Texture Mapping 3D Models of Real-World Scenes

FREDERICK M. WEINHAUS

ESL

AND

VENKAT DEVARAJAN

University of Texas at Arlington

Texture mapping has become a popular tool in the computer graphics industry in the last few years because it is an easy way to achieve a high degree of realism in computer-generated imagery with very little effort. Over the last decade, texture-mapping techniques have advanced to the point where it is possible to generate real-time perspective simulations of real-world areas by texture mapping every object surface with texture from photographic images of these real-world areas. The techniques for generating such perspective transformations are variations on traditional texture mapping that in some circles have become known as the *Image Perspective Transformation* or IPT technology. This article first presents a background survey of traditional texture mapping. It then continues with a description of the texture-mapping variations that achieve these perspective transformations of photographic images of real-world scenes. The style of the presentation is that of a resource survey rather than an in-depth analysis.

Categories and Subject Descriptors: I.3.3 [**Computer Graphics**]: Picture/Image Generation—*antialiasing*; I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling–*curve*, *surface*, *solid*, and object representations; *hierarchy and geometry transformations*; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism–*color*, *shading*, *shadowing*, *and texture*; *hidden line/surface removal*; *raytracing*

General Terms: Algorithms

Additional Key Words and Phrases: Anti-aliasing, height field, homogeneous coordinates, image perspective transformation, image warping, multiresolution data, perspective projection, polygons, ray tracing, real-time scene generation, rectification, registration, texture mapping, visual simulators, voxels

INTRODUCTION

The computer graphics industry has made dramatic advances during the last decade in the creation of ever more realistic looking images. Techniques have been developed to generate illumination-based shading effects, shadows, re-

Authors' addresses: F.M. Weinhaus, 925 Wolfe Rd. #52, Sunnyvale, CA 94086; email ⟨fmw@alink.net⟩; V. Devarajan, Electrical Engineering Department, University of Texas, Arlington, TX 76019; email ⟨venkat@uta.edu⟩.

ESL is now TRW, Sunnyvale, CA 94088.

flection, refraction, motion blur, lens defocus, and such natural phenomena as sky, clouds, haze, fire, trees, grass, and water waves. Some of these techniques have even been incorporated into flight and vehicle simulators such as those built by General Electric Controls & Guidance Systems Division (now Lockheed-Martin Co. Orlando, FL), Evans & Sutherland, and others. Unfortunately, many of these techniques are too computationally intensive for such real-time applications. For a survey of synthetic scene generation techniques, see, for example, Magnenat-Thalmann and Thalmann [1987].

Today, more demands are being placed upon visual simulators to achieve yet a higher level of realism [Fischetti and Truxal 1985; Tucker 1984]. In particular, mission planning and rehearsal systems are now striving for truly faithful representations so that ground troops can become intimately familiar with important regions of the world [Geer and Dixon 1988]. This level of detail and realism is possible only by using digitized photographs of the areas of interest.

The technology is available today. Examples have already been presented to the general public in the form of movies at conferences and on public television, and the like. The Jet Propulsion Laboratory has made several of these movies. Two of these are fly-overs, called "LA—The Movie" [Hussey et al. 1987] and "Mars—The Movie," [Hall 1989], which were presented at SIGGRAPH conferences and on PBS television. A third one depicts travel along the San Andreas fault in California [Stanfill 1991]. It has been shown as part of the film, "The Blue Planet," at IMAX® and OMNIMAX® theaters worldwide. A different fly-around of California was generated by ESL (now TRW, Sunnyvale, CA) and was shown at the 1988 World's Fair in Australia.[1] Another example is the video of the Calgary area that was created by The Analytical Sciences Corporation for ABC News and Sports [McMillan 1988; Gelberg et al. 1988]. It was shown during the 1988 Winter Olympics. Also a fly-around of Mt. Everest was created by Visual Information Technologies (VITec, now Connectware Inc.). It was shown as part of the 1991 NOVA program, "Taller Than Everest?" Finally, Volotta Interactive Video has put together a system, called the Mars Navigator [Volotta 1991], to present low altitude, computer-generated fly-bys of the surface of Mars prepared by Apple Computer [Hughes 1991] from Viking orbiter imagery. This system is on display at the Tech Museum of Innovation in San Jose.

The techniques used to make such images and movies are variations of the computer graphics technique called texture mapping. They have also come to be known by such names as image perspective transformations (IPT), 2.5D image transformations, and perspective photo mapping. The first name comes from its origins in image processing, remote sensing, and photogrammetry. The second name reflects the fact that the transformation is between two-dimensional (2D) and three-dimensional (3D); that is, the resulting views account for perspective depth, but are not truly three-dimensional in the sense of full stereoscopic viewing. The third name has been coined to show its extension from traditional texture mapping.

Some of the salient challenges that must be met to achieve these images and movies and to create effective visual simulations include:

(1) How to create high resolution models from heterogeneous input sources such as magazine pictures, direct photography, digital imagery, artists renderings, and the like.
(2) How to align or register the imagery with the geometry models and smoothly blend different models from different sources so that nothing stands out unnaturally.

[1] Produced under the technical leadership of John Thomas at ESL.

Table 1. Comparison of Traditional Texture Mapping with IPT

Traditional Texture Mapping	Image Perspective Transformation
• Texture maps are frequently synthetic	• Texture maps are actual photographs or remote sensing images of real-world areas
• Relatively few small texture maps are used	• Many large images are used as texture maps
• Texture maps are frequently repeated on multiple objects or faces	• Textures are unique per object face
• Texture maps are typically face-on views	• Texture maps are typically oblique views
• Textures are often arbitrarily mapped onto objects without concern for distortions	• Photogrammetry techniques are often used to project the textures onto object faces to correct for camera acquisition geometry
• Alignment of texture maps with the object faces is generally done manually	• Photogrammetric/remote sensing techniques are frequently used to automatically align the textures with the object faces
• 3-D object models are typically built independently from the texture maps	• Photogrammetry techniques are frequently used to build 3-D object models of terrain and urban structures from the imagery
• Polygon object models and rendering approaches are typically used	• Height field, voxel and polygon object models and rendering approaches are all used

(3) How to deal efficiently with very large databases to keep rendering times down while maintaining realism.
   —Geometry models may include both natural terrain and urban features, where the former may contain millions of (regularly or irregularly spaced) elevation values.
   —Many large, oblique, high-resolution images may be needed to cover the vast expanse of natural terrain.

(4) How to correct source imagery distortions so that resulting output views are faithful.
   —Source images are often oblique views.
   —Source images are formed by various camera and sensor types often with nonstandard geometries.

(5) How to prewarp the output images so they project naturally onto domes, pancake windows, and the like.

(6) How to perform these myriad tasks fast enough for real-time or high speed nonreal-time display.

For the purposes of discussion in this article, we refer to the texture-mapping variations that attempt to solve some or all of these challenges as the image perspective transformation. Although the distinction between the IPT texture-mapping variation and traditional texture mapping may be subtle, these differences are worth noting. Some people may argue that they are fundamentally just differences in terminology arising from the perspectives of the disciplines that originated them; namely, image processing, remote sensing, and photogrammetry in the former case and computer graphics in the latter case. To a large extent this is true, especially with regard to the basic objectives, namely, to transfer image texture onto the surfaces of 3D models. Moreover, the two technologies are rapidly converging. Nevertheless, some of the conceptual differences are worth emphasizing as well as some of the novel algorithmic approaches that have been developed that are quite different from the basic polygon-based approach used in traditional texture mapping. Consequently, some of the important differences between the image perspective transformation and the traditional texture-mapping technique are summarized in the following and listed in Table 1. A mathematical treatise of traditional and projective texture mapping is presented in Weinhaus and Devich [1997].

With traditional texture-mapping techniques, texture patterns are mathematically transferred first onto the surfaces of 3D models that are used to represent either fictitious or real-world scenes. Then, from this intermediate format, they are perspectively projected onto the output viewing plane. Usually, these texture patterns are computer-synthesized ones representing generic surface materials such as grass or wood, or they are digitized photographs of representative materials viewed face-on such as brick or concrete. Sometimes, they are even photographs of real-world scenes used to represent backdrops, views out of windows, reflection patterns of the surrounding scene, or pictures to be incorporated on interior walls of rooms. These texture patterns typically are mapped orthogonally onto planar object faces with a simple affine transformation. However, for certain applications, they either may be prewarped or treated as a parametric representation of a coordinate system such as cylindrical or spherical in order to wrap them about an object conforming to the same geometry. Texture tiling [Dungan et al. 1978], cell texturing [Economy and Bunker 1984; Bunker et al. 1984], decal mapping [Barr 1984], reflection mapping [Blinn and Newell 1976], and environment mapping [Green 1986] are some of the specific names used to describe variations of the texture mapping technique.

With the IPT technique, the textures also come from digitized photographs; however, in this case, the texture patterns are the complete and often *oblique* photographs of real-world areas that are represented by the 3D models. The objective is to start from a set of source photographs and create new views of the real-world areas from arbitrary vantage (eye) points and, if possible, to do so in real-time (i.e., 30 or 60 frames per second).

For the traditional texture-mapping case, one generally uses only a limited number of relatively small texture patterns. Usually, one texture pattern is defined to span the space of the surface in question, since there is often a simple correspondence between the polygonal bounds of the texture pattern and the polygonal shape of the surface. Sometimes, however, a single texture pattern may be repeated to fill out the coverage of a surface or the same pattern may be assigned to many neighboring surfaces. In this context, a surface can be a planar or nonplanar polygon, a nonplanar patch, or even the complete surface of a quadratic object such as a cylinder or sphere. Thus, for example, it may be that a texture pattern is arbitrarily stretched using a "rubber sheet" (affine or bilinear) transformation onto a planar quadrilateral or parametrically wrapped about the surface of a cylindrically symmetric or spherical object. As a consequence of the rather arbitrary mapping onto nonplanar objects, the resulting appearance of the texture covering for the surface may be distorted. Nevertheless, this may not be noticeable due to the lack of real-world correspondence between the texture and the model.

For the IPT case, many large photographic texture images may be needed to cover large expanses of territory. Also, different portions of one or more photographs must be mapped onto the numerous surfaces that form the complex 3D models of the real-world area. This photographic information must align with the 3D models without distortion and with minimal manual intervention. This is often accomplished as a preprocessing step using photogrammetry and remote sensing registration techniques. More optimally, the texture-mapping process should automatically take into account the acquisition geometry of the system that took the photographs. For pictures acquired by a frame (snap-shot) camera system, this mapping must correct for perspective distortion inherent in the source imagery. This distortion causes obliquely viewed rectangles to appear as quadrilaterals. For this camera system, two perspective transformations are involved: one to map the texture onto the 3D models and one to project this intermediate result onto the output viewing plane. Photogrammetry equations are often used to represent both of these perspective transformations.

The objectives of this article are threefold: to present the historical background for the image perspective transformation technique starting with its origin in texture mapping, to introduce its mathematical foundation (a detailed mathematical treatise of texture mapping can be found in Weinhaus and Devich [1997]), and to identify the various approaches that have been taken, giving credit to the many people and organizations that have contributed to it. The emphasis of this article is primarily on the geometry of the transformation; however, the issue of rendering quality is also addressed. Our approach to this survey, therefore, is to provide only a simple description of the many techniques that have been developed, citing references so that the user subsequently may review them in more depth and formulate his or her own judgment on their merits. Where appropriate, we point out the strengths and weaknesses of these approaches. However, it is not our intention to provide our biased selections of the "best" ones. Moreover, an algorithm that works well in a software environment may be quite inappropriate for hardware implementation.

In the material that follows, Section 2 first reviews the history of traditional texture mapping. It then describes the mathematical foundation and the development of texture-mapping approaches. Section 3 starts with a similar historical review for the image perspective transformation technology. It then reviews the IPT-specific approaches that have been developed, subdividing them into three categories according to the way the 3D model is represented: height fields, voxels, and polygons/patches. In Section 4, nonreal-time and real-time systems are discussed with emphasis on the technology arising from the flight simulator community.

2. TEXTURE MAPPING

2.1 Texture Mapping Background

The pioneering work in texture mapping is generally attributed to Catmull [1975]. He was probably the first to demonstrate the mapping of a (brick) texture pattern onto arbitrary planar and cylindrical surfaces. Blinn and Newell [1976] then took the concept one step further by mapping photographic textures onto a bicubic patch representation of the now famous teapot. Aoki and Levine [1978] and Feibush et al. [1980] independently used the technique to render synthetic textures onto the faces of 3D models of houses. Feibush and Greenberg [1980] also developed a texture mapping system to be used for architectural design. Later, Economy and Bunker [1984] and Bunker et al. [1984] introduced the concept of computer-synthesized "cell" textures to cover ground terrain in visual flight simulators, and Dungan et al. [1978] applied actual photographic texture patterns to cover the ground terrain. In the latter case, they acquired vertical photographs of representative tree-covered terrain from low-flying aircraft and photographs of grass textures from model boards. These texture patterns were then mapped onto 3D polygonal terrain elevation models in their simulated scenes. These authors were among the first to publish examples using textures with multiple levels of resolution as a method of anti-aliasing. Later, the scientists at General Electric [Economy and Bunker 1984; Bunker et al. 1984] and Honeywell [Scott 1983; Erickson et al. 1984] developed applications of texture mapping of photographs of individual trees onto planar "billboards" and onto multifaceted 3D models of individual trees. Transparency effects were introduced to make the texture regions around the trees and between the leaves invisible. Bunker et al. [1984] described similar translucency techniques for inserting texture patterns of cloud layers into simulated scenes.

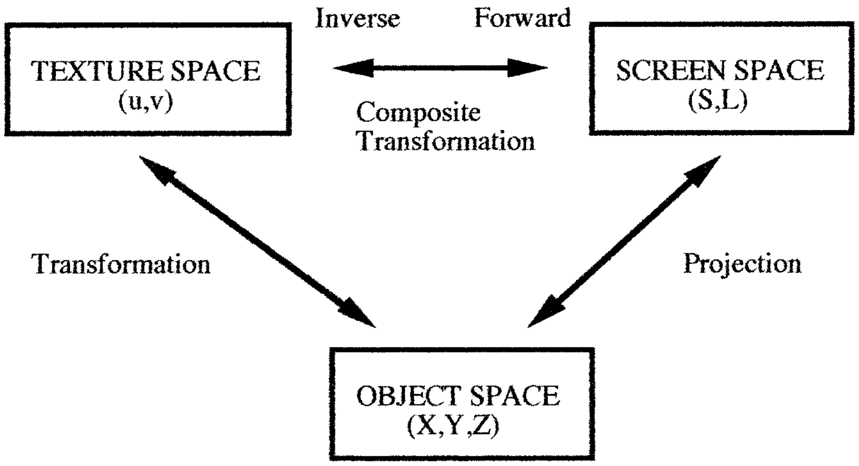

Figure 1. Texture-mapping geometry.

The geometry of texture mapping is depicted in Figure 1. Two transformations are involved. One transformation is between object space and texture (input) space and the other is between object space and screen (output) space. The object space is characterized by the equation of each surface of the 3D model. The object–texture transformation may be as simple as an affine or bilinear transformation when the texture is to be mapped orthogonally onto a planar quadrilateral. Alternately, it may be a parametric transformation when the texture coordinates are used to represent nonCartesian coordinates, such as cylindrical or spherical, and the textures are to be "shrink-wrapped" about a 3D model with like symmetry. Bier and Sloan [1986] have presented a number of examples for this latter case and Heckbert [1989] has discussed at length various 2D transformations used to map textures onto planar and nonplanar polygons. The object–screen transformation is usually either an orthographic projection (sometimes called a parallel plane projection) or a (central) perspective projection. The former involves imaging with parallel rays and exhibits uniform scale. The latter involves imaging with rays that converge to a focal point and exhibits scale that varies from foreground to background, that is, perspective foreshortening. Table 2 presents the form of some of these key transformation equations fundamental to traditional texture mapping and the image perspective transformation variation. In these equations as relevant to the direction of the transformation (forward or reverse), either (x, y, and optionally, z) or (X, Y, and optionally Z) represents object space and the other represents either texture space (u, v) or screen space (S, L).

Often the object–texture and object–screen transformations are concatenated in order to save computations. The resulting composite transformation usually can be formulated either as a forward (texture-to-screen) transformation or as an inverse (screen-to-texture) transformation. Each has its own advantages and disadvantages, which have been discussed at length by Heckbert [1989] and by Wolberg [1990].

