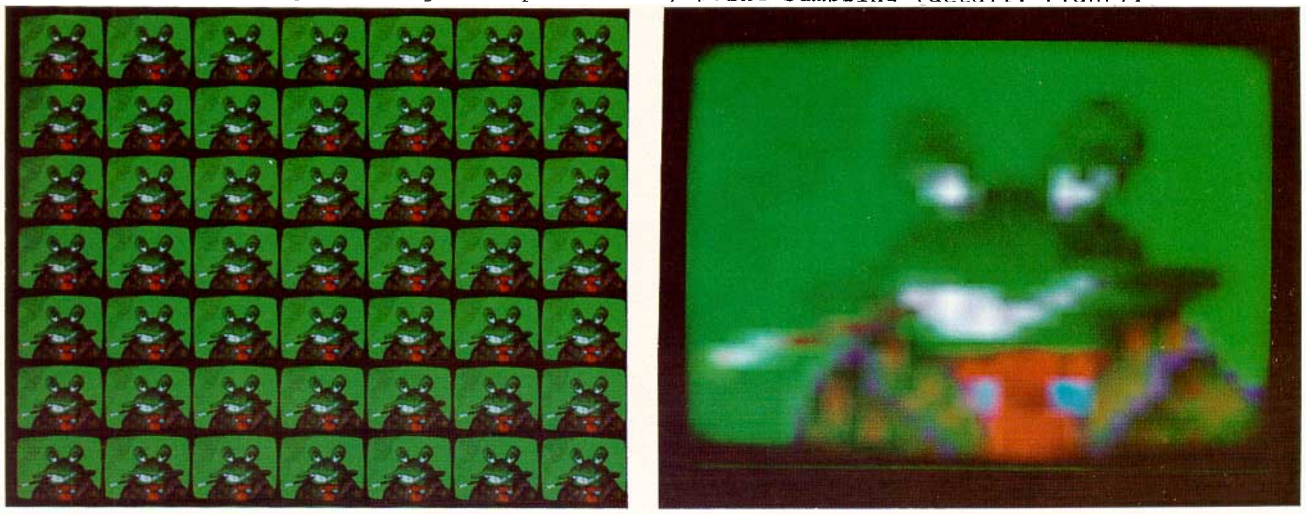
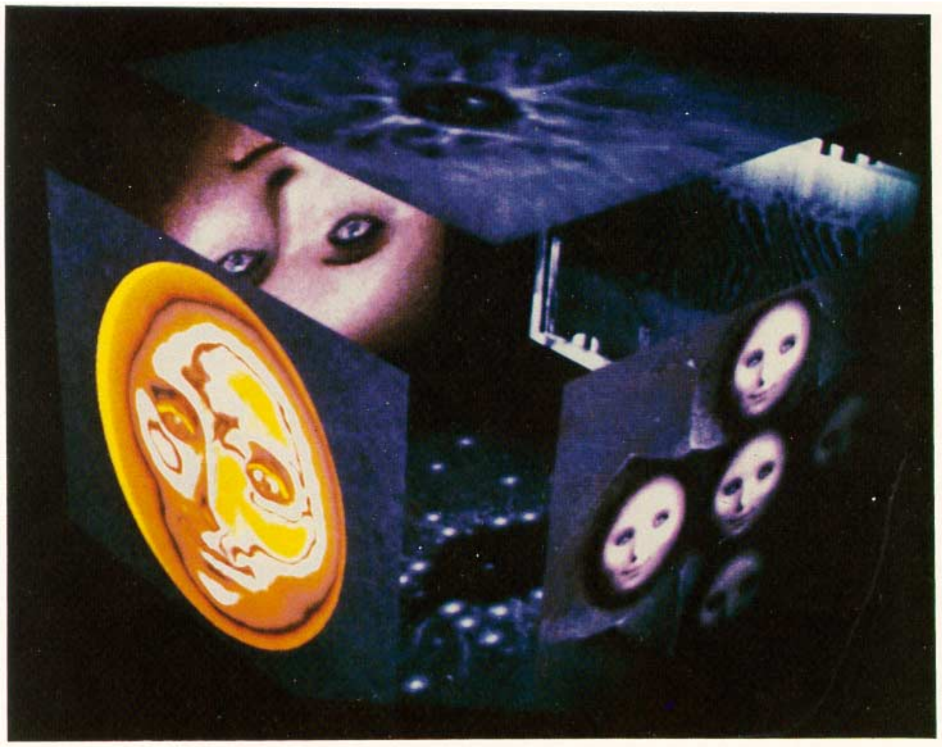
