Proceedings of SIGGRAPH 95 (Los Angeles, California, August 6-11, 1995)

Plenoptic Modeling:
An Image-Based Rendering System

Leonard McMillan† and Gary Bishop‡

Department of Computer Science
University of North Carolina at Chapel Hill

ABSTRACT

Image-based rendering is a powerful new approach for generating real-time photorealistic computer graphics. It can provide convincing animations without an explicit geometric representation. We use the "plenoptic function" of Adelson and Bergen to provide a concise problem statement for image-based rendering paradigms, such as morphing and view interpolation. The plenoptic function is a parameterized function for describing everything that is visible from a given point in space. We present an image-based rendering system based on sampling, reconstructing, and resampling the plenoptic function. In addition, we introduce a novel visible surface algorithm and a geometric invariant for cylindrical projections that is equivalent to the epipolar constraint defined for planar projections.

CR Descriptors: I.3.3 [**Computer Graphics**]: Picture/Image Generation– *display algorithms*, *viewing algorithms*; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism– *hidden line/surface removal*; I.4.3 [**Image Processing**]: Enhancement– *registration*; I.4.7 [**Image Processing**]: Feature Measurement– *projections*; I.4.8 [**Image Processing**]: Scene Analysis.

1. INTRODUCTION

In recent years there has been increased interest, within the computer graphics community, in image-based rendering systems. These systems are fundamentally different from traditional geometry-based rendering systems. In image-based systems the underlying data representation (i.e model) is composed of a set of photometric observations, whereas geometry-based systems use either mathematical descriptions of the boundary regions separating scene elements (B-rep) or discretely sampled space functions (volumetric).

The evolution of image-based rendering systems can be traced through at least three different research fields. In photogrammetry the initial problems of camera calibration, two-dimensional image registration, and photometrics have progressed toward the determination of three-dimensional models. Likewise, in computer vision, problems such as robot navigation, image discrimination, and image understanding have naturally led in the same direction. In computer graphics, the progression toward image-based rendering systems was initially motivated by the desire to increase the visual realism of the approximate geometric descriptions by mapping images onto their surface (texture mapping) [7], [12]. Next, images were used to approximate global illumination effects (environment mapping) [5], and, most recently, we have seen systems where the images themselves constitute the significant aspects of the scene's description [8].

Another reason for considering image-based rendering systems in computer graphics is that acquisition of realistic surface models is a difficult problem. While geometry-based rendering technology has made significant strides towards achieving photorealism, creating accurate models is still nearly as difficult as it was ten years ago. Technological advances in three-dimensional scanning provide some promise in model building. However, they also verify our worst suspicions— the geometry of the real-world is exceedingly complex. Ironically, the primary subjective measure of image quality used by proponents of geometric rendering systems is the degree with which the resulting images are indistinguishable from photographs.

One liability of image-based rendering systems is the lack of a consistent framework within which to judge the validity of the results. Fundamentally, this arises from the absence of a clear problem definition. Geometry-based rendering, on the other hand, has a solid foundation; it uses analytic and projective geometry to describe the world's shape and physics to describe the world's surface properties and the light's interaction with those surfaces.

This paper presents a consistent framework for the evaluation of image-based rendering systems, and gives a concise problem definition. We then evaluate previous image-based rendering methods within this new framework. Finally, we present our own image-based rendering methodology and results from our prototype implementation.

2. THE PLENOPTIC FUNCTION

Adelson and Bergen [1] assigned the name *plenoptic* function (from the latin root *plenus*, meaning complete or full, and *optic* pertaining to vision) to the pencil of rays visible from any point in space, at any time, and over any range of wavelengths. They used this function to develop a taxonomy for evaluating models of low-level vision. The plenoptic function describes all of the radiant energy that can be perceived from the point of view of the observer rather than the point of view of the source. They postulate

> *"… all the basic visual measurements can be considered to characterize local change along one or two dimensions of a single function that describes the structure of the information in the light impinging on an observer."*

Adelson and Bergen further formalized this functional description by providing a parameter space over which the plenoptic function is valid, as shown in Figure 1. Imagine an idealized eye which we are free to place at any point in space (V_x, V_y, V_z). From there we can select any of the viewable rays by choosing an azimuth and elevation angle

†‡CB 3175 Sitterson Hall, Chapel Hill, NC 27599
† (919) 962-1797 mcmillan@cs.unc.edu http://www.cs.unc.edu/~mcmillan
‡ (919) 962-1886 gb@cs.unc.edu http://www.cs.unc.edu/~gb

(θ,ϕ) as well as a band of wavelengths, λ, which we wish to consider.

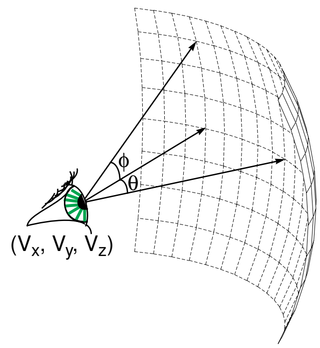


FIGURE 1. The plenoptic function describes all of the image information visible from a particular viewing position.

In the case of a dynamic scene, we can additionally choose the time, *t*, at which we wish to evaluate the function. This results in the following form for the plenoptic function:

$$p = P(\theta, \phi, \lambda, V_x, V_y, V_z, t) \qquad (1)$$

In computer graphics terminology, the plenoptic function describes the set of all possible environment maps for a given scene. For the purposes of visualization, one can consider the plenoptic function as a scene representation. In order to generate a view from a given point in a particular direction we would need to merely plug in appropriate values for (V_x, V_y, V_z) and select from a range of (θ,ϕ) for some constant t.

We define a complete sample of the plenoptic function as a full spherical map for a given viewpoint and time value, and an incomplete sample as some solid angle subset of this spherical map.

Within this framework we can state the following problem definition for image-based rendering. *Given a set of discrete samples (complete or incomplete) from the plenoptic function, the goal of image-based rendering is to generate a continuous representation of that function.* This problem statement provides for many avenues of exploration, such as how to optimally select sample points and how to best reconstruct a continuous function from these samples.