In general, anti-aliasing techniques for texture-mapping applications have been more elaborate than for the more traditional coloring and/or illumination-based rendering techniques in order to accommodate the combination of fine detail in photographic imagery and spa-

Table 2. Key Transformation Equations*

Name	Properties	Form
AFFINE (LINEAR)	• Translation, Scale, Rotation And Skew • Preserves Straight Lines, Parallelism And Scale Homogeneity	$x = A_0 + A_1X + A_2Y$ $y = B_0 + B_1X + B_2Y$
BILINEAR	• Preserves Straightness Of Horizontal & Vertical Lines • Maps Other Straight Lines Into Hyperbolic Curves	$x = A_0 + A_1X + A_2Y + A_3XY$ $y = B_0 + B_1X + B_2Y + B_3XY$
QUADRATIC	• Maps Straight Lines Into Quadratic Curves	$x = A_0 + A_1X + A_2Y + A_3XY + A_4X^2 + A_5Y^2$ $y = B_0 + B_1X + B_2Y + B_3XY + B_4X^2 + B_5Y^2$
CUBIC	• Maps Straight Lines Into Cubic Curves	$x = A_0 + A_1X + A_2Y + A_3XY + A_4X^2 + A_5Y^2 + A_6X^2Y + A_7XY^2 + A_8X^3 + A_9Y^3$ $y = B_0 + B_1X + B_2Y + B_3XY + B_4X^2 + B_5Y^2 + B_6X^2Y + B_7XY^2 + B_8X^3 + B_9Y^3$
PROJECTION • Perspective • Orthographic	• Preserves Straight Lines	$\begin{pmatrix} wx \\ wy \\ wz \\ w \end{pmatrix} = \mathbf{PRT}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$ where Matrices **P**=projection **R**=rotation **T**=translation w is eliminated by dividing the 4th component into the other three $\mathbf{P} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/f & 1 \end{pmatrix}$ f = focal length for the perspective case $(1/f) = 0$ and $w = 1$ for the orthographic case

* Shows the form of some of the transformation equations fundamental to traditional texture mapping and the image perspective transformation variation. In these equations, as relevant to the direction of the transformation (forward or reverse), either (x, y, and optionally, z) or (X, Y, and optionally, Z) represent object space and the other represents either texture space (u, v) or screen space (S, L).

tially varying scale when viewing in perspective. Some of the more sophisticated techniques that have been used to anti-alias texture-mapped images are cited here for completeness. They include the following: trilinear interpolation on hierarchical resolution texture patterns (called pyramids [Tanimoto and Pavlidis 1975; Burt 1981] and MIP maps [Williams 1983]), elliptically weighted averaging (EWA) [Greene and Heckbert 1986], spatially variant filtering [Fournier and Fiume 1988], summed area tables [Crow 1984; Glassner 1986], repeated integration [Heckbert 1986a], clamping [Norton et al. 1982], super-resolution sampling, stochastic and jittered sampling [Cook 1986; Dippe 1985], adaptive sampling [Painter and Sloane 1989], A-buffering [Carpenter 1984], accumulation-buffering [Haeberli and Akeley 1990], and spatial transformation lookup tables [Walterman and Weinhaus 1991]. Heckbert [1986b] has presented a review and comparison of many of these anti-aliasing techniques. Some of these are discussed later in the context of the description of a particular texture mapping or IPT algorithm.

2.2 Texture Mapping Approaches

The fundamental, nontexture-mapped rendering technique used by most 3D graphics systems typically projects the vertices of each primitive (usually planar polygons) from world coordinates into screen coordinates using a perspective projection equation as shown in Table 2. It then linearly interpolates the color (and perspective depth) at each edge and interior pixel of this projected primitive. Usually, an algorithm, such as the digital differential analyzer (DDA) [Swanson and Thayer 1986; Pineda 1988] or forward difference technique [Lien et al. 1987; Shantz and Lien 1987] is used to do the interpolation from the color and perspective depth values stored for each vertex. Such algorithms are very efficient, since they exploit incremental computations along each scan-line of the projected primitive. The well-known Gouraud shading technique is an example of this type of interpolation where the shading information associated with the vertices is determined by an illumination model in conjunction with the vertex normals. The Phong shading technique is similar, but interpolates the vertex normals before determining the shading value at each interior pixel. See, for example, Rogers [1985] or Foley et al. [1993] for more details about these two specific shading techniques.

As pointed out by Heckbert and Moreton [1991] and by Blinn [1992], a naive approach to texture mapping would store the corresponding texture coordinates for each vertex and try to interpolate these coordinates in a similar manner. The interpolated texture coordinates would then be used to index into the texture image to extract the appropriate color values. Unfortunately, this procedure introduces distortions that may be noticeable, because the technique approximates the perspective projection with a linear transformation. Moreover, polygons are usually subdivided into combinations of triangles and quadrilaterals that have one or two edges aligned along the scan direction, respectively. Each triangle or quadrilateral ultimately will use a different linear transformation. This may create sharp bends or folds in transformed linear features as they pass from one subpolygon to another. A demonstration of this effect is presented in Figure 2(b) where the quadrilateral has been subdivided along one of its diagonals into two triangles. Since the other diagonal must pass through the midpoint of the first diagonal, it will be transformed into two joined line segments, each with different slopes. Figure 2(a) shows the original texture map.

To avoid such discontinuities for quadrilaterals, one might try using a bilinear transformation rather than a linear one to interpolate the texture coordinates. Unfortunately, the bilinear transformation only preserves straight lines originally parallel to the coordinate axes of the texture map. Consequently, the diagonals will appear curved in the resulting interpolated polygon. A proper perspective transformation will preserve all straight lines. Demonstrations of these characteristics are presented in Figures 2(c) and 2(d). These three transformations also affect scale differently. The linear and bilinear transformations preserve scale across the texture pattern better than the perspective transformation. This effect can be observed in the figures by examining the spacing of the grid pattern, especially between those lines that correspond to the horizontal ones in the original texture image. In Figure 2(d), this spacing progressively increases from top to bottom, whereas, in Figures 2(b) and 2(c), it remains constant.

Note that the linear approximation gets progressively better for smaller polygons. Thus, Catmull's [1975] rendering algorithm, which adaptively subdivides surfaces into pixel-sized micropolygons, is well suited to handle this problem. Following this premise, Oka et al. [1987] used a local linear approximation to perform texture mapping of photographs of people onto

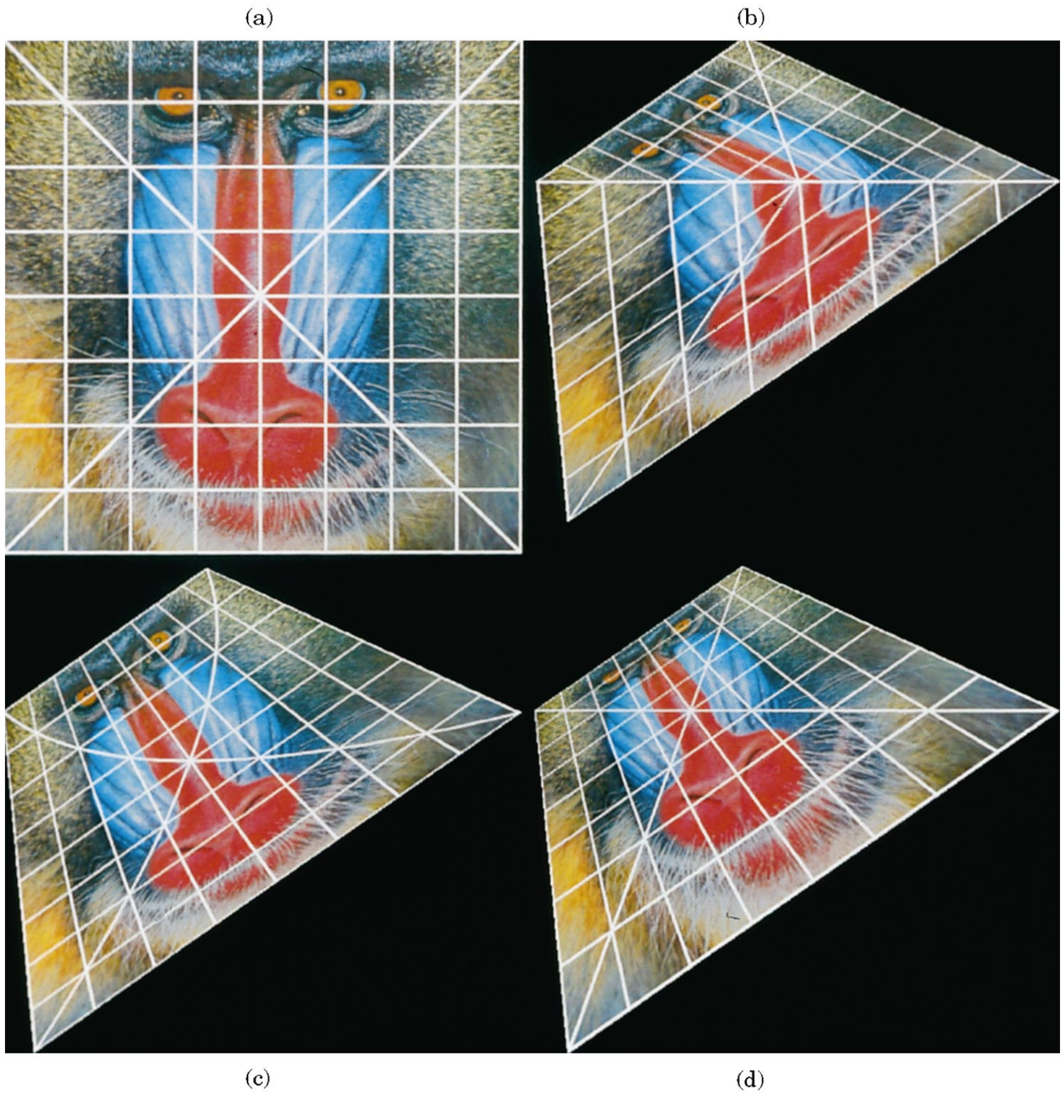

Figure 2. Comparison of several texture-mapping transformations: (a) original texture map; (b) after linear transformation; (c) after bilinear transformation; (d) after perspective transformation.

quadrilateral mesh models of human faces. They were then able to modify facial expressions by interactively manipulating the mesh.

Kirk and Voorhies [1990] have demonstrated the use of a quadratic interpolation function rather than a linear one to approximate the perspective transformation for texture-mapping applications. Other examples of the use of quadratic as well as cubic approximations for texture mapping in perspective have been presented by Wolberg [1990]. However, polygon subdivision may still be necessary to achieve acceptable results.

The proper mathematical basis for texture mapping onto planar polygons has been described in detail by Aoki and Levine [1978], Gangnet et al. [1982], Perny et al. [1982], Heckbert [1989; 1983], Heckbert and Moreton [1991], and Blinn [1992] for the case in which the texture is to be mapped with an affine transformation and where the output viewing transformation is a perspective projection. These two transformations can be concatenated into a single fractional linear form that is applicable in either direction. Thus, it can be used to describe either forward mapping from texture space to screen space or inverse mapping from screen space to texture space. For the inverse mapping case, this transformation can be expressed as follows.

$$u = \frac{P_0 + P_1S + P_2L}{R_0 + R_1S + R_2L}$$
$$v = \frac{Q_0 + Q_1S + Q_2L}{R_0 + R_1S + R_2L}, \tag{1}$$

where the pixels in the texture image are represented by the (u, v) coordinates and those in the output image are represented by the (S, L) coordinates. Note that without any loss of generality, the zeroth order coefficient in the denominator can be set to unity; that is, the set of equations can be normalized by dividing both numerators and the common denominator by this term. From Equation (1), it is easy to see that the two numerator terms and the one denominator term are the proper expressions to interpolate, because each of them is a linear polynomial in S and L. This is often done simply by incrementing them by P_1, Q_1, and R_1, respectively, along a given scan-line. Since the texture coordinates are composed of ratios of these linear polynomials, they are not good choices for the linear interpolation technique. Blinn [1992] refers to interpolation with this fractional linear form as hyperbolic interpolation.

Equation (1) is generally used for texture mapping onto planar rectangles, because it requires a minimum of four pairs of corresponding texture and screen coordinates to solve for the eight unknown coefficients. The four vertices of the rectangle projected into screen space in the form of a planar quadrilateral and the four corresponding bounding corners of the rectangular texture image provide a convenient set. The solution can be achieved by multiplying through by the denominator and recasting the equation in the following form:

$$(1)P_0 + (S_i)P_1 + (L_i)P_2 - (S_iu_i)R_1 - (L_iu_i)R_2 = u_i$$
$$(1)Q_0 + (S_i)Q_1 + (L_i)Q_2 - (S_iv_i)R_1 - (L_iv_i)R_2 = v_i, \tag{2}$$

where the zeroth order coefficient in the denominator has been set to unity. Substitution of each corresponding set of coordinates ($i = 1$ to 4) into this normalized pair of equations provides a set of eight linear equations in eight unknowns that can be solved, for example, by the unit_square-to-quadrilateral technique described by Heckbert [1989]. As Heckbert also points out, this approach can be extended to handle the case of mapping a quadrilateral region of texture onto a planar rectangle, since the solution and its inverse can be concatenated easily to form a quadrilateral-to-unit_square-to-quadrilateral transformation.

Hourcade and Nicolas [1983] have taken the development a step further. They used a bilinear transformation rather than an affine one to map the texture onto the polygon and then viewed it in perspective. The use of the bilinear form, which includes a crossterm, permits a rectangular texture image to be mapped onto more general planar or even nonplanar quadrilaterals, neither of which are compatible with the affine transformation. For the forward (texture-to-screen) mapping case, Hourcade showed that the composite transformation can be expressed as the ratio of polynomials up to and including the (uv) crossterm. However, for the inverse (screen-to-texture) transformation, a quadratic equation must be solved for each screen pixel. (The quadratic characteristic of the inverse bilinear transformation also has been described by Heckbert [1989] and by Wolberg [1990].)

The preceding analysis follows what is commonly called a single-pass, 2D approach. Catmull and Smith [1980] and Smith [1987] have taken a different tack. They developed a flexible and efficient forward mapping, two-pass, 1D approach. In the first pass, the texture pattern is warped only in one dimension (horizontally or vertically) to form an intermediate result. In the second pass, the intermediate result is warped only in the orthogonal dimension. These authors presented equations for texture mapping onto planar and bilinear quadrilaterals, biquadratic and bicubic patches, and superquadrics. For the case of texture mapping onto a planar rectangle and perspective viewing, they showed that the two 1D transformations are also fractional linear in form. For the other cases, ratios of higher-order polynomials are involved.

In the review previously presented, it was generally assumed that the complete rectangular texture pattern was mapped one-to-one onto a four-sided polygon or patch. However, this condition may be relaxed by using *projections* to determine the texture coordinates as well as the screen coordinates. In other words, the affine object-texture transformation depicted in Figure 1 is replaced with an orthographic projection. This will permit portions of the texture pattern to be mapped onto planar polygons with more than four sides. It is only necessary that the projection of the polygon onto the texture pattern does not extend outside the texture domain. In this way, a corresponding polygonal subsection of the texture pattern will be used to fill the interior of the polygon (i.e., a "cookie-cutter" technique). The texture coordinates corresponding to the projected polygon vertices can be predetermined and saved or computed at the time the coefficients of Equation (1) are evaluated. Then four of the vertices of the planar polygon can be projected into screen space and Equation (2) solved for the coefficients.

But what about triangles? One simple solution is to project the centroid of the triangle along with its vertices into texture and screen space so that four conjugate sets of points are available. Alternately, a more general solution can be developed. This is achieved by going back one step in the development of Equation (1), before the perspective divide, where the composite transformation can be expressed in matrix form using homogeneous coordinates as

$$\begin{pmatrix} u' \\ v' \\ w \end{pmatrix} = \begin{pmatrix} wu \\ wv \\ w \end{pmatrix} = \begin{pmatrix} P_1 & P_2 & P_0 \\ Q_1 & Q_2 & Q_0 \\ R_1 & R_2 & 1 \end{pmatrix} \begin{pmatrix} WS \\ WL \\ W \end{pmatrix}$$
$$= \begin{pmatrix} P_1 & P_2 & P_0 \\ Q_1 & Q_2 & Q_0 \\ R_1 & R_2 & 1 \end{pmatrix} \begin{pmatrix} S' \\ L' \\ W \end{pmatrix}. \qquad (3)$$

Here, w and W are the corresponding third homogeneous coordinate components for any object space point projected into texture and screen spaces, respectively. For more information about homogeneous coordinates, see, for example, Foley et al. [1993]. Note that w may be set to unity when the object–texture transformation is represented by an orthographic projection, so that $u' = u$ and $v' = v$ are true pixel coordinates in the texture pattern. Substitution of any three sets of these corresponding coordinates into Equation (3) results in the readily solvable set of nine linear equations in eight unknowns.

This approach is easily generalized to the case where a perspective projection is used for the object–texture transformation. In this case, w will take on values other than unity. The disadvantage is that three floating-point texture (and screen) homogeneous coordinate components are required for each vertex. This would increase the number of bits of vertex information that would have to be carried with the 3D models, if one desired to precompute and store them rather than compute them only when needed.

An important characterization of Equation (1) has been described by Fuchs et al. [1985]. They showed that for texture mapping with an affine object–texture transformation the denominator term in Equation (1) is equivalent to inverse depth in the viewing or eye coordinate system, that is, $R_0 + R_1S + R_2L = (1/Z_e)$. Here Z_e is the coordinate component in the eye coordinate system along the look direction. This implies that inverse depth should be interpolated and used in depth buffer algorithms rather than simple depth. The numerator terms can then be equated to the product of inverse depth with the texture coordinate components, namely, $(u/Z_e) = P_0 + P_1S + P_2L$ and $(v/Z_e) = Q_0 + Q_1S + Q_2L$, and these products can be precomputed for each vertex of the polygon. Equation (1) can be evaluated subsequently for each screen pixel within a polygon simply by interpolating these precomputed products (u/Z_e) and (v/Z_e), representing the numerator terms, and then dividing the interpolated results by the interpolated denominator term $(1/Z_e)$. The advantage of this approach is that the coefficients of the fractional linear equation never need to be evaluated. Note that the equivalence between the denominator term and inverse depth is not justified for the case in which the object–texture transformation is a perspective projection.

An analogous approach, but expressed using homogeneous coordinates, was first described by Heckbert and Moreton [1991] and later by Blinn [1992]. Heckbert and Moreton showed that each of the terms (u/ω), (v/ω), and $(1/\omega)$ are linear in S and L and so should be interpolated rather than simply u and v. Here ω is the fourth homogeneous coordinate in the transformation of eye coordinates $(X_e, Y_e, Z_e, 1)$ to homogeneous eye coordinates $(\omega X_e, \omega Y_e, \omega Z_e, \omega)$. Then, dividing the first two interpolated terms pixel by pixel by the fourth interpolated term recovers the interpolated texture coordinates u and v. The equivalence of this approach with the previous one is apparent, since $\omega = (Z_e/f)$ [Foley et al. 1993], where f is the distance between the eye point and picture plane in the viewing pyramid.