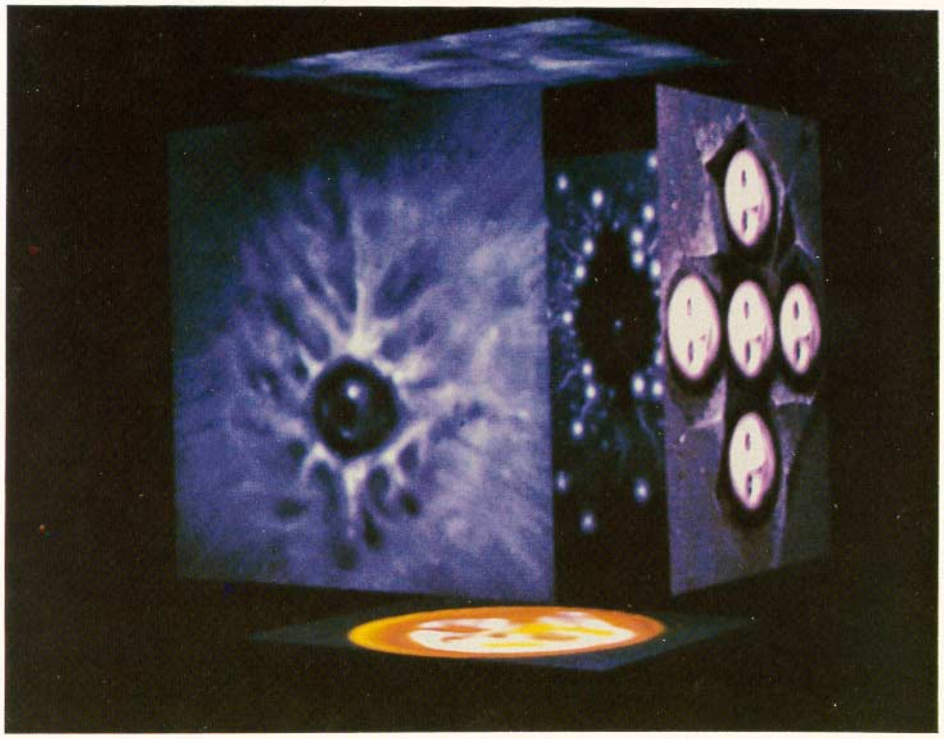
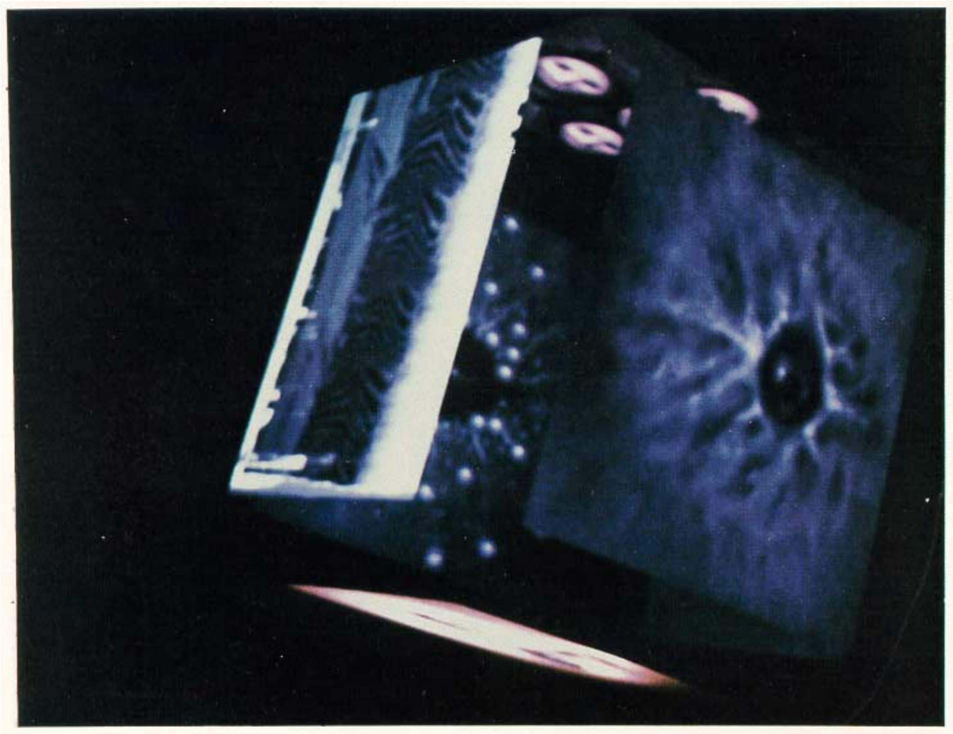


