Sea of Images

A Dense Sampling Approach for Rendering Large Indoor Environments

Daniel G. Aliaga and Thomas Funkhouser
Princeton University

Dimah Yanovsky
Harvard University

Ingrid Carlbom
Lucent Bell Labs

Computer simulation of real-world environments is one of the great challenges of computer graphics. Ultimately, computer simulation technologies should let an untrained operator walk through a city or building with a handheld device that captures a digital model, which can provide the realistic visual experience of walking through the environment interactively. Applications include remote education, virtual heritage, specialist training, electronic commerce, and entertainment. For instance, students can "visit" famous historical sites, archeologists can capture excavation sites as they evolve over time, and soldiers can train in simulated environments.

The sea of images approach uses novel methods for capturing, rendering, and managing large data sets to provide interactive and photorealistic walk-throughs of complex indoor environments.

The research challenge is to develop methods to capture, represent, and render large environments with photorealistic imagery from arbitrary viewpoints at interactive rates. Navigation in these environments is critical, and, in many cases, the application should convey the environment's scale and duplicate its visual richness. For example, a virtual visit to the Louvre is not particularly interesting if only one room is available, if the user is constrained to move along a preset sequence of viewpoints, or if the sculptures are displayed as synthetic renderings of coarsely detailed 3D models.

Current computer graphics techniques fall short of meeting these challenges. (See the "Related Work in Simulated Environments" sidebar for a discussion of some current methods.) Our sea of images approach provides new methods for acquiring, analyzing, representing, and rendering photorealistic models of complex indoor environments. Figure 1 gives an overview of our image-based rendering (IBR) walk-through system based on the sea of images approach. In this article, we describe the system and give results for its implementation in three environments of different sizes and types. Although we've described several of the components in conference proceedings,[1–4] this is the first article to provide a beginning-to-end description of the entire system.

System overview

A sea of images is a collection of images captured every couple of inches in a large environment. The representation provides a densely sampled 4D approximation to the plenoptic function, which describes the radiance leaving or arriving any point (x,y) from any direction (ϕ,θ). To provide real-time rendering, we address several data-management issues. In particular, we compress the acquired data into a multiresolution hierarchy so users can access it efficiently for continuous sequences of viewpoints, and we use time-critical algorithms to prefetch relevant image data and feature-based warping methods to reconstruct novel images. These techniques enable interactive walk-throughs of large environments with subtle viewpoint-dependent visual effects.

The advantages of our approach are fourfold:

- It enables accurate image reconstructions for novel views in environments with specular surfaces, extensive geometrical detail, and complex visibility changes.
- It doesn't require an accurate geometric model to produce novel images over a wide range of viewpoints.
- It provides a method for image-based modeling of a large, concave environment without sophisticated gaze planning.
- It supports rendering of inside-looking-out images in an IBR interactive walk-through system.

We believe no other IBR approach includes all of these features.

Figure 2 (see p. 24) outlines the system. The remainder of this article focuses on the algorithms we've developed for the three main components: *capture*, *image warping*, and *data management*.

Capture

The first step in our system is capturing a dense sea of images. We pursue a set of design goals that would

Related Work in Simulated Environments

Research in several fields—computer graphics, geometric modeling, computer vision, and so on—has attempted to capture and reproduce 3D environments.

Traditional computer graphics systems represent a scene as a set of 3D surfaces, light sources, and material-shading properties. After constructing a model, a standard computer graphics system can render it (for example, with ray tracing or scan conversion) to synthesize images of the scene from arbitrary viewpoints. This approach readily handles dynamic scenes and can leverage commonly available hardware for rendering polygons with hidden surface removal. However, constructing a computer-based model that faithfully represents the real world's visual richness is difficult, despite recent advances in interactive modeling tools and global illumination algorithms.

Image-based rendering (IBR) methods represent a scene by its radiance distribution without necessarily relying on a model of its geometry, lighting, and reflectance properties. Movie maps[1] and panoramas[2] are early image-based systems that sample the environment from a small, sparse set of viewpoints. More recent IBR systems take photographs of a static scene as input and construct a sample-based representation of the plenoptic function, which can be resampled to render photorealistic images for novel viewpoints.

View interpolation systems create novel views by interpolating pixel data between images from nearby viewpoints.[3] They often require dense depth or pixel correspondence, which are difficult to obtain robustly.

IBR approaches don't necessarily require geometric or surface reflectance models, and rendering times are usually independent of scene complexity. However, most IBR representations do not adequately represent scenes with specular surfaces and complex occlusions. Moreover, they usually support only a limited range of novel viewpoints in rendered images.