3. PREVIOUS WORK

3.1 Movie-Maps

The Movie-Map system by Lippman [17] is one of the earliest attempts at constructing an image-based rendering system. In Movie-Maps, incomplete plenoptic samples are stored on interactive video laser disks. They are accessed randomly, primarily by a change in viewpoint; however, the system can also accommodate panning, tilting, or zooming about a fixed viewing position. We can characterize Lippman's plenoptic reconstruction technique as a nearest-neighbor interpolation because, when given a set of input parameters $(V_x, V_y, V_z, \theta, \phi, t)$, the Movie-Map system can select the nearest partial sample. The Movie-Map form of image-based rendering can also be interpreted as a table-based evaluation of the plenoptic function. This interpretation reflects the database structure common to most image-based systems.

3.2 Image Morphing

Image morphing is a very popular image-based rendering technique [4], [28]. Generally, morphing is considered to occur between two images. We can think of these images as endpoints along some path through time and/or space. In this interpretation, morphing becomes a method for reconstructing partial samples of the continuous plenoptic function along this path. In addition to photometric data, morphing uses additional information describing the image flow field. This information is usually hand crafted by an animator. At first glance, this type of augmentation might seem to place it outside of the plenoptic function's domain. However, several authors in the field of computer vision have shown that this type of image flow information is equivalent to changes in the local intensity due to infinitesimal perturbations of the plenoptic function's independent variables [20], [13]. This local derivative behavior can be related to the intensity gradient via applications of the chain rule. In fact, morphing makes an even stronger assumption that the flow information is constant along the entire path, thus amounting to a locally linear approximation. Also, a blending function is often used to combine both reference images after being partially flowed from their initial configurations to a given point on the path. This blending function is usually some linear combination of the two images based on what percentage of the path's length has been traversed. Thus, morphing is a plenoptic reconstruction method which interpolates between samples and uses local derivative information to construct approximations.

3.3 View Interpolation

Chen's and Williams' [8] view interpolation employs incomplete plenoptic samples and image flow fields to reconstruct arbitrary viewpoints with some constraints on gaze angle. The reconstruction process uses information about the local neighborhood of a sample. Chen and Williams point out and suggest a solution for one of the key problems of image-based rendering— determining the visible surfaces. Chen and Williams chose to presort the quadtree compressed flow-field in a back-to-front order according to its (geometric) z-value. This approach works well when all of the partial sample images share a common gaze direction, and the synthesized viewpoints are restricted to stay within 90 degrees of this gaze angle.

An image flow field alone allows for many ambiguous visibility solutions, unless we restrict ourselves to flow fields that do not fold, such as rubber-sheet local spline warps or thin-plate global spline warps. This problem must be considered in any general-purpose image-based rendering system, and ideally, it should be done without transporting the image into the geometric-rendering domain.

Establishing flow fields for a view interpolation system can also be problematic. Chen and Williams used pre-rendered synthetic images to determine flow fields from the z-values. In general, accurate flow field information between two samples can only be established for points that are mutually visible to both samples. This points out a shortcoming in the use of partial samples, because reference images seldom have a 100% overlap.

Like morphing, view interpolation uses photometric information as well as local derivative information in its reconstruction process. This locally linear approximation is nicely exploited to approximate perspective depth effects, and Chen and Williams show it to be correct for lateral motions relative to the gaze direction. View interpolation, however, adds a nonlinearity by allowing the visibility process to determine the blending function between reference frames in a closest-take-all (a.k.a. winner-take-all) fashion.

3.4 Laveau and Faugeras

Laveau and Faugeras [15] have taken advantage of the fact that the epipolar geometries between images restrict the image flow field in such a way that it can be parameterized by a single disparity value and a fundamental matrix which represents the epipolar relationship. They also provide a two-dimensional raytracing-like solution to the visibility problem which does not require an underlying geometric description. Their method does, however, require establishing correspondences for each image point along the ray's path. The Laveau and Faugeras system also uses partial plenoptic samples, and results are shown only for overlapping regions between views.

Laveau and Faugeras also discuss the combination of information from several views but primarily in terms of resolving visibility. By relating the reference views and the desired views by the homogenous transformations between their projections, Laveau and Faugeras can compute exact perspective depth solutions. The reconstruction process again takes advantage of both image data and local derivative information to reconstruct the plenoptic function.

3.5 Regan and Pose

Regan and Pose [23] describe a hybrid system in which plenoptic samples are generated on the fly by a geometry-based rendering system at available rendering rates, while interactive rendering is provided by the image-based subsystem. At any instant, a user interacts with a single plenoptic sample. This allows the user to make unconstrained changes in the gaze angle about the sample point. Regan and Pose also discuss local reconstruction approximations due to changes in the viewing position. These approximations amount to treating the objects in the scene as being placed at infinity, resulting in a loss of the kinetic depth effect. These partial updates can be combined with the approximation values.

4. PLENOPTIC MODELING

We claim that all image-based rendering approaches can be cast as attempts to reconstruct the plenoptic function from a sample set of that function. We believe that there are significant insights to be gleaned from this characterization. In this section, we propose our prototype system in light of this plenoptic function framework.

We call our image-based rendering approach Plenoptic Modeling. Like other image-based rendering systems, the scene description is given by a series of reference images. These reference images are subsequently warped and combined to form representations of the scene from arbitrary viewpoints. The warping function is defined by image flow field information that can either be supplied as an input or derived from the reference images.

Our discussion of the plenoptic modeling image-based rendering system is broken down into four sections. First, we discuss the representation of the plenoptic samples. Next, we discuss their acquisition. The third section covers the determination of image flow fields, if required. And, finally, we describe how to reconstruct the plenoptic function from these sample images.

4.1 Plenoptic Sample Representation

The most natural surface for projecting a complete plenoptic sample is a unit sphere centered about the viewing position. One difficulty of spherical projections, however, is the lack of a representation that is suitable for storage on a computer. This is particularly difficult if a uniform (i.e. equal area) discrete sampling is required. This difficulty is reflected in the various distortions which arise in planar projections of world maps in cartography. Those uniform mappings which do exist are generally ill-suited for systematic access as a data structure. Furthermore, those which do map to a plane with consistent neighborhood relationships are generally quite distorted and, therefore, non-uniform.

