Image-Based Modeling and Rendering Techniques: A Survey

Manuel M. Oliveira[1]

Resumo: A recente convergência entre computação gráfica e visão computacional levou à criação de um conjunto de técnicas denominadas *modelagem e rendering baseados em imagens (MRBI)*. Modelagem baseada em imagens se refere ao processo de utilização de imagens para reconstrução de modelos geométricos 3D. Rendering baseado em imagens busca a obtenção de um alto grau de realismo, redução do tempo de processamento e simplificação da tarefa de modelagem, através do uso de imagens como primitivas de modelagem e rendering. Tais abordagens tem o potencial de simplificar a representação de cenas complexas e permitir a obtenção de efeitos subtos presentes em cenas reais, mas de difícil reprodução por meio de técnicas convencionais. Este artigo apresenta um levantamento sistemático do estado-da-arte em MRBI, discutindo os princípios básicos de cada técnica, suas vantangens e limitações.

Palavras-Chave: Modelagem e Rendering Baseados em Images, Computação Gráfica, Realismo.

Abstract: A recent convergence of computer graphics and computer vision has produced a set of techniques collectively known as *image-based modeling and rendering* (IBMR). Image-based modeling refers to the use of images to drive the reconstruction of 3D models. In the case of image-based rendering, the goal is to achieve more realistic and faster renderings and to simplify the modeling task by using images, as opposed to polygons, as both modeling and rendering primitives. Image-based approaches can potentially eliminate the labor-intensive task usually required for modeling detailed geometric structures. They can also handle subtle real-world effects captured by images, but difficult to reproduce with conventional graphics techniques. This paper surveys the state-of-the-art in image-based modeling and rendering techniques, discussing the fundamental principles behind each technique, their strengths and limitations.

Keywords: Image-Based Modeling and Rendering, Computer Graphics, Realism.

---

[1] Instituto de Informática, UFRGS, Caixa Postal 15064, CEP 91501-970, Porto Alegre, RS, Brasil
{oliveira@inf.ufrgs.br}

1 Introduction

In recent years, image-based modeling and rendering (IBMR) techniques have gained considerable attention in the graphics community because of their potential to create very realistic images. One of the major benefits of these techniques is the ability to capture subtle real-world effects and details related to the imperfections of the real world that graphics researchers still do not know how to model and render [12]. By using images as both modeling and rendering primitives, image-based approaches can help to alleviate two important and long standing problems in computer graphics: the need for simpler modeling techniques suitable for representing complex scenes, and the ever need for rendering acceleration. The former can be achieved by replacing conventional (geometric) models with image-based representations. Rendering speedups are obtained by detaching rendering time from scene complexity, and by re-sampling pre-shaded images.

Image-based rendering (IBR) uses images, as opposed to polygons, as modeling and rendering primitives. In practice, many IBR approaches correspond to image-geometry hybrids, with the corresponding amount of geometry ranging from per-pixel depth to hundreds of polygons. Image-based modeling (IBM), on the other hand, refers to the use of images to drive the reconstruction of three-dimensional geometric models. Despite their potential, IBMR techniques are still in their infancy and several challenges still need to be overcome. This tutorial surveys the state-of-the-art in image-based modeling and rendering techniques, discussing their strengths and limitations, and enumerating some possible research opportunities.

The structure of the survey is organized around the diagram presented in Figure 1. The techniques are first classified as image-based rendering (IBR) or image-based modeling (IBM) and further refined according to their relative positions along the image-geometry spectrum (indicated by a dot). Thus, for instance, pure image-based approaches (*i.e.*, pure IBR and pure IBM) fall on the left part of the spectrum, whereas hybrid techniques are positioned according to the amount of geometric information required.

The goal of providing an intuitive description of the fundamental ideas behind most current IBR and IBM techniques in a single document is very ambitious. Its hope is to offer a useful reference for students and researchers interested in a solid introduction to the underlying principles of this young field. Unfortunately, due to space limitations, some techniques had to be left out in order to guarantee proper coverage of the material treated here. The choice of the presented techniques took into account their fitness in a progression that helps the reader to gradually assimilate concepts and understand the field evolution. This by no means imply the lack of importance of the uncovered methods. Techniques such as *Image-Based Visual Hulls* [24, 47], *Surface Light Fields* [45], *Light Field Mapping* [8] and *Voxel Coloring* [48] deserve proper recognition but could not be included in this survey due to the lack of space. They appear, however, in the diagram shown in Figure 1.

2 Pure Image-Based Rendering Techniques

Pure IBR techniques utilize only intensity images[2] of the scenes for rendering novel views. Samples of the environment are captured as sets of photographs (or videos sequences) and resampled during rendering time. In this case, no 3D modeling is required and the rendering speed is not affected by scene complexity. An early example of this approach, called *Movie-Maps*, was developed Lipmann [21]. In this pioneering work, a database of images from the streets of a city, acquired at periodic intervals, was stored in videodisk. A drive through these streets could be simulated by playing back, at controlled rates, individual frames acquired from viewpoints close to the current position of the virtual driver. In recent years, several other rendering approaches based exclusively on image resampling have been proposed [6, 18, 39, 2] and will be discussed next.

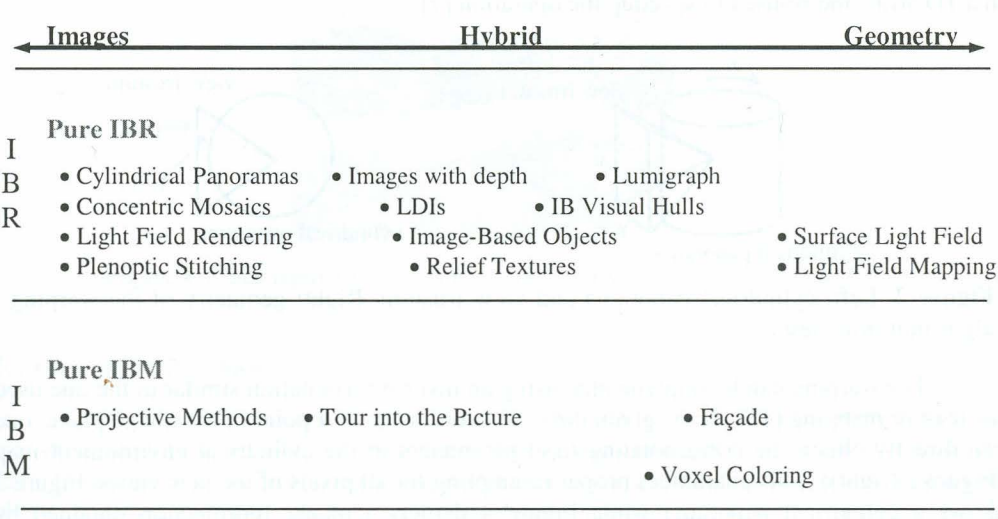

Figure 1. Image-Based Modeling and Rendering techniques in the image-geometry spectrum.

2.1 Cylindrical Panoramas

Cylindrical panoramas (or cylindrical environment maps) are used to provide horizontal orientation independence when exploring an environment from a single point. The choice for cylindrical panoramas is justified by the facility to obtain such images and by the fact that a cylindrical map only curves in one direction, greatly simplifying the warping required to reconstruct new views. Cylindrical panoramas can be created using specialized panoramic cameras [37, 13], stitching together photographs acquired with a regular camera or using computer renditions. QuickTime VR®, developed by Apple Computers, supports the

[2] Regular, photographlike images encoding light intensity [41].

rendering from cylindrical panoramas and was the first commercial product to use image-based technology [6].

After QuickTime VR®, several viewers for rendering from cylindrical environment maps were implemented. These viewer support continuous panning in the horizontal direction and zooming. Panning in the vertical direction is limited by the vertical field of view of the cylindrical panoramas (usually 50 degrees). During panning, the visible portion of the panorama is warped to produce correct planar perspective images. Figure 2 shows a depiction the cylindrical panorama with the associated view frustum and illustrates the basic geometry of the warping algorithm [7]. Since the cylindrical panorama does not curve in the vertical direction, the warping function reduces to a scaling operation applied to the vertical scanlines (columns) intersected by the view frustum. Note that the scaling factor varies from column to column. For a given field of view, the scaling factors can be pre-computed, stored in a 1D array and re-used to speedup the operation [7].

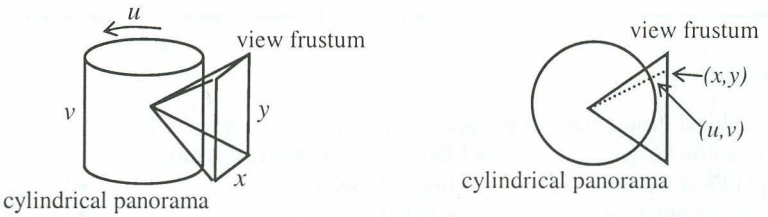

Figure 2. Left: cylindrical panorama and view frustum. Right: geometry of the warping algorithm (top view).