Computer vision techniques partially address these problems by creating geometric models of scenes automatically from sensor data. Researchers often combine structure-from-stereo, structure-from-motion, and/or laser-range scanning with texture images to produce photorealistic models. For instance, Koch et al. built visual models of several sites from video sequences using feature tracking and bundle adjustment to calibrate camera viewpoints, and dense stereo to recover geometry.[4]

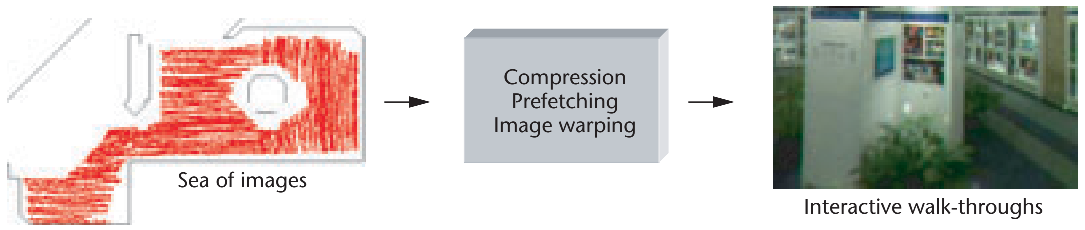
Teller et al. captured hundreds of high-resolution omnidirectional images in MIT's Technology Square and reconstructed coarsely detailed outdoor models of several buildings with textures.[5] Nyland et al. are constructing a 3D model of Monticello from multiple images registered with laser range scans.[6]

These approaches partially automate acquisition of the 3D model, enabling geometric scene analysis and higher quality image reconstruction for a wider range of viewpoints. However, current computer vision algorithms are generally not robust enough to produce accurate 3D models of large interior environments with specular surfaces, arbitrary textures, and complex visibility effects.

In short, no current system provides photorealistic walk-throughs of large interior spaces rendered from arbitrary viewpoints at interactive rates as does the sea of images approach.

References

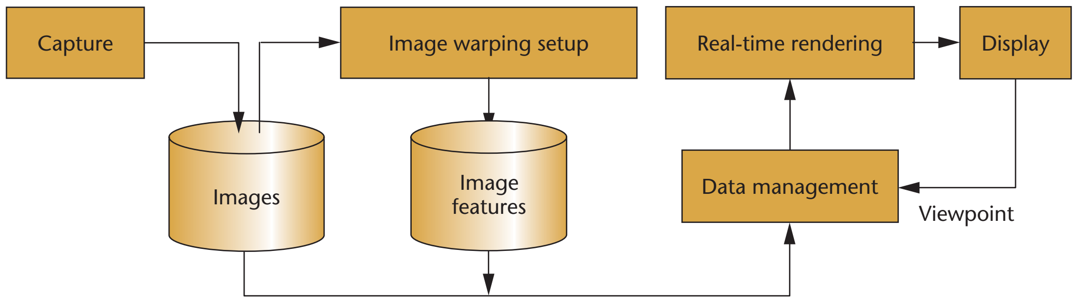
1. A. Lippman, "Movie-Maps: An Application of the Optical Videodisc to Computer Graphics," *Proc. ACM Siggraph*, ACM Press, 1980, pp. 32-42.
2. S.E. Chen, "Quicktime VR—An Image-Based Approach to Virtual Environment Navigation," *Proc. ACM Siggraph*, ACM Press, 1995, pp. 29-38.
3. S.E. Chen and L. Williams, "View Interpolation for Image Synthesis," *Proc. ACM Siggraph*, ACM Press, 1993, pp. 279-288.
4. R. Koch et al., "Calibration of Hand-Held Camera Sequences for Plenoptic Modeling," *Proc. IEEE Int'l Conf. Computer Vision* (ICCV 99), IEEE CS Press, 1999, pp. 585-591.
5. S. Teller et al., "Calibrated, Registered Images of an Extended Urban Area," *Proc. IEEE Computer Vision and Pattern Recognition* (CVPR 01), IEEE CS Press, 2001, pp. 813-820.
6. L. Nyland et al., "Capturing, Processing, and Rendering Real-World Scenes," *Proc. Videometrics and Optical Methods for 3D Shape Measurement Techniques, Electronic Imaging, Photonics West*, vol. 4309, SPIE, 2001.



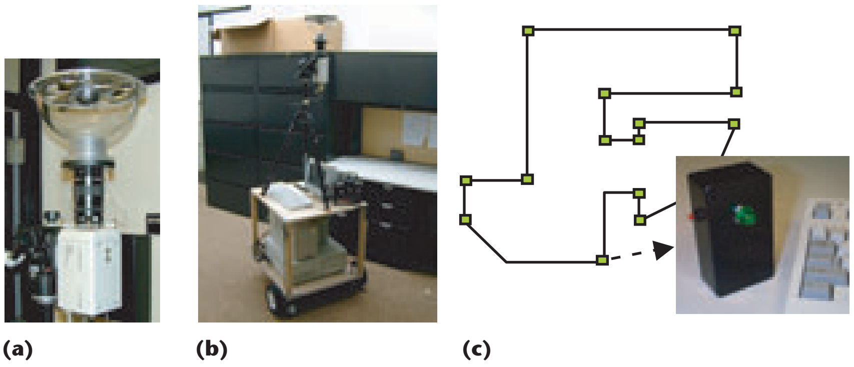
1 Sea of images. We capture a dense sea of images of a 3D environment, apply compression algorithms, prefetching schemes, and image-warping methods, and get interactive walk-throughs of large indoor environments.

let an untrained technician reliably capture images in a large indoor environment. Because site caretakers frown on complex installations, we also want a practical and easy-to-deploy approach.

