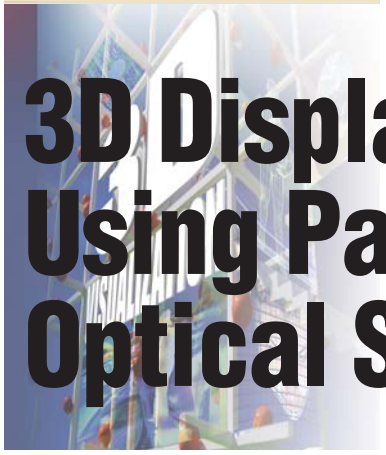


3D Display Using Passive Optical Scatterers

Shree K. Nayar and Vijay N. Anand
Columbia University

A new class of 3D displays uses a digital projector and a cloud of passive optical scatterers etched in a glass cube with laser-induced damage. These inexpensive devices can be used to display simple 3D objects, extend videogames to three dimensions, and create 3D avatars.

Systems for displaying images and videos have become part of our everyday lives. However, most systems in use today can only display 2D images. Since we live in a 3D physical world, a system that can display static and dynamic 3D images would provide viewers with a more immersive experience.

Over the past century, researchers have worked vigorously to create a device that can display realistic high-resolution 3D content.[1,2] As an offshoot of that, we are developing an inexpensive class of volumetric displays that can present certain types of 3D content, including simple 3D objects, extruded objects, and 3D surfaces that appear dynamic when projected with time-varying images, at relatively low resolution. The "Previous Approaches to Volumetric Display" sidebar describes related research efforts.

Our displays use a simple *light engine* and a cloud of *passive optical scatterers*. The basic idea is to trade off the light engine's 2D spatial resolution to gain resolution in the third dimension. One way to achieve such a tradeoff is to use a stack of planar grids of scatterers where no two stacks overlap each other with respect to the light engine's projection rays.

Such a semiregular 3D grid suffers from poor visibility. As the viewer moves around the point cloud, the fraction of visible points varies dramatically and is very small for some viewing directions. However, randomizing the point cloud in a specific manner consistent with the light engine's projection geometry produces a remarkably stable visibility function.

We used a technology called *laser induced damage* (LID)[3,4] that can efficiently, precisely, and at a very low cost embed a desired point cloud in a solid block of glass or plastic. Each scatterer is a physical crack in the block that is created by focusing a laser beam at a point. When such a crack is lit by ambient light it is barely visible, but when it is lit by a focused source it glows brightly.

To illuminate the scatterers, we developed an *orthographic light engine* that uses an off-the-shelf digital projector and inexpensive optics to create parallel rays with a large footprint. While orthographic projection isn't required, it allows us to use point clouds without resolution biases and with relatively straightforward calibration of the display.

We developed several versions of our volumetric display, each designed to meet the needs of a specific class of objects or a specific application. We implemented point clouds with

- 10,000 points to display true 3D objects,
- 190,500 points to display extruded objects with arbitrarily textured top surfaces,
- 180,500 points to display purely extruded objects,
- 83,866 points to extend the *Pac-Man* game to 3D, and
- 127,223 points to create a 3D avatar.

Since our displays render content in a physical volume, the content's three-dimensionality is naturally perceived with all its cues—*binocular parallax*, *motion*

Published by the IEEE Computer Society
0018-9162/07/$25.00 © 2007 IEEE

Previous Approaches to Volumetric Display

B.G. Blundell and A.J. Schwarz presented a comprehensive survey of volumetric displays in their 2000 book.[1] In a swept-volume display, a physical surface's mechanical motion creates the display volume. The speed of surface motion is kept high enough so that the observer doesn't perceive it. Early versions of this approach used spinning[2] and varifocal[3] mirrors to reflect images displayed on cathode-ray tubes[2] and a rotating phosphor-coated screen.[4] Perspecta, a recent commercial product,[5] uses a diffuse spinning screen and a high-speed digital projector. Swept-volume displays produce impressive results, but their use of large moving parts, such as a screen or source, is a drawback.

A static volumetric display can create a display volume without using mechanical motion. Previous implementations have used fluorescence excitation of gases[6] and infrared laser excitation of fluorescent metallic particles.[7]

The closest work to ours is Duncan MacFarlane's static volumetric display,[8] which used a controllable light source and a 3D array of voxels doped with a fluorescent dye. Optical fibers guide light from the source elements to the voxels. The voxels and fibers are immersed in a refractive index-matching liquid to avoid refraction artifacts. This system is an impressive engineering feat, but implementing it is complex and expensive compared to our approach using passive light scatterers.