Heckbert and Moreton [1991] took this approach a step further and showed that the correct terms to interpolate when the object–texture transformation is a perspective projection rather than an affine transformation are $(u\omega'/\omega)$, $(v\omega'/\omega)$, and (ω'/ω), where ω' is analogous to ω, but is for the texture projection coordinate system. Segal et al. [1992] have described an implementation of this projective texture-mapping approach to demonstrate shadows, spot lighting, and slide projection effects in computer-generated images.

One can list numerous past and present research and commercial computer graphics systems as well as the commercial flight simulators mentioned earlier that use or have used texture mapping to a limited extent. A few of these are hardware and software systems developed by Hewlett-Packard, Pixar (Reyes Image Rendering Architecture [Cook et al. 1987] and RenderMan software [Upstill 1990]), Schlumberger Palo Alto Research [Deering et al. 1988], Silicon Graphics Inc. (SGI), Stardent Computers (now Kubota Graphics), Sun Microsystems, and the University of North Carolina (Pixel-Planes [Fuchs et al. 1985]). These systems rely or relied primarily on traditional 3D graphics techniques such as Gouraud and Phong shading. By using special-purpose hardware, many are capable of rendering hundreds of thousands of polygons per second. All of these systems can map textures onto surfaces, but most of them limit the amount of texture data that can be used so that rendering rates are not degraded by either disk-to-memory or memory-to-processor bandwidths. A few systems that use parallel processors store the same texture patterns in each processor's own local memory so that real-time rendering rates can be maintained. Some of the advanced simulators recently developed by Evans & Sutherland, Lockheed-Martin Co. (Orlando, FL), Lockheed-Martin-Vought Systems (formerly LTV), and SGI have been designed specifically to address applications requiring high quality texture mapping of large quantities of texture data.

3. IMAGE PERSPECTIVE TRANSFORMATION

3.1 Image Perspective Transformation Background

The techniques used in the perspective transformation of images have evolved readily from those used to perform texture mapping. *The most important difference between these two applications, however, is the need to handle larger, more numerous and frequently oblique photographic texture data in the former case.* This implies that larger memory capacity, faster disks and higher bandwidth I/O channels are desirable. New algorithmic approaches also are needed to accommodate the increased load.

Another important difference is the use of photogrammetry techniques to represent the texture imagery acquisition geometry. The collinearity condition [Gosh 1979; Thompson 1966] in photogrammetry is the basis for characterizing the acquisition geometry. It states that a point on an object, its corresponding picture point, and the focal or perspective center all lie on the same line. For a frame camera, this may be expressed in forward and inverse form as

$$\begin{pmatrix} x \\ y \\ f \end{pmatrix} = \mu \mathbf{R} \begin{pmatrix} X - X_c \\ Y - Y_c \\ Z - Z_c \end{pmatrix} \tag{4a}$$

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} + \lambda \mathbf{M} \begin{pmatrix} x \\ y \\ f \end{pmatrix}, \tag{4b}$$

where (x, y, f) represent the picture point, (X, Y, Z) represent the object point, (X_c, Y_c, Z_c) represent the camera location, and $\mathbf{R} = \mathbf{M}^T$ is a 3D rotation matrix. As in the case of the perspective projection equation in Table 2, the third components here must be divided into the other two components in order to eliminate the unknown parameters μ and λ. In fact, Equation (4) is an alternate representation for perspective projection. Imaging systems such as panoramic [Case 1967], strip [Case 1967], and multispectral scanners [Baker 1977] (e.g., LANDSAT, SPOT, etc.) will have other mathematical forms. Nevertheless, they all relate an object point to an image point. Having such equations permits the imagery to be "projected" onto the surfaces of the object models, automatically correcting for the imaging distortions inherent in these systems. Moreover, they facilitate constructing the 3D object models, since a 3D point on an object may be identified by intersecting two such "rays," one each passing through corresponding feature points on two different images. Utilization of the collinearity equations requires knowledge of the camera location and orientation parameters. These parameters are deduced typically by nonlinear least squares fit either for each image independently or simultaneously for a group of overlapping images. In the former case (called space resection), the least squares fit requires conjugate image and world "control" points. In the latter case (called block or bundle adjustment), the least square fit uses a combination of the same along with conjugate image "pass" points located on overlapping images.

Rather than photogrammetrically projecting textures onto the surfaces of the terrain models, remote sensing registration and resampling techniques are often used instead to preprocess the images to align them with the terrain model. In effect, this preprocessing step removes the imaging distortion so as to rectify the imagery to a vertical view. Several different approaches have been used. One method performs a least squares fit of a set of arbitrarily spaced control points using a low (typically second or third) order 2D polynomial transformation (see Table 2) to facilitate the registration and resampling operation globally across the image. A second method divides the image into a rectangular grid of control points and then uses a different bilinear transformation locally within each grid area to resample the imagery [Rifman 1973; Bernstein 1976; Van Wie and Stein 1977]. A third method divides the image into a triangular mesh of control points and then uses a different affine transformation locally within each triangle to resample the imagery [Goshtasby 1986, 1987].

Another approach, called orthorectification [Hood et al. 1989; Friedman 1981; Peled and Shmutter 1984], uses the collinearity equation and the terrain elevation model to correct for terrain relief distortion as well as the imaging distortion. In effect this technique is a special (reverse) case of traditional texture mapping, where the object–texture transformation is a perspective projection (for a frame camera system) and the object–screen transformation is a (vertical) orthographic projection (represented by an affine transformation). It also differs in that the output image is typically much higher resolution (larger) than normal. It is essentially a case of a photogrammetry-based image perspective transformation used to preprocess the imagery so that a simpler (affine) object–texture transformation can be used later in the final texture-mapping operation.

Subsequent sections describe the various algorithmic approaches that have been developed for the IPT case. These approaches are categorized in several ways. One grouping characterizes them according to the type of hidden object removal (or visibility) technique that is used in the rendering (e.g., depth-priority, depth-buffer, scan-line, ray tracing, etc.). For a general review of these visibility techniques, see, for example, Magnenat-Thalmann and Thalmann [1987], Rogers [1985], or Foley et al. [1993]. A second grouping characterizes them according to scene content, for example, urban versus rural (cultural versus natural). A third grouping characterizes them by the class of 3D model primitive used to represent the setting, for example, a regular array of point features, voxels, polygons, or higher-order surfaces and patches.

Perhaps the earliest example of a perspective transformation of an image of a real-world scene was that presented by Tanaka [1979]. He transformed a LANDSAT image of Mt. Fuji into a low altitude horizontal view. The type of 3D model that he used was a regular grid of elevation values interpolated from a contour map.

In 1980, Lippman [1980] described the "Movie-Map" concept of "surrogate travel" being developed at the Massachusetts Institute of Technology. In one study, frames of movie film were collected at 10 ft spacing while driving down the streets of Aspen, Colorado. Then selected portions of some of these frames corresponding to face-on views of building facades were mapped onto the faces of 3D models of the buildings. Distant mountains were also textured to add a sense of realism. Computer-synthesized images simulating driving down the streets as well as low altitude views were then generated in segments and stored on video disk for later playback under joystick control. During playback, turns could only be initiated at selected locations such as street intersections.

Devich and Weinhaus [1980, 1981a] at ESL (now TRW, Sunnyvale, CA), also in 1980, presented another example of a perspective transformation of an urban setting. Their example converted two oblique and one nadir-looking aerial photographs of downtown San Jose, CA into multiple street-level horizontal views. These authors constructed 3D planar polygon models for the buildings and terrain from the source photographs using photogrammetry techniques rather than from blueprints or dimensional drawings. This was perhaps the first demonstration where the object–texture transformation was a perspective projection, expressed in the form of the preceding collinearity equation, rather than an affine or orthographic projection and where it was also concatenated with the perspective object–screen transformation to form a single composite texture mapping transformation. These authors showed that this composite transformation was also characterized by Equation (1). Using this composite texture mapping transformation along with the photogrammetric calibration of the images, these authors were able to automatically map the appropriate areas of the images onto each model face without manual intervention to cut-out and rectify them first to face-on views.

In 1981, they also presented perspective transformations of both LANDSAT images and digitized high-altitude aircraft photographs of the Grand Canyon area in Arizona [Devich and Weinhaus 1981b]. Like Tanaka, they used a regular grid of elevation values for the 3D model of the terrain; however, this grid was purchased as a standard product from the United States Geological Survey. In addition to showing transformed images in perspective, panoramic, and orthographic formats, they also presented a pair of transformed images that could be viewed stereoscopically in 3D.

Another noteworthy early development was that by Dungan [1979]. However, he transformed a computer-synthesized image of a rural setting. The pseudophotographic texture information for this synthesized image was generated from a regular grid of elevation values using a combination of processing techniques that included illuminated shading based upon a diffuse (Lambertian) reflection model. Subsequently, several other authors created perspective views of rural terrain settings using the diffuse shading technique either as a preprocessing step, like Dungan, or including it as an integral part of the rendering process [Coquillart and Gangnet 1984; Miller 1986; Robertson 1989].

Since the early 1980s, most examples have concentrated on rural settings. These so-called terrain rendering approaches have been very amenable to the development of novel and efficient algorithms. Only a few examples have been published that have dealt with the more complex urban or mixed rural and urban setting.

Most IPT examples have focused on rural settings because the terrain surface often can be treated as a single-valued function of elevation, that is, a height field. Vertical overhangs are not allowed or are ignored. This permits much freedom in the form that can be used to represent the 3D model of the terrain. The simplest form treats the terrain by an array of equally spaced elevation points or vertical posts. Sometimes, the posts are considered to have a width equivalent to their spacing. Consequently, the terrain surface is represented as a uniform step function. Another representation that has gained popularity is the voxel. Voxels are essentially volume elements (cubes or rectangular parallelepiped columns). In their most general usage, they are linked in an octree hierarchical structure and all sides of the voxel are important [Samet and Webber 1988a, b]. They have found perhaps their most significant application in volumetric rendering for the medical imaging field where they are often used to represent the 3D form of the human body. However, when they are used as a single-valued function to represent the surface of natural terrain, the sides are either ignored or coded with the same texture value as the top. Therefore, such voxels also represent the terrain as a step function, but with horizontal step dimensions that may not be uniform. In other words, where the terrain is relatively flat, larger voxels and wider steps may be used. The terrain may also be represented in more traditional forms by a mesh of planar triangles, bilinear rectangles, or higher-order patches. Fig-

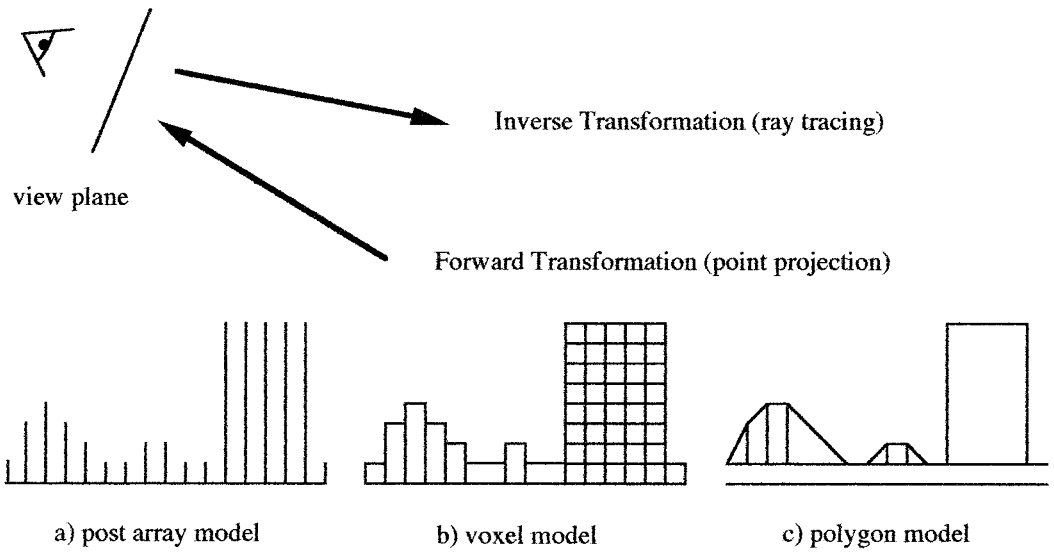
Inverse Transformation (ray tracing)
view plane
Forward Transformation (point projection)
a) post array model
b) voxel model
c) polygon model

Figure 3. Various representations of 3D models for a mixed rural and urban region with the terrain on the left side and a building on the right side.

ure 3 depicts a representative mixed rural and urban area for each of these types of 3D models.

3.2 Height Field-Based Approaches

One can find many examples of perspective transformations of images of rural settings based upon the point-array, post-array, or uniform column-type voxel representation of the terrain elevation. This is by far the most frequently used representation because it accommodates the simplest and most efficient rendering algorithms. This efficiency is achieved (1) by a preprocessing step to coregister and resample the image and the terrain elevation array to a common (map projection) coordinate system and scale and (2) by keeping both the image and terrain array in memory when the data sets are small enough. In addition, both this approach and the voxel-based approach discussed in the following are amenable to massive parallelization.

The registration and resampling operations account for the texture-to-object transformation. This is particularly advantageous when generating multiple views for motion sequences, since such computations need not be repeated for each frame. However, there is also a disadvantage associated with registration. When high-altitude satellite data such as LANDSAT or SPOT are used, there may be only a minimal affine transformation required to scale, rotate, skew, and translate the image. However, if multiple oblique images are used, then more complicated techniques such as orthorectification must be performed to get these images into a common vertical format and to correct them for terrain elevation relief distortion. Furthermore, these images must be mosaicked together. This may be quite time consuming, since it is not uncommon for the mosaicked image to be tens of thousands of pixels on a side. Another drawback is that the terrain elevation array often must be enlarged to match the finer scale of the mosaicked imagery, thereby causing an increased burden on memory and/or disk storage.

Two categories of approaches can be found in the literature for the height-field terrain representation. The first is generically called ray-tracing, but is

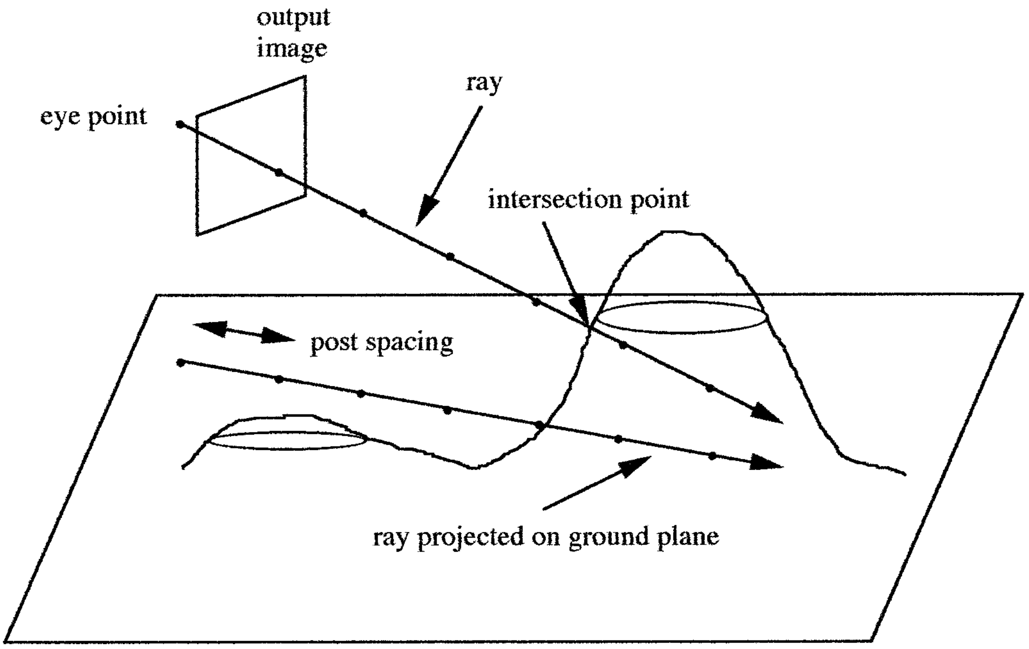

Figure 4. Ray-tracing methodology.

usually only ray-casting, because secondary rays for reflection and/or refraction are not generally spawned. This is an inverse (screen-to-object) transformation approach. The second is what might be called point projection, which is a forward (object-to-screen) transformation approach. These two approaches are depicted in Figure 3.

3.2.1 *Ray-Tracing Approaches for Height Fields.* With the ray-tracing approach, every pixel in the screen (output view) is processed in sequence. This might be described as output data driven and it grows in proportion to the size of the screen image generated and to a lesser extent in proportion to the size and complexity of the 3D terrain model. A ray from the (output) eye point is cast through each screen pixel in sequence and its intersection with the terrain is determined. Dungan's [1979] shaded terrain work, described earlier, used a ray-tracing technique. Butler and Harville-Hendrix [1988] and Butler [1989] at Visual Information Technologies (VITec, now Connectware Inc.), and Nack et al. [1989a, b] at Image Data Corp. (now called Core Software Technology) have independently presented ray-tracing examples of image perspective transformations using variations on an extremely efficient algorithm called accelerated ray-tracing (ARTS) [Fujimoto et al. 1986]. This technique uses a 3D DDA concept, which steps along the projection of the ray on the ground plane in horizontal units equivalent to the terrain post spacing, decrements the elevation of the ray by a constant amount precomputed from the slope of the ray, and compares it with the corresponding elevation of the terrain. When the ray height makes the transition from above the terrain elevation to below the terrain elevation, the ray has pierced the terrain. See Figure 4. Then a nearest-neighbor or higher-order interpolation, such as bilinear, cubic convolution, or windowed sinc function, can be used to compute the (X, Y) ground coordinates of the intersection. Once these coordinates are determined, the corresponding location in the texture image is addressed and the color of this pixel or interpolated color from neighboring pixels is transferred to the screen. In certain circumstances, speedups can be achieved for each subsequent pixel along a screen column by starting the search each time from the last intersection rather than from the (*X*, *Y*) coordinate of the eye point.