Most previous IBR-capture methods rely on special-purpose hardware, specific environment content, or careful path and gaze planning. Levoy and Hanrahan,[5] for example, use a custom gantry to capture a dense set



2 System organization. Flow of data from scene capture to novel image display.



3 Image capture. (a) Our paraboloidal catadioptric camera. (b) The camera, a computer, and a battery sit on a mobile platform controlled via radio remote control. (c) We place fiducials (such as small portable light boxes) throughout the environment to quickly and reliably compute camera pose.

of images uniformly over a plane. Teller et al.[6] capture and calibrate outdoor images spaced by several to tens of meters over a campus-size environment and obtain initial measurements from a global positioning system. This approach doesn't work in indoor environments, however.

Large-scale (optical) trackers allow image capture but require a significant hardware installation. Gaze-planning algorithms for image capture in larger spaces are another option, but their success depends on accurate a priori knowledge of the environment's geometry. All of these methods are practical for small objects, single rooms, or outdoor environments, but don't easily extend to a wide range of dense viewpoints in complex interior environments of unknown geometry.

We capture a sea of images by moving an omnidirectional camera on a motorized cart through an environment, continuously capturing image data to disk. We built the capture system entirely from off-the-shelf components.

Camera

Our goal for the camera system is to capture large field-of-view (FOV) images to create an immersive experience for the observer. During image capture, the camera system should have the ability to simultaneously navigate through the environment, thus reducing capture time. With these requirements in mind, we chose a paraboloidal catadioptric camera,[7] shown in Figure 3a, and placed the camera, a computer, and a battery on a mobile platform controlled via radio remote control (Figure 3b). The camera captures a hemispherical FOV of the environment with a single center-of-projection (COP) and 1,024 × 1,024 pixel images. A FOV spanning 360 × 180 degrees in front of the mirror is reflected onto an image plane that is parallel to and lies in front of the mirror. Because the horizon is at the mirror's border, the vertical FOV spans 0 to –90 degrees. Because the camera has a single COP, applying simple transformations gives us cylindrical and planar reprojections.

We calibrate the intrinsic camera parameters using a custom calibration method.[1] Traditional methods assume the lens projection is perfectly orthographic. By relaxing this assumption, we obtain intrinsic parameters that are more accurate by an order of magnitude.

Camera pose

To facilitate image capture in large, multiroom environments, we need a simple and fast method for automatically determining images' camera poses—to do so manually would be prohibitive. Capturing an image every few inches inside a nontrivial environment (for example, a small museum) should take no more than an afternoon.

Researchers have proposed many approaches to camera pose estimation. For example, structure-from-motion techniques[8] track environmental features during an image sequence and obtain both camera pose and 3D information about the scene. Occlusion changes and drift, particularly in large interconnected spaces, hinder accurate and robust camera pose estimation. Taylor describes a system for computing pose from feature correspondences in panoramic images.[9] Because the user manually selects points and edges as features in several keyframes, this approach is not practical for large environments. Hardware trackers such as Polhemus, Ascension, and 3rd Tech compute the position and orientation of a sensor within a (small) environment, but they require complex and invasive installations to yield high-precision results.

Our pose-estimation algorithm tracks fiducials in the captured images and triangulates camera position and orientation from the fiducial projections. To ensure reliability and accuracy, we use small battery-powered light bulbs that the operator distributes in approximately known locations throughout the environment (Figure 3c). After the operator sets the initial camera position, the pose-estimation algorithm determines which fiducials might be visible in the current image using the current position estimate and a floor plan, tracks the fiducials from image to image, and triangulates camera position and orientation in real time. When capture is complete, we determine a globally consistent set of camera poses and fiducial locations using bundle adjustment (a nonlinear global optimization method) and get a dense set of omnidirectional images with calibrated camera parameters on a plane at eye height.

To determine pose estimation accuracy, we compute a bound on pose error using an analytic error model.[4] Given a fiducial configuration in an environment, an image sequence through the environment, and tracked fiducial projections, the model computes for each image the region of the environment that might contain the camera's COP. This error model also lets us formulate a tradeoff between fiducials and pose estimation accuracy. Consequently, this algorithm can suggest fiducial locations to produce a desired pose estimation error bound.

Image warping

Given a dense sampling of viewpoints with omnidirectional images, the next step is to produce high-quality views and support real-time rendering (20 frames per second or more). The algorithm should work with or without approximate geometry information. Moreover, to support a wide range of users, the rendering scheme should work on most PCs and graphics hardware.

Several approaches to image warping in an IBR system exist. The simplest approach is to interpolate images without warping.[5] This approach can easily lead to ghosting, however. A second approach is to reproject image samples according to depths derived from an approximate 3D model (or proxy). This approach requires accurate geometric models, which are often difficult to obtain.

A third approach is to estimate the depth value or pixel correspondence for every pixel in every image. Although this approach produces the most accurate warps, current methods for automatic camera calibration and depth acquisition calculate poor depth values in large environments with specular and occluding surfaces. Laser-scanning devices produce accurate depth estimates but are impractical for dense sets of viewpoints because of their long acquisition time (see the "Related Work in Simulated Environments" sidebar).

Another approach is to define a warp based on the correspondences between a relatively small set of distinctive image features (corners, for example). Unlike view morphing,[10] which simply produces a natural transition from image to image, this approach attempts to create plausible novel 3D views. Koch et al.,[8] for example, detect features in images from a video sequence and track them to earlier or later images while relabeling corresponding features. Their approach exploits the redundancy of images in a linear sequence but not in multiple directions, such as the images in our dense data set.