The DepthCube,[9] a recent commercial product, is a static volumetric display that uses a stack of 20 scattering LCD sheets that a high-speed digital projector illuminates in sequence. Although this display produces compelling 3D content, it is expensive compared to our approach, since it uses a high-speed light engine and electrically controlled LCD scatterers.

Finally, New York University's Ken Perlin and Jeff Han proposed using dust particles suspended in air to project 3D images.[10] The idea is to scan the dust using an infrared time-of-flight detector to find the locations of the particles and then use a visible light-scanning beam to light up the appropriate particles.

References

1. B.G. Blundell and A.J. Schwarz, *Volumetric Three-Dimensional Display Systems*, John Wiley & Sons, 2000.
2. E.J. Parker and P.A. Wallis, "Three-Dimensional Cathode-Ray Tube Displays," *J. IEE*, vol. 95, 1948, pp. 371-390.
3. A.C. Traub, "Stereoscopic Display Using Varifocal Mirror Oscillations," *Applied Optics*, vol. 6, no. 6, 1967, pp. 1085-1087.
4. B.G. Blundell, A.J. Schwarz, and D.K. Horrell, "Cathode Ray Sphere: A Prototype System to Display Volumetric Three-Dimensional Images," *Optical Eng.*, vol. 33, 1994, pp. 180-186.
5. G.E. Favalora et al., "100 Million-Voxel Volumetric Display," *Proc. SPIE*, vol. 4712, SPIE, 2002, pp. 300-312.
6. A.J. Schwarz and B.G. Blundell, "Considerations Regarding Voxel Brightness in Volumetric Displays Utilizing Two-Step Excitation Processes," *Optical Eng.*, vol. 32, no. 11, 1993, pp. 2818-2823.
7. E.A. Downing et al., "A Three-Color, Solid-State Three-Dimensional Display," *Science*, vol. 273, 1991, pp. 1185-1189.
8. D.L. MacFarlane, "A Volumetric Three-Dimensional Display," *Applied Optics*, vol. 33, no. 31, 1994, pp. 7453-7457.
9. A. Sullivan, "A Solid-State Multiplanar Volumetric Display," *SID Digest*, vol. 58, no.3, 2003, pp. 354-356.
10. K. Perlin and J. Han, *Volumetric Display with Dust as the Participating Medium*, US patent application 20040218148 to New York Univ., Patent and Trademark Office, 2004.

parallax, *accommodation*, and so on.[5] Using a physical volume also allows multiple observers to view the content from a wide range of directions simultaneously. On the other hand, our displays share a limitation inherent to most volumetric displays: They cannot render view-dependent effects such as *specularities* and *occlusions*.