When higher order interpolation is used to locate the exact terrain intersection, the terrain is essentially represented by a smooth surface rather than by a step-function. Therefore, this interpolation anti-aliases ridgelines; that is, it smooths out these "jaggies." Furthermore, when higher-order interpolation is used to resample the input image, some texture anti-aliasing can be achieved (by virtue of the larger interpolating kernel size) to mitigate the artifacts produced by undersampling. This anti-aliasing reduces the amount of speckling in single scenes and blinking or shimmering in motion sequences that are typical of ray-traced oblique perspective views. However, due to the finite size of the interpolating kernel, the mitigating effects are of limited benefit. In fact, they fail where there is a large change in scale between the input and output images, for example, in the background regions of a low oblique perspective view. In such cases, an output pixel would have a very elongated "footprint" if projected into the input image, covering many or even hundreds of pixels. Thus, a bilinear, cubic convolution, or windowed sinc function kernel (nominally, 2×2, 3×3, and 7×7 pixels on a side, resp.) would not adequately sample the data within this "footprint."

The height-field approach also has been applied to the urban setting where the elevations of the terrain posts are raised locally to account for the heights of buildings. However, the faces of buildings rendered in this manner often display a vertically striped appearance, since the texture values along the boundaries of the roofs are simply extended down the sides of the buildings where no source texture is available. In other words, this approach simply assigns the texture that corresponds to the top of the elevation posts to the sides of the (visible) posts.

The approach used by the Jet Propulsion Laboratory to make their early movies was also a ray-tracing method. It has been described by Hussey et al. [1986] and Mortensen et al. [1988]. Subsequently, Stanfill [1991] has introduced multiresolution (pyramid) imagery techniques into JPL's approach to reduce the sparkling artifact typical of aliasing. A precomputed lookup table based upon the range from the eye point to the terrain is used to select the appropriate image resolution level to use during rendering. Each output pixel is then rendered as a weighted blend from two different input image resolution levels so that no sudden spatial or temporal transitions in resolution are apparent. Figure 5 is an example image generated by this multiresolution ray-casting technique using a height-field representation for the terrain.

Such pyramid techniques are much better at mitigating the aliasing artifacts previously discussed because the input data are more completely represented in any sampling for an output pixel. The downside is that they tend to overcompensate; that is, they include too many input data in their "footprint" sampling, since they are usually generated with a square rather than a rectangular averaging filter. Often the result is too much blurring rather than too much aliasing. See Williams [1983] and Heckbert [1989, 1983] for more details.

Devich and Weinhaus [1981b] also presented a novel variation on the ray-tracing concept. They recognized that, for a camera with a horizontally oriented optical axis, columns and rows on the screen correspond to azimuth and elevation angles. Therefore, before ray-tracing, they processed both the terrain elevation array and the image into a polar coordinate system centered at the (*X*, *Y*) position of the eye point. They

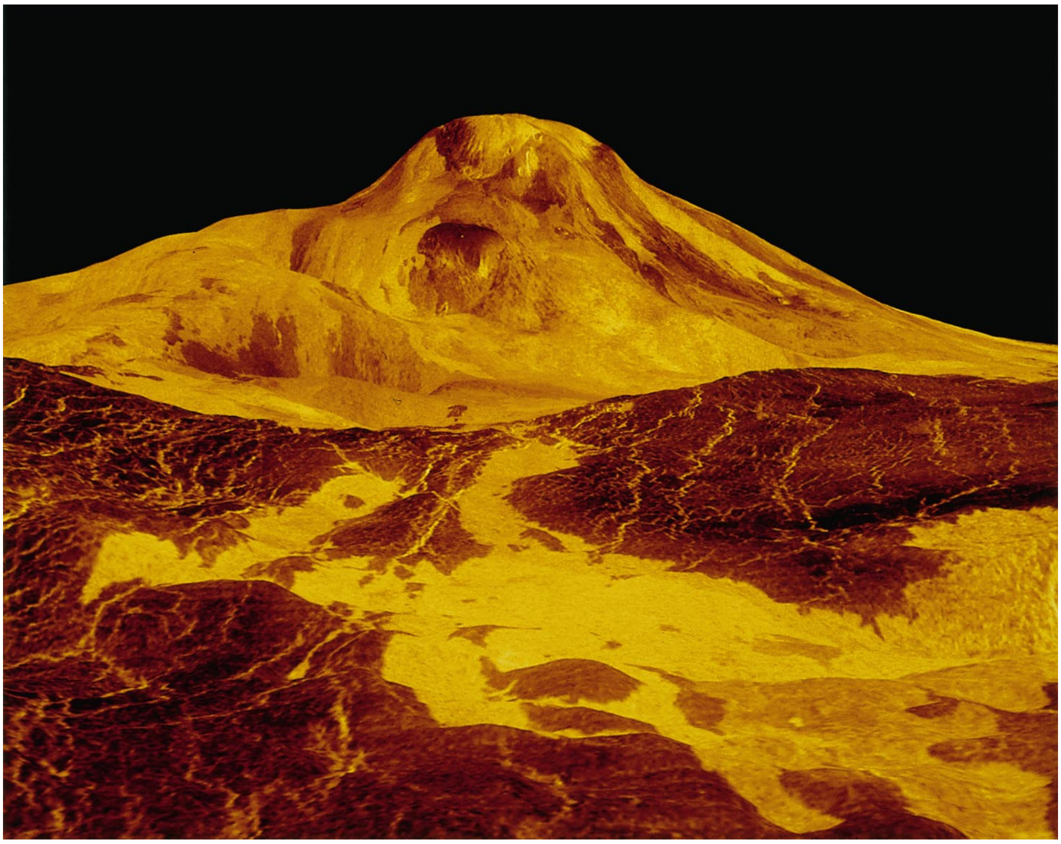

Figure 5. Computer-generated view of the 5 mi high volcano, Maat Mons, on Venus. Source data included colorized radar imagery acquired by the Magellan spacecraft and altimeter elevation data in height-field format. Perspective view generated by JPL/NASA and photo obtained from Newell Color Lab.

also processed the elevation values in the terrain array, first to correct for earth curvature and then into values representing screen-line coordinates which are proportional to the tangent of the elevation angles. Ray-tracing was thereby transformed from incrementing both X and Y ground coordinates to incrementing simply radial distance, that is, along a single dimension. Furthermore, no decrement in the ray elevation was required, because of the conversion of the terrain elevation values to a quantity more related to elevation angle. Thus, extra computations associated with the Cartesian-to-polar and elevation-to-elevation-angle processing were traded for more efficient ray-tracing. Weinhaus and Walterman [1990] have pointed out that the Cartesian-to-polar transformation need only be set up once in the form of a spatial transformation lookup table and can be used for all subsequent translations of the eye point (but not for changes in view direction). For the case of tilted views, Devich and Weinhaus implemented an extra one-dimensional transformation along each row of the screen space image to correct the geometry from that of a horizontal view. Anti-aliasing was achieved by processing multiple resolution versions of the input into range annuli in the output and by using higher-order resampling techniques. Ridgelines were specially anti-aliased by blending information from foreground and background pixels using area coverage weighting factors.

Paglieroni and Peterson [1992] have presented a ray-tracing method that adaptively optimizes each step size during the ARTS algorithm. Their method involves preprocessing the elevation data using distance transforms [Borgefors 1986; Paglieroni 1992]. For each possible height value in the digital elevation array, they extract a horizontal cross-sectional slice in the form of a binary image. This binary image codes regions above and regions equal to or below the given elevation value. The *X–Y* Euclidean distance from each "above" pixel to the closest "equal to or below" pixel in a plane is then computed and stored. In concept, this would form a set of hierarchical distance, images, one image for each possible height value within the data. During ray-tracing, the (X, Y, Z) position at a given step could be used to look up the next step's horizontal increment from within the set of hierarchical distance images. However, to save on disk storage, for each pixel in the elevation model, they form a parametric representation for the hierarchical data. This approximation sets a lower bound on the step size. Thus, in place of the full sequence of distances as a function of height values, only a slope and intercept are stored for each pixel in the digital elevation array. The distance to use for any given step is then computed from these slope and intercept values rather than from an actual lookup into the hierarchical distance images. For the data tested, Paglieroni and Peterson found speedup factors for their algorithm of about 40 and 6 when compared to a uniform step, accelerated ray-tracing method, and to an hierarchical step, octree-like ray-tracing method, respectively.

3.2.2 *Point Projection Approaches for Height Fields.* In the point projection approach, the registered image and terrain elevation array are divided into corresponding profiles and these dual profiles are processed one pair at a time as follows. First, each pixel in the terrain elevation profile is projected from object space to screen space where it is usually truncated or rounded to the closest integer coordinate. Then the texture value from the corresponding pixel in the image profile is assigned to the specified screen coordinate. This process is input data driven and grows in proportion to the size of the registered 3D terrain model and source (texture) image.

One generally finds two common forms of hidden pixel removal used in conjunction with the point projection approach: the painter's algorithm and the floating horizon algorithm. However, in either case, the profiles typically are linear slices through the registered digital elevation array and texture image oriented orthogonal to the projection of the output view's optical axis onto these data sets. Alternately, they may simply be rows or columns of this dual data, if properly sequenced. Figure 6 depicts both kinds of point projection algorithms as discussed in the following.

In the painter's algorithm, the profiles are processed starting with the most distant profile from the eye point and advance towards the profile closest to the eye point. When textures corresponding to different terrain elevation profiles contend for the same screen pixel, the current one (associated with the closest profile to the eye point) always overwrites the previous one.

In the floating horizon technique, the profiles are processed in the opposite order; that is, they advance from the profile closest to the eye point towards the profile furthest from the eye point. A floating horizon buffer is maintained as a mechanism to eliminate hidden pixels. As each profile is processed, this buffer is potentially updated and saves the screen line coordinate associated with the current horizon for each screen sample coordinate. Any projected point that falls below the current horizon is ignored. The current horizon is only up-

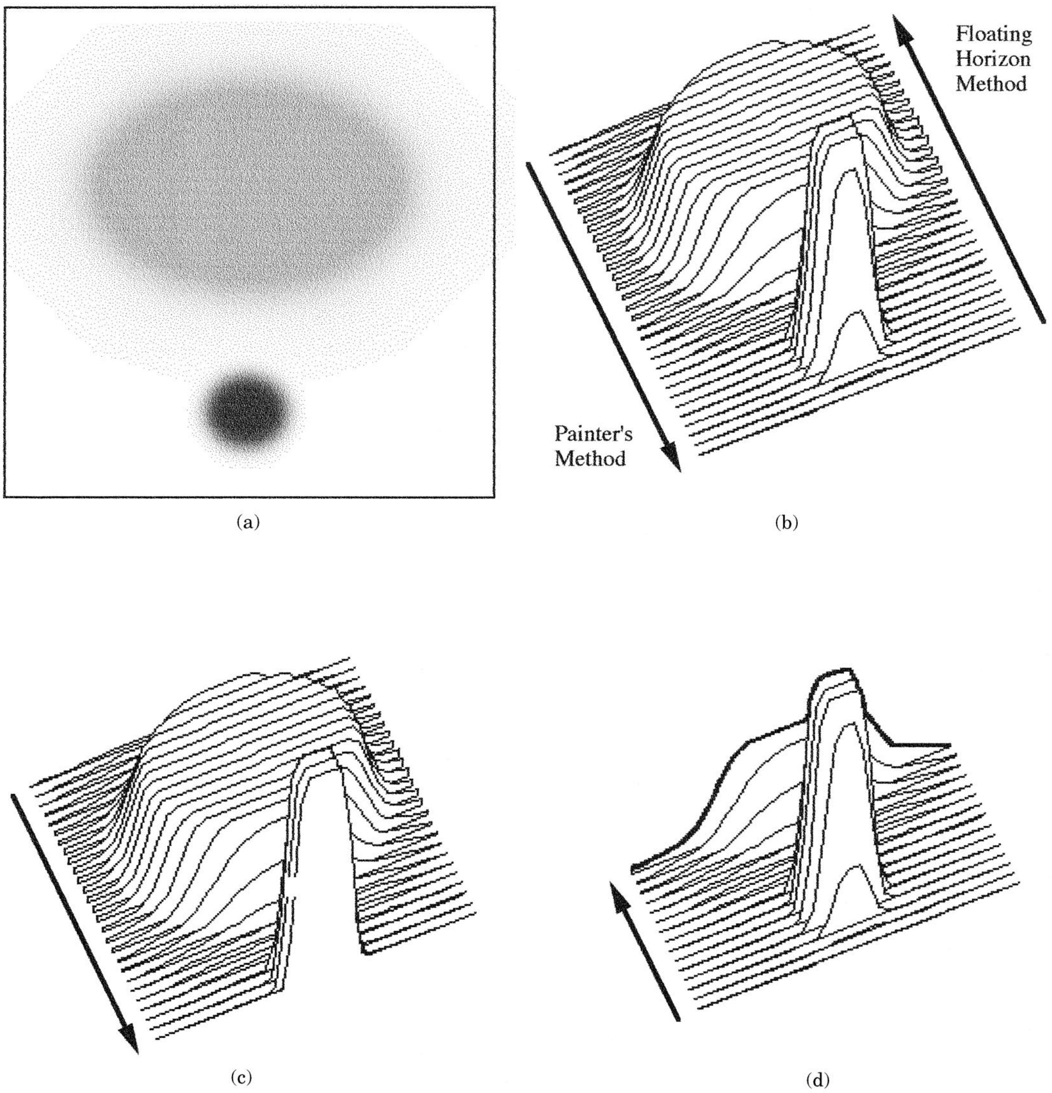

Figure 6. Point projection approaches. (a) Terrain image: brightness encoded elevation. Higher elevations are darker; (b) perspective view (every 8th row): profiles processed back to front for Painter's algorithm; profiles processed front to back for Floating Horizon algorithm; (c) Painter's algorithm: partially processed; (d) Floating Horizon algorithm: partially processed showing current location of floating horizon.

dated whenever a projected point falls above the current horizon. When texture pixels corresponding to different terrain elevation profiles contend for the same screen pixel on this horizon, the current one is usually ignored in favor of the previous one (associated with the closer profile to the eye point). Alternately, an area weighted averaging (*A*-buffer) technique may be used for each horizon screen pixel to store and blend values from texture pixels associated with more than one profile.

A disadvantage common to the point projection technique occurs when transforming terrain elevation and texture arrays for close-range viewing. In this case, without special techniques such as described by Fant [1986], undersampling in screen space will leave unfilled pixels. For nearly horizontal and low-altitude viewing conditions, this often occurs in the foreground part of the output image where the source data are magnified. The opposite situation or oversampling in screen space is a common occurrence in the background part of the output image or for overall distant viewing where source data are minified. In this case, many texture pixels are transformed to the same screen location. This is definitely inefficient unless multiresolution source imagery techniques are used. One advantage typical of the point projection technique is that ridgeline anti-aliasing is easily achieved with the floating horizon approach when it uses an *A*-buffer.

Tanaka's [1979] example described earlier used the painter's method. Junkin [1982] at NASA also has presented examples of point projections of LANDSAT images in perspective, but he used the floating horizon method. Other examples of the point projection method have been presented by Fishman and Schachter [1980] at General Electric and Smedley et al. [1989] at Lockheed. However, their examples transformed synthetic images generated by a terrain elevation diffuse shading algorithm. Smedley's examples used bathymetric elevation data. Bernstein et al. [1984] and Pareschi and Bernstein [1989] at IBM presented transformations of images of the San Francisco Bay area and of Mt. Etna, respectively, but projected them, orthographically rather than perspectively. Carlotto and Hartt [1989] at TASC presented orthographic projections of images of Mars using a variation on the floating horizon technique that they implemented in the form of a preprocessing step to identify hidden pixels. They obtained their elevation data using a shape-from-shading technique.

A novel orthographic point projection approach also based upon horizon information has been developed by Dubayah and Dozier [1986] at the University of California Santa Barbara. In this technique, the terrain elevation grid is preprocessed to compute an array whose intensity at each element represents the angle to the local horizon. During point projection, the elevation angle for a given terrain point is tested against this preprocessed horizon angle to determine whether it is visible. These authors also developed an accumulation technique to address the problem of unfilled screen pixels. Each registered terrain elevation and texture pixel is projected to the screen, but the coordinates are kept to floating point precision rather than simply truncated or rounded to the integers. Then a fraction of the color is assigned to each of the four closest screen pixels according to an inverse distance weighting scheme. Each screen pixel can accumulate fractional color from more than one texture pixel. Finally, the accumulated color at a screen pixel is normalized by the sum of the inverse distances accumulated for all the texture pixels that contributed to it.

Another point projection approach that mitigates hole-like artifacts has been presented by Petersen et al. [1990] at Brigham Young University. In their back-to-front approach, the complete elevation post is rendered to the screen by projecting both its top and bottom and subsequently connecting a line between the two points. Then each screen pixel along the line is assigned the color of the source texture pixel that corresponds to the elevation post.

