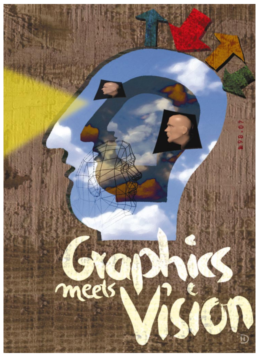
The Convergence of Graphics and Vision



Approaching similar problems from opposite directions, graphics and vision researchers are reaching a fertile middle ground. The goal is to find the best possible tools for the imagination. This overview describes cutting-edge work, some of which will debut at Siggraph 98.

Jed Lengyel
Microsoft
Research

At Microsoft Research, the computer vision and graphics groups used to be on opposite sides of the building. Now we have offices along the same hallway, and we see each other every day. This reflects the larger trend in our field as graphics and vision close in on similar problems.

Computer graphics and computer vision are inverse problems. Traditional computer graphics starts with input geometric models and produces image sequences. Traditional computer vision starts with input image sequences and produces geometric models. Lately, there has been a meeting in the middle, and the center—the prize—is to create stunning images in real time.

Vision researchers now work from images backward, just as far backward as necessary to create models that capture a scene without going to full geometric models. Graphics researchers now work with hybrid geometry and image models. These models use images as partial results, reusing them to take advantage of similarities in the image stream. As a graphics researcher, I am most interested in the vision techniques that help create and render compelling scenes as efficiently as possible.

GOALS AND TRENDS

Jim Kajiya, assistant director of Microsoft Research, proposes that the goal for computer graphics is to create the best possible tool for the imagination. Computer graphics today seeks answers to the question, "How do I take the idea in my head and show it to you?" But imagination must be grounded in reality. Vision provides the tools needed to take the real world back into the virtual.

There are several current trends that make this an exciting time for image synthesis:

- The combined graphics and vision approaches have a hybrid vigor, much of which stems from *sampled representations*. This use of captured scenes (enhanced by vision research) yields richer rendering and modeling methods (for graphics) than methods that synthesize everything from scratch.
- Exploiting temporal and spatial *coherence* (similarities in images) via the use of layers and other techniques is boosting runtime performance.
- The explosion in PC graphics performance is making powerful computational techniques more practical.

VISION AND GRAPHICS CROSSOVER: IMAGE-BASED RENDERING AND MODELING

What are vision and graphics learning from each other? Both deal with the image streams that result when a real or virtual camera is exposed to the physical or modeled world. Both can benefit from exploiting image stream coherence. Both value accurate knowledge of the surface reflectance properties. Both benefit from the decomposition of the image stream into layers.

The overlapping subset of graphics and vision goes by the somewhat unwieldy description of "image-based." In this article, I use *graphics* to describe the forward problem (*image-based rendering*) and *vision* for the inverse problem (*image-based modeling*). Inverse problems have been a staple of computer graphics from the beginning. These inverse problems range from the trivial (mapping user input back to model space for interaction) to the extremely difficult (finding the best path through a controller state space to get a desired animation). This article discusses only forward and inverse *imaging* operations (that is, light transport and projection onto a film plane).

Image-based rendering and modeling has been a fruitful field for the past several years, and Siggraph 98 devotes two sessions to the subject. The papers I discuss later come from those two sessions.