A set of six planar projections in the form of a cube has been suggested by Greene [10] as an efficient representation for environment maps. While this representation can be easily stored and accessed by a computer, it provides significant problems relating to acquisition, alignment, and registration when used with real, non-computer-generated images. The orthogonal orientation of the cube faces requires precise camera positioning. The wide, 90 degree field-of-view of each face requires expensive lens systems to avoid optical distortion. Also, the planar mapping does not represent a uniform sampling, but instead, is considerably oversampled in the edges and corners. However, the greatest difficulty of a cube-oriented planar projection set is describing the behavior of the image flow fields across the boundaries between faces and at corners. This is not an issue when the six planar projections are used solely as an environment map, but it adds a considerable overhead when it is used for image analysis.

We have chosen to use a cylindrical projection as the plenoptic sample representation. One advantage of a cylinder is that it can be easily unrolled into a simple planar map. The surface is without boundaries in the azimuth direction, which simplifies correspondence searches required to establish image flow fields. One shortcoming of a projection on a finite cylindrical surface is the boundary conditions introduced at the top and bottom. We have chosen not to employ end caps on our projections, which has the problem of limiting the vertical field of view within the environment.

4.2 Acquiring Cylindrical Projections

A significant advantage of a cylindrical projection is the simplicity of acquisition. The only acquisition equipment required is a video camera and a tripod capable of continuous panning. Ideally, the camera's panning motion would be around the exact optical center of the camera. In practice, in a scene where all objects are relatively far from the tripod's rotational center, a slight misalignment offset can be tolerated.

Any two planar perspective projections of a scene which share a common viewpoint are related by a two-dimensional homogenous transform:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2}$$

$$x' = \frac{u}{w} \qquad y' = \frac{v}{w}$$

where x and y represent the pixel coordinates of an image I, and x' and y' are their corresponding coordinates in a second image I'. This well known result has been reported by several authors [12], [28], [22]. The images resulting from typical camera motions, such as pan, tilt, roll, and zoom, can all be related in this fashion. When creating a cylindrical projection, we will only need to consider panning camera motions. For convenience we define the camera's local coordinate system such that the panning takes place entirely in the x-z plane.

In order to reproject an individual image into a cylindrical projection, we must first determine a model for the camera's projection or, equivalently, the appropriate homogenous transforms. Many different techniques have been developed for inferring the homogenous transformation between images sharing common centers of projection. The most common technique [12] involves establishing four corresponding points across each image pair. The resulting transforms provide a mapping of pixels from the planar projection of the first image to the planar projection of the second. Several images could be composited in this fashion by first determining the transform which maps the Nth image to image N-1. These transforms can be catenated to form a mapping of each image to the plane of the first. This approach, in effect, avoids direct determination of an entire camera model by performing all mappings between different instances of the same camera. Other techniques for deriving these homogeneous transformations without specific point correspondences have also been described [22], [25].

The set of homogenous transforms, $\boldsymbol{H}_i$, can be decomposed into two parts which will allow for arbitrary reprojections in a manner similar to [11]. These two parts include an intrinsic transform, $\boldsymbol{S}$, which is determined entirely by camera properties, and an extrinsic transform, $\boldsymbol{R}_i$, which is determined by the rotation around the camera's center of projection:

$$\bar{u} = \boldsymbol{H}_i \bar{x} = \boldsymbol{S}^{-1} \boldsymbol{R}_i \boldsymbol{S} \bar{x} \tag{3}$$

This decomposition decouples the projection and rotational components of the homogeneous transform. By an appropriate choice of coordinate systems and by limiting the camera's motion to panning, the extrinsic transform component is constrained to a function of a single parameter rotation matrix describing the pan.

$$\boldsymbol{R}_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{4}$$

Since the intrinsic component's properties are invariant over all of the images, the decomposition problem can be broken into two parts: the determination of the extrinsic rotation component, $\boldsymbol{R}_i$, followed by the determination of an intrinsic projection component, $\boldsymbol{S}$. The first step in our method determines estimates for the extrinsic panning angle between each image pair of the panning sequence. This is accomplished by using a linear approximation to an infinitesimal rotation by the angle θ. This linear approximation results from substituting $1 + \boldsymbol{O}(\theta^2)$ for the cosine terms and $\theta + \boldsymbol{O}(\theta^3)$ for the sine terms of the rotation matrix. This infinitesimal perturbation has been shown by [14] to reduce to the following approximate equations:

$$\begin{aligned} x' &= x - f\theta - \frac{\theta\,(x - C_x)^2}{f} + \boldsymbol{O}(\theta^2) \\ y' &= y - \frac{\theta\,(x - C_x)\,(y - C_y)}{f} + \boldsymbol{O}(\theta^2) \end{aligned} \tag{5}$$

where f is the apparent focal length of the camera measured in pixels, and (C_x, C_y) is the pixel coordinate of the intersection of the optical axis with the image plane. (C_x, C_y) is initially estimated to be at the center pixel of the image plane. A better estimate for (C_x, C_y) is found during the intrinsic matrix solution.

These equations show that small panning rotations can be approximated by translations for pixels near the image's center. We require that some part of each image in the sequence must be visible in the successive image, and that some part of the final image must be visible in the first image of the sequence. The first stage of the cylindrical registration process attempts to register the image set by computing the optimal translation in x which maximizes the normalized correlation within a region about the center third of the screen. This is first computed at a pixel resolution, then refined on a 0.1 subpixel grid, using a Catmull-Rom interpolation spline to compute subpixel intensities. Once these translations, t_i, are computed, Newton's method is used to convert them to estimates of rotation angles and the focal length, using the following equation:

$$2\pi - \sum_{i=1}^{N} \operatorname{atan}\left(\frac{t_i}{f}\right) = 0 \tag{6}$$

where N is the number of images comprising the sequence. This usually converges in as few as five iterations, depending on the original estimate for f. This first phase determines an estimate for the relative rotational angles between each of the images (our extrinsic parameters) and the initial focal length estimate measured in pixels (one of the intrinsic parameters).