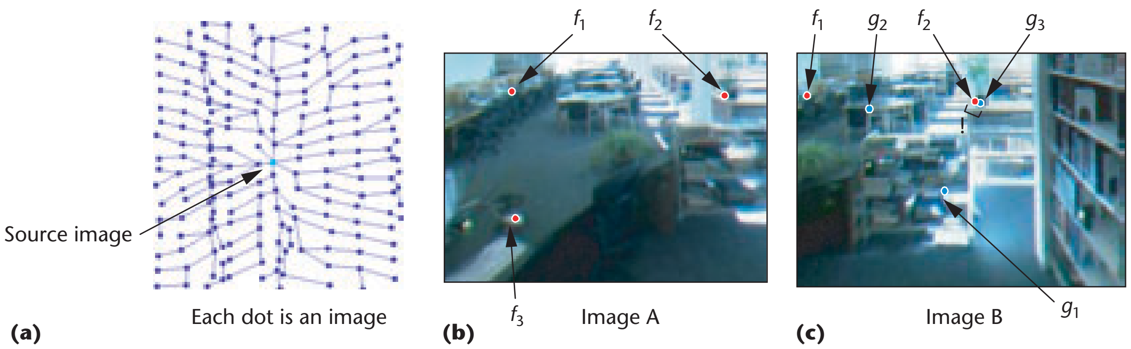
In any case, finding feature correspondences over a wide region of viewpoints is difficult because features drift during tracking and because a feature detector can select significantly different features in images of the same scene, even when the viewpoints are close.

Our approach exploits the redundancy within the dense image data set to establish feature correspondences over a wide region of viewpoints and creates a globally consistent set of image features.[3] We then use these features to warp and combine omnidirectional images, creating novel 3D views of the environment. This process involves several steps:

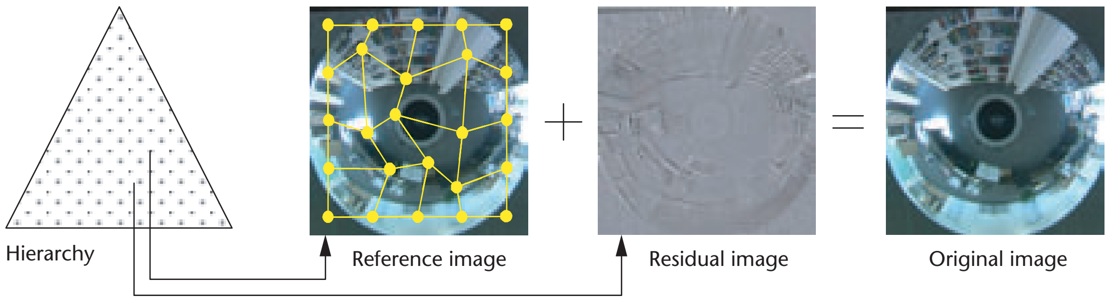
1. A feature detection algorithm selects initial image features in a subset of the reference images.
2. A feature tracking algorithm propagates them to other images within a local region. If features track to the same location in the same image, they are flagged as potentially corresponding.
3. A greedy graph-labeling algorithm selects feature correspondences and provides a consistent labeling in best-first order rather than the order in which they are detected.

This approach finds feature correspondences across a wide range of viewpoints in difficult scenes (such as scenes with occlusions), where a typical feature tracker would lose many features. Moreover, we can produce high-quality novel images in a walk-through system, even when reference images are separated by large distances or have poor camera pose estimates. Figure 4 (next page) shows an example propagation and matching sequence during feature globalization.

A key feature of our approach is that we consider two features potentially identical if they track to the same location in any destination image via any of multiple viewpoint paths. For example, consider trying to find the correspondences for two scene features, X and Y, between two images, A and B, captured from opposite ends of a room. Occlusion by an obstacle in the middle of the room might make it difficult to track feature X in A to feature Y in B along any single path of viewpoints between the two images. Our algorithm can find the correspondence of X and Y because it matches features that track the same location in any image (that is, not just correspondences along a single path, but along any path). Because of this redundancy, our algorithm finds potential feature correspondences more robustly and across a wider range of viewpoints than traditional feature-tracking methods.



4 Feature globalization. (a) Using the edges of a 2D Delaunay triangulation of the image viewpoints, we track outward from each source image along disjoint paths. (b) For a source image A, we detect features, such as f_1, f_2, and f_3, and add them to A's list of untracked features. (c) We iteratively track all untracked features to the next neighboring image B along each disjoint path. If a successfully tracked feature (such as f_1 or f_2) is within ε of a feature in B, the pair (f_i, g_j) is potentially corresponded.



5 Image hierarchy and compression. We combine a spatial image hierarchy with a compression algorithm. Original images are extracted from the hierarchy by warping reference images and adding in the residual images.

Data management

The third system component manages the large captured data sets, which usually contain thousands of images and require gigabytes of storage. To support interactive walk-throughs, we require image extraction along arbitrary contiguous paths through the environment—that is, we don't want to restrict image access to the capture order. Further, images near the novel viewpoint should always be available for image warping.