0018-9162/98/$10.00 © 1998 IEEE

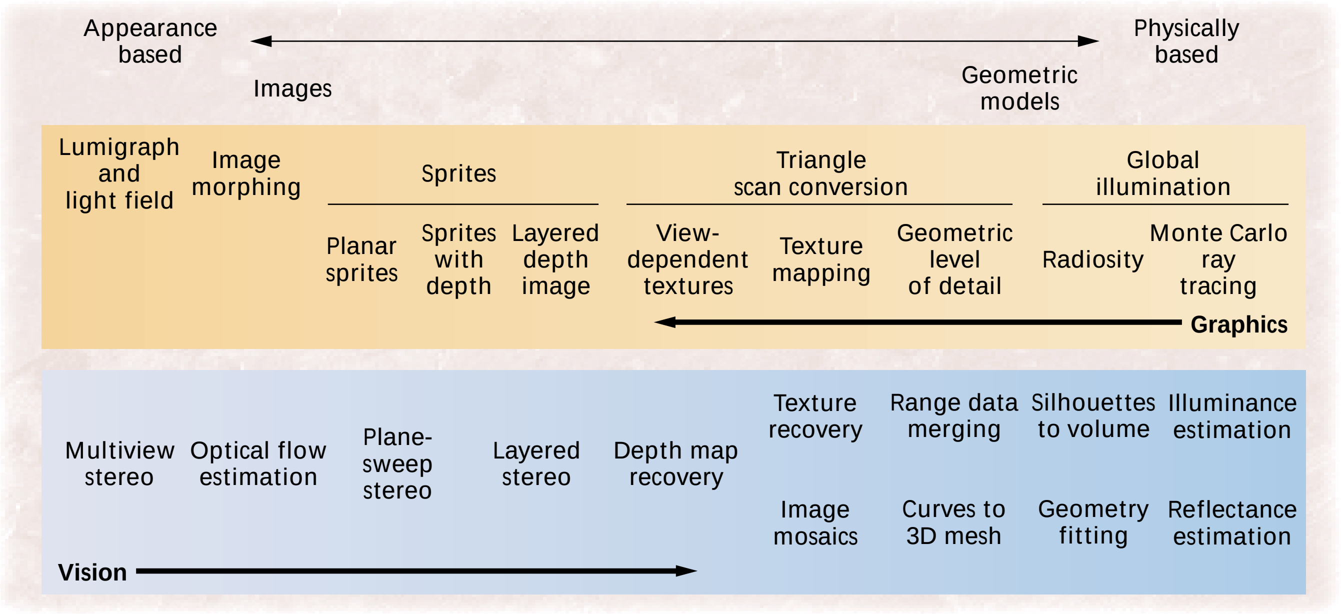


Figure 1. Graphics and vision techniques, on a spectrum from more image-based to more physical- or geometry-based. Traditional graphics starts on the right with geometric models and moves to the left to make images. Traditional vision starts on the left with images and moves to the right to make geometric models. Lately, there has been a meeting in the middle.

Images have been used to increase realism since Ed Catmull and Jim Blinn first described texture mapping from 1974 to 1978. In a 1980 survey, Blinn described the potential of combining vision and graphics techniques (http://research.microsoft.com/~blinn/imagproc.htm). Renewed interest in the shared middle ground arose just a few years ago.

The new research area is split into roughly three overlapping branches. Here I touch on the graphics articles in the field. (A longer bibliography is available at http://computer.org/computer/co1998/html/r7046.htm.) For the vision perspective, see the survey articles by Sing Bing Kang[1] and Zhengyou Zhang.[2] Figure 1 provides a schematic illustration of the spectrum of these research interests.

Image-based rendering

Given a set of images and correspondences between the images, how do you produce an image from a new point of view?

In 1993, Eric Chen and Lance Williams described how to interpolate, or flow, images from one frame to the next instead of having to do full 3D rendering.[3] Chen then introduced QuickTime VR in 1995, showing how an environment map sampled from a real scene could be warped to give a strong sense of presence. Independently, Richard Szeliski described the use of image mosaics for virtual environments[4] in 1996 and, the following year with Harry Shum, described improved techniques for combining multiple images into a single panoramic image.

In 1995, Leonard McMillan and Gary Bishop introduced the graphics community to plenoptic modeling, which uses the space of viewing rays to calculate the appropriate warp to apply to image samples; they also introduced the term "image-based rendering."[5] In 1996, Steven Seitz and Charles Dyer showed that the proper space for interpolation between two images is in the common coordinate system defined by the synthetic camera view.[6] That same year, Steven Gortler and his colleagues at Microsoft Research introduced the *lumigraph* (http://research.microsoft.com/msrsiggraph), and Marc Levoy and Pat Hanrahan of Stanford University introduced light field rendering (http://www-graphics.stanford.edu/papers/light). Both systems describe a dense sampling of the radiance over a space of viewing rays.

This year, Jonathan Shade and colleagues describe new image-based representations that allow multiple depths per pixel,[7] and Paul Rademacher and Gary Bishop describe a representation that combines pixels from multiple cameras into a single image.[8]

Image-based 3D-rendering acceleration

Given a traditional 3D texture-mapped geometric model, how can you use image caches (also known as *sprites*) to increase the frame rate or complexity of the model?

In 1992, Steve Molnar and colleagues described hardware to split the rendering of a scene into 3D rendering plus 2D compositing with z.[9] In 1994, Matthew Regan and Ronald Pose built inexpensive hardware to show how image caches with independent update rates exploit the natural coherence in computer graphics scenes.[10]

Paolo Maciel and Peter Shirley introduced in 1995 the idea of using image-based "imposters" to replace the underlying geometric models.[11] More recently, Jonathan Shade and colleagues[7] and, independently, Gernot Schaufler,[12] describe techniques that use depth information for better reconstruction.

In 1996, Jay Torborg and Jim Kajiya introduced Microsoft's Talisman architecture at Siggraph. (See http://research.microsoft.com/msrsiggraph for electronic versions of MSR Siggraph submissions.) The following year, John Snyder and I showed how to handle dynamic scenes and lighting factorization on Talisman, and this year, we describe how to sort a layered decomposition into sprites without splitting.[13]

Image-based modeling

Given an input set of images, what is the most efficient representation that will allow rendering from new points of view?