The second stage of the registration process determines the $\mathbf{S}$, or structural matrix, which describes various camera properties such as the tilt and roll angles which are assumed to remain constant over the group of images. The following model is used:

$$\boldsymbol{S} = \Omega_x \Omega_z \boldsymbol{P} \tag{7}$$

where $\mathbf{P}$ is the projection matrix:

$$\boldsymbol{P} = \begin{bmatrix} 1 & \sigma & -C_x \\ 0 & \rho & -C_y \\ 0 & 0 & f \end{bmatrix} \tag{8}$$

and (C_x, C_y) is the estimated center of the viewplane as described previously, σ is a skew parameter representing the deviation of the sampling grid from a rectilinear grid, ρ determines the sampling grid's aspect ratio, and f is the focal length in pixels as determined from the first alignment stage.

The remaining terms, Ω_x and Ω_z, describe the combined effects of camera orientation and deviations of the viewplane's orientation from perpendicular to the optical axis. Ideally, the viewplane would be normal to the optical axis, but manufacturing tolerances allow these numbers to vary slightly [27].

$$\Omega_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\omega_x & -\sin\omega_x \\ 0 & \sin\omega_x & \cos\omega_x \end{bmatrix} \tag{9}$$

$$\Omega_z = \begin{bmatrix} \cos\omega_z & -\sin\omega_z & 0 \\ \sin\omega_z & \cos\omega_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{10}$$

In addition, the ω_z term is indistinguishable from the camera's roll angle and, thus, represents both the image sensor's and the camera's rotation. Likewise, ω_x, is combined with an implicit parameter, ϕ, that represents the relative tilt of the camera's optical axis out of the panning plane. If ϕ is zero, the images are all tangent to a cylinder and for a nonzero ϕ the projections are tangent to a cone.

This gives six unknown parameters, $(C_x, C_y, \sigma, \rho, \omega_x, \omega_z)$, to be determined in the second stage of the registration process. Notice that, when combined with the θ_i and f parameters determined in the first stage, we have a total of eight parameters for each image, which is consistent with the number of free parameters in a general homogeneous matrix.

The structural matrix, $\mathbf{S}$, is determined by minimizing the following error function:

$$\mathrm{error}(C_x, C_y, \sigma, \rho, \omega_x, \omega_z) = \sum_{i=1}^{n} 1 - \mathrm{Correlation}(I_{i-1}, S^{-1} R_{y_i} S I_i) \tag{11}$$

where I_{i-1} and I_i represent the center third of the pixels from images i-1 and i respectively. Using Powell's multivariable minimization method [23] with the following initial values for our six parameters,

$$\begin{gathered} C_x = \frac{\text{image width}}{2} \qquad C_y = \frac{\text{image height}}{2} \\ \sigma = 0 \qquad \rho = 1 \qquad \omega_x = 0 \qquad \omega_z = 0 \end{gathered} \tag{12}$$

the solution typically converges in about six iterations. At this point we will have a new estimate for (C_x, C_y) which can be fed back into stage one, and the entire process can be repeated.

The registration process results in a single camera model, $\mathbf{S}(C_x, C_y, \sigma, \rho, \omega_x, \omega_z, f)$, and a set of the relative rotations, θ_i, between each of the sampled images. Using these parameters, we can compose mapping functions from any image in the sequence to any other image as follows:

$$\boldsymbol{I}'_i = \boldsymbol{S}^{-1} \boldsymbol{R}_{y_{i+1}} \boldsymbol{R}_{y_{i+2}} \boldsymbol{R}_{y_{i+3}} \ldots \boldsymbol{R}_{y_j} \boldsymbol{S} \boldsymbol{I}_j \tag{13}$$

We can also reproject images onto arbitrary surfaces by modifying $\mathbf{S}$. Since each image pixel determines the equation of a ray from the center-of-projection, the reprojection process merely involves intersecting these rays with the projection manifold.

4.3 Determining Image Flow Fields

Given two or more cylindrical projections from different positions within a static scene, we can determine the relative positions of centers-of-projection and establish geometric constraints across all potential reprojections. These positions can only be computed to a scale factor. An intuitive argument for this is that from a set of images alone, one cannot determine if the observer is looking at a model or a full-sized scene. This implies that at least one measurement is required to establish a scale factor. The measurement may be taken either between features that are mutually visible within images, or the distance between the acquired image's camera positions can be used. Both techniques have been used with little difference in results.

To establish the relative relationships between any pair of cylindrical projections, the user specifies a set of corresponding points that are visible from both views. These points can be treated as rays in space with the following form:

$$\bar{x}_a(\theta, v) = \bar{C}_a + t\bar{D}_a(\theta, v) \qquad \bar{D}_a(\theta, v) = \begin{bmatrix} \cos(\phi_a - \theta) \\ \sin(\phi_a - \theta) \\ k_a\left(C_{v_a} - v\right) \end{bmatrix} \tag{14}$$

where $\bar{C}_a = (A_x, A_y, A_z)$ is the unknown position of the cylinder's center of projection, ϕ_a is the rotational offset which aligns the angular orientation of the cylinders to a common frame, k_a is a scale factor which determines the vertical field-of-view, and C_{v_a} is the scanline where the center of projection would project onto the scene (i.e. the line of zero elevation, like the equator of a spherical map).