The warping can be implemented using an inverse formulation similar to the one used for texture mapping [15]. Thus, given the *(x,y)* coordinates of a point in the image plane, one can directly obtain the corresponding *(u,v)* parameters in the cylindrical environment map (Figure 2 (right)). This guarantees proper resampling for all pixels of the new views. Figure 3 shows a cylindrical panorama while Figure 4 depicts a planar reprojection obtained by warping a portion of Figure 3. Antialiasing is implemented by averaging neighbor pixels in the panorama. To improve performance, antialiasing can be disregarded during panning. Vertical panning is implemented as a two-step process [7]: first, the visible portion of the panorama is mapped to a plane parallel to the axis of the cylinder using the warping technique described above; next, the resulting image is mapped to the desired image plane

Figure 3. Cyllindrical Panorama. Courtesy of Philip Morgan, UK.

using a planar projective transformation. During a vertical panning, the images undergo two resamplings, which tends to degrade the quality of the final image.

Zooming is obtained by changing the field of view of the frustum (Figure 2 (left)). In order to provide more detail when zooming in and to avoid aliasing during when zooming out, a pyramidal structure can be used to interpolated between the appropriate levels of resolution [6], similarly to the use of a mip-map pyramid [43].

The major limitation of systems based on environment maps (cylindrical, spherical or cubic) is to constraint the viewer to a single location (the viewpoint from where the images were acquired). Although the viewer is free to rotate around that point, he/she is not allowed to move, since translation promotes changes in visibility. As a result, the use of environment maps is not appropriate for walkthroughs of virtual environments.

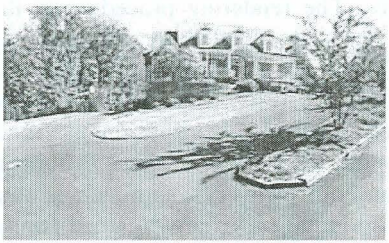

Figure 4. Planar reprojection obtained from the panorama shown in Figure 2.

2.2 Concentric Mosaics

Concentric mosaics are a generalization of cylindrical panoramas that allows the viewer to explore a circular region and experience horizontal parallax and lighting effects [39]. In this case, instead of using a single cylindrical image, slit cameras[3] are rotated along planar concentric circles. A series of concentric manifold mosaics [33] are created by composing the slit images acquired by each camera along their circular paths [39] (Figure 5). Thus, a cylindrical panorama is equivalent to a single mosaic for which the axis of rotation passes through the camera's center of projection, such as the case of camera C_0 in Figure 5. In a set of concentric mosaics, all slit images associated to a given column are acquired at the same angle (Figure 5). The use of slit images significantly reduces the amount of data required to approximate the plenoptic function [1] when compared to other IBR approaches such as Light Fields [18] and Lumigraph [14].

[3] A camera captures a single vertical-line image. It can be simulated with a regular camera by keeping only the central column of pixels of each frame.

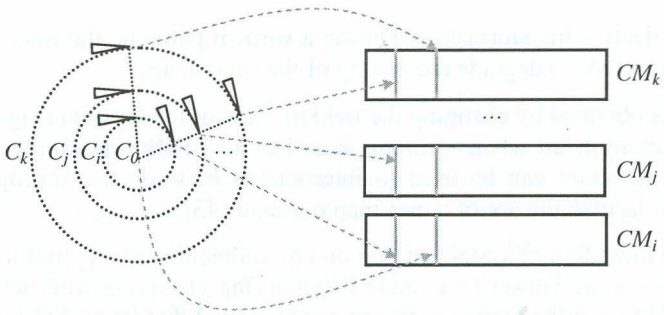

Figure 5. A concentric mosaic CM_l is created by composing all the slit images acquired by camera C_l along its curcular path.

During rendering time, the viewer is allowed to move inside the physical circular region covered by the mosaics. The rendering procedure is illustrated in Figure 6 for a desired viewpoint *P*. Novel views are created by composing (entire) slit images in the direction of the desired frustum (Figure 6). Note that this procedure implicitly assumes that all imaged surfaces are infinitely far away. Since this assumption is not satisfied in most cases, this algorithm introduces vertical distortions in the rendered images. The authors discuss some strategies to minimize the effect of these distortions in the final images [39]. Another limitation of the Concentric Mosaics approach is the lack of vertical parallax, resulting from the fact that all original views are obtained from a horizontal planar region. Horizontal parallax, however, seems to be, by itself, a strong enough clue for 3D perception, possibly due to the fact that our eyes tend to remain relatively at constant level as we walk [39].

As the radial distance between two adjacent circles increases, the number of slit images available for reconstruction of a given novel view is reduced, requiring more interpolation of the original data to fill in the image plane of the novel view (Figure 6). In practice, acquiring dense concentric samples is obtained with the use of a regular camera moved along a circular path [39, 33]. In this case, the camera can be oriented either tangent or normal to the circular path. The captured images are used to construct multiple concentric mosaics through resampling. When this acquisition strategy is used, the region that the viewer can explore is determined by the horizontal field of view of the camera [39].

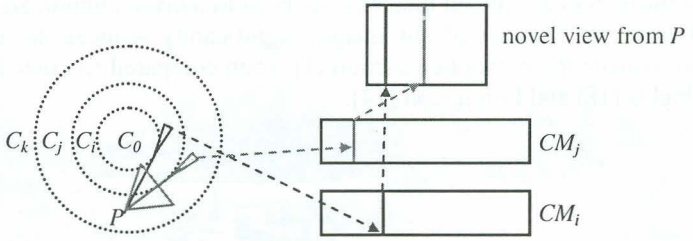

Figure 6. Rendering with concentric mosaics (after [Shum99]). For a given viewpoint *P*, the novel view is created by composing slit images captured by cameras C_l , $l = 0..k$, at the angles in which a line through *P* is tangent to the circle define by C_l .

2.3 Plenoptic Stitching

Plenoptic stitching [2] is a clever technique that gives the viewer the ability to explore (walkthough) unobstructed environments of arbitrary sizes and shapes. In order to provide appropriate sampling for most viewpoints in the environment, an omnidirectional video camera[4] [27] is moved over a grid. Along these paths, the position and orientation of the camera are tracked and stored in synchrony with the corresponding video frames. The intersections among the several paths define *image loops* [2] (Figure 7). Each loop is segmented as part of a pre-processing. During a walkthrough, the image loop containing the current viewpoint is used to reconstruct the desired view. Figure 7 (left) shows a schematic representation of a grid pattern used to explore a particular environment. Each line segment represents an individual path. Groups of segments delimiting a closed portion of the environment define an image-loop. The triangle (Figure 7) indicates the camera position and field of view.

Novel views are constructed by warping and composing columns of pixels obtained by resampling the original omnidirectional images (Figure 7 – right). The planes (represented as dashed line in Figure 7 (right)) defined by the novel viewpoint and each of the columns of the desired image plane are used to select columns of pixels from omnidirectional images located in front and behind the current viewpoint. If such a plane does not pass through the center of projection of any omnidirectional camera (*e.g.*, dotted line in Figure 7 (right)), the color information is interpolated from the two closest images.

The omnidirectional images are acquired at eye-level, preventing the system from exploring vertical parallax. However, the judicious combination of feature tracking and images acquired both in front and behind of the desired viewpoint allows the approach to warp the resampled columns, avoiding the undesirable vertical distortions observed in renderings produced with concentric mosaics.

Omnidirectional cameras use some sort of convex mirror (paraboloidal, for the case of catadioptric cameras [27]) mounted in front of the lens in order to focus the 360-degrees

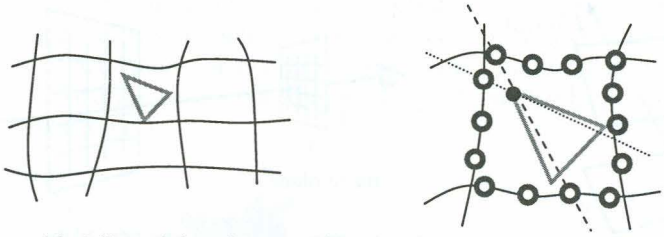

Figure 7. Left: grid followed by the omnidirectional camera during the sampling of the environment. Right: zoom in of the image loop containing the camera. The small circles represent the sampling locations.

[4] A camera with a 360° field of view.

horizontal field of view (FOV) onto each frame[5] of the video sequence. This has some practical implications. For instance, each frame records a large amount of information, reducing the spatial resolution at which the scene is sampled at each sampling position, which results in undesirable blurring. The geometry of the mirror lends to uneven resolution across the vertical FOV. Also, a single hemisphere is imaged, constraining the viewer from looking at portions of the environment located above the horizon line defined by the border of the mirror. The use of two omnidirectional cameras can be used to cover the entire FOV [2], but registering images acquired from two different centers of projection requires extra care.

2.4 Light Field and Lumigraph

The light field is a function that describes, for any given point, the radiance perceived in a particular direction in free space [18] (this is equivalent to the definition of plenoptic function [1]). Light field [18] and lumigraph [14] rendering create novel views of scenes/objects by resampling a database of images representing a discrete sample of the plenoptic function. In this representation, a ray is parameterized by its intersections with two parallel planes, a structure known as a *light slab* [18] and depicted in Figure 8 (left).

In order to create the image database, a regular orthogonal lattice is used to define the camera positions for capture. Such a lattice is associated to the *uv* plane, which is also known as the *camera plane*. At each lattice position, a picture is taken using the *st* plane as image plane (Figure 9 - left). Since the positions of both planes are fixed, the field of view becomes progressively skewed as the camera moves from the center of the camera plane towards its borders (Figure 9 - right).