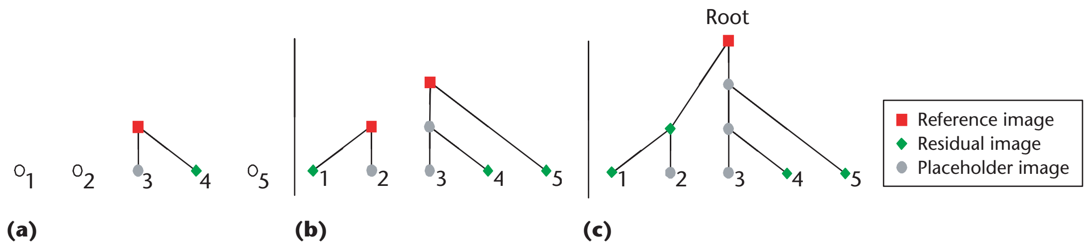
Previous work on data management for IBR representations has often focused on reducing overall storage requirements, assuming that the system will decompress and store the entire data set (or large subsets) in memory before rendering any novel image. Of course, this assumption is unrealistic in situations like ours where the size of the IBR representation exceeds the capacity of host memory. We assume that disk storage is sufficient to hold the entire data set. Thus, our representation should allow efficient disk access.

The sea of images data structure, illustrated in Figure 5, combines a multiresolution spatial image hierarchy with a compression algorithm to provide quick access to images during interactive walk-throughs and enable efficient compression of high-resolution images irregularly sampled over a plane. This approach ensures that images near the observer are always available in memory at runtime.

Compression

Our compression algorithm takes advantage of the data redundancy while simultaneously optimizing the data layout for cache coherence in a walk-through system. In today's computer architecture, hard disk storage is relatively abundant but bandwidth from disk into memory is limited. Hence, our compression strategy emphasizes data reduction for incremental decompression of nearby images rather than reduction of the total database stored on disk.

Although several IBR systems use schemes for organizing and compressing images, they rarely provide high compression and interactive access to thousands of high-resolution images in a manner suitable to walk-through applications. For example, the original Lightfield paper[5] describes a method that uses vector quantization and Lempel-Ziv coding. Follow-up work has investigated other compression schemes. Magnor and Girod,[11] for example, describe a discrete cosine transform (DCT) coder for regular Lightfields using data sets of only 1,024 images at 256 × 256 pixels each. Peter and Strasser describe a wavelet-based compression algorithm for the same Lightfield data sets that yields semi-interactive access using a complex caching scheme, which on failure yields approximately 3 frames per second.[12] Wilson et al. create a system for interactive walk-throughs of synthetic models and use spatially encoded video to represent image-based impostors.[13]



6 **Tree building. The five initial images are arranged in a line. From (a) to (c), image pairs (edges) are collapsed, producing a tree with reference images, residual images, and placeholder images (pointers to reference images higher up the tree).**

Our objective is to compress approximately 10,000 images—each at 1,024 × 1,024 pixels—of a real-world environment irregularly sampled over a plane, and provide high-compression performance, scalability, and interactive image access along arbitrary contiguous viewpoint paths. Moreover, we wish to take advantage of the interimage redundancy over the entire capture plane. None of the previous systems described support all these features.

Our compression algorithm arranges and compresses reference and residual images into a multiresolution hierarchy, as follows:

1. The algorithm creates a set of nodes representing each captured image at its viewpoint (Figure 6a).
2. Using a (Delaunay) triangulation of the nodes, it collapses the next highest priority edge, replaces the nodes of the edge end points with a new common parent node containing a reference image and pointing to a residual image and a placeholder image—that is, a pointer to an image higher up in the tree (Figure 6b).
3. The algorithm repeats the process for the remaining edges (Figure 6c).

Original captured images are the sum of a reference image (warped to the COP of a residual) and the residual image. Image warping uses either a globalized feature set or a simple 3D geometric proxy of the environment to map one image into another's view space. The tree is built bottom up using half edge-collapse operations. An edge-collapse operation consists of removing an edge by collapsing its two vertices into one vertex and locally retessellating the geometry.

Several options for creating our tree exist. We've found that a good compromise between compression, quality, and processing time occurs when we use inverse Euclidean image distance as the edge-collapse priority, and distribute reference images throughout the tree (for example, every five levels). To better register the proxy to each pair of images and reduce the residual image's energy, we optimize translation and rotation offsets. Moreover, image prediction directly relative to an ancestor reference image in the tree instead of an ancestor residual image does not give optimal compression, but it improves the working set size and decompression time.

Prefetching

Our prefetching algorithm should guarantee that the cache always contains a set of captured images surrounding the viewpoint. If the observer outpaces the prefetcher, the algorithm should support a graceful degradation of reconstruction quality by providing the runtime algorithm with a sparse set of images. When the observer moves more slowly or stands still, the algorithm will eventually catch up to the observer's viewpoint and display the best quality imagery.

Several systems support interactive rendering of large environments. Some approaches are geared toward synthetic models and use geometric simplification and visibility culling to obtain real-time rates. Another class of systems, specialized for outdoor models, uses image-based methods to render acceleration. In the same spirit as these systems, our research challenge is to build an interactive rendering system that supports quick access to an out-of-core data set. Unlike previous systems, we have a huge collection of images irregularly distributed over a plane that we must organize and load from disk in real time.