A novel variation similar in concept to the 1D approach of Devich and Weinhaus has been developed by Robertson [1987] at CSIRO, but is based upon the point projection approach. Robertson used a combination of rotational and rhombic deformations of the texture image and terrain elevation array, rather than the Cartesian-to-polar one, to transform to a domain where the projection became one-dimensional (i.e., along rows or columns in the transformed domain). However, he used a back-to-front projection technique rather than a ray-tracing method to render the output.

3.3 Octree/Voxel-Based Approaches

Perspective transformations of images using octree encoded voxels also have been used. One finds both forward projections from the voxels to the screen and inverse projections from the screen to the voxels. The forward transformation technique is similar to the point projection technique previously described, except all nodes of the octree are projected at their appropriate levels of detail. Indexing and projection of the nodes of the octree may progress either from the back towards the front or from the front towards the back. The inverse transformation technique uses the ray-tracing concept, but must intersect and identify the appropriate node of the octree.

Not only does the voxel carry the 3D structure, but it also can be coded with source image color or other attributes about the model that would be useful for simulating views by such nonvisual sensors as infrared and radar. Furthermore, when multiresolution source imagery is used, a color can be assigned to each node of the structure using the pixel color at the corresponding image resolution level. The octree hierarchy allows the voxel approach to deal with vertical faces of urban 3D objects in addition to the surface of the terrain. However, unless very fine voxels are used, this technique will not represent sloped surfaces of urban models very smoothly.

Nack et al. [1989a, b] used a software-based ray-tracing approach and have presented perspective transformations showing the sides of individual buildings at high resolution as well as those of rural settings. Figure 7 is an example image generated by this type of ray-tracing approach using a column voxel representation for the terrain. On the other hand, Scuderi and Strome [1988] have presented voxel projection transformations of both rural and urban settings. These output images were generated on prototype hardware developed by Hughes Aircraft Co. However, their results for the urban environment also displayed the same striping artifacts on the sides of the buildings as was described earlier, due to the use of simple column voxels and vertical photography.

3.4 Polygon/Patch-Based Approaches

The traditional polygon and patch representation of 3D models has also been used to perform perspective transformations of images. In this case, the following techniques similar to those used for ordinary texture mapping can be applied: back-to-front or front-to-back depth-priority, depth-buffer, scan-line, polygon subdivision, and ray-tracing.

An example of an early polygon-based approach was Sun Microsystem's MAPVU demo [1989] which ran on their TAAC-1 accelerator. It processed a regular 3D mesh of triangles in a back-to-front (painter's) fashion. Each vertex of a triangle was assigned the color of the corresponding pixel in the image texture. Triangle vertices were projected to the screen and the color for each screen pixel within a projected polygon was interpolated using the traditional color method. A quality parameter controlled subsampling of the data. It permitted coarse 3D mesh and texture data to be used thereby increasing the rendering rates for rapid previews. However, this occurred at the expense of poorer textural detail, since no other source image data were used to fill the triangles.

Subsequent polygon-based approaches such as those of SGI and Stardent (now Kubota Graphics) could fill the interior of each triangle in a coarse mesh with high resolution texture from the source image. The approach taken in this case properly interpolated the texture coordinates for each projected triangular polygon from texture coordinates stored at the vertices. Then the interpolated texture coordinates were used to look up the color in the source image. A depth-buffer was used to remove hidden pixels.

In many high resolution, nonreal-time polygon-based terrain rendering appli-

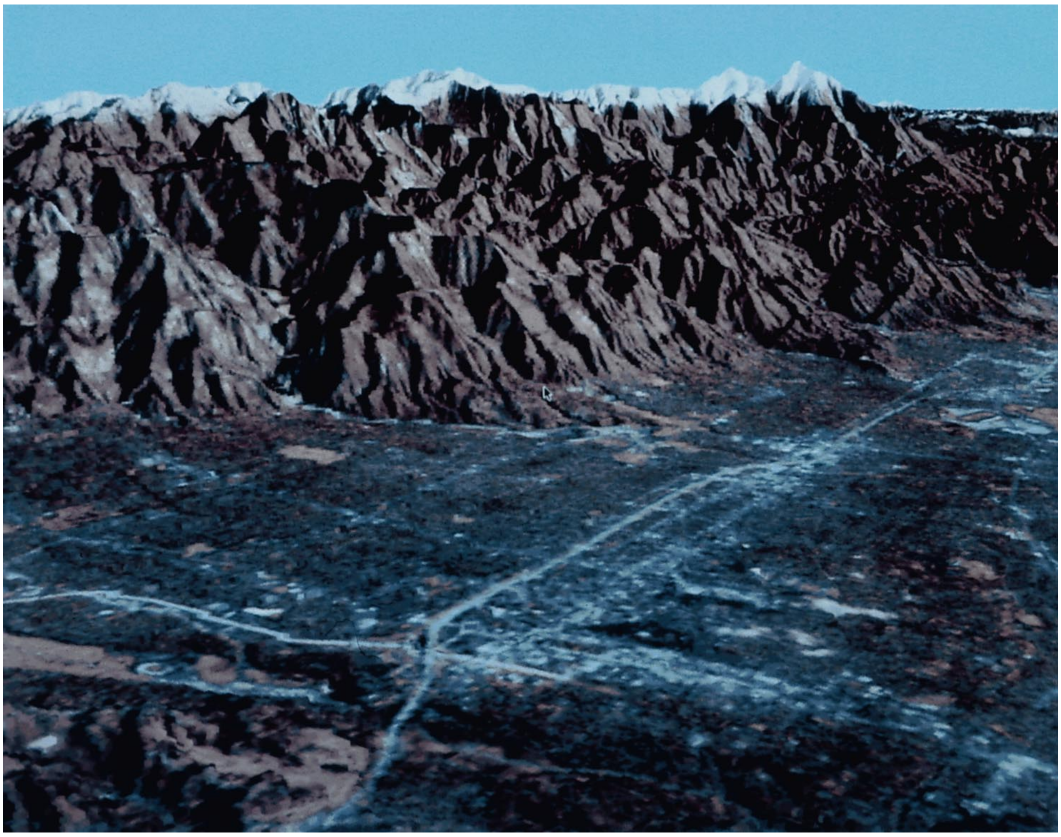

Figure 7. Perspective view of the area, around Pasadena, CA, computer-generated from coregistered 30 m LANDSAT Thematic Mapper imagery and voxel format terrain elevation data. Courtesy of Image Data Corp. (now Core Software Technology ©).

cations, the grid of elevation values is preprocessed to resample it to the same resolution as the (rectified) image. This is advantageous in the following ways. First, it means that texture coordinate interpolation is unnecessary, since there are no interior texture pixels to access. It suffices in this case simply to assign the colors of the image pixels to their corresponding mesh vertices and interpolate them across the polygon using the traditional color method. Second, if higher-order interpolation techniques are used in the terrain resampling process, then the terrain will display a smoother appearance, although not truly any more accurate than in its original representation. Other advantages associated with resampling the texture data and 3D model geometry to a common coordinate system have been discussed by Cook et al. [1987]. The main disadvantage is that the complexity (i.e., number of polygons in the terrain 3D model) rises dramatically. Thus, rendering can become extremely time consuming for large, high resolution data sets. Techniques that generate and use multiresolution terrain elevation data are therefore necessary to mitigate this problem.

Moreover, when the data sets are too large to be totally contained in memory, tiling (i.e., partitioning the data into blocks) also becomes important. For example, Blinn [1990] has explained tiling and paging strategies that he has used to perform efficient texture mapping of astronomical photographs of the planets onto spherical models.

Gelberg et al. [1988] and Hughes [1991] each used combined tiled and multiresolution (pyramid) approaches to generate TASC's Calgary video and Apple's Mars Navigator videos, respectively. Each started with a terrain elevation grid that was registered to a very large rectified image (36 million pixels in Gelberg's example and 145 million pixels in Hughes' example). Gelberg used a simple tiled pyramid approach and Hughes used the MIP map pyramid approach.

In a simple tiled pyramid approach [Tanimoto and Pavlidis 1975; Burt 1981], multiple versions of the source terrain elevation grid and/or imagery are created usually with a factor of two steps in resolution, stopping at a specified block size. Each version is then subdivided into blocks, reformatted in scan-line order and stored as a hierarchy of tiles, each containing one block of data. In the MIP map approach [Williams 1983], the source terrain elevation grid and/or imagery are first divided into blocks. A pyramid is then formed for each block, typically stopping when the size is 2×2 pixels. Finally all resolution versions of a given block are reformatted in scan-line order and stored successively in the same tile. In the simple tiled pyramid, the number of resolution levels is given by $(1 + \log_2$ (original image size/block size)). In the MIP map pyramid, the number of resolution levels is given by $(1 + \log_2$ (block size/2)).

An advantage of the MIP map approach over the simple tiled pyramid approach is that all resolutions of the data are always readily available for any MIP map tile in memory. This is especially convenient when it is desired to blend data from two successive resolutions so that sudden transitions in resolution are not detected. When used in conjunction with bilinear resampling, this technique is commonly referred to as trilinear interpolation. On the other hand, a disadvantage is that memory is less efficiently allocated with the MIP map approach. This occurs because the higher resolution data also must be carried along for those MIP-mapped tiles corresponding to the background areas of the output scenes where only low resolution data are needed. When memory is limited, more tile paging is required due to the smaller amount of low resolution data available in any one tile. A compromise approach might be to store a sequence of MIP maps for each block of the image where each MIP map contains only two successive resolutions. However, this would increase the total data load from a factor of 4/3 times the original image and/or terrain elevation array size to 5/3 times the original image and/or terrain elevation array size.

In Hughes' approach, only those terrain elevation tiles that were found to lie at least partially within the output image's ground "footprint" (projection onto the X, Y ground plane) were rendered. The terrain elevation model and imagery resolutions were selected as a function of distance between the terrain elevation tile and the eye point so that coarse resolution could be used for more distant regions. Terrain elevation tiles were then tessellated into triangular meshes at the specified resolution. Then color values from the corresponding resolution imagery pixels were assigned to each triangle vertex. An SGI computer was used to perform the rendering by simply interpolating the color values across the triangles and by using a depth-buffer to remove hidden pixels.

In Gelberg's approach, terrain elevation tiles were processed by ordering them according to distance in a back-to-front manner after eliminating those tiles that were not at least partially within the viewing frustum of the output image. The appropriate resolution data were determined from the ratio of the number of quadrilateral polygons to the number of screen pixels. A Pixar computer was used to perform the rendering using the ChapReyes software [Cook et al. 1987]. This software subdivides each 3D model primitive into quadrilateral-shaped planar micropolygons and renders them using a depth-buffer to remove hidden pixels.

Kaneda et al. [1989] in Japan also generated perspective transformations of rural settings using a quadrilateral mesh to represent the 3D model of the terrain. Their algorithm used a back-to-front depth-priority technique for rendering. However, they used a specially preprocessed, radially organized quadrilateral mesh rather than one in a rectilinear format. This was done to generate uniformly sized quadrilaterals in screen space. Fewer quadrilaterals are needed this way, since they are correspondingly larger in object space as they progress farther into the distance. These authors also created multiple resolution versions of the vertically oriented (aerial) image and selected the appropriate resolution to use for each quadrilateral so that high resolution texture was not wasted for the background regions of the scene.

Another quadrilateral-based approach has been presented by Whiteside et al. [1987] and Whiteside [1989] at DBA, but it used a combination of depth-buffer and accumulation-buffer techniques for rendering. In this implementation, more than one source image was used to render the area of interest, but they were first orthorectified and mosaicked to a vertical format. Multiple resolution versions of the mosaicked image were created subsequently and used as required to avoid oversampling of the input texture. This was achieved by projecting each 2×2 set of terrain elevation posts into input (texture) space and screen space and selecting the texture resolution that nearly matched the quadrilateral areas in the two domains. Then a forward projecting, bilinear transformation rather than a true perspective transformation was used to map the rectangular-bounded input texture areas to quadrilateral-bounded output screen areas. Screen-space anti-aliasing was nicely achieved for the oversampling case using an accumulation buffer. (No discussion of the undersampling case was presented. However, the resampling scheme described by Fant [1986] should be applicable to the approach advocated by these authors and would mitigate this problem.) These authors also were able to render 3D cultural features with rectangular faces, such as buildings. The textures for these faces, however, had to be cropped from the images and geometrically preprocessed into face-on views before they could be used.

In general, quadrilaterals are especially difficult 3D primitives to use, because they are not necessarily planar. Even when their edges are constrained so that they form bilinear surfaces, they may still present ridgelines for a particular view. Thus, back-facing portions of the surfaces must be taken into account if they are to be handled rigorously. Generally, triangles (and planar quadrilaterals) are much easier to deal with, because testing for backfacing conditions need only be done once per surface rather than at every interior pixel. It has therefore been necessary when dealing with nonplanar quadrilaterals to use approximations and other special techniques, such as polygon subdivision, in order to avoid these difficulties.

3.5 Approach Comparison

Table 3 presents a comparison of the various categories of approaches that have been discussed. The approaches have been categorized as mentioned earlier by the geometry primitives and rendering methods used. The comparison identifies the major advantages and disadvantages of each category. Too many algorithmic variations within categories have been presented to go into details for each. The advantages and disadvantages that have been presented here focus on issues such as resulting quality (e.g., aliasing), processing efficiency, and the ability to handle urban (man-made) as well as rural (natural terrain) features.

Table 3. Algorithmic Comparison

Geometry	Rendering	Advantages	Disadvantages
Height field	Painter's	● Algorithmic simplicity	● Minimal antialiasing ● No handling of (urban) vertical faces
Height field	Floating horizon	● Good ridgeline antialiasing	● Minimal other antialiasing ● No handling of (urban) vertical faces
Height field	Ray trace	● Efficient handling of large terrain sets ● Amenable to speed optimizations ● Amenable to parallelization	● Minimal to moderate antialiasing ● Poor handling of (urban) vertical faces
Voxel	Ray trace	● Can handle (urban) vertical faces ● Amenable to speed optimizations ● Amenable to parallelization	● Geometry data base increases substantially to handle (urban) objects
Polygon	Depth buffer	● Good antialiasing using multires data ● Handles urban & terrain objects equally	● Limited parallelization

4. IPT SYSTEMS

4.1 Nonreal-Time Systems

The image perspective transformation capability is relevant to a number of disciplines: urban planning, architecture, law enforcement, industrial visualization, and military training to name just a few. When used, for example, in mission planning and rehearsal systems, IPT techniques must accurately portray the real-world scene. Any distortions in the transformed images, such as inaccurately placed doorways on building faces or misplaced roads on the terrain, cannot be tolerated. Furthermore, these systems must be easy to use and require as little manual intervention as possible. For mission planning systems, real-time scene generation is desirable, but not mandatory. Consequently, increased output scene generation times may be traded off against higher output scene quality and accuracy.

Although much research has been conducted towards fully automated extraction of 3D models of cultural features[2] this task still remains primarily a manual one. However, it can be made user-friendly in several ways. The first way is to use a modern man–machine interface that permits the modeler to construct 3D models interactively and simultaneously displays them as overlays on the source images. The second way is to use computer-assisted or semiautomated techniques to extract 3D models from the imagery [Mueller and Olson 1993]. The third way is to eliminate potentially unnecessary preprocessing operations, especially those that can be folded into the basic projective geometry transformations used during the rendering. These include: orthorectification and mosaicking of ground coverage images, resampling of the terrain elevation data to match the imagery resolution, and rectification of images of the sides of urban features into face-on views. Such rectifications introduce potentially unnecessary resampling of the imagery that, in addition, degrade the

[2] See, for example, Herman and Kanade [1986], Irvin and McKeown [1989], Walker and Herman [1988], and DARASISTO [1992, 1993].

results. Several nonreal-time, workstation-based systems have made advances in these areas. These include systems developed by TRW/ESL, General Dynamics Electronics Corp., SRI, and the University of California at Berkeley.

TRW's system renders the output scenes by computing texture coordinates for every screen pixel using an incremental scan-line method to evaluate the fractional linear perspective transformation of Equation (1). This transformation is able to account for the oblique geometry of the source texture images as well as the viewing perspective associated with the output. The articles by Devich and Weinhaus [1980, 1981a] and Weinhaus and Devich [1997] showed how to derive the fractional linear transformation coefficients for a planar polygon independent of the number of its vertices without the need to store u, v (or w) texture coordinates for each polygon. The only information needed is the camera parameter model for the texture image and the equation of the plane for the polygon. In fact, the four coefficients defining the plane need not be carried with the polygon, since they can be generated from the world coordinates of the vertices.

In this system, none of the imagery, either for the texture for the ground covering or for the sides of urban features, has to be rectified or mosaicked. A simple tiled pyramid is created for each source image and the appropriate tile (or tiles) is automatically selected to cover each surface at an appropriate resolution for anti-aliasing. A deferred texturing, depth-buffer technique is used and various resampling techniques may be selected to trade quality versus speed, including supersampling and/or EWA on a pyramid. The EWA technique [Greene and Heckbert 1986] is a very good method for anti-aliasing, since it samples the input data within an elliptical "footprint" region that represents the projection of a given (circular) output pixel. The size and orientation of the ellipse adapts to the geometry and depends upon the orientation and location of both the input and output cameras and the orientation of the 3D model surface upon which it is projected.

The terrain elevation data, which originate from a regular grid of elevation values, are automatically reformatted into a multiresolution triangular surface mesh for each output frame. This allows coarser triangles to be used in the background of output images and finer ones to be used in the foreground. However, before rendering any frame, surfaces from two levels are blended. This prevents sudden transitions in level of detail from being observed in motion sequences. Urban features are modeled by interactively adjusting "elastic" wireframe models on one or more source images simultaneously. Figures 8 and 9 show examples of an image perspective transformation for an urban and rural area, respectively produced by this system using the polygon representation for the terrain elevation model and the cultural objects.