Conceptually, the rendering of novel views is illustrated in Figure 8 (right). For each pixel of the novel view, one computes the intersections of the corresponding viewing ray with the camera and focal planes. The *(u,v)* coordinates of the intersection are used to select

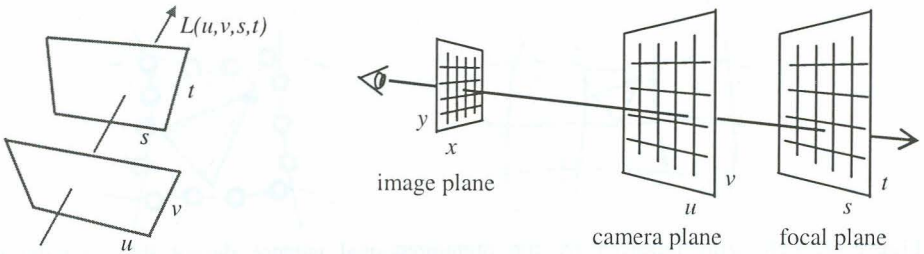

Figure 8. Left: Light slab. A light ray is parameterized by its intersections with two parallel planes. Right: Geometric representation of the image-database resampling (right) (after [18])

[5] Such a frame is called an omnidirectional image.

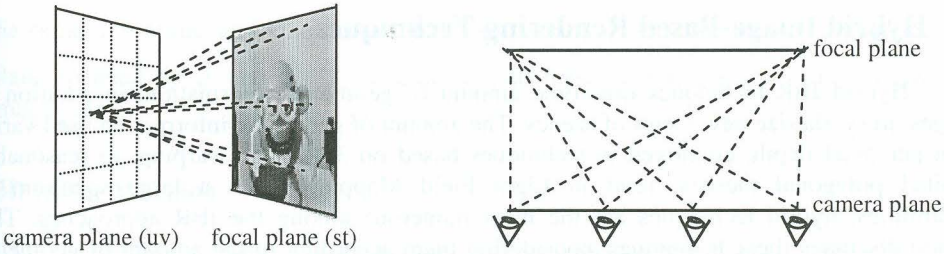

Figure 9. Left: each grid element in the camera plane corresponds to an image. Right: as the viewpoint moves from the center towards the borders of the camera plane, the view frustum becomes progressively skewed (after [18])

the image(s) to be used for resampling; the *(s,t)* coordinates of the intersection are used to select the actual pixel(s) from the selected image(s). The resampling strategy can range from nearest neighbors to quadralinear interpolation [18]. Note from the geometry of Figure 8 (right) that the transformations of coordinates from the image plane to both the camera and the focal planes are planar projective transformations and, therefore, can be efficiently implemented using texture hardware. Figure 10 shows an image rendered with the light field approach [18].

By imaging an object from several viewpoints, light field and lumigraph rendering can account for view-dependent effects, such as highlights. However, the large number of images required to avoid excessive blurring (due to the interpolation step performed during the resampling) significantly increase the storage requirements when compared to other IBR techniques. In the Lumigraph system, approximate geometry can be used during rendering time to perform depth correction [14], thus reducing the amount of blurring in the final images. Light fields and lumigraphs are not effective for representing inside-looking-out views of environments.

Although the original light field and lumigraph concepts assumed fixed scene illumination, interactive changes in the lighting conditions can be achieved by acquiring and processing images under different lighting conditions [44, 22]. As a viewing ray falls outside the valid range of camera plane in two-plane light-slab configuration, no information is returned for that ray. An arrangement containing multiple light-slab is necessary to cover the entire surface of a given object.

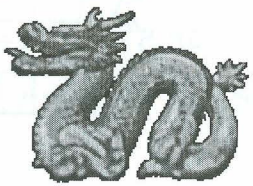

Figure 10. Stanford dragon: single slab 32 x 32 x 256 x 256 light field [20].

3 Hybrid Image-Based Rendering Techniques

Hybrid IBR techniques use some amount of geometric information, in addition to images, to synthesize new views of scenes. The amount of geometric information used varies from per pixel depth, employed in techniques based on 3D image warping, to reasonably detailed polygonal meshes, used in Light Field Mapping. With a large spectrum of possibilities, hybrid techniques are the most numerous among the IBR approaches. This section discusses these techniques, considering them according to the amount of geometric information required.

3.1 Techniques Based on Range Images

Range images, or images with depth, are used by a large class of IBR techiniques, including 3D image warping [26], post-rendering warping [23], layered depth images (LDIs) [38], image-based objects (IBOs) [30], LDI trees [46] and relief texture mapping [31]. An image with depth is a pair $\{i_d, K\}$, where i_d is a digital image and K is a camera model associated with i_d. Each element of the color space of i_d is augmented to include a scalar value per pixel representing the distance, in Euclidean space, between the sampled point and a reference entity. If K is a perspective-projection camera model, the image is called a *perspective projection image with depth* and the reference entity is K's center of projection. When K is a parallel-projection camera model, the image is called a *parallel-projection image with depth* and the reference entity is K's image plane. In an image with depth, the geometric content of the scene is represented implicitly by combining the per-pixel depth information with the camera model associated with the image.

Figure 11 (left) shows a model of a perspective projection camera. Vectors $\vec{a}$ and $\vec{b}$ form a basis for the image plane. The lengths of these vectors are, respectively, the horizontal and vertical sample spacing in Euclidean space. $\dot{C}$ is the center of projection (COP) of the camera, and $\vec{c}$ is a vector from the COP to the origin of the image plane [26]. Figure 11 (right) depicts the model for a parallel-projection camera. Vectors $\vec{a}$ and $\vec{b}$ have

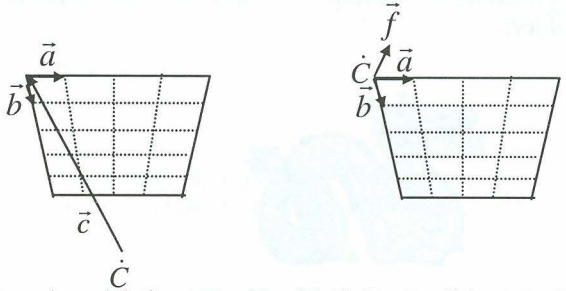

Figure 11 - Perspective pinhole camera model (left). Parallel projection camera representation (right).

the same definitions as in the perspective case. Vector $\vec{f}$ is a unit vector orthogonal to the plane spanned by $\vec{a}$ and $\vec{b}$ [31]. The tails of all these vectors are at $\dot{C}$, the origin of the image plane.

3D Image Warping

3D image warping [26] is a geometric transformation that maps a source image with depth i_s onto an arbitrary target view i_t. Thus, let $\dot{x}$ be a point in 3D Euclidean space whose projection into the image plane of i_s has coordinates (u_s, v_s). Using the perspective projection camera model of Figure 11 (left), the coordinates of $\dot{x}$ can be expressed as $\dot{x} = \dot{C}_s + (\vec{c} + u_s\vec{a} + v_s\vec{b})t_s(u_s, v_s)$, where the scalar $t_s(u_s, v_s) = \left|\dot{x} - \dot{C}_s\right| / \left|\vec{c} + u_s\vec{a} + v_s\vec{b}\right|$ is the ratio between the distances from $\dot{C}_s$ to $\dot{x}$ and from $\dot{C}_s$ to pixel (u_s, v_s), respectively (2). In matrix notation:

$$\dot{x} = \dot{C}_s + \begin{bmatrix} a_i & b_i & c_i \\ a_j & b_j & c_j \\ a_k & b_k & c_k \end{bmatrix} \begin{bmatrix} u_s \\ v_s \\ 1 \end{bmatrix} t_s(u_s, v_s) = \dot{C}_s + P\vec{x}\, t_s(u_s, v_s)$$

While the projection of $\dot{x}$ on an arbitrary target view can be obtained by explicitly recovering $\dot{x}$'s 3D coordinates and projecting them on the target image plane, such an intermediate step is actually unnecessary. The target pixel coordinates can be directly obtained by expressing the coordinates of $\dot{x}$ in both camera systems [26] (Figure 12):

$$\begin{aligned} \dot{C}_t + P_t\vec{x}_t t_t(u_t, v_t) = \dot{x} &= \dot{C}_s + P_s\vec{x}_s t_s(u_s, v_s) \\ \vec{x}_t t_t(u_t, v_t) &= P_t^{-1}[P_s\vec{x}_s t_s(u_s, v_s) + (\dot{C}_s - \dot{C}_t)] \\ \vec{x}_t &\doteq P_t^{-1}[P_s\vec{x}_s t_s(u_s, v_s) + (\dot{C}_s - \dot{C}_t)] \end{aligned} \qquad (1)$$

where $\doteq$ is projective equivalence, *i.e.*, the same except for a scalar multiple. Note from Equation (1) that arbitrary target views can be obtained from a single source image without knowledge about the depth associated with pixels from the desired views. Dividing Equation (1) by $t_s(u_s, v_s)$ and distributing P_t^{-1} gives:

$$\vec{x}_t \doteq P_t^{-1}P_s\vec{x}_s + P_t^{-1}(\dot{C}_s - \dot{C}_t)\delta_s(u_s, v_s) \qquad (2)$$

where $\delta_s(u_s, v_s) = 1/t_s(u_s, v_s)$ is called the *generalized disparity* of source pixel (u_s, v_s) [26].