A pair of tiepoints, one from each image, establishes a pair of rays which ideally intersect at the point in space identified by the tiepoint. In general, however, these rays are skewed. Therefore, we use the point that is simultaneously closest to both rays as an estimate of the point's position, $\bar{p}$, as determined by the following derivation.

$$\bar{p}(\theta_a, v_a, \theta_b, v_b) = \frac{\bar{x}_a - \bar{x}_b}{2} \tag{15}$$

where (θ_a, v_a) and (θ_b, v_b) are the tiepoint coordinates on cylinders A and B respectively. The two points, $\bar{x}_a$ and $\bar{x}_b$, are given by

$$\begin{aligned} \bar{x}_a &= \bar{C}_a + t\bar{D}_a(\theta_a, v_a) \\ \bar{x}_b &= \bar{C}_b + s\bar{D}_b(\theta_b, v_b) \end{aligned} \tag{16}$$

where

$$\begin{aligned} t &= \frac{\mathrm{Det}\left[\bar{C}_a - \bar{C}_b, \bar{D}_b(\theta_b, v_b), \bar{D}_a(\theta_a, v_a) \times \bar{D}_b(\theta_b, v_b)\right]}{\left|\bar{D}_a(\theta_a, v_a) \times \bar{D}_b(\theta_b, v_b)\right|^2} \\ s &= \frac{\mathrm{Det}\left[\bar{C}_b - \bar{C}_a, \bar{D}_a(\theta_a, v_a), \bar{D}_a(\theta_a, v_a) \times \bar{D}_b(\theta_b, v_b)\right]}{\left|\bar{D}_a(\theta_a, v_a) \times \bar{D}_b(\theta_b, v_b)\right|^2} \end{aligned} \tag{17}$$

This allows us to pose the problem of finding a cylinder's position as a minimization problem. For each pair of cylinders we have two sets of six unknowns, $[(A_x, A_y, A_z, \phi_a, k_a, C_{va}), (B_x, B_y, B_z, \phi_b, k_b, C_{vb})]$. In general, we have good estimates for the k and C_v terms, since these values are found by the registration phase. The position of the cylinders is determined by minimizing the distance between these skewed rays. We also choose to assign a penalty for shrinking the vertical height of the cylinder in order to bring points closer together. This penalty could be eliminated by accepting either the k or C_v values given by the registration.

We have tested this approach using from 12 to 500 tiepoints, and have found that it converges to a solution in as few as ten iterations of Powell's method. Since no correlation step is required, this process is considerably faster than the minimization step required to determine the structural matrix, ***S***.

The use of a cylindrical projection introduces significant geometric constraints on where a point viewed in one projection might appear in a second. We can capitalize on these restrictions when we wish to automatically identify corresponding points across cylinders. While an initial set of 100 to 500 tiepoints might be established by hand, this process is far too tedious to establish a mapping for the entire cylinder. Next, we present a geometric constraint for cylindrical projections that determines the possible positions of a point given its position in some other cylinder. This constraint plays the same role that the epipolar geometries [18], [9], used in the computer vision community for depth-from-stereo computations, play for planar projections.

First, we will present an intuitive argument for the existence of such an invariant. Consider yourself at the center of a cylindrical projection. Every point on the cylinder around you corresponds to a ray in space as given by the cylindrical epipolar geometry equation. When one of the rays is observed from a second cylinder, its path projects to a curve which appears to begin at the point corresponding to the origin of the first cylinder, and it is constrained to pass through the point's image on the second cylinder.

This same argument could obviously have been made for a planar projection. And, since two points are identified (the virtual image of the camera in the second projection along with the corresponding point) and, because a planar projection preserve lines, a unique, so called epipolar line is defined. This is the basis for an epipolar geometry, which identifies pairs of lines in two planar projections such that if a point falls upon one line in the first image, it is constrained to fall on the corresponding line in the second image. The existence of this invariant reduces the search for corresponding points from an **O**(N^2) problem to **O**(N).

Cylindrical projections, however, do not preserve lines. In general, lines map to quadratic parametric curves on the surface of a cylinder. Surprisingly, we can completely specify the form of the curve with no more information than was needed in the planar case.

The paths of these curves are uniquely determined sinusoids. This *cylindrical epipolar geometry* is established by the following equation.

$$v(\theta) = \frac{N_x \cos(\phi_a - \theta) + N_y \sin(\phi_a - \theta)}{N_z k_a} + C_{v_a} \tag{18}$$

where

$$\bar{N} = (\bar{C}_b - \bar{C}_a) \times \bar{D}_a(\theta_a, v_a) \tag{19}$$

This formula gives a concise expression for the curve formed by the projection of a ray across the surface of a cylinder, where the ray is specified by its position on some other cylinder.

This cylindrical epipolar relationship can be used to establish image flow fields using standard computer vision methods. We have used correlation methods [9], a simulated annealing-like relaxation method [3], and the method of differences [20] to compute stereo disparities between cylinder pairs. Each method has its strengths and weaknesses. We refer the reader to the references for further details.

4.4 Plenoptic Function Reconstruction

Our image-based rendering system takes as input cylindrically projected panoramic reference images along with scalar disparity images relating each cylinder pair. This information is used to automatically generate image warps that map reference images to arbitrary cylindrical or planar views that are capable of describing both occlusion and perspective effects.

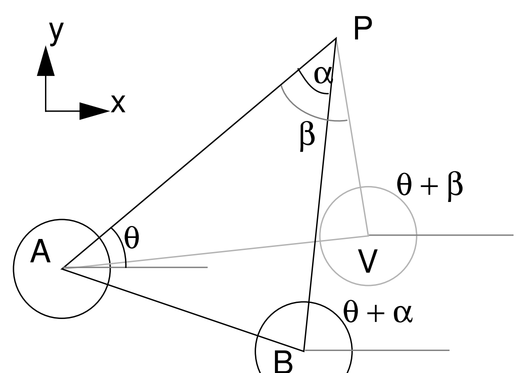


FIGURE 2. Diagram showing the transfer of the known disparity values between cylinders A and B to a new viewing position V.

We begin with a description of cylindrical-to-cylindrical mappings. Each angular disparity value, α, of the disparity images, can be readily converted into an image flow vector field, $(\theta + \alpha, v(\theta + \alpha))$ using the epipolar relation given by Equation 18 for each position on the cylinder, (θ, v). We can transfer disparity values from the known cylindrical pair to a new cylindrical projection in an arbitrary position, as in Figure 2, using the following equations.

$$
\begin{aligned}
a &= (B_x - V_x)\cos(\phi_A - \theta) + (B_y - V_y)\sin(\phi_A - \theta) \\
b &= (B_y - A_y)\cos(\phi_A - \theta) - (B_x - A_x)\sin(\phi_A - \theta) \\
c &= (V_y - A_y)\cos(\phi_A - \theta) - (V_x - A_x)\sin(\phi_A - \theta) \\
\cot(\beta(\theta, v)) &= \frac{a + b\cot(\alpha(\theta, v))}{c}
\end{aligned}
\tag{20}
$$

By precomputing $[\cos(\phi_i - \theta), \sin(\phi_i - \theta)]$ for each column of the cylindrical reference image and storing $\cot(\alpha)$ in place of the disparity image, this transfer operation can be computed at interactive speeds.