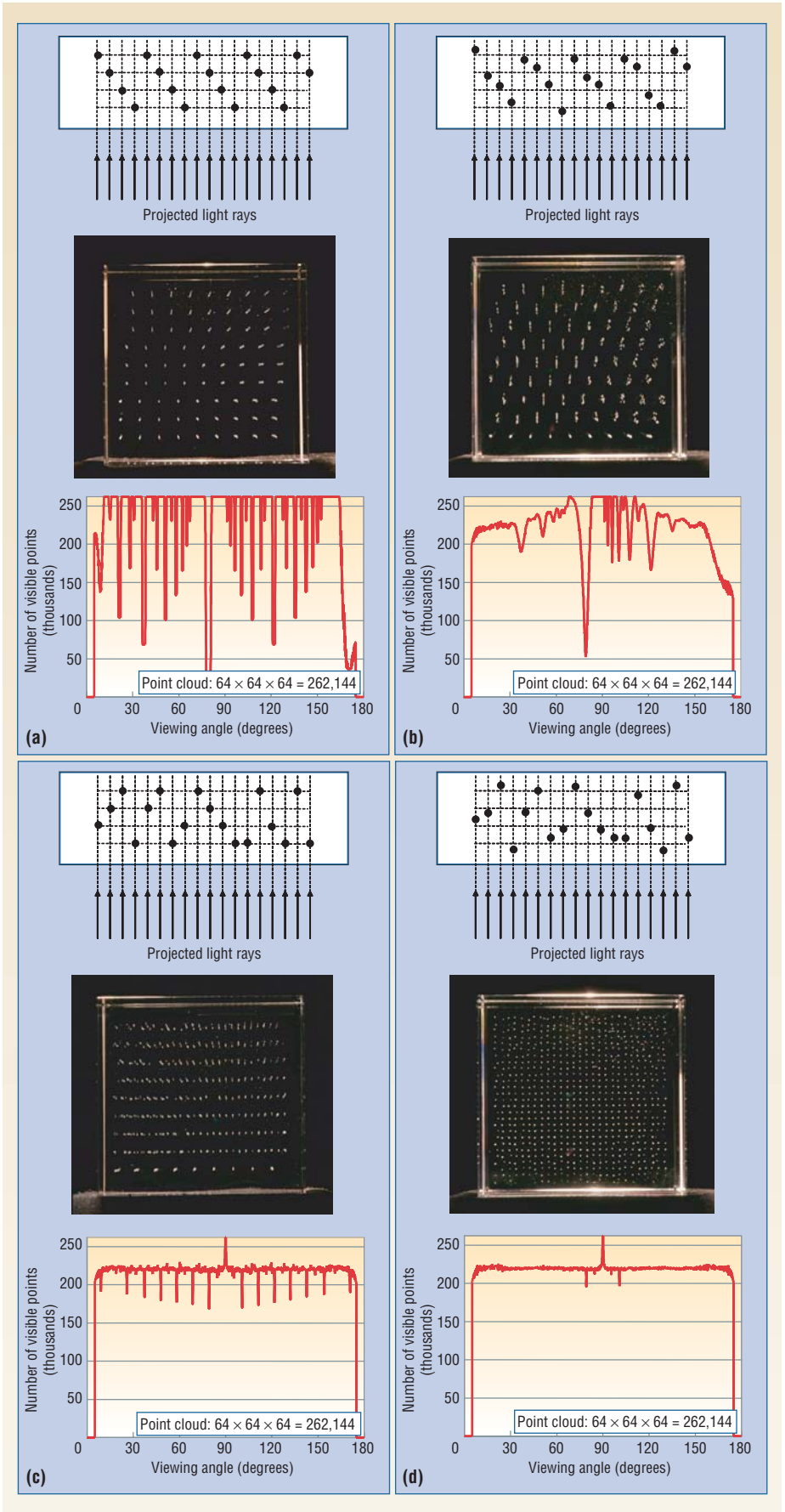
INDEXABLE POINT CLOUDS AND VISIBILITY

In an indexable point cloud, a light engine can illuminate each of the points (passive scatterers) without lighting up other points. Such a configuration is essential for displaying arbitrary 3D objects.

There are many ways to create an indexable point cloud. The notion of visibility is critical to choosing the point cloud. While moving around the display, the percentage of points that the viewer can see, at least in part, shouldn't change dramatically with the viewing direction. Ideally, the indexable point cloud should have high and constant visibility with respect to the viewing direction.

To compare the visibilities of different point-cloud configurations, we conducted OpenGL simulations using n^3 points where $n = 64$. We assumed each point cloud to be within a glass cube (refractive index of 1.5), and modeled each point as a sphere with a 0.5 mm diameter.

We assumed that an orthographic projector that produces parallel light rays illuminates the point cloud. We denote the planes parallel to the projector's image plane as XY planes, and the third dimension, which is the direction of light projection, as Z. When we project all the spheres in a cloud orthographically onto a single XY plane, the distance between adjacent spheres in the X and Y directions is 0.6 mm.



Therefore, the average distance p between adjacent spheres on any given XY plane is $0.6\sqrt{n} = 4.8$ mm. We also used this as the distance q (along Z) between adjacent XY planes. We assumed the viewer would be far from the display and move along a single plane perpendicular to the XY planes. We varied the viewing angle from 5 to 175 degrees in steps of 0.2 degrees, where at 90 degrees the viewing direction is exactly opposite the direction of light projection.

For each viewing angle, we rendered an image of the point cloud with each sphere assigned a unique color; a sphere was deemed visible if a viewer could see more than f percent of its entire unobstructed area.

Semiregular indexable cloud

Figure 1a shows a 2D version of the simplest indexable cloud. The orthographic projector illuminates the cloud from below, and it is viewed from the upper hemisphere. In this cloud, the XY planes have regular grids of points, and the grid for one plane is simply offset in X and Y by a fixed distance with respect to the next plane. The offsets ensure that the projector can uniquely index each point in the cloud.