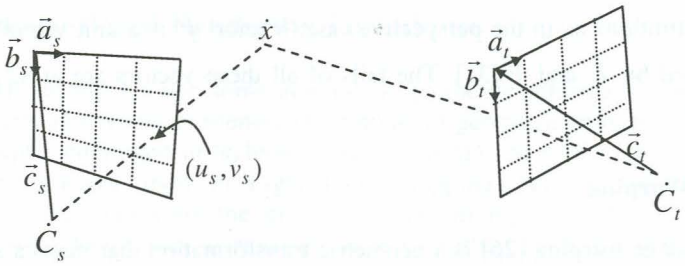

Figure 12. Point $\dot{x}$ in Euclidean space projected onto both source and target image planes.

Equation (2) is called the *3-D image-warping equation* [26] and produces correct reprojections of the scene sampled by the source image from arbitrary viewpoints. Here, the term "correct" is defined as being consistent with the projections of the corresponding static three-dimensional models represented in Euclidean geometry [26]. Figure 14 (left) shows a computer generated image with depth of a cathedral. Figures 14 (center) and (right) show two views of the same cathedral obtained by warping Figure 14 (left) to new viewpoints. Note that the samples present on the source image have been correctly reprojected on the new views. The existence of black areas in Figures 14 (center) and (right), usually referred to as *disocclusion artifacts*, correspond to surfaces not visible in the source image, but exposed in the novel views. In order to minimize the artifacts introduced by disocclusion events, one can warp multiple images to the desired view or use depth images with multiple color and depth values along each sampling ray (pixel). Such images are known as layer depth images (LDIs) and will be discussed in the next section. From Equation (2), one observes that texture mapping onto a polygon is a special case of 3D image warping for which all pixels in the source image have the same depth.

During the warping of a single image, the visibility problem can be solved using a list-priority algorithm [26]. The algorithm specifies possible orders for warping the pixels of a perspective projection image with depth, so that correct visibility is achieved for arbitrary viewpoints without explicit depth comparison. Algorithm 3-1 summarizes the procedure, which is illustrated in Figure 13.

Figure 13 (left) shows two pinhole cameras. The white and gray regions represent the sheets defined by the epipolar geometry of the camera arrangement [11]; the arrows define

Find the projection of the target COP into the source image plane (the epipole);
Divide the source image into at most four sheets based on the coordinates of the epipole;
If the target COP is behind the source COP then
  for each resulting sheet
    warp its samples from the epipole towards the borders of the sheet;
else
  for each resulting sheet
    warp its samples from the borders of the sheet towards the epipole;

Algorithm 1. Occlusion-compatible order for perspective projection source images.

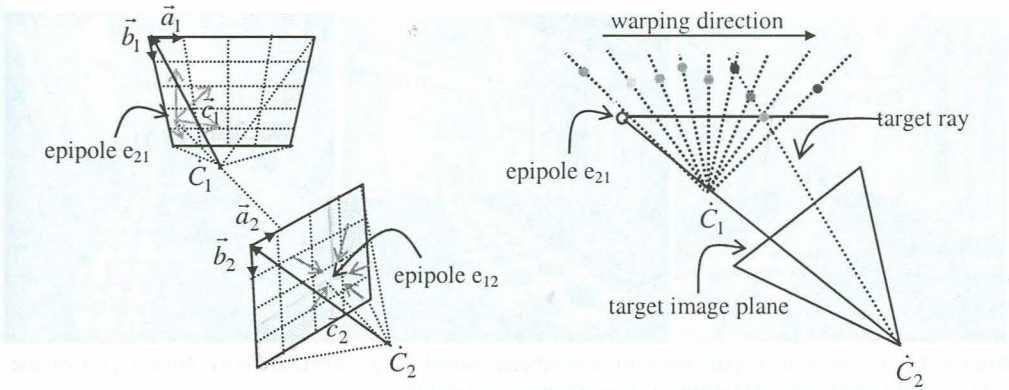

Figure 13. Two projective pinhole cameras. Left: the epipoles divide the images into at most four regions (shaded). The gray arrows specify the order in which the samples should be warped, if the image is the source one. Right: as multiple samples fall along the same ray, the closest to the target COP is warped last.

the warping order (see Algorithm 1). The intuition behind the occlusion-compatible order algorithm is illustrated in Figure 13 (right) for the case in which the target center of projection is behind the source COP: whenever multiple samples fall along a target ray, the one whose corresponding pixel is furthest from the epipole is the closest to the desired center of projection and, therefore, may safely overwrite previously warped samples. Thus, source pixels should be warped from the epipole towards the borders of the image.

Images with depth of real environments can be constructed by combining range images acquired with laser rangefinders and photographs. Nyland et al. [28, 25, 29] have built a system for acquiring and rendering image-based representations of real environments and objects. The results exhibit an extremely high degree of realism.

The regular structure of an image grid allows 3D image warping to be implemented using incremental computation, significantly speeding up the warping process. By reprojecting the original samples, 3D image warping makes the implicit assumption that all imaged surfaces are diffuse. The occurrence of view-dependent effects in the original image introduces rendering artifacts.

In order to avoid or minimize the occurrence of disocclusion artifacts (see Figure 14 (center) and (right)), one may want to warp several images with depth to the same target view. In general, however, the occlusion-compatible algorithm order cannot be used with multiple source images. In this case, alternatives for solving visibility issues may include the construction and use of an image with multiple samples per ray (LDIs) or, a more general solution, the use of a depth buffer.

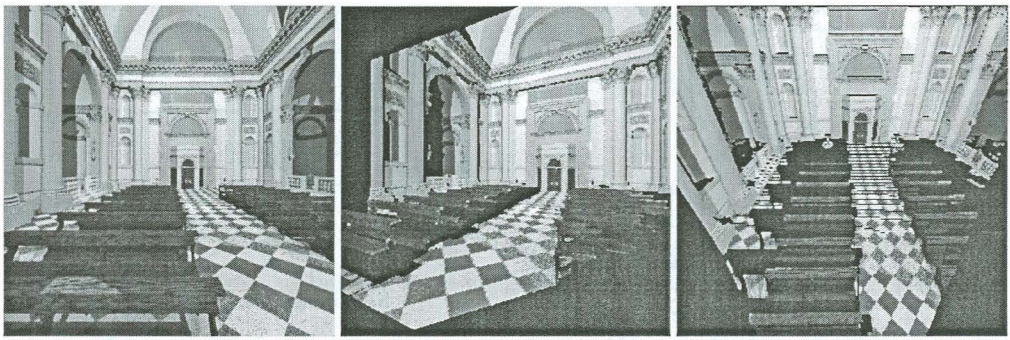

Figure 14. Image with depth: view of a synthetic model of a cathedral (left). New views of the cathedral obtained using 3D image warping (center and right).

Layered Depth Images

A layered depth image (LDI) [38] is an image with depth that supports multiple samples (color and depth information) per sampling ray (Figure 15). In this case, each element of the image consists of an ordered list of samples. LDIs that can be warped using Equation (2) in occlusion compatible order. In this case, as a "pixel" is ready for warping, all samples along the corresponding ray are warped [38]. The sample furthest from the novel COP is warped first and the closest is warped last.

By sampling multi-layer surfaces from a single center of projection, LDIs can reduce the occurrence of disocclusion artifacts while retaining the efficiency of 3D image warping. Such a feature can be exploited in combination with polygonal techniques to achieve rendering speedups [35]. While LDIs can reduce the occurrence of disocclusion artifacts, they cannot complete eliminate them. Like single-layer depth images, their effectiveness tend to be reduced as the desired viewpoint moves away from the LDI's COP. Another issue (also present in single-layer images) is the fixed sampling density defined at the time of the construction of the LDI. In order to reduce aliasing, some filtering is required. Avoiding aliasing in texture mapping has been efficiently solved with the use of mip mapping [43]. The equivalent of mip mapping in the context of LDIs is known as an LDI Tree [46].

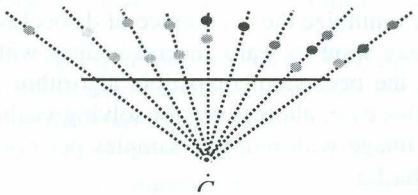

Figure 15. Layered Depth Image. Each samp ing ray may contain multiple samples.

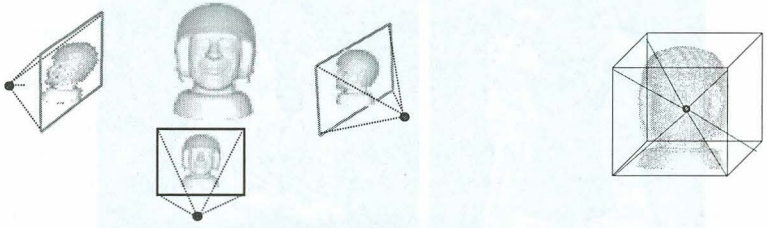

Figure 16. Left: object sampled from multiple COPs. Right: the samples are registered and reprojected onto perpendicular image planes (faces of a cube) sharing a single COP.

Image-Based Objects

While images with depth and LDIs can be warped producing correct reprojections of the represented surfaces, sampling the entire surface of a 3D object requires imaging it from several viewpoints. Image-based objects (IBOs) [30] provide a compact image-based representation for 3D objects that can be rendered in occlusion-compatible order (OCO). An image-based object is constructed by acquiring multiple views of the object, registering and resampling them from a single COP onto the faces of a parallelepiped, as illustrated in Figure 16. Thus, each image-based object is represented by six LDIs [30]. The use of a parallelepiped allows such a representation to be decomposed into parameterized planar regions (faces of the parallelepiped) for which a warper can be efficiently implemented.