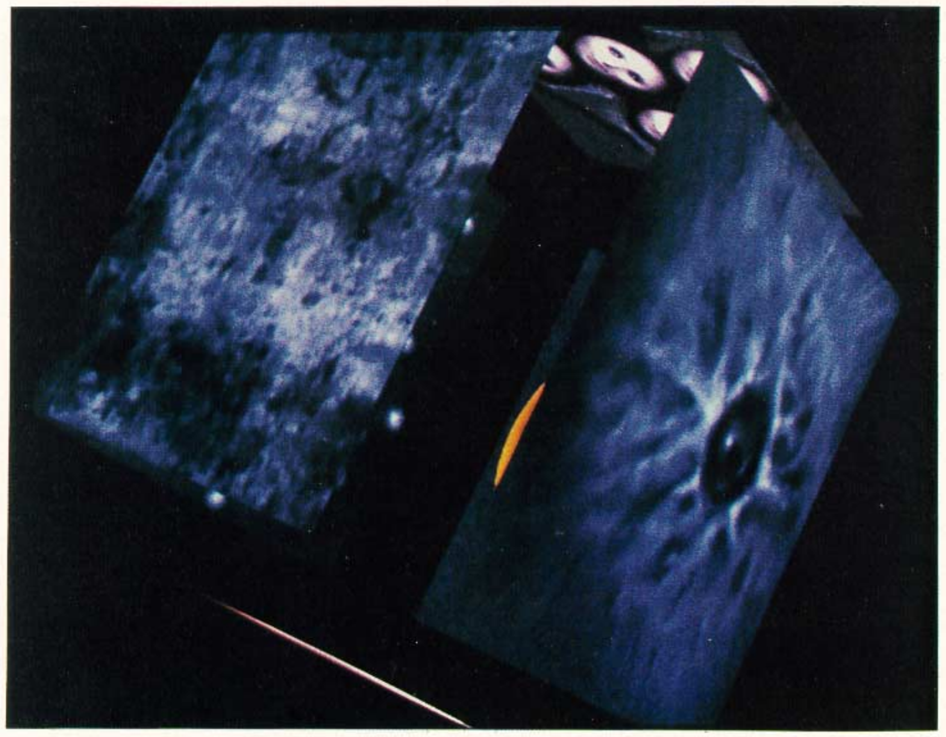
Figure (6)
Downsampling: compression by pyramidal interpolation (detail, right).