The SOCET SET system, built by General Dynamics Electronics Corp. [Inman and McMillan 1990] uses stereo image pairs as the source of its 3D information. A combination of automatic stereo compilation followed by manual editing can be used to generate the terrain elevation model. The construction of 3D cultural models is performed interactively on this system using a stereoscopic monitor for visualization or by using semi-automated techniques [Mueller and Olson 1993]. For example, the vertices of the top of a rectangular parallelepiped-shaped building can be identified manually with a "floating" cursor. Then a wire-frame graphic is draped automatically about the boundary of the building with the bottom portion "buried" below the ground.

The Cartographic Modeling Environment [Hanson et al. 1987; Hanson and Quam 1988], developed by Hanson and Quam at SRI, was created to support research on interactive, semi-automatic, and automatic computer-based cartographic activities. This system uses

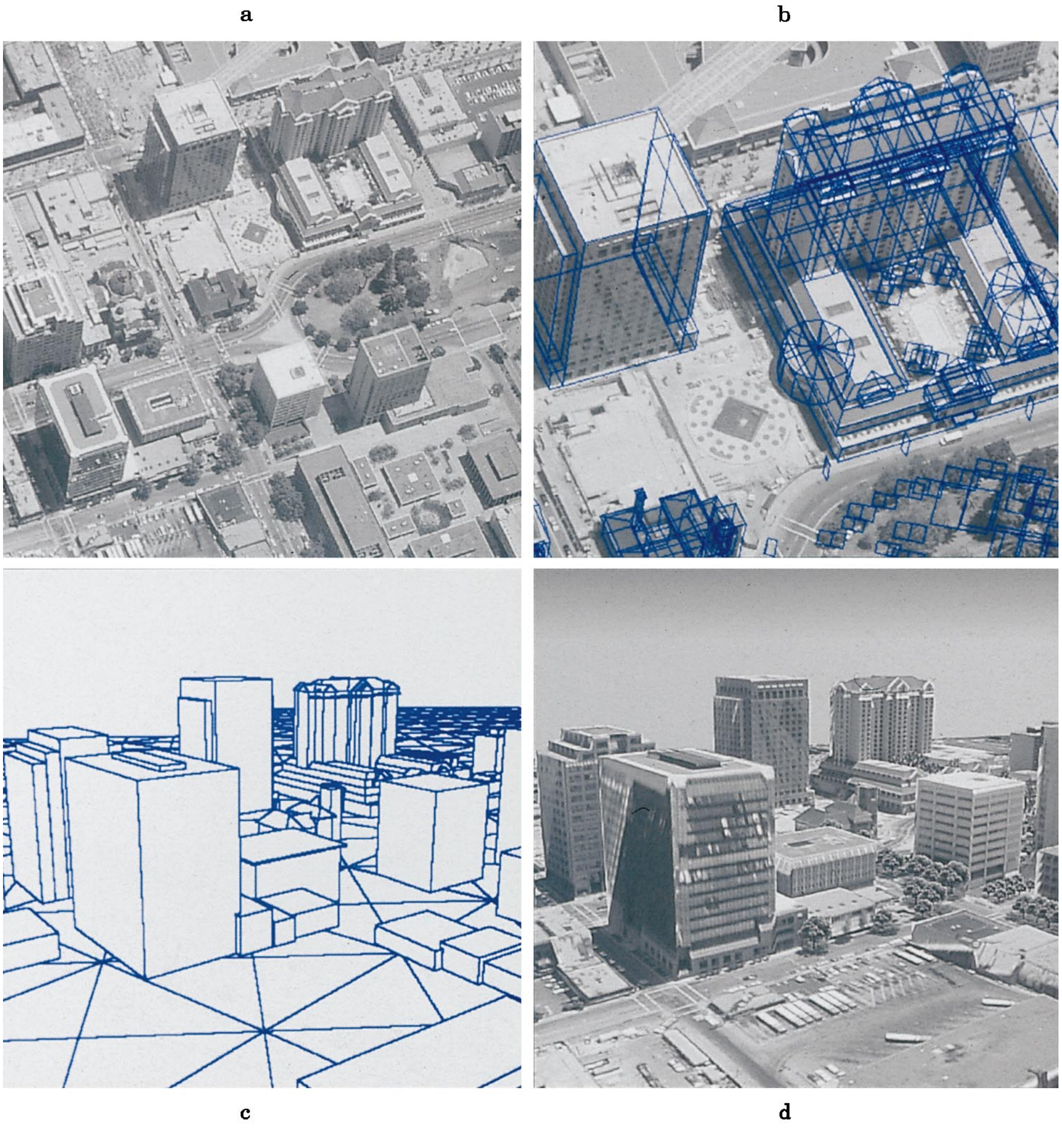

Figure 8. Computer-generated perspective transformation of downtown San Jose, CA. Perspective view generated by ESL (now TRW Sunnyvale, CA). (a) high oblique aerial source image (1 ft resolution); (b) subsection of the source image with 3D models overlaid; (c) low oblique wire-frame perspective view of 3D models; (d) low oblique image perspective transformation output. © TRW, Inc.

MIP-mapped imagery to cover the terrain and urban models. Gridded terrain elevation data are converted into a multiresolution mesh of triangular surfaces; however, before rendering any frame in a motion sequence, elevation data from two levels are blended to present a smooth transition. Terrain rendering is achieved using a floating horizon method similar to that described in Anderson [1982] or Wang and Staudhammer [1990] in combination with an *A*-buffer. A depth-buffer also is maintained so that cultural models may be rendered subsequently.

This system has a very extensive

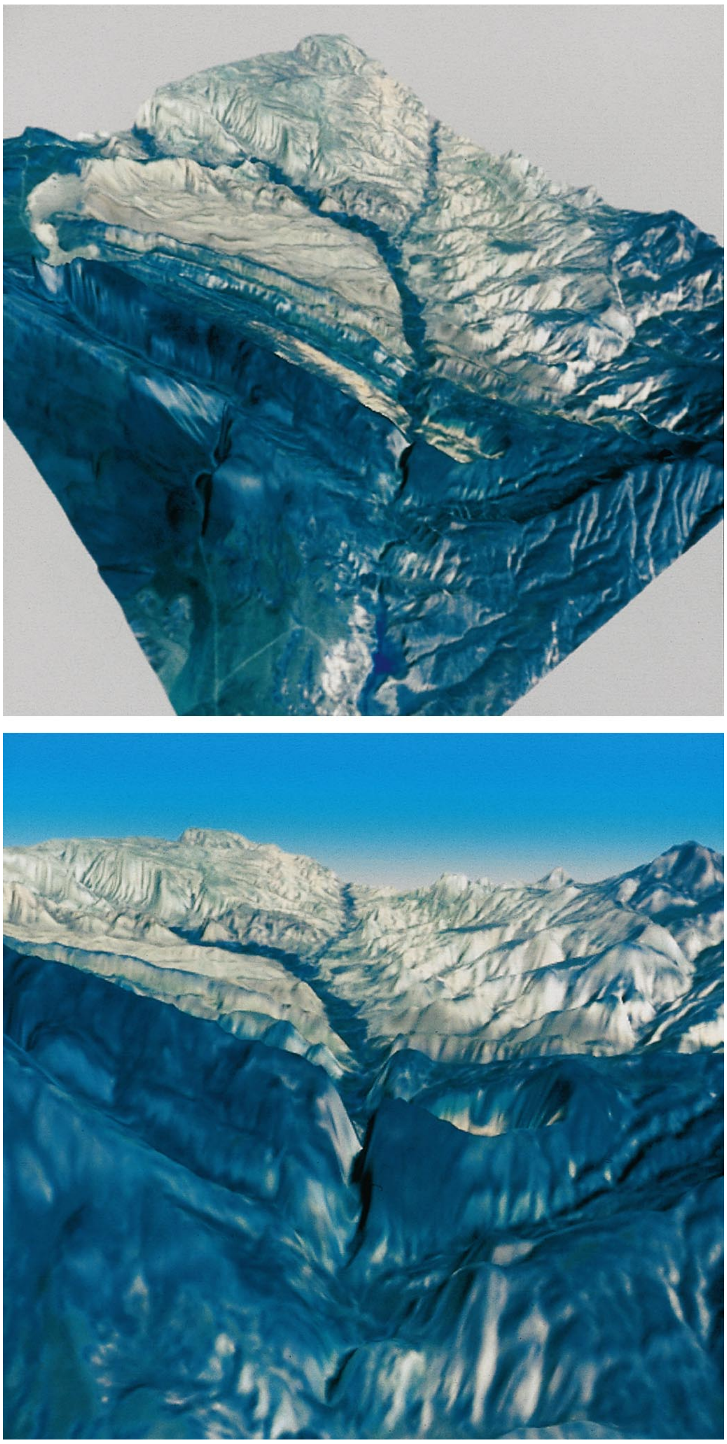

Figure 9. High and low oblique, computer-generated perspective views of the area around Irish Canyon, CO. Source data were composed of 20 m SPOT imagery and polygonalized 10 m terrain elevation data that were vertically exaggerated by a factor of 5. Source data courtesy of STX Corp. Perspective views generated by ESL (now TRW Sunnyvale, CA).© TRW, Inc.

menu of interactive modeling and manipulation tools. For example, wireframe models may be rotated about world axes, about any object edge, or about a line from an object vertex to the sun's position. It may even be moved along the ray from the eye point to the object. Shadow mensuration is an option that can be used to help in 3D model construction. Also, shadows can be rendered into a graphic overlay on the transformed view. Semi-automated 2D feature extraction tools, such as road following and the detection of boundaries of buildings whereby the user starts the extraction process with a manual cue, also are included. SRI also has experimented with automatic extraction of 3D models of buildings and used their system to portray the results [Fua and Hanson 1989].

The University of California at Berkeley's Facade System developed by Debevec et al. [1996] was designed to model and render architectural scenes from multiple photographic inputs. Its objective is to create high quality outputs of a small number of detailed architectural structures rather than large areas of rural and/or urban content. In this system, 3D models composed of parametric primitives such as blocks, pyramids, and the like are manually composited to approximate the geometry of the architectural structure using the imagery as a visual guide. The focus is on linking the right shapes together without putting significant effort into setting the exact size and orientation. Then a few corresponding edges are located in the source imagery and on the 3D models. A nonlinear least square fit is then performed to recover simultaneously the parameters (size, orientation, etc.) of the primitives in the 3D architectural model as well as the camera location and orientation parameters for each of the source images.

A novel view-dependent texture-mapping technique is used to render the architectural model. Multiple photographs are projected onto the model in order to texture its surface completely. (An image-space shadow-mapping algorithm based on a z-buffer is used to keep track of obscurations.) However, since the photographs overlap, the rendering algorithm must decide which photograph or photographs to use at each output pixel. Here, a weighted average of the textures from the overlapping images is used, where the weights are the angular deviations of the viewing vectors of each source image from that of the output view. Moreover, to avoid visible "seams," the weights are ramped near the boundaries of source photographs. Optionally, a model-based stereo correlation algorithm may be used to refine the 3D model to include finer detail such as recessed windows, and the like, and to increase the fidelity of the texture-mapped rendering. To achieve this result, the base 3D model is first used to create an output view similar to the orientation of one of the existing source images. These two images are then used as input to the automated stereo correlation algorithm. The stereo correlation produces a disparity image that can be converted into a depth map which in turn is used as a refined z-buffer for re-rendering the output view.

4.2 Real-Time Systems

In contrast, the training industries and in particular the commercial and military flight and ground vehicle simulation industries have traditionally required real-time scene generation rates in their visual systems and, historically, have traded off quality and realism to achieve these rates. These visual systems produce interactively controlled moving images that depict on the simulator display what the pilot or driver might see if he or she were looking "out-the-window" or at a monitor for a sensor mounted on the vehicle. In the interest of speed, both data and algorithms are trimmed to the bare-bones level for use by such systems. The development of this technology historically took two different paths, direct and hybrid. The direct approaches were simply real-time implementations of the previously described polygon-based computer graphics techniques. The hybrid approaches merged computer graphics techniques with prestored digital images. However, both approaches ultimately converged on the use of certain fundamental procedures, notably the use of digital orthophoto mosaics and registered digital terrain elevation data.

In the early 1980s, researchers at Honeywell [Scott 1983; Erickson et al. 1984; Baldwin et al. 1983] developed one hybrid approach that they called the Computer Generated Synthesized Imagery (CGSI) technique. This approach combined the best qualities at the time of the Computer Graphics Imagery (CGI) technique and the Computer Synthesized Imagery (CSI) technique. The CGI technique, namely, coloring and illumination-based shading of polygon models permitted full freedom of motion through the gaming area or database. However, it lacked realism. The CSI technique used high resolution digitized photographs as backgrounds, for example, for artillery training with computer-generated target models moving around the scene. It attained a high degree of realism, but basically was limited to a single viewpoint, although some motion could be simulated by panning and zooming the background image. The CGSI technique combined the full freedom of motion of the CGI technique with the realism of the CSI technique. This was achieved by using the computer graphics-generated imagery as the framework and inserting digitized photographs of real-world objects, such as buildings, trees, and targets at the appropriate locations in the database. Multiple views of these objects were photographed, digitized, and prestored on optical video disks. During real-time image generation, when a particular object was needed, the most appropriate view of it was retrieved and warped to proper perspective according to Equation (1) for the specific output image. Then it would be inserted into the previously computed CGI background with proper occlusions and border blending. In effect, this combination of image warping and insertion is equivalent to texture mapping the image onto a planar 3D rectangle that has been placed in the database at the desired location and orientation and subsequently viewing it from the desired eye point.

Meanwhile, researchers at LTV Missiles and Electronics [Hooks and Devarajan 1981a] (now Lockheed-Martin-Vought Systems (LMVS)) developed another type of hybrid approach that they called the "pseudo-film" approach. This monochrome image technique interpolated new output views at real-time rates from a small set of specially collected, oblique perspective photographs. These photographs were acquired at locations corresponding to a 3D matrix of eye points throughout some area of interest in the real world. Before real-time interpolation by the image generator part of the system, these oblique perspective photographs were converted into large (typically thousands of pixels on a side) electronic images using a high resolution, digital scanning process and then were stored on analog video disks. When the real-time simulation was started, the system would know at any given time what the aircraft position and attitude were. It would then retrieve the most appropriate oblique perspective image stored in the video disk bank and then warp it according to Equation (1) to simulate the pilot's or sensor's view of the gaming area at that instance in time. Since these operations were accomplished in real-time, it would give the illusion of smooth interactive motion through the gaming area. An improved version of this system using color imagery and displays as well as multiple output channels has been described by Hooks and Devarajan [1981b] and by Mudunuri and Hooks [1985].

The warping operation used to interpolate between successive oblique source images in the LTV simulations (on their TOPSCENE Model 3500) consisted primarily of zooming, panning, and 2.5D perspective changes. The zooming operation approximated forward motion, the panning operation approximated lateral motion, and the 2.5D perspective changes allowed smooth transitions between successive input oblique images. Since the oblique input images were acquired in proper perspective, they were constructed to have correct elevation information built in for both the terrain and urban features. Therefore, the process of warping the source image according to Equation (1) to achieve limited changes in the perspective of the output views was assumed to result in a good approximation of the true perspective view for the given output eye point. As the simulation proceeded through the gaming area, the system would make the transition at the appropriate time from using one oblique source image to the next one in sequence. The warping process used here is again equivalent to texture mapping the source image onto a single planar 3D surface, in this case, the oblique imaging plane of the source camera, but seen from a slightly different eye point than for that of the original image.

Eventually, the researchers at LTV [Devarajan and Chen 1986; Devarajan 1989] generated the large oblique "pseudo-film" perspective images from mosaics of overhead photographs rather than by collecting actual oblique photography. Likewise, the researchers at Honeywell [Hopkins and Cooper 1988; Mudunuri 1989] (now Hughes Training Inc.) eventually replaced the background imagery generated by the CGI technique with a sequence of computer-generated oblique "viewplane" images similar in nature to LTV's computer-generated "pseudo-film" images. A sequence of these "viewplane" images typically was generated to represent one or more "corridor" approaches to a target area. The overhead source images in both applications were first orthorectified to remove any terrain elevation relief distortion and then mosaicked together to be in registration with the terrain elevation array. The terrain elevation array was then reformatted into a mesh of triangles. It would be used in this form along with the registered orthophoto mosaic to generate the requisite oblique "pseudo-film" or "viewplane" images for a desired gaming area. The rendering techniques used to create these oblique images were similar to those described in the preceding sections for nonreal-time, polygon-based, perspective scene generation. Since this rendering was performed by a nonreal-time preprocessing step, there was no need to limit the number of polygons used to represent the terrain nor were there any restrictions as to which anti-aliasing techniques could be used. Consequently, these oblique perspective images suffered no appreciable loss of quality or realism.

In the late 1970s and early 1980s, another set of researchers, notably from General Electric and Evans & Sutherland, started developing direct techniques for real-time rendering of polygonal models of the terrain and urban features. These techniques were based upon the tools evolving from the computer graphics industry. In the beginning, a very limited number of polygonal surfaces were either simply assigned colors or were flat shaded according to an illumination model. Strict limits on the number of polygons that could be processed in any one output scene had to be maintained. In order to keep the polygon load to a minimum, these systems typically represented the terrain using a mesh of irregularly sized triangles, called a triangulated irregular net or TIN. These TINs were often generated using the technique of Delaunay triangulation [Lee and Schachter 1980] or by a hierarchical subdivision technique [Bunker and Heartz 1976] starting from a regular grid of elevation values. The objective was to use larger triangles where the terrain was generally smooth and finer triangles where the terrain undulated more rapidly. A summary of many of these early flight simulator systems and techniques has been presented by Schachter and Ahuja [1980], Schachter [1981], and Yan [1985].