List-Priority Rendering. Consider a spherical reference image and a desired viewpoint. The viewing ray passing through the center of the spherical image intersects its surface at two points. The one closest to the viewer (or behind him/her, if the viewer is inside the sphere) is called the *positive epipole* (e+); The other one is called the *negative epipole* (e-). An occlusion compatible order for spherical reference images consists of warping samples from the negative epipole towards the positive one [26]. Since the IBO representation is topologically equivalent to a sphere, IBOs can be warped in occlusion-compatible order. In the case of IBOs, the following properties can be observed [30]: (1) Once a relative order among the LDIs has been established, each image can be independently warped using 3-D image warping algorithm for LDIs; (2) the positive and the negative epipoles fall within

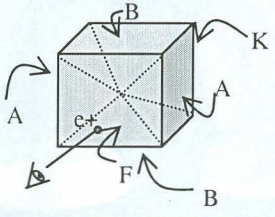

Figure 16. Faces of the parallelepiped are labeled with respect to the desired viewpoint. F contains the positive epipole (e+), whereas K contains the negative epipole.

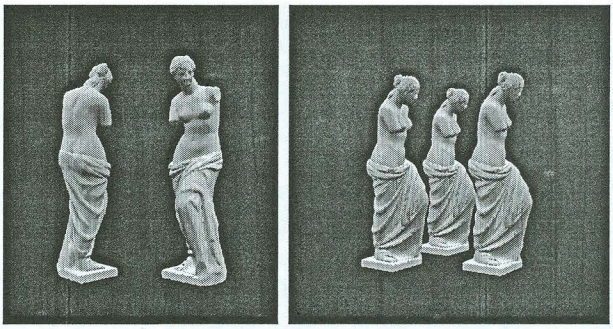

Figure 18. Left: two views of an IBO of the status of Venus. Right: multiple instantiations of the IBO in the same scene.

opposite faces; (3) There is no redundancy among images, *i.e.*, no sample is seen in more than one face; (4) The whole field of view is covered.

The line connecting the desired and the IBO COPs intersects the cube at opposite faces (Figure 17) and defines an occlusion compatible order for warping the whole object. The face containing the negative epipole (e-) must be warped first, while the face containing the positive epipole (e+) must be warped last. In Figure 17, the faces containing the positive and negative epipoles, are called F, and K, respectively. Notice that this classification is relative to desired viewpoint. The other two pairs of opposite faces are called (A, A'), and (B, B'). Thus, warping the faces of the cube in the order (K, B, A, A', B', F), or (K, B, A', A, B', F) produce correct visibility from the desired view position [30]. Since the set of rays emanating from the object COP covers a solid angle of 4π steradians, some care should be taken in order to guarantee that multiple samples along a ray are always warped from back to front with respect to the desired viewpoint.

Figure 18 shows several views of an IBO of the statue of Venus consisting of 6 150x150 LDIs and created from four images of a 90,044 polygonal model rendered in 3D Studio MAX. Figure 19 shows an IBO of a real object generated using four images acquired with a laser rangefinder [30]. IBOs can be explored from arbitrary viewpoints and can be combined with other IBOs or instantiated multiple times (Figure 18) in order to create more

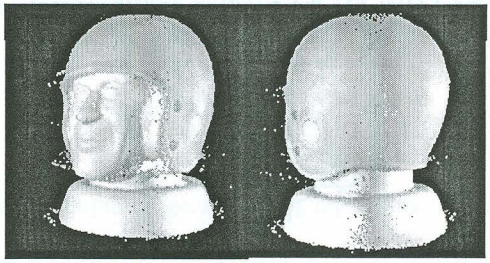

Figure 19. Views of an IBO create by scanning a real object with a laser rangefinder.

complex scenes. Visibility among multiple non-interpenetrating IBOs can also be solved in occlusion-compatible order. In case the IBO bounding boxes interpenetrate, a depth-buffer solution is required for solving visibility.

Relief Textures

Relief texture mapping is an extension to conventional texture mapping that supports the representation of 3D surface detail and view-motion parallax [31]. This effect is achieved by pre-warping the so-called *relief textures* and mapping the resulting images onto flat polygons. Figure 20 shows the relief texture-mapping pipeline. Relief textures, in turn, are parallel projection images with depth (*i.e.*, textures extended with a parallel-projection camera model and orthogonal displacements per texel). The use of parallel-projection images with depth greatly simplifies the pre-warping, and the rendering of complete 3D objects [31]. Figures 21 (left) illustrates the construction process of a relief texture: the corresponding surface is ortografically projected onto a reference plane and depth is measured as the per-texel distance from the plane to the sampled point. Figure 21 (right) shows the corresponding color and depth components of the resulting relief texture.

Figure 20. Relief texture mapping pipeline: pre-warping followed by conventional texture mapping.

The pre-warping depends on the relative position of the observer with respect to the polygon to be relief texture-mapped, and guarantees that the final rendering is equivalent to the view of the actual 3D surfaces. Figure 22 illustrates the steps of the relief texture-mapping pipeline for the case of a single polygon. The pre-warping step takes as input a

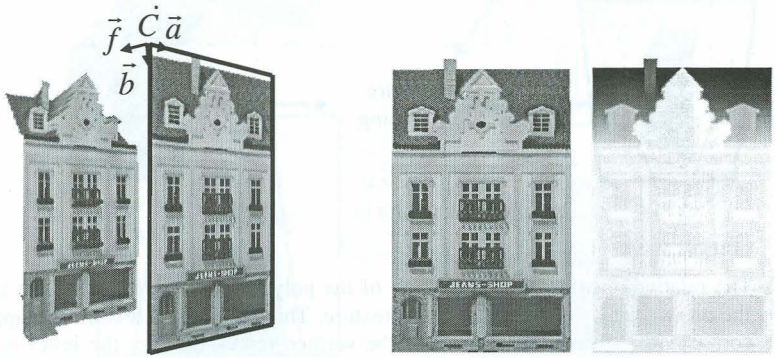

Figure 21. Left: Surface sampled to build a relief texture (orthographic projection). Right: Color image and depth map associated with a relief texture. Darker regions in the depth map indicate more distant surfaces.

relief texture and the parameters (relative position and orientation) of the polygon to be relief texture-mapped. As a result, it generates a pre-warped texture, which is conventionally mapped onto the polygon producing a correct view of the scene.

Pre-Warping. One needs to find a pre-warp p so that the composition $m \circ p$, where m is a standard texture-mapping transformation, is consistent with the projections of the original surfaces in 3D. Thus, the *pre-warping equations* can be obtained by factoring the texture mapping component from the 3D warping equation. The new coordinates (u_i, v_i) obtained after pre-warping a source pixel (u_s, v_s) are then given by:

$$u_i = \frac{u_s + k_1 displ(u_s, v_s)}{1 + k_3 displ(u_s, v_s)}$$
$$v_i = \frac{v_s + k_2 displ(u_s, v_s)}{1 + k_3 displ(u_s, v_s)} \qquad (3)$$

where k_1, k_2 and k_3 are constants for the given viewing configuration, expressed by

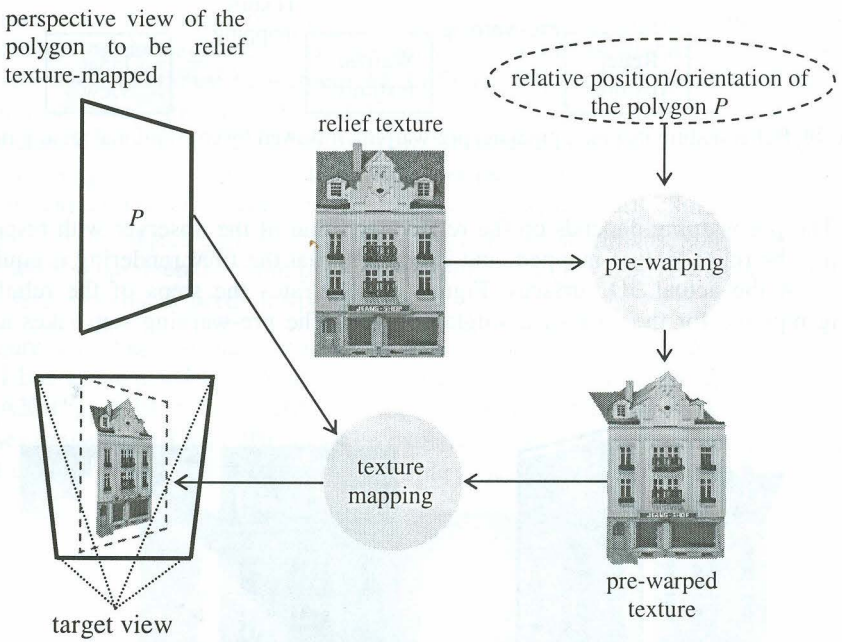

Figure 22. The relative position and orientation of the polygon P with respect to the target view are used by the pre-warping step to create a new texture. This pre-warped texture is mapped onto P to create a correct view (lower left corner) of the surface represented by the relief texture. The limits of the projection of P onto the novel view are highlighted with dashed lines.

$$k_1 = \frac{\vec{f} \cdot (\vec{b} \times \vec{c})}{\vec{a} \cdot (\vec{b} \times \vec{c})},\ k_2 = \frac{\vec{f} \cdot (\vec{c} \times a)}{\vec{a} \cdot (\vec{b} \times \vec{c})} \text{ and } k_3 = \frac{1}{\vec{c} \cdot \vec{f}}.$$

Together with *displ*(u_s, v_s), these coefficients determine the amount of shift applied to each source texel. Such a factorization proves to have many desirable properties. In particular, the coordinates of a texel in the pre-warped image can be computed independently from each other, *i.e.*, u_i does not depend on v_s and v_i does not depend on u_s. Also, when *displ*(u_s, v_s)=0, no computation is required. Although, at first, it may seem surprising that the pre-warping equations are one-dimensional, the geometric intuition behind this fact is actually very simple and follows from the separability of the perspective projection into orthogonal components. A rigorous treatment of this subject is presented in [32].