Figures (8)-(9)
"Sunstone" by Ed Emshwiller, segment animated by Alvy Ray Smith
Pyramidal parametric texture mapping on polygons.





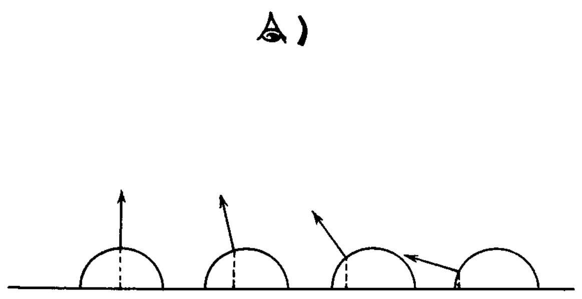
Figures (10)-(11)
"Sunstone" by Ed Emshwiller, segment animated by Alvy Ray Smith
Pyramidal parametric texture mapping on polygons.

4. Highlight Antialiasing

As small or highly curved objects move across a raster, their surface normals may beat erratically with the sampling grid. This causes the shading values to flash annoyingly in motion sequences, a symptom of illumination aliasing. The surface normals essentially point-sample the illumination function.

Figure (12) illustrates samples of the surface normals of a set of parallel cylinders. The cylinders in the diagram are depicted as if from the edge of the image plane; the regularly-spaced vertical line segments are the samples along a single axis. The arrows at the sample points indicate the directions of the surface normals. Depending on the shading formula invoked, there may be very high contrast between samples where the normal is nearly parallel to the sample axis, and samples where the normal points directly at the observer's eye.

Figure (12)



The shading function depends not only on the shape of the surface, but its light reflection properties (characterized by the shading formula), the position of the light source, and the position of the observer's eye. Hanrahan [7] expresses it in honest Greek:

$$\int_x \int_y \varphi(E,N,L) \frac{\partial(u,v)}{\partial(x,y)} \, dxdy$$

where the normal, N, the light sources, L, and the eye, E, are vectors which may each be functions of U and V, and the limits of integration are the X, Y boundaries of the pixel.

Figure (13) illustrates highlight aliasing on a perfectly flat surface. The viewing conventions of the diagram are the same as in Figure (12). "L" is the direction vector of the light source; the surface is a polygon at an angle to the image plane; the dotted bump is a graph of the reflected light, characteristic of a

Figure (13)

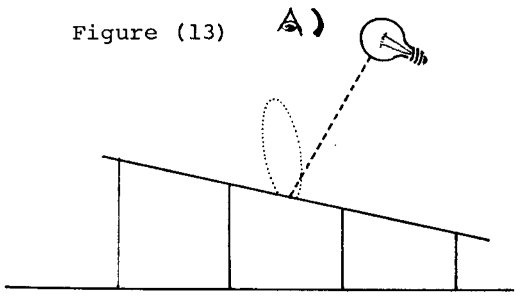
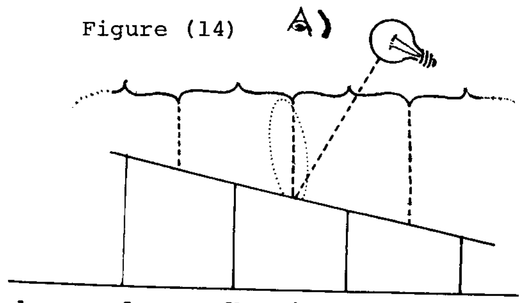


Figure (14)



specular surface reflection function. The highlight indicated by the bump falls entirely between the samples. (Note that this is only possible on a flat surface if either the eye or the light is local, a point in space rather than simply a direction vector. Some boring shading formulae exclude the possibility of highlight aliasing on polygons by requiring all flat surfaces to be flat in shading.)

A first attempt to overcome the limitations of point-sampling the illumination function is to integrate the function over the projected area represented by each sample point. This approach is illustrated in Figure (14). The brackets at each sample represent the area of the surface over which the illumination function is integrated. This procedure is analogous to area-averaging of sampled edges or texture [3].