Paul Debevec, Camillo Taylor, and Jitendra Malik enhanced architectural modeling with simple modeling primitives aligned to images with vision techniques. They added realism and detail using recovered view-dependent texture maps and depth displacements, and they captured high-dynamic-range radiance maps from photographs. At Siggraph 98, Debevec adds synthetic objects to the photographed scenes,[14] and Yizhou Yu and Malik estimate reflectance properties and environmental radiance of architectural scenes.[15]

Another longstanding trend in computer graphics is to pursue *coherence* wherever it can be found.

COHERENCE AND LAYERS

Another longstanding trend in computer graphics is to pursue *coherence* wherever it can be found. Strategies for accelerating image rendering center on one of two types of coherence:

- temporal, from frame to frame, or
- spatial, within the same frame but from pixel to pixel.

Basically coherence implies the question "Why do the same work twice?" Researchers have typically pursued coherence in the spatial domain; a good example is in the hardware for rasterizing triangles. The operations change little from pixel to pixel, so the hardware stores partial results for each output row and column and uses these results to incrementally calculate the next output value. Researchers are beginning to explore temporal coherence. The section on sprites that follows addresses some of this work.

Coherence is everywhere in computer graphics; geometry and image sequences both contain coherently changing values. At Microsoft Research, we have been working on exploiting the coherence in real-time computer graphics. Hugues Hoppe's work, for instance, takes advantage of coherence in geometry by encoding shape as a sequence of coarse to fine representations. The granularity of the representation depends on the object's current distance from the viewer. Others are working on exploiting coherence in dynamic image streams, as I describe later.

Scenes naturally factor into layers, since empty space typically surrounds objects. Each layer—a bouncing ball as separate from its background, for example—has much more coherence than the combined scene. For vision, partitioning a scene into layers permits the independent analysis of each layer. Doing so avoids the difficulties in scene analysis that stem from overlapping layers and occlusion between layers. For graphics, partitioning a scene into layers allows rendering algorithms to take better advantage of spatial and temporal coherence.

3D rendering with sprites

Taking advantage of temporal coherence requires storing the partial results of a given frame for use in a later frame; in other words, trading off memory for computation. The partial results are image samples, which are used to interpolate (or extrapolate) from a given image to subsequent images.

The necessity of storing results has spawned research into efficient image caches. The ones considered to date include a set of z-buffered images,[10] textured 3D geometry (quadrilaterals or triangular meshes),[11] 2D sprites,[13] and sprites with z.[7,9,12] Researchers will present two new image-based representations at Siggraph 98: layered depth images[7] and multiple-center-of-projection images.[8]

A sprite is an image with the standard three components (red, green, and blue) for colors and an additional channel—the alpha channel, invented by Alvy Ray Smith and Ed Catmull in 1977—that encodes the image's shape. Thomas Porter and Tom Duff introduced the sprite-compositing algebra in 1984. Compositing permits individual renderings of each sprite, which are then combined to create a final image. When alpha is 1, the sprite defines the color. When alpha is 0, the sprite leaves the color value "undefined" or "clear" so that the background shows through. An alpha between 0 and 1 represents partial coverage of the pixel, which is useful for smooth antialiased edges.

Two-dimensional sprites are the basis of the Talisman architecture and related layer-based rendering techniques. Sprites are useful in their own right as 2D imaging primitives and will be put to good use in Chrome, a multimedia browser aimed at raising the quality of Web multimedia content. Microsoft's DirectX and Chrome both use image-based rendering primitives, and 2D imaging operations are considered to be as important as 3D rendering.

2D sprites. Talisman is a reference design for a low-cost hardware platform that incorporates compressed textures (to exploit spatial coherence) and the use of 2D image sprites for 3D rendering acceleration (to exploit temporal coherence).

In 1997, John Snyder and I discussed the algorithms needed to harness the Talisman architecture. To take advantage of coherence in animated-image streams, our method finds a best-match affine transform. Each lighting effect and appropriate geometric element goes

Figure 2. Barnyard scene with 119 sprite layers, such as the silo, barn, fences, fields, and so on. By factoring the scene into sprites, the relative motion between sprites can be handled by image warps instead of 3D rendering.