As both hardware and computer graphics technologies matured, the number of polygons processed in an output scene has increased to represent more faithfully the undulations of the terrain. Also, the flat shading technique was replaced by Gouraud and Phong interpolated shading techniques that smoothed over and disguised the faceted appearance of many of the 3D models. Eventually, the demand for more realism resulted in the application of synthetic and photo-derived textures. Some of the techniques used to generate the synthetic texture maps as well as the ways these texture maps were used to achieve various simulator effects have been summarized by Robinson and Zimmerman [1985]. For the photo-derived case, nongeographic-specific texture maps of typical terrain cover such as grassland, forests, marsh, and deserts were created from actual photographs of such areas. These generic texture maps were then "painted" onto the surfaces of the terrain polygons when needed to represent similar areas of the world. Later, actual photographs of geographic-specific regions were used to create real-time perspective fly-throughs of corresponding real-world regions. Usually, these photographs had to be orthorectified, mosaicked together, and registered with the terrain elevation model before they could be used. Similarly, when 3D models of real-world urban features were to be included, most of the time specially acquired face-on photographs of each side of the object were used. Alternately, the texture patterns for the faces of the 3D models were extracted from oblique photographs and rectified into face-on views. Examples of this technology have been presented by General Electric [Economy et al. 1988], Evans & Sutherland [Geer and Dixon 1988], Star Technologies [Rich 1989] (now AAI Visual Systems), Thomsen CSF [Allain and Boidin 1986], and Boeing [Ross 1990].

Some of today's more advanced visual systems, such as Evans & Sutherland's [1990] ESIG-4500 [Cosman et al. 1990; Abascal 1996], Lockheed-Martin's COMPUSCENE VI [Abascal 1996], and Flight Safety International's VITAL VIII [Abascal 1996], generate images for cockpit display windows using multiresolution, geographic-specific digital imagery. However, these real-time systems still make compromises. Although many of these systems use supersampling or trilinear interpolation, aliasing artifacts, nonetheless, may occasionally be observed even with the use of multiresolution data. This aliasing is primarily due to the difference in longitudinal (along the viewing direction) and transverse (across the viewing direction) resolution requirements associated with highly oblique output perspective views of the orthophoto mosaic. This type of aliasing can be dramatically reduced when techniques such as elliptically weighted averaging are used for adaptive resampling; however, the computation burden associated with this technique currently is too high to maintain real-time rates. Nevertheless, to the users familiar with generic photo-derived textures or monoresolution imagery, the use of multiresolution data is a significant step forward.

The data rates and memory requirements of the multiresolution digital orthophoto mosaic approach have been described in a paper by Hooks et al. [1990]. This paper assumed that, for a given output image eye point and field-of-view, all relevant pixel data from the orthophoto mosaic would be available at the appropriate resolution so that the resulting system was display resolution-limited. The authors presented formulae that characterized the data rates and memory requirements of such a system as functions of the number of different resolution levels used, the scale factor between successive resolution levels, and the field-of-view corresponding to the output image. The calculations presented in this article identified two very interesting facts: the optimum scaling factor between successive resolution levels is about 1.12 rather than 2; and simulations for sensors with narrow fields-of-view (i.e., a telephoto lens) and high slew rates place considerably more severe demands upon these real-time systems with respect to resolution, memory, and I/O bandwidth than do those for out-the-window displays.

A variation of the "pseudo-film" technique has been used by LTV to overcome this latter problem and to mitigate the omnidirectional filtering problem. In this case, the "pseudo-film" images were created from the orthophoto mosaic to simulate a full 360-degree horizontal and 100-degree vertical field-of-view rather than one more typical of a planar central perspective image, namely, about 70 degrees in both dimensions. During their generation, these fish-eye-view images were also prefiltered to have roughly the same resolution in both transverse and longitudinal directions. The real-time system then used the fish-eye-view "pseudo-images" in conjunction with the polygonalized terrain elevation model to interpolate the output images corresponding to the narrow field-of-view sensor's display. This type of hybrid technique allowed full freedom of flight around the area of interest, produced high quality output images due to the off-line antialiasing, and maintained high data throughput rates associated with rapid slewing of narrow field-of-view sensors.

As mentioned earlier, the workstation-based nonreal-time rendering systems and the entirely custom-built high performance real-time systems have been inching towards each other to the point where there no longer is a huge void of capability between the two. This is primarily due to SGI's series of high performance CPUs and graphics engines and to the Pixel-Planes/Pixel-Flow technology [Fuchs et al. 1985; Molnar et al. 1992] developed at the University of North Carolina and commercialized by Division Ltd. and IVEX Corporation. For instance, an SGI Onyx system (preferably with multiple CPUs) and a Reality Engine-2 or Infinite Reality graphics engine can provide real-time rendering of a high resolution mosaic that is stored in high-speed texture memory. This has been demonstrated by several third party companies, most notably by Cambridge Research Associates, Gemini Corp., Autometric Inc., and LMVS (on their TOPSCENE Model 4000). On the other side, Evans & Sutherland and Lockheed-Martin Co. (Orlando, FL) have been engaged in developing smaller and leaner versions of their custom-built Image Generators (IGs). For example, Lockheed-Martin's REAL3D, which is a derivative of their COMPUSCENE product, is now being used by the computer gaming industry. Also, Evans & Sutherland is now offering a workstation-based system called iNTegrator. This system provides optional real-time rendering running on an Intel Pentium when accelerated by their custom Harmony IGs.

5. CONCLUSION

The technology to generate arbitrary views of real-world areas from photographic information has evolved from both the computer graphics and flight simulation industries. Two decades ago representations of real or fictitious scenes were generated using simple wire frames. Then illumination-based shading techniques were developed that over time have evolved to create increasingly more realistic effects. This was followed by the introduction of texture mapping—a technique that adds realistic-looking synthetic or even photographic texture patterns to faces of objects. Over a decade ago, the first true perspective transformations of images were published that completely texture mapped every 3D model in the scene with actual photographic information about the real-world model to which it corresponded. The popularity of this technology has grown dramatically over the last few years. Low-cost workstation-based systems exist today that can present moderate quality 2.5D transformed views of natural terrain environments in near real-time (approximately one frame per second). This is achieved using modern, general-purpose processors and efficient software, but limited texture information must be specially preprocessed to a vertical format, mosaicked together, and then stored in memory. Other low-cost, workstation-based systems are available that can create high quality 2.5D views for arbitrary mixes of natural terrain and cultural models. They can handle extremely large amounts of imagery with minimal preprocessing, but they are typically slower, since these data must be transferred from disk storage. Furthermore, the algorithms must be general enough to handle urban models as well as the terrain and therefore are not quite as efficient. Commercial and military flight simulators as well as some high-end workstations can achieve real-time perspective transformation rates for mixed rural and urban areas, because they use special-purpose parallel hardware and make compromises with respect to algorithms and data loads.

Rapid advances are currently being made in commercial computer technology in the areas of computational speed (e.g., superscalar chips) and texture memory capacity. Advances in disk and bus technology, however, have progressed at a slightly slower pace. Nevertheless, some advances have been achieved in the areas of formatting and storage of data for efficient processing. For example, multiresolution data and special tiling and paging strategies have become increasingly popular and necessary for enhanced rendering times. Other advances have been made in disk and disk controller technology. For example, disk arrays and parallel transfer disks (the so-called real-time disks) are available and have been used for several years on image processing systems. It is inevitable that all these limitations eventually will be solved and low-cost, well integrated, real-time, IPT workstations based upon commercial technology will be available, perhaps even within the next few years.

ACKNOWLEDGMENTS

The authors would like to thank the following people for providing information about their algorithms and systems: Kurt Akeley, Dorothy Baldwin, Avi Bleiweiss, Tim Butler, Nick Dalton, Paul Debevec, Kay Edwards, Jay Feuquay, Robert Fraser, Larry Gelberg, Carl Graf, Stan Hack, Wyndham Hannaway, Glen Johnson, Don McArthur, Chris Migdal, Kris Mudunuri, Myron Nack, Lynn Quam, H.B. Siegel, Dan Stanfill, Steve Wilson, Herman Towles, and Patrick Wong. Unfortunately, due to length limitations, not all the provided information could be included. We express our apologies to those whose information was not included. The First author especially thanks Robert Devich for motivating this work.

REFERENCES

ABASCAL, R. 1996. *The 1996 Image Society Resource Guide and Image Generator Survey.* The Image Society, Inc., PO Box 6221, Chandler, AZ 85246-6221.

ALLAIN, G. AND BOIDIN, P. 1986. VISA 4: A computed image generator for ground training. In *Proceedings of the Eighth Interservice/Industry Training Systems Conference* (Washington, DC, Nov. 18–20), 70–77.

ANDERSON, D. P. 1982. Hidden line elimination in projected grid surfaces. *ACM Trans. Graph. 1,* 4 (Oct.), 274–288.

AOKI, M. AND LEVINE, M. D. 1978. Computer generation of realistic pictures. *Comput. Graph. 1,* 149–161.

BAKER, J. R. 1977. Linearized collinearity equations for scanned data. *J. Surveying Mapping Division, Proc. Am. Soc. Civil Eng. 103,* SU1 (Sept.), 25–35.

BALDWIN, D. M., GOLDIEZ, B. F., AND GRAF, C. P. 1983. A hybrid approach to high fidelity visual/sensor simulation. In *Proceedings of the International Conference on Simulators,* Conference Publication 226, Institution of Electrical Engineers, London and New York, 1–6.

BARR, A. 1984. Decal projections. *Course 15: Mathematics of Computer Graphics, SIGGRAPH Conference, ACM SIGGRAPH* (July).

BERNSTEIN, R. 1976. Digital image processing of Earth observation sensor data. *IBM J. Res. Dev.* (Jan.), 40–57.

BERNSTEIN, R., LOTSPIECH, J. B., MYERS, H. J., KOLSKY, H. G., AND LEES, R. D. 1984. Analysis and processing of LANDSAT-4 sensor data using advanced image processing techniques and technologies. *IEEE GE 22,* 3 (May), 192–221.

BIER, E. A. AND SLOAN, K. R. 1986. Two-part texture mappings. *IEEE Comput. Graph. Appl.* (Sept.), 40–53.

BLINN, J. 1990. The truth about texture mapping. *IEEE Comput. Graph. Appl.* (March), 78–83.

BLINN, J. F. 1992. Hyperbolic interpolation. *IEEE Comput. Graph. Appl.* (July), 89–94.

BLINN, J. F. AND NEWELL, M. E. 1976. Texture and reflection in computer generated images. *Commun. ACM 19,* 10 (May), 542–547.

BORGEFORS, G. 1986. Distance transforms in digital images. *Comput. Vis. Graph. Image Process. 34,* 344–371.

BUNKER, W. AND HEARTZ, R. 1976. Perspective display simulation of terrain. Tech. Rep. AF-HRL-TR-76-39, Defense Technical Information Center, Cameron Station, Alexandria, VA.

BUNKER, M., ECONOMY, R., AND HARVEY, J. 1984. Cell texture—Its impact on computer image generation. In *Proceedings of the Sixth Interservice/Industry Training Equipment Conference* (National Security Industrial Association, Washington, DC, Oct.), 149–155.

BURT, P. J. 1981. Fast filter transforms for image processing. *Comput. Graph. Image Process. 16,* 20–51.

BUTLER, T. 1989. Three approaches to terrain rendering. In *Proceedings of SPIE* (Bellingham, WA), Vol. 1075, 217–225.

BUTLER, T. AND HARVILLE-HENDRIX, L. 1988. Image ray tracing: Rendering real world 3-D data. *Adv. Imag.* (Nov./Dec.), 54, 55, 67.

CARLOTTO, M. AND HARTT, K. 1989. Connection machine system for planetary terrain reconstruction and visualization. In *Proceedings of SPIE* (Bellingham, WA, Nov. 5–10), Vol. 1192, 220–229.

CARPENTER, L. 1984. The A-buffer, an antialiased hidden surface method. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 18,* 3 (July), 103–108.

CASE, J. B. 1967. The analytical reduction of panoramic and strip photography. *Photogrammetria,* 127–141.

CATMULL, E. 1975. Computer display of curved surfaces. In *Proceedings of the Conference on Computer Graphics, Pattern Recognition and Data Structures* (IEEE Computer Society, New York, May 14–16), 11–17.

CATMULL, E. AND SMITH, A. R. 1980. 3D transformations of images in scanline order. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 14,* 3 (July), 279–285.

COOK, R. 1986. Stochastic sampling in computer graphics. *ACM Trans. Graph. 5,* 1 (Jan.), 51–72.

COOK, R., CARPENTER, L., AND CATMULL, E. 1987. The Reyes image rendering architecture. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 21,* 4 (July), 95–102.

COQUILLART, S. AND GANGNET, M. 1984. Shaded display of digital maps. *IEEE Comput. Graph. Appl.* (July), 35–42.

COSMAN, M., MATHISEN, A. E., AND ROBINSON, J. A. 1990. A new visual system to support advanced requirements. In *Proceedings of IMAGE Conference V,* The IMAGE Society (Tempe, AZ, June 19–22), 371–380.

CROW, F. 1984. Summed-area tables for texture mapping. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH, 18,* 3 (July), 207–212.

DARASISTO 1992. *Proceedings: Image Understanding Workshop* (Jan.). Defense Advanced Research Agency Software and Intelligent Systems Technologies Office, Morgan Kaufmann, San Mateo, CA.

DARASISTO 1993. *Proceedings: Image Understanding Workshop* (April). Defense Advanced Research Agency Software and Intelligent Systems Technologies Office, Morgan Kaufmann, San Mateo, CA.

DEBEVEC, P. E., TAYLOR, C. J., AND MALIK, J. 1996. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *Computer Graphics,* Annual Conference Series, ACM SIGGRAPH, New York, 11–20.

DEERING, M., WINNER, S., SCHEDIWY, B., DUFFY, C., AND HUNT, N. 1988. The triangle processor and normal vector shader: A VLSI system for high performance graphics. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 22,* 4 (July), 21–30.

DEVARAJAN, V. 1989. Image processing in visual systems for flight simulation. In *Proceedings of SPIE* (Bellingham, WA), Vol. 1075, 208–216.

DEVARAJAN, V. AND CHEN, Y. P. 1986. Advanced data and picture transformation system. In *Proceedings of Electronic Imaging '86.* The Institute for Graphic Communications (Boston, MA), 767–772.

DEVICH, R. N. AND WEINHAUS, F. M. 1980. Image perspective transformations. In *Proceedings of SPIE* (Bellingham, WA, July 29–Aug. 1), Vol. 238, 322–332.

DEVICH, R. N. AND WEINHAUS, F. M. 1981a. Image perspective transformations—urban scenes. *Optical Eng. 20,* 6, 912–921.

DEVICH, R. N. AND WEINHAUS, F. M. 1981b. Rural image perspective transformations. In *Proceedings of SPIE* (Bellingham, WA) Vol. 303, 54–66.

DIPPE, M. AND WOLD, E. 1985. Antialiasing through stochastic sampling. *Computer*

Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 19, 3 (July), 69–78.

DUBAYAH, R. O. AND DOZIER, J. 1986. Orthographic terrain views using data derived from digital elevation models. *Photogram. Eng. Remote Sens. 52,* 4 (April), 509–518.

DUNGAN, W. 1979. A terrain and cloud computer image generation model. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 13,* 2 (Aug.), 143–150.

DUNGAN, W., STENGER, A., AND SUTTY, G. 1978. Texture tile considerations for raster graphics. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 12,* 3 (Aug.), 130–134.

ECONOMY, R. AND BUNKER, M. 1984. Advanced video object simulation. In *Proceedings of the National Aerospace and Electronics Conference* (IEEE, New York, May), 1065–1071.

ECONOMY, R., ELLIS, J. R., AND FERGUSON, R. 1988. The application of aerial photography and satellite imagery to flight simulation. In *Proceedings of the Tenth Interservice/Industry Training Systems Conference* (Washington, DC, Nov. 29–Dec. 1), 280–287.

ERICKSON, C., FAIRCHILD, K. M., AND MARVEL, O. 1984. Simulation gets a new look. *Defense Electron.* (Aug.), 76–87.

Evans & Sutherland in Multiple Launch. 1990. *Military Simul. Train.* 4, 41.

FANT, K. M. 1986. A nonaliasing, real-time spatial transform technique. *IEEE Comput. Graph. Appl.* (Jan.), 71–80.

FEIBUSH, E. AND GREENBERG, D. 1980. Texture rendering system for architectural design. *Comput. Aided Des. 12,* 2 (March), 67–71.

FEIBUSH, E., LEVOY, M., AND COOK, R. 1980. Synthetic texturing using digital filters. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 14,* 3 (July), 294–301.

FISCHETTI, M. A. AND TRUXAL, C. 1985. Simulating the right stuff. *IEEE Spectrum* (March), 38–47.

FISHMAN, B. AND SCHACHTER, B. 1980. Computer display of height fields. *Comput. Graph. 5,* 53–60.

FOLEY, J. D., VAN DAM, A., FEINER, S. K., AND HUGHES, J. F. 1993. *Computer Graphics Principles and Practice.* 2nd ed., Addison-Wesley, Reading, MA.

FOURNIER, A. AND FIUME, E. 1988. Constant-time filtering with space-variant kernels. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 22,* 4 (Aug.), 229–238.

FRIEDMAN, D. E. 1981. The use of digital terrain models in the rectification of satellite-borne imagery. In *Fifteenth International Symposium on Remote Sensing of Environment* (Ann Arbor, MI, May), 653–662.