In order to generalize this approach to curved surfaces, the "sample interval" over which illumination is integrated must be modified according to the local curvature of the surface at a sample. In Figure (15), the area of a surface represented by a pixel has been projected onto a curved surface. The solid angle over which illumination must be integrated is approximated by the volume enclosed by the normals at the pixel corners. The distribution of light within this volume will sum to an estimate of the diffuse reflection over the pixel. If the surface exhibits undulations at the pixel level, however, aliasing will result.

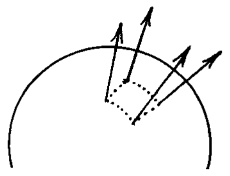


Figure (15)

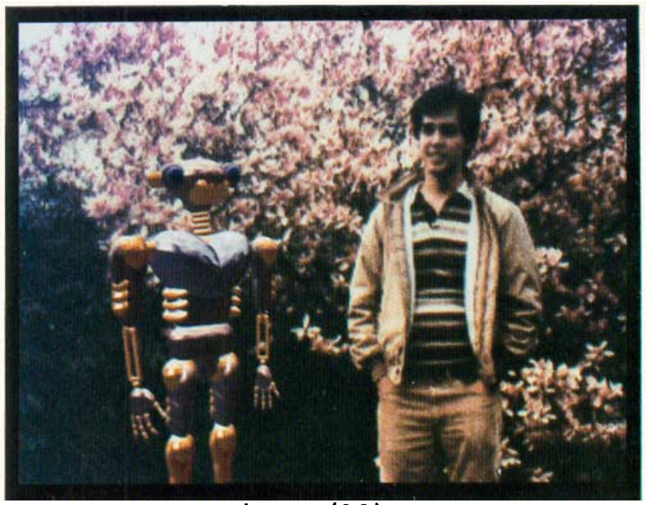


Figure (16)
Michael Chou (right) poses with an imaginary companion. Reflectance maps can enhance the realism of synthetic shading.

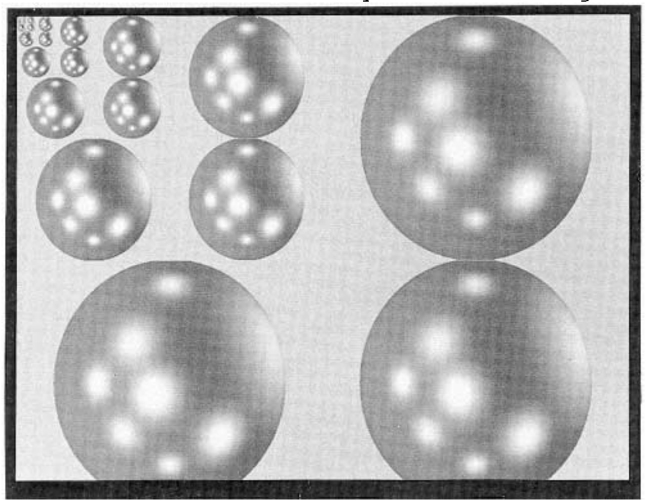


Figure (17)
A pyramidal parametric reflectance map, containing 9 light sources. The region outside the "sphere" is unused.

We might divide the surface up into regions of relatively low curvature (as is done in some patch rendering algorithms), and rely on "edge antialiasing" to integrate the different surfaces within a pixel. Alternatively, we may develop some mechanism for limiting the local curvature of surfaces before rendering. This possibility is explored in the next section.

If we represent the illumination of a scene as a two-dimensional map, highlights can be effectively antialiased in much the same way as textures. Blinn and Newell [1] demonstrated specular reflection using an illumination map. The map was an image of the environment (a spherical projection of the scene, indexed by the X and Y components of the surface normals) which could be used to cast reflections onto specular surfaces. The impression of mirrored facets and chrome objects which can be achieved with this method is striking; Figure (16) provides an illustration. Reflectance mapping is not, however, accurate for local reflections. To achieve similar results with three dimensional accuracy requires ray-tracing.