The middle of Figure 1a shows a physical glass cube with a semiregular indexable cloud consisting of $9 \times 9 \times 9$

Figure 1. Visibility functions (bottom row) for point clouds (top row). (a) Semiregular, (b) Z-dithered, (c) XY-dithered, and (d) XYZ-dithered grids. The middle row in each section shows real 9×9×9 point clouds etched in glass cubes using laser-induced damage.

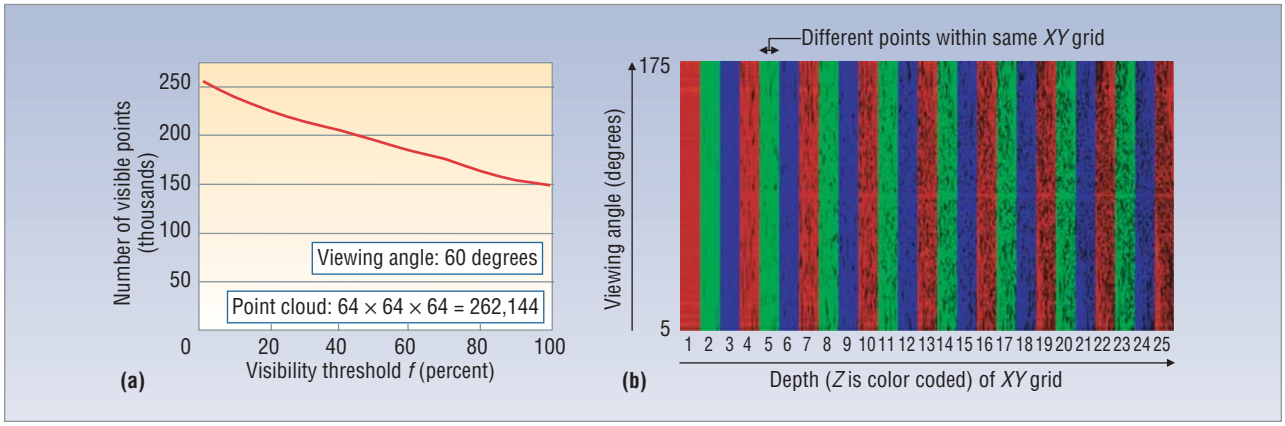


Figure 2. More simulations related to visibility. (a) Visibility plotted as a function of the threshold f used for determining point visibility. (b) A color-coded image that illustrates how visibility varies with the depth Z of the XY grid.

points (scatterers). The visibility plot (with $f = 25$ percent) for the 64^3 cloud used in our simulation, shown at the bottom in Figure 1a, clearly illustrates the simulation problem with using such a semiregular cloud. As the viewing angle changes, a single XY grid can occlude one or several other XY grids beneath it. Therefore, the visibility fluctuates dramatically with viewing direction, indicating poor display performance.

Z-dithered indexable cloud

This point cloud, illustrated in Figure 1b, is identical to the semiregular cloud, except that a random distribution dithers each point's Z coordinate. In our simulation, we used a uniform distribution over ±1 mm. This simple modification to the regular grid greatly improves the visibility function. For instance, it is possible to see many more points in the glass cube's image with $9 \times 9 \times 9$ points. Overall, the visibility function for the 64^3 cloud is much smoother, but the visibility is very low for some viewing angles, for example at 80 degrees.

XY-dithered indexable cloud

Consider the first four (from left) points in the semiregular cloud in Figure 1a. These points lie within a vertical tube through the point cloud. In the case of an n^3 cloud, the tube would have an XY cross section of $p \times p$ and a length along Z of $(n-1)q$. Within this tube, there is no reason to offset the points in XY in a regular fashion, as in Figure 1a. In fact, such an offset gives the point cloud a strong structure that can distract the viewer. We only need to ensure that we include the full range of Z values within the tube.

Therefore, within each tube, for each Z value we can randomly assign (without repetition) the XY coordinate from the ones in the semiregular grid. Since there are n points in the tube, there are $n!$ possible XY assignments (permutations). For any reasonable value of n, it is unlikely that any two tubes in the cloud would have the same assignment.

Note that such a random assignment of XY coordinates is not the same as the pure randomization of the Z coordinate in the Z-dithered case. A pure randomization of XY coordinates would place points outside the regular structure of the light rays the light engine produces, while a random assignment does not.