FUA, P. AND HANSON, A. J. 1989. Objective functions for feature discrimination: Applications to semiautomated and automated feature extraction. In *Proceedings of the Image Understanding Workshop* (May 23–26), Morgan Kaufmann, San Mateo, CA, 676–690.

FUCHS, H., GOLDFEATHER, J., HULTQUIST, J., SPACH, S., AUSTIN, J., BROOKS, F., EYLES, J., AND POULTON, J. 1985. Fast spheres, shadows, textures, transparencies, and image enhancements in pixel-planes. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 19,* 3 (Aug.), 111–120.

FUJIMOTO, A., TANAKA, T., AND IWATA, K. 1986. ARTS: Accelerated ray-tracing system. *IEEE Comput. Graph. Appl.* (April), 16–26.

GANGNET, M., PERNY, D., AND COUEIGNOUX, P. 1982. Perspective mapping of planar textures. In *Proceedings Eurographics '82,* North-Holland, 57–71.

GEER, R. W. AND DIXON, D. 1988. Mission rehearsal: Its impact on data base technology. In *Proceedings of the Tenth Interservice/Industry Training Systems Conference,* National Security Industrial Association, Washington, DC (Nov. 29–Dec. 1), 213–220.

GELBERG, L. M., SCHOLTEN, D. K., STEPHENSON, T. P., WILLIAMS, T. A., AND WOLFE, G. J. 1988. Urban and terrain scene generation and animation for the ABC coverage of the Calgary Winter Olympics. *Course 20, SIGGRAPH Conference* (Aug.).

GLASSNER, A. 1986. Adaptive precision in texture mapping. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 20,* 4 (Aug.), 297–306.

GOSH, S. K. 1979. *Analytical Photogrammetry.* Pergamon Press, New York.

GOSHTASBY, A. 1986. Piecewise linear mapping functions for image registration. *Pattern Recogn. 19,* 6, 459–466.

GOSHTASBY, A. 1987. Piecewise cubic mapping functions for image registration. *Pattern Recogn. 20,* 5, 525–533.

GREEN, N. 1986. Environment mapping and other applications of world projections. *IEEE Comput. Graph. Appl.* (Nov.), 21–29.

GREENE, N. AND HECKBERT, P. 1986. Creating raster Omnimax images from multiple perspective views using the elliptical weighted average filter. *IEEE Comput. Graph. Appl.* (June), 21–27.

HAEBERLI, P. AND AKELEY, K. 1990. The accumulation buffer: Hardware support for high-quality rendering. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 24,* 4 (Aug.), 309–318.

HALL, B. A. 1989. Mars—the movie, *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 23,* 3 (July), 339.

HANSON, A. J. AND QUAM, L. H. 1988. Overview of the SRI cartographic modeling environment. In *Proceedings of the Image Understanding Workshop,* Morgan Kaufmann, San Mateo, CA, 576–582.

HANSON, A. J., PENTLAND, A. P., AND QUAM, L. H. 1987. Design of a prototype interactive cartographic display and analysis environment. In *Proceedings of the Image Understanding Workshop,* (Feb.), Morgan Kaufmann, San Mateo, CA, 475–482.

HECKBERT, P. 1983. Texture mapping polygons in perspective. Tech. Memo No. 13, Computer Graphics Lab, New York Institute of Technology, April 28.

HECKBERT, P. 1986a. Filtering by repeated integration. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 20,* 4 (Aug.), 315–321.

HECKBERT, P. S. 1986b. Survey of texture mapping. *IEEE Comput. Graph. Appl.* (Sept.), 56–67.

HECKBERT, P. S. 1989. Fundamentals of texture mapping and image warping. Masters Thesis, Computer Science Division, University of California, Report No. UCB/CSD 89/516.

HECKBERT, P. S. AND MORETON, H. P. 1991. Interpolation for polygon texture mapping and shading. In *State of the Art in Computer Graphics: Visualization and Modeling,* D. F. Rogers and R. A. Earnshaw, Eds., Springer-Verlag, New York, 101–111.

HERMAN, M. AND KANADE, T. 1986. Incremental reconstruction of 3D scenes from multiple, complex images. *Artif. Intell. 30,* 289–341.

HOOD, J., LADNER, L., AND CHAMPION, R. 1989. Image processing techniques for digital orthophotoquad production. *Photogram. Eng. Remote Sens. 55,* 9 (Sept.), 1323–1329.

HOOKS, J. T. AND DEVARAJAN, V. 1981a. Simulated FLIR imagery using computer animated photographic terrain views. In *Proceedings of IMAGE Conference II,* The IMAGE Society (Tempe, AZ, June 10–12), 25–34.

HOOKS, J. T. AND DEVARAJAN, V. 1981b. Digital processing of color photography for visual simulations. In *Proceedings of the Third Interservice/Industry Training Equipment and Exhibition,* National Security Industrial Association (Washington, DC, Nov. 30–Dec. 2), 47–55.

HOOKS, J. T., MARTINSEN, G. J., AND DEVARAJAN, V. 1990. On 3-D perspective generation from a multi-resolution photo mosaic data base. In *Proceedings of IMAGE Conference V,* The IMAGE Society (Tempe, AZ, June 19–22), 132–143.

HOPKINS, R. R. AND COOPER, M. B. 1988. Making clouds. In *Proceedings of the Tenth Interservice/Industry Training Systems Conference,* National Security Industrial Association (Washington, DC, Nov. 29–Dec. 1), 209–212.

HOURCADE, J. C. AND NICOLAS, A. 1983. Inverse perspective mapping in scanline order onto non-planar quadrilaterals. In *Proceedings Eurographics '83,* North-Holland, 309–319.

HUGHES, P. 1991. Building a terrain renderer. *Comput. Phys.* (July/Aug.), 434–437.

HUSSEY, K. J., HALL, J. R., AND MORTENSEN, R. A. 1986. Image processing methods in two and three dimensions used to animate remotely sensed data. In *Proceedings of IGARSS '86 Symposium,* IEEE Service Center Cat. No. 86CH2268-1 (Piscataway, NJ, Sept. 8–11) 771–776.

HUSSEY, K., MORTENSEN, B., AND HALL, J. 1987. LA—the movie. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 21,* 6 (Nov.), Appendix E, ViSC Videotape: List of Contributors, E-2.

INMAN, R. AND MCMILLAN, D. 1990. The state of the art in image processing: Visualization. *Aerospace Defense Sci.* (Nov./Dec.), 45–48.

IRVIN, R. B. AND MCKEOWN, D. M. 1989. Methods for exploiting the relationship between buildings and their shadows in aerial imagery. In *Proceedings of SPIE* (Bellingham, WA), Vol. 1076, 156–164.

JUNKIN, B. G. 1982. Development of three-dimensional spatial displays using a geographically based information system. *Photogram. Eng. Remote Sens. 48,* 4 (April), 577–586.

KANEDA, K., KATO, F., NAKAMAE, E., NISHITA, T., TANAKA, H., AND NOGUCHI, T. 1989. Three dimensional terrain modeling and display for environmental assessment. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 23,* 3 (July), 207–212.

KIRK, D. AND VOORHIES, D. 1990. The rendering architecture of the DN10000VS. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 24,* 4 (Aug.), 299–307.

LEE, D. AND SCHACHTER, B. 1980. Two algorithms for constructing a Delaunay triangulation. *Int. J. Comput. Inf. Sci. 9,* 3, 219–242.

LIEN, S., SHANTZ, M., AND PRATT, V. 1987. Adaptive forward differencing for rendering curves and surfaces. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 21,* 4 (July), 111–118.

LIPPMAN, A. 1980. Movie-maps: An application of the optical videodisk to computer graphics. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 14,* 3 (July), 32–42.

MAGNENAT-THALMANN, N. AND THALMANN, D. 1987. *Image Synthesis—Theory and Practice.* Springer-Verlag, Tokyo.

MCMILLAN, T. 1988. Graphics for the task. *Comput. Graph. World* (Aug.), 69–71.

MILLER, G. 1986. The definition and rendering of terrain maps. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 20,* 4 (Aug.), 39–48.

MOLNAR, S., EYLES, J., AND POULTON, J. 1992. Pixel Flow: High-speed rendering using image composition. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 26,* 2 (July), 231–240.

MORTENSEN, H. B., HUSSEY, K. J., AND MORTENSEN, R. A. 1988. Image processing methods used to simulate flight over remotely sensed data. In *Proceedings of the 20th Annual Summer Computer Simulation Conference,* (San Diego, CA, July 25–28), 719–722.

MUDUNURI, B. K. 1989. Photogrammetric applications in visual data base generation for part task trainers. In *Proceedings of the 12th Biennial Workshop on Color Aerial Photography and Videography in the Planetary Sciences and Related Fields,* American Society for Photogrammetry and Remote Sensing (Bethesda, MD, May 23–26), 227–235.

MUDUNURI, B. K. AND HOOKS, J. T. 1985. An algorithm for generation of super wide field of view scanned images. In *Proceedings of the 38th Annual SPSE Conference* (Springfield, VA, May 12–16), 154–158.

MUELLER, W. J. AND OLSON, J. A. 1993. Model-based feature extraction. In *Proceedings of SPIE* (Bellingham, WA), Vol. 1994, paper #26.

NACK, M. L., BATTAGLIA, M. P., AND NATHAN, A. 1989a. Real time perspective image generation. In *Proceedings of SPIE* (Bellingham, WA), Vol. 1075, 244–249. Also in *SPIE/SPSE Symposium on Electronic Imaging* (Bellingham, WA, Jan.) 15–20.

NACK, M. L., NATHAN, A. R., AND NATHAN, R. 1989b. Parallel processing applied to image processing and perspective scene generation. In *Proceedings of Electronic Imaging West 1989 Symposium,* The Institute for Graphic Communications (Boston, MA, April), 1–6.

NORTON, A., ROCKWOOD, A. P., AND SKOLMOSKI, P. T. 1982. Clamping: A method of antialiasing textured surfaces by bandwidth limiting in object space. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 16,* 3 (July), 1–8.

OKA, M., TSUTSUI, K., OHBA, A., KURAUCHI, Y., AND TAGO, T. 1987. Real-time manipulation of texture-mapped surfaces. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 21,* 4 (July), 181–188.

PAGLIERONI, D. W. 1992. A unified distance transform algorithm and architecture. *Mach. Visioin Appl. 5,* 47–55.

PAGLIERONI, D. W. AND PETERSON, S. M. 1992. Parametric height field ray tracing. In *Proceedings Graphics Interface '92* (Toronto, May), 192–200.

PAINTER, J. AND SLOAN, K. 1989. Antialiased ray tracing by adaptive progressive refinement. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 23,* 3 (Aug.), 281–286.

PARESCHI, M. T. AND BERNSTEIN, R. 1989. Modeling and image processing for visualization of volcanic mapping. *IBM J. Res. Dev. 33,* 4 (July), 406–416.

PELED, A. AND SHMUTTER, B. 1984. Analytical generation of orthophotos from panoramic photographs. *Photogram. Eng. Remote Sens. 50,* 2 (Feb.), 184–192.

PERNY, D., GANGNET, M., AND COUEIGNOUX, P. 1982. Perspective mapping of planar textures. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 16,* 1 (May), 70–100.

PETERSEN, S. M., BARRETT, W. A., AND BURTON, R. P. 1990. A new morphological algorithm for automated interpolation of height grids from contour images. In *Proceedings of SPIE* (Bellingham, WA), Vol. 1256, 62–72.

PINEDA, J. 1988. A parallel algorithm for polygon rasterization. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 22,* 4 (Aug.), 17–20.

RICH, H. 1989. Tradeoffs in creating a low-cost visual simulator. In *Proceedings of the 11th Interservice/Industry Training Systems Conference* (Washington, DC, Nov.), 214–223.

RIFMAN, S. 1973. Digital rectification of ERTS multispectral imagery. In *Proceedings of the Symposiuim on Significant Results Obtained from the ERTS-1,* NASA SP-327, Vol. 1, Sec. B (March), 1131–1142.

ROBERTSON, P. 1989. Spatial transformations for rapid scan-line surface shadowing. *IEEE Comput. Graph. Appl.* (July), 35–42.

ROBERTSON, P. K. 1987. Fast perspective views of images using one-dimensional operations. *IEEE Comput. Graph. Appl.* (Feb.), 47–56.

ROBINSON, J. AND ZIMMERMAN, S. 1985. Exploiting texture in an integrated training environment. In *Proceedings of the Seventh Interservice/Industry Training Systems Conference* (Washington, DC, Nov. 19–21), 113–121.

ROGERS, D. 1985. *Procedural Elements for Computer Graphics.* McGraw-Hill, New York.

ROSS, M. K. 1990. Imaging and graphics meet on the battlefield. *Adv. Imag.* (Feb.), 52–54, 75.

SAMET, H. AND WEBBER, R. 1988a. Hierarchical data structures and algorithms for computer graphics—part 1: Fundamentals. *IEEE Comput. Graph. Appl.* (May), 48–68.

SAMET, H. AND WEBBER, R. 1988b. Hierarchical data structures and algorithms for computer graphics—part 2: Applications. *IEEE Comput. Graph. Appl.* (July), 59–75.

SCHACHTER, B. J. AND AHUJA, N. 1980. A history of visual flight simulation. *Comput. Graph. World 3,* 3 (May), 16–31.

SCHACHTER, B. J. 1981. Computer image generation for flight simulation. *IEEE Comput. Graph. Appl.* (Oct.), 29–63.

SCOTT, W. B. 1983. Image system to unite computer, video. *Aviation Week Space Technol.* (May 16), 70–73.

SCUDERI, L. AND STROME, W. M. 1988. Bringing remote sensing down to earth. *Adv. Imag.* (Sept.), 44–50.

SEGAL, M., KOROBKIN, C., VAN WIDENFELT, R., FORAN, J., AND HAEBERLI, P. 1992. Fast shadows and lighting effects using texture mapping. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 26,* 2 (July), 249–252.

SHANTZ, M. AND LIEN, S. 1987. Shading bicubic patches. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 21,* 4 (July), 189–196.

SMEDLEY, K., HAINES, B., VAN VACTOR, D., AND JORDAN, M. 1989. Digital perspective generation and stereo display of composite ocean bottom and coastal terrain images. In *Proceedings of SPIE* (Bellingham, WA), Vol. 1083, 246–250.

SMITH, A. R. 1987. Planar 2-pass texture mapping and warping. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 21,* 4 (July), 263–272.

STANFILL, D. F. 1991. Using image pyramids for the visualization of large terrain data sets. *Int. J. Image Syst. Technol.* 157–166.

SUN MICROSYSTEMS. 1989. MAPVU demo package documentation. *TAAC-1 Software Reference Manual*, Apr. 15, Section 18, 36–40.

SWANSON, R. AND THAYER, L. 1986. A fast shaded-polygon renderer. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 20,* 4 (Aug.), 95–101.

TANAKA, S. 1979. Landscape drawings from LANDSAT MSS data. *Photogram. Eng. Remote Sens. 45,* 10 (Oct.), 1345–1351.

TANIMOTO, S. L. AND PAVLIDIS, T. 1975. A hierarchical data structure for picture processing. *Comput. Graph. Image Process. 4,* 2, 104–119.

THOMPSON, M. M., ED. 1966. *Manual of Photogrammetry.* American Society of Photogrammetry, Falls Church, VA.

TUCKER, J. B. 1984. Visual simulation takes flight. *High Technol.* (Dec.), 34–47.

UPSTILL, S. 1990. *The Renderman Companion.* Addison-Wesley, Reading, MA.

VAN WIE, P. AND STEIN, M. 1977. A LANDSAT digital image rectification system. *IEEE Geosci. Electron. GE-15,* 3 (July), 130–137.

VOLOTTA, T. A. 1991. Mars navigator. *Comput. Phys.* (May/June), 283–284.

WALKER, E. L. AND HERMAN, M. 1988. Geometric reasoning for constructing 3D scene descriptions from images. *Artif. Intell. 37,* 275–290.

WALTERMAN, M. AND WEINHAUS, F. 1991. Antialiasing warped imagery using look-up table based methods for adaptive resampling. In *Proceedings of SPIE* (Bellingham, WA, July), Vol. 1567, 204–214.

WANG, S. L. AND STAUDHAMMER, J. 1990. Visibility determination on projected grid surfaces. *IEEE Comput. Graph. Appl.* (July), 36–43.

WEINHAUS, F. AND WALTERMAN, M. 1990. A flexible approach to image warping. In *Proceedings of SPIE* (Bellingham, WA, Feb.), Vol. 1244, 108–122.

WEINHAUS, F. M. AND DEVICH, R. N. 1997. Photogrammetric texture mapping onto planar polygons. *Graph. Models Image Process.* (Submitted for publication).

WHITESIDE, A. E. 1989. Preparing data bases for perspective scene generation. In *Proceedings of SPIE* (Bellingham, WA), Vol. 1075, 230–237.

WHITESIDE, A., ELLIS, M., AND HASKELL, B. 1987. Digital generation of stereoscopic perspective scenes. In *Proceedings of SPIE* (Bellingham, WA), Vol. 761, 172–178.

WILLIAMS, L. 1983. Pyramidal parametrics. *Computer Graphics (Proceedings of SIGGRAPH), ACM SIGGRAPH 17,* 3 (July), 1–11.

WOLBERG, G. 1990. *Digital Image Warping.* IEEE Computer Society Press, Los Alamitos, CA.

YAN, J. K. 1985. Advances in computer-generated imagery for flight simulation. *IEEE Comput. Graph. Appl.* (Aug.), 37–51.

Received February 1996; revised April 1997; accepted May 1997