Figure 3. This Chicken Crossing *sequence uses 80 layers, some of which are shown separately (left and bottom) and displayed in the final frame with colored boundaries (middle). The sprite sizes reflect their actual rendered resolutions relative to the final frame. The rest of the sprites (not shown separately) were rendered at 40 to 50 percent of their display resolution. Since the chicken wing forms an occlusion cycle with the tailgate, the two were placed in a single sprite (bottom).*

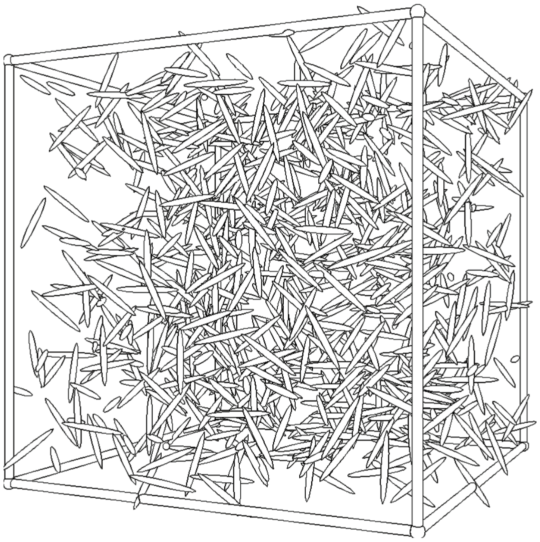


Figure 4. Simulation of "toothpicks" tumbling in a cubic volume shows the scene complexity that can be handled with software visibility sorting. Each of the 800 toothpicks contains 328 polygons and forms one part.

into a separate layer and can be updated with independent resolution in space and time.

Figure 2 depicts one frame of a barnyard scene through which a camera moves. The scene is factored into 119 sprite layers, including the contiguous landscape geometry. The shadows of the silo and barn are also separate layers.

Sprite independence allows rendering resources to be distributed precisely to those parts of the scene that need the most bandwidth and processing cycles. For example, Figure 3 shows a frame from *Chicken Crossing*, a short 3D animated film. Each part of the scene has a different spatial resolution. (For clarity, the figure shows just the sprites that make up the chicken.)

Each sprite also has an independent temporal resolution. In other words, each sprite is redrawn with an update rate independent of the other sprites. The sprites are then warped (based on the actual geometric motion) to interpolate the image through time. Sprite independence is the key benefit that permits graceful image degradation under varying processor and memory system loads.

Sprite independence is particularly useful in a parallel graphics architecture.[16] The only synchronization required is among sprites before the final composition. Each sprite may be rendered with a different 3D rendering engine: One sprite may be raytraced while others are rendered from a lumigraph or as pen-and-ink illustrations. All the sprites in a scene combine to produce the final result. We can also apply 2D imaging operations, such as focus blur, to each sprite. Separating objects into sprites also lets us apply the full precision of the z-buffer to only the objects in the sprite rather than the whole scene.

Sprite independence comes with a cost, however: Sprites must be sorted to account for the visibility between sprites. At Siggraph 98, John Snyder and I describe a new take on one of the earlier visibility-sorting approaches in computer graphics, that of Newell, Newell, and Sancha from 1972.

In the original algorithm, failing to find separating planes meant that the geometry had to be split—a costly operation for large and dynamic geometry. The new algorithm avoids splitting. Instead, an analysis of the visibility graph structure can yield a compositing expression useful in resolving visibility cycles. Alternatively, we can merge the geometry into a single layer for visibility resolution with the same z-buffered rendering used by a typical layer.

By reusing the sort from the previous frame and caching collision information, the enhanced algorithm exploits temporal coherence and can handle hundreds of objects, each of which can contain thousands of polygons. The "toothpicks" in Figure 4 are an example of the complex visibility ordering that this algo-