The independence between the u and v coordinates allows the pre-warp to be implemented as a two-step process using only 1D operations along rows and columns of the texture and using only two texels at any time. This is illustrated in Figure 23. The original relief texture is transformed into an intermediate representation by shifting each texel to its final column along its original row (top right). In the second step, texels are moved to their final rows (bottom right). The entire pre-warping is implemented in occlusion-compatible order using an adaptation of the original OCO algorithm for parallel projection images with depth [31]. The implementation of the pre-warp as a series of 1D operations over pairs of texels greatly simplifies the reconstruction task. It should also lead to a simple and efficient hardware implementation.

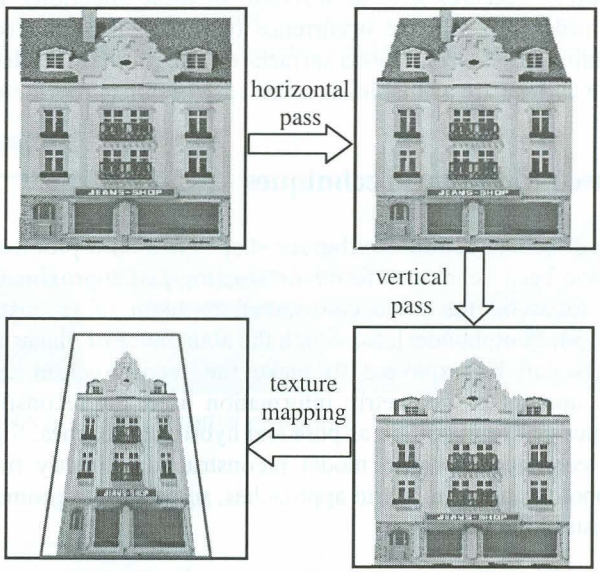

Figure 23. The pre-warp is implemented as a sequence of horizontal and vertical 1D warps.

Figure 24. Object represented by six relief textures associated with the faces of its bounding box (left). Rendering of the statue as two relief texture-mapped polygons shown with borders (center) and without borders (right).

Relief textures can also be used to model and render three-dimensional objects. An object representation consists of six relief textures associated with the faces of the object's bounding box. Figure 24 (left) shows a relief-texture representation of a statue originally modeled with 35,280 polygons. New views of an object can be obtained by pre-warping the relief textures and mapping the resulting images onto the faces of the box. Figures 24 (center) and (right) shows the rendering of the relief textured model for a given viewpoint with and without highlighting the relief texture-mapped polygon borders. Note the striking illusing of tridimensionality.

A relief texture is a single-layer representation and, therefore, not appropriate for rendering multi-layer structures such as foliages. In these situations, LDIs are probably a better choice. In order to avoid the occurrence of disocclusion artifacts, the surfaces are assumed to be continuous. Discontinuous surfaces can be properly rendered if multiple views covering the entire surface are available, as in the case of object representation.

4 Image-Based Modeling Techniques

Image-based reconstruction of arbitrary shapes is a hard problem and, so far, only a few techniques have been developed for reconstructing just approximate models [Seitz-17]. Most techniques focus on the more constrained problem of reconstructing architectural models [10, 19, 3, 34, Photobuilder], for which the abundance of planar surfaces, parallel and perpendicular lines can be exploited to make the reconstruction considerably simpler. According to the amount of geometric information used for reconstruction, image-based modeling techniques can be classified as pure and hybrid approaches. Similarly to the case in IBR, pure IBM techniques perform model reconstruction directly from the set of input images (usually photographs). In hybrid approaches, some simple geometric models are used to guide the reconstruction process.

4.1 Pure Image-Based Modeling Techniques

Projective Methods

These techniques exploit projective properties of the scene to reconstruct geometric models directly from a set of photographs. Due to the difficult to reconstruct arbitrary shapes and the simplicity of planar projective transformations, essentially all projective approaches are constrained to reconstruction of planar surfaces. The completeness of the resulting models depends on the surface coverage by the set of pictures.

Photo3D

Photo3D [3] is a commercial product that supports reconstruction of complete 3D models from multiple photographs, or incomplete models from a single image. It assumes that the intrinsic parameters of the camera (*i.e.*, focal distance, skew, aspect ratio and principle point) do not change from image to image. Furthermore, the camera is assumed to have no skew and known aspect ratio. The intrinsic parameters of the camera can be obtained from three vanishing points for three orthogonal directions. Given a known length, Euclidean reconstruction[6] of the model can be obtained. Figure 25 (left) shows a snapshot of Photo3D's interface taken during a modeling session when a partial model is being fit onto a new image.

Photo3D is relatively easy to use. However, the incremental construction of a model from multiple photographs causes the error introduced at each step to accumulate, usually resulting in inconsistent models. Figure 25 (right) shows a VRML model for the Computer Science Building of SUNY at Stony Brook created from 50 photographs using Photo3D. Inconsistencies in the coordinates of the geometric model were fixed by hand.

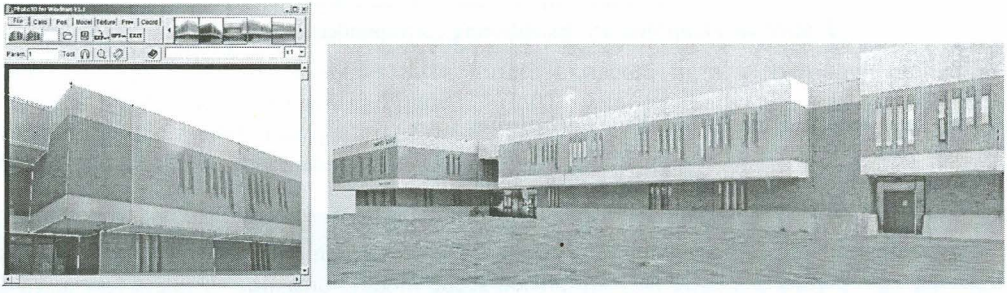

Figure 25. Photo3D interface: fitting a partial model onto a new image (left). A VRML model created using Photo3D (right).

PhotoModeler

[6] A Euclidean reconstruction has the actual dimensions of the original model, but no information about it absolute position and orientation. Position and orientation are defined relatively to the camera.

PhotoModeler [34] is another commercial product and requires a set of features correspondences among pairs of images as well as the intrinsic parameters of the camera used. Once the intrinsic parameters of the camera are known, the system can produce a metric reconstruction. Figure 26 shows PhotoModeler interface for establishing correspondences among features from a pair of images.

PhotoBuilder

PhotoBuilder [9] assumes the existence of several detectable parallel and perpendicular lines (in 3D) in the architectural elements to be reconstructed. The system recovers a 3D model and a set of camera matrices, one for each input image, from a set of uncalibrated photographs. The reconstruction is performed in three steps: for each image, the user selects a set of image edges, either parallel or perpendicular to each other in 3D; vanishing points implied by the sets of parallel lines are recovered and used to compute the intrinsic parameters and the corresponding projection matrix for each pose; the projection matrices are then used to determine the epipolar constraints [11] between images. The final model, represented as a textured 3D mesh, is obtained by triangulation using some specified corresponding points in different images.

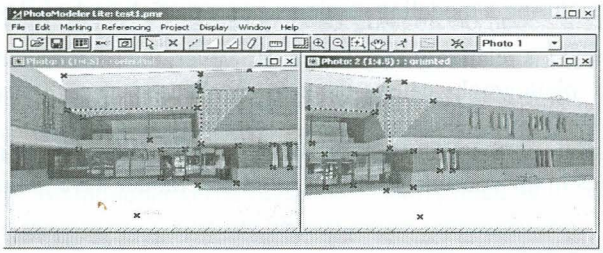

Figure 26. PhotoModeler. Establishing correspondences.

4.2 Hybrid Image-Based Modeling Techniques

Tour into the Picture

Tour into the picture [16], the simplest image-based modeling technique, recovers, from a single picture, an extremely simplified scene model consisting of just a few texture-mapped polygons. Despite the roughness of the underlying model, the technique can be used to create some convincing fly-through animations.