A pyramidal parametric illumination map permits convenient antialiasing of highlights as long as a good measure of local surface curvature is available. The value of "D" used to index the map is proportional to the solid angle subtended by the surface over the pixel being computed; this may be estimated by the same formula used to compute D for ordinary texture mapping. Nine light sources of varying brightness glint raggedly from the test object in Figure (18); the reflectance map in Figure (17) provided the illumination. In Figure (19), convincing highlight antialiasing results from the full pyramidal parametric treatment.

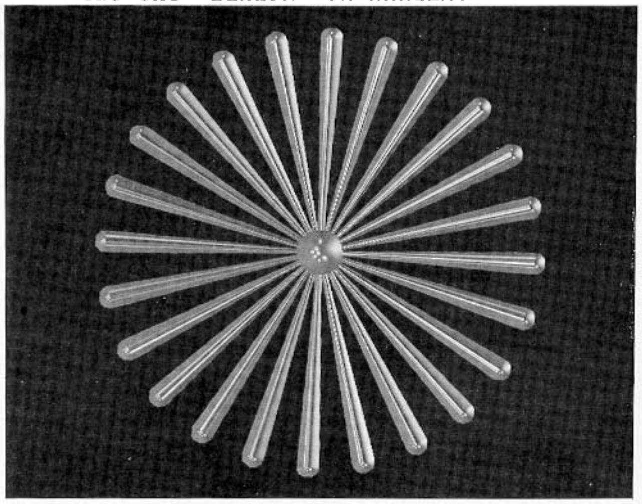


Figure (18) Before

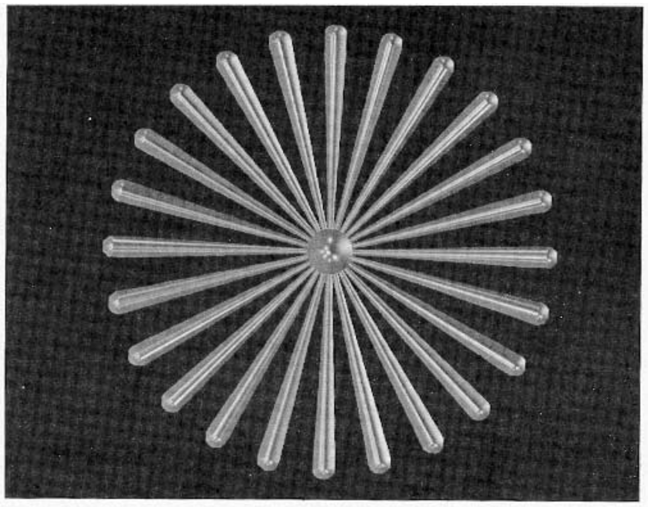
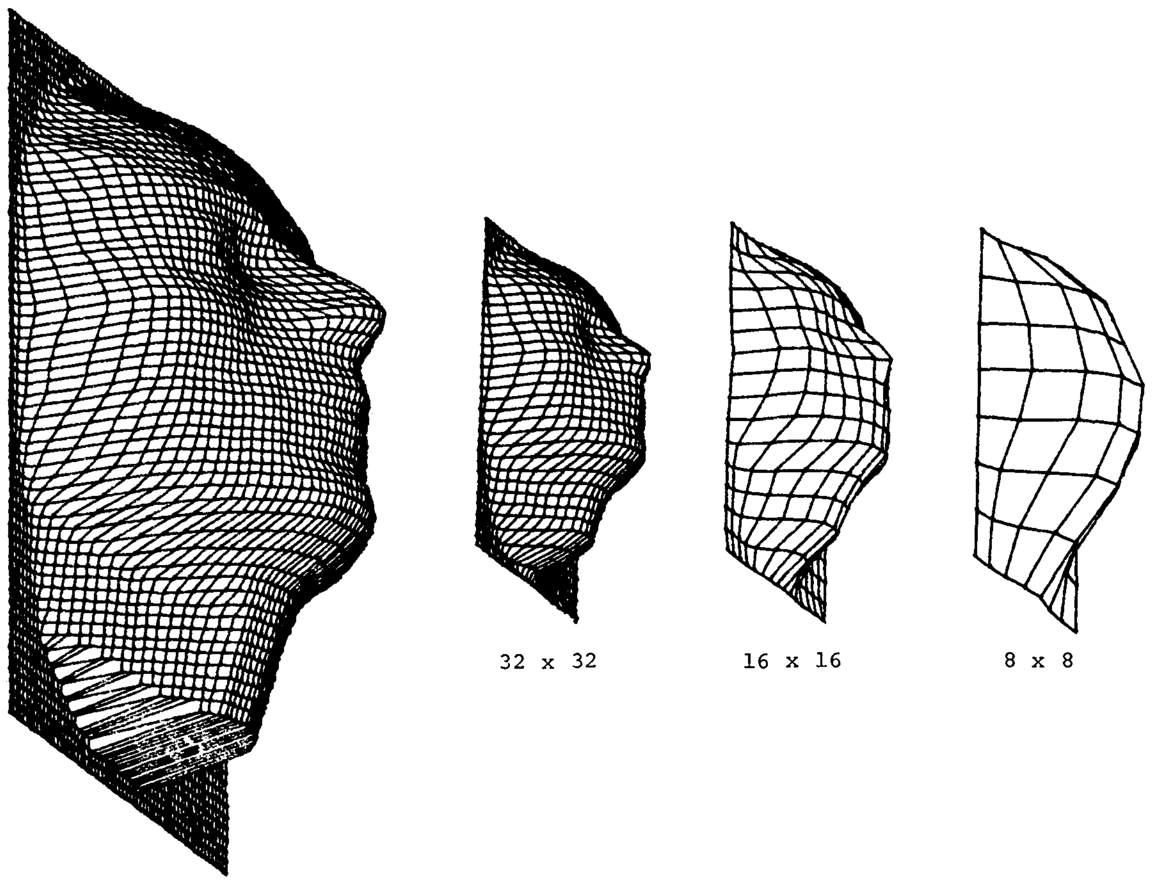


Figure (19) After



Figures (20-23) Different resolution meshes.

5. Levels of Detail in Surface Representation

In addition to bandlimiting texture and illumination functions for mapping onto a surface, pyramidal parametrics may be used to limit the level of detail with which the surface itself is represented. The goal is to represent an object for graphic display as economically as its projection on the image plane permits, without boiling and sparkling aliasing artifacts as the projection changes.

The expense of computing and shading each pixel dominates the cost of many algorithms for rendering higher-order surfaces. For meshes of polygons or patch control points which project onto a small portion of the image, however, the vertex (or control-point) expense dominates. In these situations it is desirable to reduce the number of points used to represent the object.

A pyramidal parametric data structure the components of which are spatial coordinates (the X-Y-Z of the vertices of a rectangular mesh, for example, as opposed to the R-G-B of a texture or illumination map) provides a continuously-variable filtered instance of the surface for sampling at any desired degree of resolution.

Figures (20) through (23) illustrate a simple surface based on a human face model developed by Fred Parke at the University of Utah. As the sampling density varies, so does the filtering of the surface. These faces are filtered and sampled by the same methods previously discussed for texture and reflectance maps. Pyramidal parametric representations such as these appear promising for reducing aliasing effects as well as systematically sampling very large data bases over a wide range of scales and viewing angles.

6. Conclusions

Pyramidal data structures are of proven value in image analysis and have interesting application to image bandwidth compression and transmission. "Pyramidal parametrics," pyramidal data structures with intra- and inter-level interpolation, are here proposed for use in image synthesis. By continuously varying the detail with which data are resolved, pyramidal parametrics provide economical approximate solutions to filtering problems in mapping texture and illumination onto surfaces, and preliminary experiments suggest they may provide flexible surface representations as well.