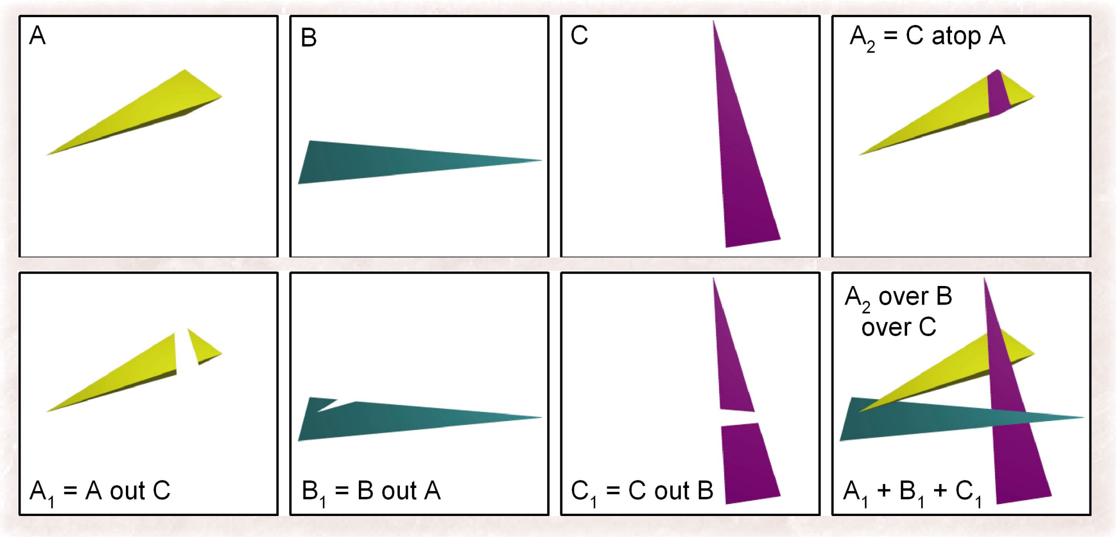


Figure 5. Cyclic occlusions (the bottom right corner shows the desired image) can be handled with either of two compositing expressions. The source sprite images are shown as A, B, C. The standard compositing expression, A over B over C, would give the incorrect result of A covering C. Instead, we first compute A combined with that part of C that covers A, and then composite. An alternate formulation is to first remove the occluding parts from each sprite and then add (bottom row).

rithm can calculate interactively. New compositing expressions also handle cyclic occlusions, such as the one shown in the bottom right corner of Figure 5. The occlusion is cyclic because object A occludes B, B occludes C, and C occludes A.

Sprites with depth. In 1996, Jonathan Shade and colleagues and, independently, Gernot Schaufler and Wolfgang Stürzlinger, presented an image cache technique based on 3D texture-mapped quadrilaterals. This technique calls for projecting the geometry inside a given bounding box to a plane in the center of the bounded area and then uses the resulting image to approximate later renderings. In 1997, Schaufler presented a technique called *nailboards* that augmented the planar 3D-image cache with depth values at each pixel. These values were encoded as a delta from the surface of the quadrilateral.[12] Having an associated depth value at each pixel improves image quality by making the per-pixel warp more accurate. The following describes two new representations for encoding depths per pixel in the image cache. Both of these techniques are also useful in capturing real-world objects.

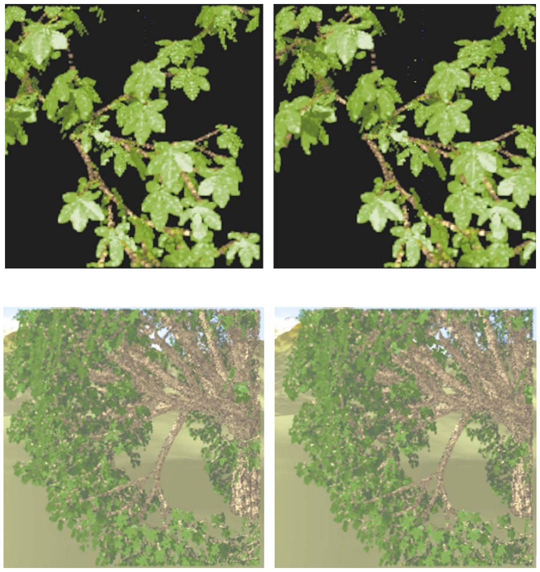


Figure 6. Two cross-eye stereo pairs of a chestnut tree, with (top) only the near segment displayed and (bottom) both segments in front of an environment map. An environment map uses a single image to capture the far-away environment behind foreground objects.

At Siggraph 98, Paul Rademacher and Gary Bishop describe *multiple-center-of-projection images*,[8] a representation combining a series of images from different camera views into a single image, which can then be used to reconstruct new projections. They focus on a strip camera, where each column of pixels in the image is from a different viewing position and orientation. This is an encoding of the 3D position of the image sample points. The key benefit of this representation is the economy of storage and the coherence and connectivity of the neighboring samples.

Jonathan Shade and his colleagues present techniques for quickly rendering sprites with depth and describe a new multivalued representation, called a *layered depth image* (LDI), in which each pixel can have multiple depths.[7] The key advantage of these representations is that they permit using the McMillan-Bishop warp-ordering technique[5] without a z-buffer. Eliminating the z-buffer permits the rendering of complex models at interactive frame rates on standard PCs.

The trees shown in Figure 6 as stereo pairs are represented as an LDI. An LDI has a depth value that allows each pixel to be drawn in the proper location (in space) to yield the correct parallax. A correct parallax enables more realistic 3D animations. The information stored in an LDI also permits more depth complexity in an animation—the movement of a leaf in the foreground can uncover other leaves in the background.