Typically, once the disparity images have been transferred to their target, the cylindrical projection would be reprojected as a planar image for viewing. This reprojection can be combined with the disparity transfer to give a single image warp that performs both operations. To accomplish this, a new intermediate quantity, δ, called the *generalized angular disparity* is defined as follows:

$$
\begin{aligned}
d &= (B_x - A_x)\cos(\phi_A - \theta) + (B_y - A_y)\sin(\phi_A - \theta) \\
\delta(\theta, v) &= \frac{1}{d + b\cot(\alpha(\theta, v))}
\end{aligned}
\tag{21}
$$

This scalar function is the cylindrical equivalent to the classical stereo disparity. Finally, a composite image warp from a given reference image to any arbitrary planar projection can be defined as

$$
\begin{aligned}
x(\theta, v) &= \frac{\bar{r} \cdot D_A(\theta, v) + k_r \delta(\theta, v)}{\bar{n} \cdot D_A(\theta, v) + k_n \delta(\theta, v)} \\
y(\theta, v) &= \frac{\bar{s} \cdot D_A(\theta, v) + k_s \delta(\theta, v)}{\bar{n} \cdot D_A(\theta, v) + k_n \delta(\theta, v)}
\end{aligned}
\tag{22}
$$

where

$$
\begin{aligned}
\bar{r} &= \bar{v} \times \bar{o} \qquad & k_r &= \bar{r} \cdot (\bar{C}_a - \bar{V}) \\
\bar{s} &= \bar{o} \times \bar{u} \qquad & k_s &= \bar{s} \cdot (\bar{C}_a - \bar{V}) \\
\bar{n} &= \bar{u} \times \bar{v} \qquad & k_n &= \bar{n} \cdot (\bar{C}_a - \bar{V})
\end{aligned}
\tag{23}
$$

and the vectors $\bar{p}$, $\bar{o}$, $\bar{u}$ and $\bar{v}$ are defined by the desired view as shown in Figure 3.

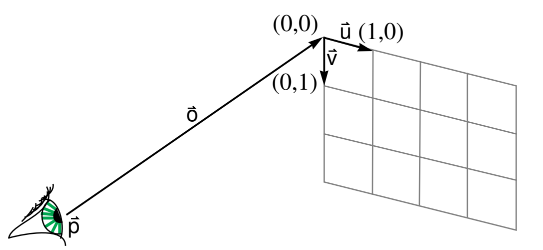


FIGURE 3. The center-of-projection, $\bar{p}$, a vector to the origin, $\bar{o}$, and two spanning vectors ($\bar{u}$ and $\bar{v}$) uniquely determine the planar projection**.**

In the case where $\delta(\theta, v) = \text{constant}$, the image warp defined by Equation 22, reduces to a simple reprojection of the cylindrical image to a desired planar view. The perturbation introduced by allowing $\delta(\theta, v)$ to vary over the image allows arbitrary shape and occlusions to be represented.

Potentially, both the cylinder transfer and image warping approaches are many-to-one mappings. For this reason we must consider visibility. The following simple algorithm can be used to determine an enumeration of the cylindrical mesh which guarantees a proper back-to-front ordering, (See Appendix). We project the desired viewing position onto the reference cylinder being warped and partition the cylinder into four toroidal sheets. The sheet boundaries are defined by the θ and v coordinates of two points, as shown in Figure 4. One point is defined by the intersection of the cylinder with the vector from the origin through the eye's position. The other is the intersection with the vector from the eye through the origin.

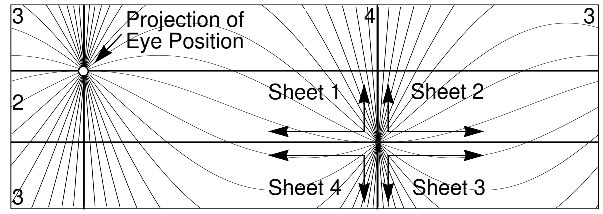


FIGURE 4. A back-to-front ordering of the image flow field can be established by projecting the eye's position onto the cylinder's surface and dividing it into four toroidal sheets.

Next, we enumerate each sheet such that the projected image of the desired viewpoint is the last point drawn. This simple partitioning and enumeration provides a back-to-front ordering for use by a painter's style rendering algorithm. This hidden-surface algorithm is a generalization of Anderson's [2] visible line algorithm to arbitrary projected grid surfaces. Additional details can be found in [21].

At this point, the plenoptic samples can be warped to their new position according to the image flow field. In general, these new pixel positions lie on an irregular grid, thus requiring some sort of reconstruction and resampling. We use a forward-mapping [28] reconstruction approach in the spirit of [27] in our prototype. This involves computing the projected kernel's size based on the current disparity value and the derivatives along the epipolar curves.

While the visibility method properly handles mesh folds, we still must consider the tears (or excessive stretching) produced by the exposure of previously occluded image regions. In view interpolation [8] a simple "distinguished color" heuristic is used based on the screen space projection of the neighboring pixel on the same scanline. This approach approximates stretching for small regions of occlusion, where the occluder still abuts the occluded region. And, for large exposed occluded regions, it tends to interpolate between the boundaries of the occluded region. These exposure events can be handled more robustly by combining, on a pixel-by-pixel basis, the results of multiple image warps according to the smallest-sized reconstruction kernel.

5. RESULTS

We collected a series of images using a video camcorder on a leveled tripod in the front yard of one of the author's home. Accurate leveling is not strictly necessary for the method to work. When the data were collected, no attempt was made to pan the camera at a uniform angular velocity. The autofocus and autoiris features of the camera were disabled, in order to maintain a constant focal length during the collection process. The frames were then digitized at a rate of approximately 5 frames per second to a resolution of 320 by 240 pixels. An example of three sequential frames are shown below.