Our approach is to maintain in cache a linked list of tree nodes (a *cut*) through the multiresolution spatial hierarchy. This cut guarantees that at any location within the walkable space we always have a set of surrounding images in cache.

Prefetching uses an asynchronous process to maintain two linked lists of tree nodes. The evict list defines a cut through the tree such that all nodes on and above the cut are in cache. The fetch list consists of all the immediate children of the evict list nodes that can be loaded from disk. The algorithm uses the observer's latest viewpoint and predicted velocity to compute for all nodes in the lists the probability that their image is needed next. The prefetcher continuously swaps nodes between the two lists based on these probabilities. At runtime, the renderer refers to the current cut to determine which images to use for reconstructing the current view.

This approach provides a multiresolution way of loading images into the cache with graceful degradation and adaptive refinement. Even if the system can't load images from disk into the cache at video frame rates, the multiresolution prefetching strategy ensures that nearby images, corresponding to viewpoints at higher levels of the hierarchy, are available.

Table 1. Capture and image statistics.*

Environment	Setup (minutes)	Capture (minutes)	Fiducial Tracking (minutes)	Pose Optimization (minutes)	Area (square feet)	Average Distance to Images (inches)	Number of Images
Office	5	10	50	25	30	0.7	3,475
Library	15	17	60	20	120	1.6	1,947
Museum	30	82	160	30	900	2.2	9,832

* The three sets of images covered a total of 1,050 square feet, with 15,254 images at an average image spacing of 1.5 inches and average capture and calibration time of 2.8 hours per environment.

Table 2. Compression analysis showing incremental improvement in compression as different parts of the compression algorithm are enabled.

	Museum		Office		Library		
Operation	Contributions %	Mbytes	Contributions %	Mbytes	Contributions %	Mbytes	Average Contributions
Raw data	0	10,931	0	6,128	0	30,929	0
Intracoding	35	376.4	34	317.9	35	1,559.7	35
Image differencing	67	195.3	72	148.8	72	754.8	70
Optimization	100	129.7	100	106.6	100	538	100
Final compression	N/A	84:1	N/A	57:1	N/A	57:1	66:1

Results

We implemented the sea of images system in C/C++ using OpenGL, the OpenGL Utility Toolkit (GLUT), and the OpenGL User Interface Library (GLUI) on both a Pentium IV 1.7-MHz computer with an NVidia GeForce3 Ti 500 graphics card and an SGI Onyx2 R10000 250-MHz computer with InfiniteReality2 graphics.

So far, we've experimented with three environments:

- The Office is a small room, about the size commonly used for IBR demonstrations.
- The Library is the lobby area of a library at Princeton University.
- The Museum is the Lucent Bell Labs Technology Museum.

The museum is our most challenging test case because it has a concave floor plan, contains many specular surfaces (glass cases and brass plaques on the wall), and has complex geometry (plants on the floor and museum pieces in the cases).

Table 1 presents some capture process statistics. The capture process for each of the three environments—including setup, image capture, and optimization—took only a couple of hours. We also created a simple proxy of each environment using a measuring tape and a simple text-based polygon editor.

Setup included making lighting changes, configuring the camera (for example, making aperture adjustments), placing fiducials, and measuring and estimating fiducial setup in the environment, which we refined later in the bundle adjustment phase. The actual capture time, during which the motorized cart moved through the environment, was relatively short (10 minutes per 1,000 images). During the optimization time, we obtained optimized pose estimates for each image.

The entire capture process yielded a sea of images covering more than 1,000 square feet of walkable area at a density such that the average distance from a random point on the eye-height plane to its nearest image was 1.5 inches. The smallest pose error bound computed by our error model for the museum was 0.0011 times the diagonal of the environment (an accuracy of 99.89 percent or an error bound of 2.4 centimeters—the theoretical worst-case positional error of pose estimation). We conjecture that few other capture processes could both cover such a large space and sample fine details within the viewpoint space typical of an interactive walkthrough.

The compression algorithm produced useful compression ratios ranging from approximately 50:1 to 150:1 (50:1 compression yields high-quality imagery; compression ratios higher than 150:1 usually have too many visible artifacts). Table 2 gives an example breakdown of the contributions from different parts of the compression algorithm for an average compression ratio of 66:1.

Overall, the three environments contain 15,254 images and require 48 Gbytes of disk space in their raw form. Roughly one-third of the cumulative compression comes from intracoding the images (as JPEG files, for example), one-third from using image differencing and residuals, and one-third from computing per-difference image-optimized translation and rotation offsets for registering the proxy to the images. Using per-difference image optimization requires approximately five seconds of optimization time per difference image (bringing the total compression time to about five hours, three hours, and 14 hours for the office, library, and museum environments). Without this optimization, we get 41 to 56 times the compression of the original data (totaling about 0.5 hour, 1 hour, and 2.5 hours). The images and sequences in Figures 7, 8, and 9 use the optimized offsets.