MAKING FACES

Humans are complicated radios: Transferring emotion and thought from one brain to another requires an elaborate modulation via face muscles, body muscles, and vocal cords. By nature, we are wonderfully adept at expressing and comprehending emotion, subjects we have studied our whole lives. We are thus extremely sensitive to any imperfection in the motion of a human figure. As Frédéric Pighin and his colleagues observe, no computer has yet passed the "facial-animation Turing test."[17]

The capture of face animation is an excellent example of the rich middle ground between graphics and vision. Computer vision provides an excellent input device, particularly for shapes and motions such as the complex changing shape of my face when I am expressing emotion symbolically. Matthew Turk and Kentaro Toyama of Microsoft Research have been studying how to efficiently analyze video sequences to capture gestures and emotion. The work mentioned here strives to capture the high-resolution motion and appearance of an individual face. The goal is to use this information to animate and render synthetic faces with high fidelity to the original.

Brian Guenter and his colleagues describe a system that uses multicolored dots placed on an actor's face for reference and a set of six calibrated video cameras to capture a high-fidelity texture map and mesh deformation.[18] The 3D positions recorded as the actor performs are mapped to a 3D face model and used to distort it in mimicry of human facial expression. After dot removal and texture-map cleanup, the model can be used to render a remarkably lifelike image from any angle, as shown in Figure 7.

Frédéric Pighin and his colleagues describe a system that uses uncalibrated cameras to record images of an

Image-based modeling

The Digital Michelangelo project (http://www-graphics.stanford.edu/projects/mich) seeks to accurately digitize the external shape and surface characteristics of Michelangelo's sculptures for a 3D computer archive. The project will employ improvements in *laser rangefinding*—the use of lasers to scan not only planar objects of uniform reflectance but also 3D and nonuniform surfaces —and Stanford-developed algorithms for combining multiple range and color images.

Digital Muybridge (http://http.cs.berkeley.edu/~bregler/bodies.html) is a project dedicated to Eadweard Muybridge, who captured the first photographic recordings of humans and animals in motion (*Animal Locomotion*, 1887). In contrast with earlier work on faces and lips, Digital Muybridge focuses on using video to capture full body motions: running, walking, and dancing.

Two projects address the visualization of buildings. The Architectural Modeling project (http://http.cs. berkeley.edu/~debevec/Research) is working on modeling and rendering architecture photorealistically from a small number of photographs. The Walkthrough project (http://www.cs.unc.edu/~walk) has the goal of developing a visualization system that will let users walk through and interact with models of buildings.

Figure 7. Lifelike 3D animation of human facial expressions captured using six calibrated video cameras and mapped to a 3D polygonal face model. The actor wears a set of multicolored fiducial dots (not shown) that give point correspondences in each of the cameras, so that the 3D position of each dot can be recovered for each frame. After digital removal of the dots, the texture map still retains the interesting wrinkles and creases that give life and character to human expressions.