In order to create a model, the user must first split the scene into background and foreground objects. After the foreground objects have been removed, the holes left in the background image are filled using the colors of nearby pixels (Figure 27 (left) and (center)). The user should then specify (guess) a vanishing point for the picture, a rectangular region in the image and split the image in at most five regions (Figure 27 (right)). One of the regions

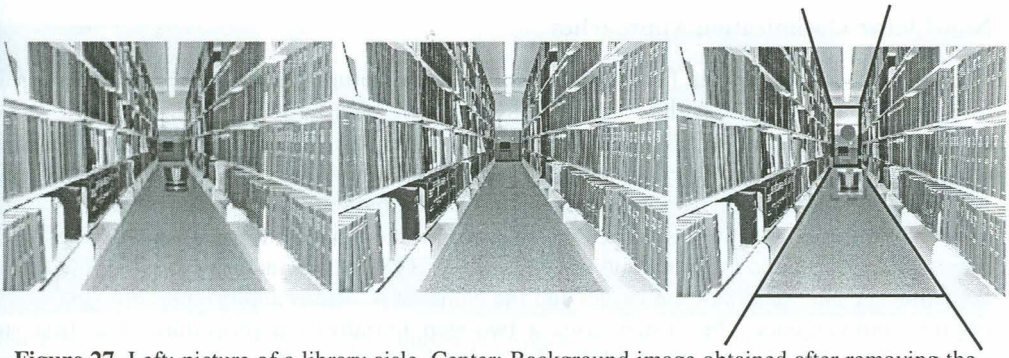

Figure 27. Left: picture of a library aisle. Center: Background image obtained after removing the foreground object (stool) and filling in the hole with the colors of adjacent pixels. Right: vanishing point (small circle). Dashed lines indicate the borders of the foreground polygon. Solid lines indicate the boundaries of image regions.

corresponds to a rectangle. The other regions are defined by radial line segments starting at the corners of the rectangle, along the direction defined by the vanishing point (Figure 27 (right). These five areas represent five faces of a parallelepiped, which will be rendered as texture-mapped polygons.

Recovering 3D coordinates for the vertices of these faces is extremely simplified by the fact that each two adjacent faces are perpendicular. The foreground objects are reintroduced in the scene as a series of texture mapped polygons, whose positions in 3D are computed with respect to the background model. Figure 27 illustrates the entire modeling procedure for a picture taken at a library. The small circle (Figure 27 (right)) represents the vanishing point. Dashed lines indicate the polygon associated with the foreground object (stool with two books) and solid lines delimit the five image regions. Note that the available textures may not cover the entire rectangles of the approximate model, as indicated in Figure 27 (right). Figure 28 shows three frames extracted from a flythrough animation created from the picture shown in Figure 27 (left). Note the remarkable realism obtained from this simple model. The quality of these results is, to some extent, due to fact that the aisle resembles the approximate box-shaped model.

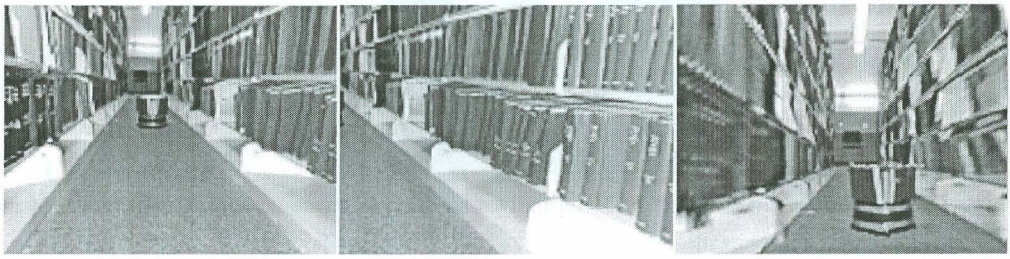

Figure 28. Views of a flythrough animation created from the picture shown in Figure 28 (left).

Non-Linear Optimization Approaches

The Façade system [10, 40] uses a non-linear optimization algorithm to reconstruct 3D textured models of architectural elements from photographs. Thus, given a set of calibrated photographs, the user provides an approximate parameterized model composed by simple 3D primitives (polyhedra) and establishes correspondences between the edges of the model, in 3D, and the corresponding edges in the photographs. The optimization consists in minimizing the sum of the disparities between the projection, into the picture, of the edges of the model and the corresponding image edges [10]. This is achieved by automatically adjusting the parameters of the model and the cameras positions and orientations. In order to improve convergence, the system uses a two-step initialization procedure. The first step estimates the camera rotation while the second estimates the camera translation and parameters of the model. Textures extracted from the original photographs are used to texture map the resulting model. Borshukov [4] extended Façade's original set of primitives to include arches and surfaces of revolution.

The system provides an intuitive interface based on correspondences between pairs of edges, features usually easily recognized by users. This removes the need to specify exact numerical values for the dimensions of the primitives, which are automatically recovered by the system. In order to enforce some properties of the model, the user can specify constraints for each individual block [10].

Canoma, produced by MetaCreations Inc. [5], was inspired by the Façade system. Although similar in spirit, Canoma uses a user-friendlier interface (Figure 29) that supports a large repertoire of 3D primitives. The primitives are superimposed to the pictures and adjusted to fit the 3D structures imaged by the photographs. Figure 29 shows a snapshot of Canoma's interface taken during a modeling session. Canoma has been recently acquired by Adobe Systems Inc. and is currently not being commercialized.

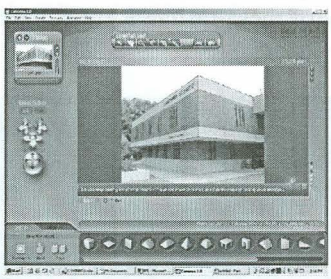

Figure 29. Canoma interface: a set of 3D primitives are used to create an approximate model of the actual environment.

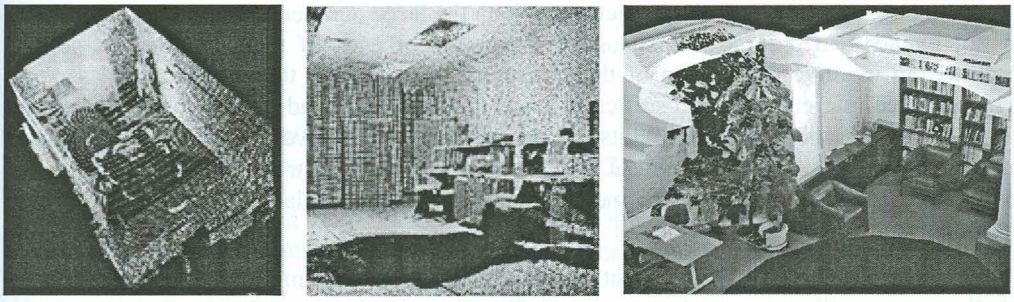

Figure 30. Left: 3D reprojection of the samples acquired by scanning an entire room. Center: detail from the scanning of a different room. Right: partial model of the UNC reading room created after registering range and coor data. The black circular areas on the floor are regions occluded by the scanner base. Images courtesy of Lars Nyland, UNC Chapel Hill.

Reconstruction from Range and Color Images

While most image-based modeling techniques are constrained to the reconstruction of approximate models built from planar surfaces or simple geometric primitives, the combined use of range images acquired with laser rangefinders and color photographs can, in principle, be used reconstruct models of arbitrary shapes [28, 25, Nylland01]. Figure 30 shows examples of real environments sampled and reconstructed using this approach. On the left, one sees the 3D reprojection of the samples acquired by scanning an entire room using a laser rangefinder. Note for instance, a person standing in one of the corners of the room. Figure 30 (center) shows an inside view of different room also scanned with a laser rangefinder. The black areas on the floor correspond to surfaces occluded from the scanner by its own base. Figure 30 (right) shows the result of combining range and color data acquired from a real environment to create very realistic renderings.

One advantage of this approach is its relative independence of the sampled geometry and short acquisition time. However, even when several sets of range and color images acquired from different viewpoints are available, occlusion and accessibility limitations may cause some surfaces not be sampled, introducing holes in the final model. These problems can be reduced using a technique described in [42]. As in the case of any other laser-based techniques, black and specular surfaces cannot be scanned directly. Ideally, the surfaces to be digitized should be a good diffuse reflector.

5 Conclusions

This paper surveys the state-of-the-art in image-based modeling and rendering techniques. The presentation was structured around a two-level classification system: first, the methods were grouped into modeling and rendering categories and further considered

according to the image-geometry spectrum. Given the large number of techniques, the survey is not intended to provide an exhaustive discussion of each of them. Rather, its goal to provide, an intuitive description of these methods, focusing on their fundamental aspects. The provided references should suffice to guide the interested reader through the specialized literature. Given the space constraints, it was not possible to cover all available techniques and a few of them had to be left out. Others, such as the LDI trees [46], were mentioned only in passing. This is not, by any means, a statement about the lack of importance of these techniques.

Despite the fast development of image-based techniqes in the last few years, many challenges are still waiting to be conquered. Creating complete models of real environments is still a difficult task due to occlusions, accessibility limitations and the reflective properties of some materials. Semi-automatic procedures for assisting the users to recover missing information (geometry and texture) and to solve possible ambiguities during reconstruction are needed.

Dealing with view-dependent effects and changes in illumination currently requires the use of massive databases [14, 18, 47, 8, 45, 44], which are not suitable for large environments. In order to make the use of these techniques practical, more compact and efficient representations are needed.

In recent years, the concept of an image has been extended in many ways to accomodate the special needs of particular algorithms. The research community became used to extensions such as images with depth [26], layered-depth images [38], multiple-center-of-projection images [36] and relief textures [31]. Adding depth to video sequences (*e.g.*, MPEG sequences) seems to be the natural next step in this exploration and a way to explore new medias.