Immediately after the collection of the first data set, the process was repeated at a second point approximately 60 inches from the first. The two image sequences were then separately registered using the methods described in Section 4.2. The images were reprojected onto the surface of a cylinder with a resolution of 3600 by 300 pixels. The results are shown in Figures 5a and 5b. The operating room scene, in Figure 5c, was also constructed using these same methods.

Next, the epipolar geometry was computed by specifying 12 tiepoints on the front of the house. Additional tiepoints were gradually added to establish an initial disparity image for use by the simulated

FIGURE 5. Cylindrical images a and b are panoramic views separated by approximately 60 inches. Image c is a panoramic view of an operating room. In image d, several epipolar curves are superimposed onto cylindrical image a.

annealing and method of differences stereo-correspondence routines. As these tiepoints were added, we also refined the epipolar geometry and cylinder position estimates. The change in cylinder position, however, was very slight. In Figure 5d, we show a cylindrical image with several epipolar curves superimposed. Notice how the curves all intersect at the alternate camera's virtual image and vanishing point.

After the disparity images are computed, they can be interactively warped to new viewing positions. The following four images show various reconstructions. When used interactively, the warped images provide a convincing kinetic depth effect.

6. CONCLUSIONS

The plenoptic function provides a consistent framework for image-based rendering systems. The various image-based methods, such as morphing and view interpolation, are characterized by the different ways they implement the three key steps of sampling, reconstructing, and resampling the plenoptic function.

We have described our approach to each of these steps. Our method for sampling the plenoptic function can be done with equipment that is commonly available, and it results in cylindrical samples about a point. All the necessary parameters are automatically estimated from a sequence of images resulting from panning a video camera through a full circle.

Reconstructing the function from these samples requires estimating the optic flow of points when the view point is translated. Though this problem can be very difficult, as evidenced by thirty years of computer vision and photogrammetry research, it is greatly simplified when the samples are relatively close together. This is because there is little change in the image between samples (which makes the estimation easier), and because the viewer is never far from a sample (which makes accurate estimation less important).

Resampling the plenoptic function and reconstructing a planar projection are the key steps for display of images from arbitrary viewpoints. Our methods allow efficient determination of visibility and real-time display of visually rich environments on conventional workstations without special purpose graphics acceleration.

The plenoptic approach to modeling and display will provide robust and high-fidelity models of environments based entirely on a set of reference projections. The degree of realism will be determined by the resolution of the reference images rather than the number of primitives used in describing the scene. Finally, the difficulty of producing realistic models of real environments will be greatly reduced by replacing geometry with images.

ACKNOWLEDGMENTS

We are indebted to the following individuals for their contributions and suggestions on this work: Henry Fuchs, Andrei State, Kevin Arthur, Donna McMillan, and all the members of the UNC/UPenn collaborative telepresence-research group.

This research is supported in part by Advanced Research Projects Agency contract no. DABT63-93-C-0048, NSF Cooperative Agreement no. ASC-8920219, Advanced Research Projects Agency order no. A410, and National Science Foundation grant no. MIP-9306208. Approved by ARPA for public release - distribution unlimited.

REFERENCES

[1] Adelson, E. H., and J. R. Bergen, *"The Plenoptic Function and the Elements of Early Vision,"* **Computational Models of Visual Processing**, Chapter 1, Edited by Michael Landy and J. Anthony Movshon. The MIT Press, Cambridge, Mass. 1991.

[2] Anderson, D., *"Hidden Line Elimination in Projected Grid Surfaces,"* **ACM Transactions on Graphics**, October 1982.

[3] Barnard, S.T. "A Stochastic Approach to Stereo Vision," SRI International, Technical Note 373, April 4, 1986.

[4] Beier, T. and S. Neely, *"Feature-Based Image Metamorphosis,"* **Computer Graphics** (SIGGRAPH'92 Proceedings), Vol. 26, No. 2, pp. 35-42, July 1992.

[5] Blinn, J. F. and M. E. Newell, *"Texture and Reflection in Computer Generated Images,"* ***Communications of the ACM***, vol. 19, no. 10, pp. 542-547, October 1976.

[6] Bolles, R. C., H. H. Baker, and D. H. Marimont, *"Epipolar-Plane Image Analysis: An Approach to Determining Structure from Motion,"* **International Journal of Computer Vision**, Vol. 1, 1987.

[7] Catmull, E., "*A Subdivision Algorithm for Computer Display of Curved Surfaces*" (Ph. D. Thesis), Department of Computer Sci-

ence, University of Utah, Tech. Report UTEC-CSc-74-133, December 1974.

[8] Chen, S. E. and L. Williams. *"View Interpolation for Image Synthesis,"* **Computer Graphics** (SIGGRAPH'93 Proceedings), pp. 279-288, July 1993.

[9] Faugeras, O., **Three-dimensional Computer Vision: A Geometric Viewpoint**, The MIT Press, Cambridge, Massachusetts, 1993.

[10] Greene, N., *"Environment Mapping and Other Applications of World Projections,"* **IEEE Computer Graphics and Applications**, November 1986.

[11] Hartley, R.I., *"Self-Calibration from Multiple Views with a Rotating Camera,"* **Proceedings of the European Conference on Computer Vision**, May 1994.

[12] Heckbert, P. S., *"Fundamentals of Texture Mapping and Image Warping,"* Masters Thesis, Dept. of EECS, UCB, Technical Report No. UCB/CSD 89/516, June 1989.

[13] Horn, B., and B.G. Schunck, *"Determining Optical Flow,"* **Artificial Intelligence**, Vol. 17, 1981.

[14] Kanatani, K., *"Transformation of Optical Flow by Camera Rotation,"* **IEEE Transactions on Pattern Analysis and Machine Intelligence**, Vol. 10, No. 2, March 1988.

[15] Laveau, S. and O. Faugeras, *"3-D Scene Representation as a Collection of Images and Fundamental Matrices,"* INRIA, Technical Report No. 2205, February, 1994.