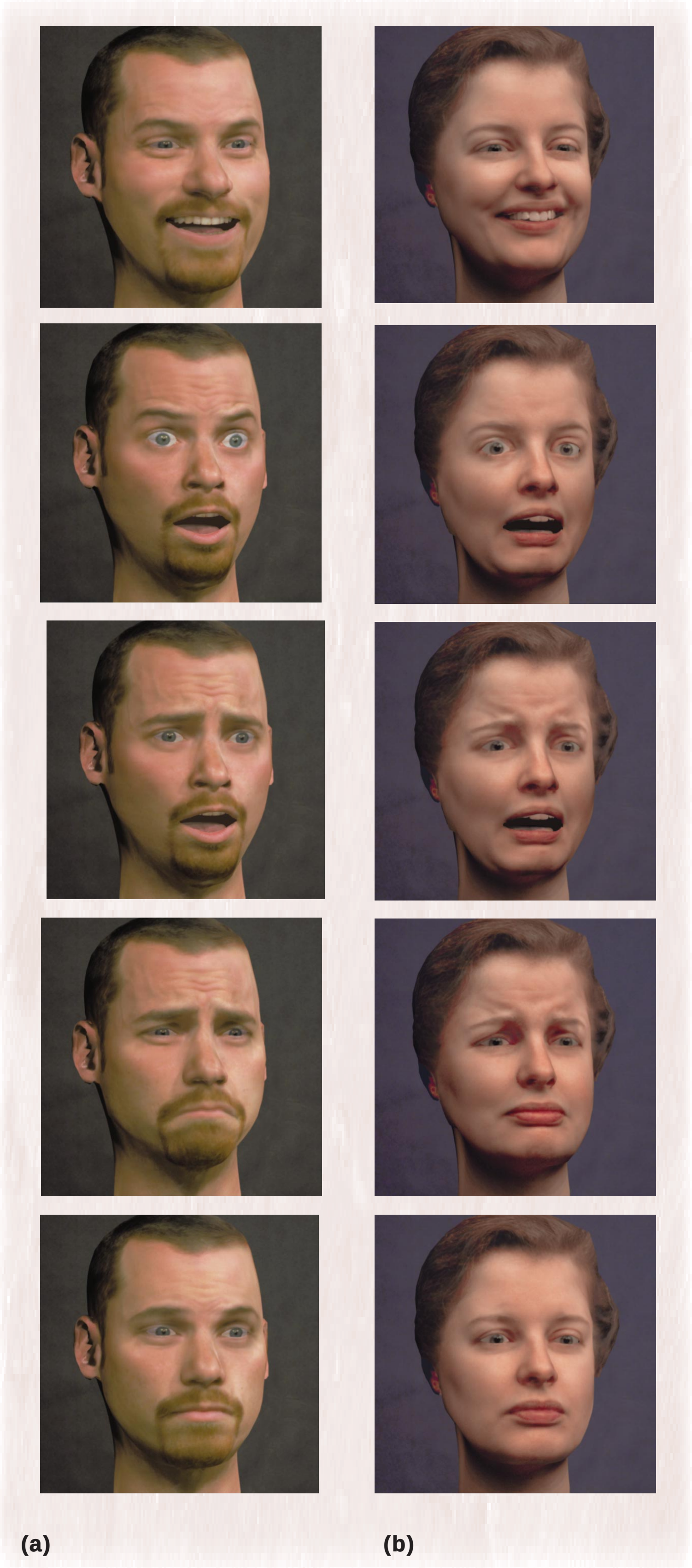


Figure 8. Sequence of expressions (a) from an interactive modeling system that blends between several expressions created by aligning a generic 3D mesh to a set of input photographs taken from several views. These views capture a view-dependent face texture. Applying the blends to a woman's face creates (b) a derivative animation.

actor's face. From the digitized images and a few facial reference points (like the tip of the nose), their method can recover 3D geometry coordinates for a given facial expression. They use this information to align a generic 3D mesh model to match the facial expression and then map view-dependent surface textures to the model.[17] An animation system also allows interpolation and extrapolation from the captured face animations; such a process generated the sequence of expressions shown in Figure 8a. The same facial expression information was applied to a model of a woman's face to produce the derivative animation sequence in Figure 8b.

Others are working to capture the nuance of human expression and thus provide improved modeling primitives for artists to use. Animating a human figure by hand is a challenging task, and great artistry is required to create compelling motion. Pixar's recent animated short, *Geri's Game*, is a lovely example. Tony DeRose, Michael Kass, and Tien Truong describe how Pixar's shape-modeling system, which uses *subdivision surfaces*, lets animators add lively wrinkles and creases to faces and fabrics.[19] Subdivision surfaces is a technique that subdivides geometric elements until the process yields a smooth surface, as opposed to a patchwork of interconnected surfaces. Another approach is to give the modeler a parameterized set of faces to work with, as described by Douglas DeCarlo, Dimitris Metaxas, and Matthew Stone.[20] One obvious application for these methods is the creation of lifelike virtual characters for film and television.

WHERE ARE WE HEADING?

A recent workshop at Stanford (Image-Based Modeling and Rendering; http://www-graphics.stanford.edu/workshops/ibr98) brought together many researchers in the overlapping field. At the workshop, Stanford's Marc Levoy observed that "the key innovation in television was the camera." He plans to use image-based techniques to capture and display digital versions of Michelangelo's sculptures.

Richard Szeliski argued that there is no "grand unified theory of image-based modeling and image-based rendering." Instead, there is a collection of image-based techniques to draw from in solving a particular engineering problem.

UC Berkeley's Paul Debevec noted that computer graphics is in the same state as audio synthesis was not long ago. The original sound synthesizers developed sounds from scratch, using FM synthesis. Now sampled waveforms are the norm. Sampled waveforms are extensively modified by the current generation of audio synthesizers but still yield a richness and timbre difficult to achieve with "pure" synthesis. In a similar vein, computer graphics increasingly relies on sampled geometry and textures. Taken from the natural and manmade landscape around us, these samples provide richness to visual images.

THE PHILOSOPHER IN THE CAVE

It's hard to project backwards from the flickering shadows on the wall of the cave to see Plato's ideal shapes. From the vision perspective, graphics has it easy; we know the shapes already. But there's a great distance from a computer representation of a complex scene to real-time display and interaction with that scene. The job of the graphics field is to find ways to render not just everything we've ever seen but everything we could ever imagine.