7. Acknowledgments

I would like to acknowledge Ed Catmull, the first (to my knowledge) to apply multiple prefiltered images to texture mapping: the method was applied to the bicubic patches in his thesis, although it was not described. Credit is also due Tom Duff, who wrote both recursive and scan-order routines for creating mip maps which preserved numerical precision over all map instances; Dick Lundin, who wrote the first assembly-coded mip map accessing routines; Ephraim Cohen, who wrote the second; Rick Ace, who translated Ephraim's PDP-11 versions for the VAX assembler; Paul Heckbert, for refining and speeding up both creation and accessing routines, and investigating various estimates of "D"; Michael Chou, for implementing highlight antialiasing and high-resolution reflectance mapping on quadric surfaces.

I owe special thanks to Jules Bloomenthal, Michael Chou, Pat Hanrahan, and Paul Heckbert for critical reading and numerous helpful suggestions in the course of preparing this text. Photographic support was provided by Michael Lehman.

8. References

[1] Blinn, J., and Newell, M., "Texture and Reflection on Computer Generated Images," CACM, Vol. 19, #10, Oct. 1976, pp. 542-547.

[2] Bui-Tuong Phong, "Illumination for Computer Generated Pictures," PhD. dissertation, Department of Computer Science, University of Utah, December 1978.

[3] Crow, F.C., "The Aliasing Problem in Computer Synthesized Shaded Images," PhD. dissertation, Department of Computer Science, University of Utah, Tech. Report UTEC-CSc-76-015, March 1976.

[4] Dungan, W., Stenger, A., and Sutty, G., "Texture Tile Considerations for Raster Graphics," SIGGRAPH 1978 Proceedings, Vol. 12, #3, August 1978.

[5] Eastman, Charles M., "Representations for Space Planning," CACM, Vol. 13, #4, April 1970.

[6] Feibush, E.A., Levoy, M., and Cook, R.L., "Synthetic Texturing Using Digital Filters," Computer Graphics, Vol. 14, July, 1980.

[7] Hanrahan, Pat, private communication, 1983.

[8] Heckbert, Paul, "Texture Mapping Polygons in Perspective," NYIT Computer Graphics Lab Tech. Memo #13, April, 1983.

[9] Klinger, A., and Dyer, C.R., "Experiments on Picture Representation Using Regular Decomposition," Computer Graphics and Image Processing, #5, March, 1976.

[10] Knowlton, K., "Progressive Transmission of Gray-Scale and Binary Pictures by Simple, Efficient, and Lossless Encoding Schemes," Proceedings of the IEEE, Vol. 68, #7, July 1980, pp. 885-896.

[11] Meagher, D., "Octree Encoding: A New Technique for the Representation, Manipulation, and Display of Arbitrary 3D Objects by Computer," IPL-TR-80-111, Image Processing Lab, Electrical and Systems Engineering Dept., Rensselaer Polytechnic Institute, October 1980.

[12] Tanimoto, S.L., and Klinger, A., Structured Computer Vision, Academic Press, New York, 1980.

[13] Tanimoto, S.L., and Pavlidis, T., "A Hierarchical Data Structure for Picture Processing," Computer Graphics and Image Processing, Vol. 4, #2, June 1975.

[14] Tanimoto, S.L., "Image Processing with Gross Information First," Computer Graphics and Image Processing 9, 1979.

[15] Warnock, J.E., "A Hidden-Line Algorithm for Halftone Picture Representation," Department of Computer Science, University of Utah, TR 4-15, 1969.

[16] Williams, L., "Pyramidal Parametrics," SIGGRAPH tutorial notes, "Advanced Image Synthesis," 1981.

[17] Yau, M.M., and Srihari, S.N., "Recursive Generation of Hierarchical Data Structures for Multidimensional Digital Images," Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing, August 1981.