[16] Lenz, R. K. and R. Y. Tsai, *"Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3D Machine Vision Metrology,"* **Proceedings of IEEE International Conference on Robotics and Automation**, March 31 - April 3, 1987.

[17] Lippman, A., *"Movie-Maps: An Application of the Optical Videodisc to Computer Graphics,"* **SIGGRAPH '80 Proceedings**, 1980.

[18] Longuet-Higgins, H. C., *"A Computer Algorithm for Reconstructing a Scene from Two Projections,"* **Nature**, Vol. 293, September 1981.

[19] Longuet-Higgins, H. C., *"The Reconstruction of a Scene From Two Projections - Configurations That Defeat the 8-Point Algorithm,"* **Proceedings of the First IEEE Conference on Artificial Intelligence Applications**, Dec 1984.

[20] Lucas, B., and T. Kanade, *"An Iterative Image Registration Technique with an Application to Stereo Vision,"* **Proceedings of the Seventh International Joint Conference on Artificial Intelligence**, Vancouver, 1981.

[21] McMillan, Leonard, *"A List-Priority Rendering Algorithm for Redisplaying Projected Surfaces,"* Department of Computer Science, UNC, Technical Report TR95-005, 1995.

[22] Mann, S. and R. W. Picard, *"Virtual Bellows: Constructing High Quality Stills from Video,"* **Proceedings of the First IEEE International Conference on Image Processing**, November 1994.

[23] Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, **Numerical Recipes in C**, Cambridge University Press, Cambridge, Massachusetts, pp. 309-317, 1988.

[24] Regan, M., and R. Pose, *"Priority Rendering with a Virtual Reality Address Recalculation Pipeline,"* **SIGGRAPH'94 Proceedings**, 1994.

[25] Szeliski, R., *"Image Mosaicing for Tele-Reality Applications,"* **DEC and Cambridge Research Lab Technical Report**, CRL 94/2, May 1994.

[26] Tomasi, C., and T. Kanade, *"Shape and Motion from Image Streams: a Factorization Method; Full Report on the Orthographic Case,"* Technical Report, CMU-CS-92-104, Carnegie Mellon University, March 1992.

[27] Tsai, R. Y., *"A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses,"* **IEEE Journal of Robotics and Automation**, Vol. RA-3, No. 4, August 1987.

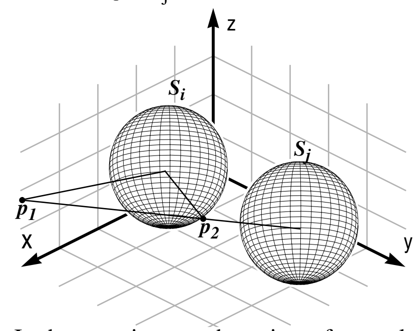
[28] Westover, L. A., *"Footprint Evaluation for Volume Rendering,"* **SIGGRAPH'90 Proceedings**, August 1990.

[29] Wolberg, G., **Digital Image Warping,** IEEE Computer Society Press, Los Alamitos, CA, 1990.

APPENDIX

We will show how occlusion compatible mappings can be determined on local spherical frames embedded within a global cartesian frame, $\boldsymbol{W}$. The projected visibility algorithm for cylindrical surfaces given in the paper can be derived by reducing it to this spherical case.

First, consider an isolated topological multiplicity on the projective mapping from $\boldsymbol{S}_\mathrm{i}$ to $\boldsymbol{S}_\mathrm{j}$, as shown below



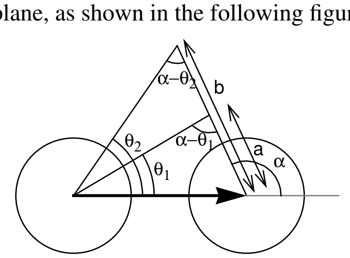
Theorem 1: In the generic case, the points of a topological multiplicity induced by a mapping from $\boldsymbol{S}_i$ to $\boldsymbol{S}_j$, and the two frame origins are coplanar.

Proof: The points of the topological multiplicity are colinear with the origin of $\boldsymbol{S}_j$ since they share angular coordinates. A second line segment connects the local frame origins, $\boldsymbol{S}_i$ and $\boldsymbol{S}_j$. In general, these two lines are distinct and thus they define a plane in three space.

Thus, a single affine transformation, $\mathbf{A}$, of $\boldsymbol{W}$ can accomplish the following results.

- Translate $\boldsymbol{S}_i$ to the origin
- Rotate $\boldsymbol{S}_j$ to lie on the x-axis
- Rotate the line along the multiplicity into the xy-plane
- Scale the system so that $\boldsymbol{S}_j$ has the coordinate (1,0,0).

With this transformation we can consider the multiplicity entirely within the xy-plane, as shown in the following figure.



Theorem 2: If $\cos\theta_1 > \cos\theta_2$ and $(\theta_1, \theta_2, \alpha) \in [0, \pi]$ then $a < b$.

Proof: The length of sides *a* and *b* can be computed in terms of the angles θ_1, θ_2 and α using the law of sines as follows.

$$\frac{a}{\sin\theta_1} = \frac{1}{\sin(\alpha-\theta_1)} \qquad \frac{b}{\sin\theta_2} = \frac{1}{\sin(\alpha-\theta_2)}$$

$$\frac{a}{b} = \frac{\sin\alpha\cot\theta_2 - \cos\alpha}{\sin\alpha\cot\theta_1 - \cos\alpha}$$

if $\cos\theta_1 > \cos\theta_2$ then $\cot\theta_1 > \cot\theta_2$, thus $a < b$

Thus, an occlusion compatible mapping, can be determined by enumerating the topological mesh defined on $\boldsymbol{AS}_i$ in an order of increasing $\cos\theta$, while allowing later mesh facets to overwrite previous ones. This mapping is occlusion compatible since, by Theorem 2, greater range values will always proceed lesser values at all multiplicities. Notice, that this mapping procedure only considers the changes in the local frame's world coordinates, and makes no use of the range information itself.