Acknowledgements

The author would like to thank the following people and institutions for their kind contributions. Jianning Wang, from Stony Brook University, provided the images illustrating the interfaces of photogrametric software (Figures 25, 26, 29). Jiahui Pan, also from Stony Brook, implemented the Tour into the Picture and provided the corresponding illustrations. Lars Nyland and the UNC IBR group for providing Figures 30 and 14 (left). The Stanford Computer Graphics Laboratory for making their LightPack software and models publicly available. Philip Morgan for making the trial version of his panorama viewer available.

References

[1] Adelson, E. and J. Bergen. The Plenoptic Function and the Elements of Early Vision. In Computational Models of Visual Processing, pp. 3-20, MIT Press, Cambrige, MA, 1991.

[2] Aliaga, Daniel and I. Carlbom. Plenoptic Stitching: A Scalable Method for Reconstructing 3D Interactive Walkthroughs. Proceedings of SIGGRAPH 2001, pp. 443-450.

[3] Apollo Software. http://www.photo3d.com/eindex.html (July 2002).

[4] Borshukov, Georgi D. New Algorithms for Modeling and Rendering Architecture from Photographs, M.S Thesis, EECS department, UC Berkeley, 1997.

[5] Canoma. http://www.canoma.com (July 2002).

[6] Chen, Shenchang Eric. QuickTime VR – An Image-Based Approach to Virtual Environment Navigation. Proceedings of SIGGRAPH 1995. pp. 29-38.

[7] Chen, Eric S. and Gavin Miller. Cylindrical to Planar Image Mapping Using Scanline Coherence. United States Patent number 5,396,583. March 7, 1995.

[8] Chen, Wei-Chao, R. Grzeszczuk, J. Bouguet. Light-Field Mapping: Hardware Accelerated Visualization of Surface Light Fields. Proceedings of SIGGRAPH 2002, pp. 447-456.

[9] Roberto Cipolla, Duncan Robertson and Edmond Boyer. Photobuilder - 3D Models of Architectural Scenes from Uncalibrated Images. Conference on Multimedia Computing and Systems, June 1999. pp.25-31.

[10] Debevec, P., C. Taylor and J. Malik. Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach. Proceedings of SIGGRAPH 1996, pp. 11-20.

[11] Faugeras, Olivier. Three-Dimensional Computer Vision: A Geometric Viewpoint. MIT Press, Cambridge, Massachusetts, 1993.

[12] Foley, J. et al. Getting There: The Ten Top Problems Left. IEEE Computer Graphics and Application, Vol. 20, No 1, January 2000, pp. 66-68.

[13] Globuscope Panoramic Camera. http://www.everent.com/globus/ (July 2002).

[14] Gortler, Steven, et al.. The Lumigraph. Proceedings of. SIGGRAPH 1996. pp. 43-54.

[15] Heckbert, P. *Fundamentals of Texture Mapping*. Master's thesis. Technical Report No. UCB/CSD 89/516. Computer Science Division, University of California, Berkeley.

[16] Horry, Youichi, Ken Ichi Anjyo, and Kiyoshi Arai. Tour into the picture: Using a spidery mesh interface to make animation from a single image. Proceedings of SIGGRAPH 1997. pp. 225-232.

[17] Kutulakos, Kiriakos and Steven Seitz. A Theory of Shape by Space Carving. International Journal of Computer Vision, 38(3):197--216, July 2000

[18] Levoy, Marc and Pat Hanrahan. Light Field Rendering. Proceedings of SIGGRAPH 1996, pp. 31-42, 1996.

[19] Liebowitz, David, A. Criminisi, and A. Zisserman. Creating Architectural Models from Images. Proceedings of Eurographics 1999, Computer Graphics Forum, Volume 18, pp. 39-50, Sept. 1999.

[20] LightPack: Light Field Authoring and Rendering Package. Stanford Computer Graphics Laboratory. http://www-graphics.stanford.edu/ software/lightpack/.

[21] Lippman, Andrew. Movie-Maps: An Application of the Optical Videodisc to Computer Graphics. Proceedings of SIGGRAPH 1980. pp. 32-43.

[22] Malzbender, Thomas, D. Gelb, H. Wolters. Polynomial Texture Maps. Proceedings of SIGGRAPH 2001. pp. 519–528.

[23] Mark, William R., Leonard McMillan and Gary Bishop. *Post-Rendering 3D Warping*. Proceedings of the 1997 Symposium on Interactive 3D Graphics, (Providence, RI), April 27-30, 1997, pp. 7-16.

[24] Matusik, W., C. Buehler, R. Raskar, S. Gortler and L. McMillan. Image-Based Visual Hulls. Proceedings of SIGGRAPH 2000. pp. 369-374.

[25] McAllister, David et al. Real Time Rendering of Real World Environments. Rendering Techniques'99, pp. 145-160, Springer-Verlag, Wien, Austria, 1999.

[26] McMillan, Leonard. An Image-Based Approach to Three-Dimensional Computer Graphics. Ph.D. Dissertation. UNC Computer Science Technical Report TR97-013, University of North Carolina, April 1997.

[27] Nayar, Shree. Catadioptric Omnidirectional Camera. Proceedings IEEE Computer Vision and Pattern Recognition, 1997. pp. 482-488.

[28] Nyland, Lars, D. McAllister, V. Popescu, C. McCue, A. Lastra, P. Rademacher, Manuel M. Oliveira, G. Bishop, G. Meenakshisundaram, M. Cutts, H. Fuchs. The Impact of Dense Range Data on Computer Graphics. Proceedings of IEEE Multi-View Modeling

and Analysis Workshop (MVIEW99) (Part of IEEE Computer Vision and Pattern Recognition 99 – CVPR99), (Fort Collins, CO), June 23-26, 1999. pp. 3-10.

[29] Nyland, Lars, A. Lastra, D. McAllister, V. Popescu, C. McCue. Capturing, Processing and Rendering Real-World Scenes. In Videometrics and Optical Methods for 3D Shape Measurement, Electronic Imaging 2001, Photonics West, January 22, 2001. SPIE Vol. 4309.

[30] Oliveira, Manuel M. and Gary Bishop. Image-Based Objects. Proceedings of 1999 ACM Symposium on Interactive 3D Graphics. pp. 191-198.

[31] Oliveira, Manuel M., Gary Bishop and David McAllister. Relief Texture Mapping. Proceedings of SIGGRAPH 2000. pp. 359-368.

[32] Oliveira, Manuel M. Relief Texture Mapping. Ph.D. Dissertation. UNC Computer Science Technical Report TR00-009, University of North Carolina, March 3, 2000.

[33] Peleg, Shmuel and Joshua Herman. Panoramic Mosaics by Manifold Projection. Proceedings of the Conference on Computer Vision and Pattern Recognition 1997. pp. 338-343.

[34] PhotoModeler. http://www.PhotoModeler.com (July 2002).

[35] Popescu, Voicu, Anselmo Lastra, Daniel Aliaga and Manuel Oliveira. Efficient Warping for Architectural Walkthroughs Using Layered Depth Images. Proceedings of the IEEE Visualization'98. pp. 211-215.

[36] Rademacher, Paul and Gary Bishop. Multiple-Center-of-Projection Images. Proceedings of SIGGRAPH 1998. pp. 199-206.

[37] Roundshot 220VR. http://www.roundshot.com/cameras220VR.html (July 2002).

[38] Shade, Johnatan., et al. Layered Depth Images. Proceedings of SIGGRAPH 1998, pp. 231-242.

[39] Shum, Heung-Yeung and Li-Wei He. Rendering with Concentric Mosaics. Proceedings of SIGGRAPH 1999, pp. 299-306.

[40] Taylor, Camilo, Paul Debevec and Jitendra Malik. Reconstructing Polyhedral Models of Architectural Scenes from Photographs. Proceeding of the European Conference on Computer Vision, 1996. pp. 659-668.

[41] Trucco, Emanuele and Alessandro Verri. Introductory Techniques for 3-D Computer Vision. Prentice Hall, Upper Saddle River, New Jersey, 1998.

[42] Wang, Jianning and Manuel M. Oliveira. *Improved Scene Reconstruction from Range Images*. Computer Graphics Forum, Volume 21 (2002) Number 3 (to appear).

[43] Williams, Lance. Pyramidal Parametrics. Proceedings of SIGGRAPH 1983. pp. 1-11.

[44] Wong, Tien-Tsin, Pheng-Ann Heng, Siu-Hang Or and Wai-Yin Ng. Image-based Rendering with Controllable Illumination. Proceedings of the 8-th Eurographics Workshop on Rendering, St. Etienne, France, June, 1997, pp 13-22.

[45] Wood, Daniel et. al. Surface Light Fields for 3D Photography. Proceedings of SIGGRAPH 2000. pp. 287-296.

[46] Chang, Chun-Fa, Gray Bishop and Anselmo Lastra. LDI Tree: A Hierarchical Representation for Image-Based Rendering. Proceedings of SIGGRAPH 1999. pp. 291-298.

[47] Matusik, W., H. Pfister, A. Ngan, P. Beardsley, R. Ziegler and L. McMillan. Image-Based 3D Photography using Opacity Hulls. Proceedings of SIGGRAPH 2002. pp. 427-437.

[48] Seitz, Steven and C. Dyer. Photorealistic Scene Reconstruction by Voxel Coloring. Proceeding of Computer Vision and Pattern Recognition Conference, 1997. pp. 1067-1073.