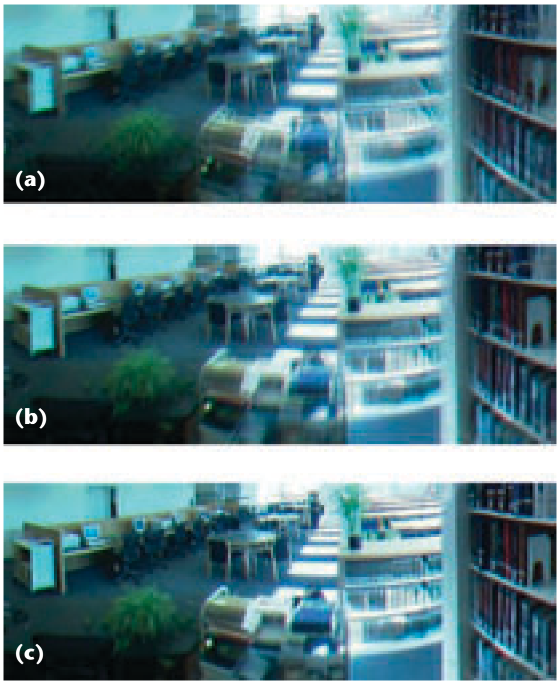
Figure 7 shows the quality of novel (cylindrical) projections reconstructed by our image-warping algorithm. Figure 7a simply blends neighboring reference images,[5] generating obvious ghosting artifacts. Figure 7b uses a proxy to warp and blend reference images,[14] yielding an improvement that depends on the proxy's accuracy. Figure 7c uses our approach and combines feature tracking with the construction and labeling of a correspondence graph, enabling correspondences over a wide range of viewpoints and producing high-quality reconstructions without requiring dense depth information or a full 3D reconstruction.

Our current runtime system reconstructs novel views on average from 20 to 30 frames per second. Figures 8 and 9 are images produced in real time. Figure 8a shows a specular highlight moving over the surface of a shiny bronze plaque. Figure 8b shows images of the same objects at several distances. Because the density of sampled viewpoints is so large, reconstruction quality is almost always similar to that of the captured omnidirectional images, as Figure 9 illustrates.

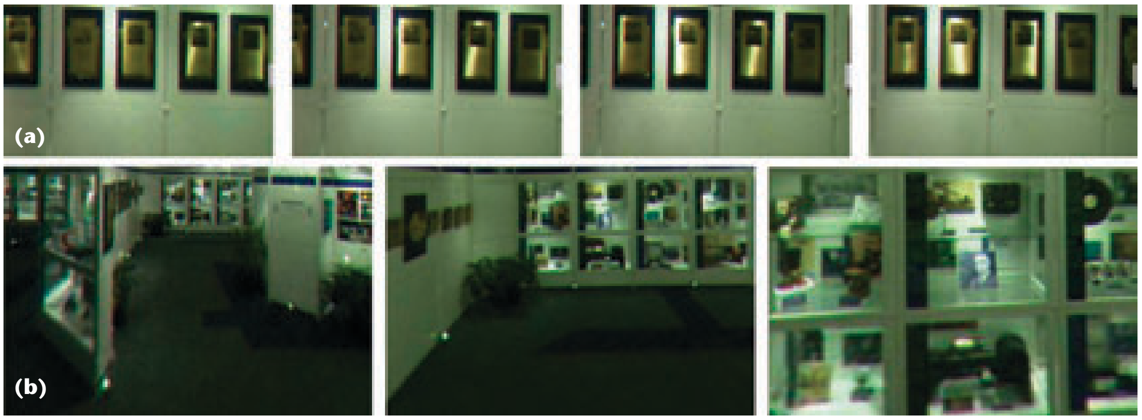
Conclusions and future work

Our system has several limitations. For example, the current capture device—a catadioptric omnidirectional camera—has limited resolution. However, the method described in this article would work with any omnidirectional video camera. We are investigating multiple-camera configurations that acquire higher resolution images.

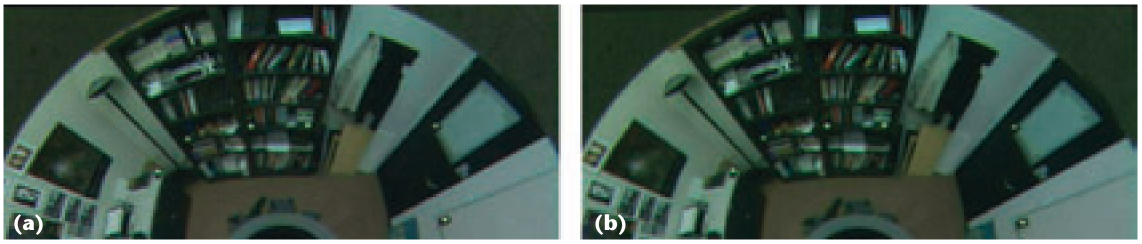
In addition, our current capture process requires some manual effort. An operator estimates fiducial locations, builds an approximate proxy model, and controls a motorized cart through the environment. Although these



7 Feature globalization. We use three methods to reconstruct novel views of a captured environment: (a) blending neighboring reference images, (b) using a proxy to warp and blend reference images, and (c) combining feature tracking with the construction and labeling of a correspondence graph.



8 Images reconstructed in real time: (a) with specular highlights moving over the surface of the bronze plaques and (b) using prefiltered multiresolution for far-to-near movement of a virtual observer.



9 Captured versus reconstructed comparison: (a) captured omnidirectional image and (b) reconstructed omnidirectional image for a novel viewpoint that is as far as possible from the surrounding images.

activities are not too burdensome (taking 15 to 112 minutes for our three environments), further automating the capture process would make the system easier to use.