The most compelling shapes are those near to our hearts: people's faces, a gracefully moving body, a natural scene with rustling leaves and flowing water. Evolution has tuned us to these sights. By combining vision and graphics, capturing and creating images of these scenes may soon be within reach. And once we have these powerful tools for creation and manipulation in hand, perhaps we will be one step closer to the best possible tool for the imagination. ❖

References

1. S. Kang, *A Survey of Image-Based Rendering Techniques*, Tech. Report CRL 97/4, Digital Equipment Corp., Cambridge Research Lab, Cambridge, Mass., 1997.
2. Z. Zhang, "Image-Based Geometrically Correct Photorealistic Scene/Object Modeling: A Review," *Proc. 3rd Asian Conf. Computer Vision*, 1998, pp. 340-349.
3. S.E. Chen and L. Williams, "View Interpolation for Image Synthesis," *Proc. Siggraph 93*, ACM Press, New York, 1993, pp. 279-288.
4. R. Szeliski, "Video Mosaics for Virtual Environments," *IEEE Computer Graphics and Applications*, Mar. 1996, pp. 22-30.
5. L. McMillan and G. Bishop, "Plenoptic Modeling: An Image-Based Rendering System," *Proc. Siggraph 95*, ACM Press, New York, 1995, pp. 39-46.
6. S.M. Seitz and C.M. Dyer, "View Morphing," *Proc. Siggraph 96*, ACM Press, New York, 1996, pp. 21-30.
7. J.W. Shade et al., "Layered Depth Images," *Proc. Siggraph 98*, ACM Press, New York, 1998.
8. P. Rademacher and G. Bishop, "Multiple-Center-of-Projection Images," to be published in *Proc. Siggraph 98*, ACM Press, New York, 1998.
9. S. Molnar, J. Eyles, and J. Poulton, "Pixelflow: High-Speed Rendering Using Image Composition," *Proc. Siggraph 92*, ACM Press, New York, 1992, pp. 231-240.
10. M. Regan and R. Pose, "Priority Rendering with a Virtual Reality Address Recalculation Pipeline," *Proc. Siggraph 94*, ACM Press, New York, 1994, pp. 155-162.
11. P. Maciel and P. Shirley, "Visual Navigation of Large Environments Using Textured Clusters," *Proc. Symp. Interactive 3D Graphics*, 1995, pp. 95-102.
12. G. Schaufler, "Nailboards: A Rendering Primitive for Image Caching In Dynamic Scenes," *Proc. 8th Eurographics Workshop Rendering*, 1997, pp. 151-162.
13. J. Snyder and J. Lengyel, "Visibility Sorting and Compositing Without Splitting for Image Layer Decompositions," to be published in *Proc. Siggraph 98*, ACM Press, New York, 1998.
14. P.E. Debevec, "Rendering Synthetic Objects Into Real Scenes: Bridging Traditional and Image-Based Graphics with Global Illumination and High Dynamic Range Photography," to be published in *Proc. Siggraph 98*, ACM Press, New York, 1998.
15. Y. Yu and J. Malik, "Recovering Photometric Properties of Architectural Scenes from Photographs," to be published in *Proc. Siggraph 98*, ACM Press, New York, 1998.
16. H. Igehy, G. Stoll, and P. Hanrahan, "The Design of a Parallel Graphics Interface," to be published in *Proc. Siggraph 98*, ACM Press, New York, 1998.
17. F. Pighin et al., "Modeling Realistic Facial Expressions from Photographs," to be published in *Proc. Siggraph 98*, ACM Press, New York, 1998.
18. B. Guenter et al., "Making Faces," to be published in *Proc. Siggraph 98*, ACM Press, New York, 1998.
19. T. DeRose, M. Kass, and T. Truong, "Subdivision Surfaces for Character Animation," to be published in *Proc. Siggraph 98*, ACM Press, New York, 1998.
20. D. DeCarlo, D. Metaxas, and M. Stone, "An Anthropometric Face Model Using Variational Techniques," to be published in *Proc. Siggraph 98*, ACM Press, New York, 1998.

Jed Lengyel works in the graphics group at Microsoft Research. His research interests focus on motion, especially creating and rendering moving images for PCs. Lengyel was educated at Caltech and at Cornell University.

Contact the author at Microsoft Research, One Microsoft Way, Redmond, WA 98052-6399; jedl@microsoft.com.

Related conferences

Siggraph 98, 25th International Conference on Computer Graphics and Interactive Techniques, Orlando, Florida, July 19-24, 1998 (See http://www.siggraph.org/s98/s98main.html).

VIS 98, IEEE Visualization Conference, October 18-23, Research Triangle Park, North Carolina (See http://www.erc.msstate.edu/vis98).

VR 99, IEEE Computer Society Virtual Reality, (formerly VRAIS) March 13-17, Houston, Texas.