Further, we'd like to expand our capture process to include images with a wider range of view directions and/or multiple viewpoint heights. We're also interested in developing algorithms to extract surface geometry and surface reflectance information from the image data set. Our dense sampling approach should facilitate this process. Given geometry and reflectance information, we could then investigate relighting issues and begin to explore editing the captured environment.

Finally, we're looking into large-scale system deployment over a network. Letting up to 1,000 simultaneous users access the same data sets and produce novel views over, for instance, the Internet will require deploying scalable systems with requirements similar to those of video-on-demand systems.

The sea of images approach is a fertile bed for future work. With it, researchers have access to a large and dense sampling of an environment. We believe it could lead to new 3D reconstruction algorithms, novel compression strategies, and new walk-through applications. ■

References

1. D. Aliaga, "Accurate Catadioptric Calibration for Real-Time Pose Estimation in Room-Size Environments," *Proc. IEEE Int'l Conf. Computer Vision* (ICCV 01), IEEE CS Press, 2001, pp. 127-134.
2. D. Aliaga et al., "Sea of Images," *Proc. IEEE Visualization* (Vis 02), IEEE CS Press, 2002, pp. 331-338.
3. D. Aliaga et al., "Interactive Image-Based Rendering Using Feature Globalization," *Proc. ACM Siggraph Symp. Interactive 3D Graphics* (I3D 03), ACM Press, 2003, pp. 163-170.
4. D. Aliaga and I. Carlbom, "Fiducial Planning for Error-Bounded Pose Estimation of a Panoramic Camera in Large Environments," *IEEE Robotics and Automation,* vol. 10, no. 2, June 2003.
5. M. Levoy and P. Hanrahan, "Light Field Rendering," *Proc. ACM Siggraph*, ACM Press, 1996, pp. 31-42.
6. S. Teller et al., "Calibrated, Registered Images of an Extended Urban Area," *Proc. IEEE Computer Vision and Pattern Recognition* (CVPR 01), IEEE CS Press, 2001, pp. 813-820.
7. S. Nayar, "Catadioptric Omnidirectional Camera," *Proc. IEEE Computer Vision and Pattern Recognition* (CVPR 97), IEEE CS Press, 1997, pp. 482-488.
8. R. Koch et al., "Calibration of Hand-Held Camera Sequences for Plenoptic Modeling," *Proc. IEEE Int'l Conf. Computer Vision* (ICCV 99), IEEE CS Press, 1999, pp. 585-591.
9. C. Taylor, "VideoPlus," *Proc. IEEE Workshop on Omnidirectional Vision*, IEEE CS Press, 2000, pp. 3-10.
10. S. Seitz and C. Dyer, "View Morphing," *Proc. ACM Siggraph*, ACM Press, 1986, pp. 21-30.
11. M. Magnor and B. Girod, "Data Compression for Lightfield Rendering," *IEEE Trans. Circuits and Systems for Video Technology,* vol. 10, no. 3, 2000, pp. 338-343.
12. I. Peter and W. Strasser, "The Wavelet Stream: Interactive Multiresolution Lightfield Rendering," *Rendering Techniques 2001: 12th Eurographics Workshop on Rendering*, 2001, pp. 127-138.
13. A. Wilson, K. Mayer-Patel, and D. Manocha, "Spatially Encoded Far-Field Representations for Interactive Walkthroughs," *Proc. ACM Multimedia*, ACM Press, 2001, pp. 348-357.
14. S. Gortler et al., "The Lumigraph," *Proc. ACM Siggraph*, ACM Press, 1996, pp. 43-54.

Daniel G. Aliaga *is an assistant professor of computer science at Purdue University. His research interests include computer graphics, image-based rendering, massive-model rendering, omnidirectional cameras, robotics, and computer vision. Aliaga has a BS in computer science from Brown University and an MS and a PhD in computer science from the University of North Carolina at Chapel Hill. He is a member of ACM Siggraph.*

Thomas Funkhouser *is an associate professor in the Department of Computer Science at Princeton University. His research interests include interactive visualization of large geometric models, acoustic modeling, image-based rendering, and 3D shape analysis. Funkhouser has a BS in biological sciences from Stanford University, an MS in computer science from the University of California, Los Angeles, and a PhD in computer science from UC Berkeley. He is a member of the IEEE Computer Society and ACM Siggraph.*

Dimah Yanovsky *is an undergraduate student at Harvard University, concentrating in computer science and economics. He has taken graduate-level courses at Princeton University and was a summer intern at Lucent Technologies Bell Labs. His research interests include 3D graphics, image-based rendering, online auctions, and economic modeling.*

Ingrid Carlbom *is a visiting scientist at Bell Labs and a visiting professor in computer science at Rutgers University. Her research interests include telepresence and virtual environments, acoustical modeling, scientific visualization, and biomedical imaging. Carlbom has a PhD in computer science from Brown University and an honorary PhD from Uppsala University, Sweden. She is a member of the editorial board of* Computers & Graphics *and co-editor in chief of* Graphical Models.

Readers may contact Daniel G. Aliaga at Computer Science Dept., Purdue Univ., 250 N. University St., West Lafayette, IN 47907; aliaga@cs.purdue.edu.