Figure 1c shows simulation results for an XY-dithered cloud with 64^3 points. There is a significant improvement in the visibility function, which is almost constant over the entire range of viewing angles except for small, sharp dips that are more or less at regular intervals.

XYZ-dithered indexable cloud

As Figure 1d shows, an XYZ-dithered indexable cloud simply combines the ideas behind the previous two clouds. It uses randomization of the Z coordinate and random assignment for the XY coordinates of each point in the cloud. When this is done, the small dips in the XY-dithered cloud's visibility function disappear, providing a remarkably stable visibility function, where viewers always see roughly 85 percent of the points.

OTHER SIMULATIONS

We conducted several other simulations as part of our visibility analysis. All the visibility functions were computed for viewing angles in a single plane that is parallel to the projection direction. We also computed visibility functions for several different rotations of this plane about the projection direction and verified that the plots remain essentially the same irrespective of the chosen plane orientation. In addition, we repeated the simulations for point clouds with $n = 25$ and $n = 100$ and found that the plots are similar to those for $n = 64$ except that the high-frequency jitter (noise) seen in all the visibility plots was greater for $n = 25$ and lesser for $n = 100$, an expected result.

We also studied visibility as a function of the percent f of the projected area of a sphere that is used as the threshold for visibility. In Figure 2a, the visibility falls almost linearly as f increases. Here, $n = 64$, and the viewing angle equals 60 degrees.

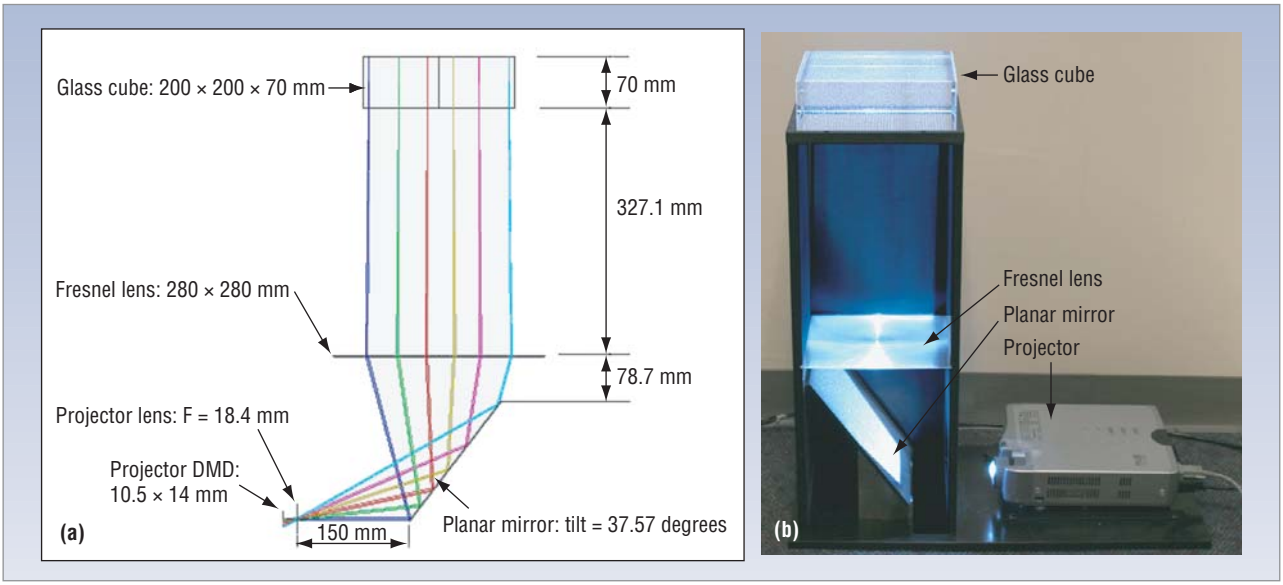


Figure 3. Orthographic light engine. (a) Optical layout. (b) The complete display system includes the light engine and an LID point cloud.

Thus far, we have defined visibility as the fraction of visible points in a cloud. Clearly, there is a greater likelihood of occlusion of the points in the *XY* layers that are deeper in the cloud. Figure 2b shows the visibility of each point in a cube with $n = 25$. The colored stripes correspond to individual *XY* grids at different depths *Z*, where $Z = 1$ is the top grid that is closest to the viewer.

The columns within each colored stripe correspond to different points in a thin slab of width *p* that is aligned with the plane of viewer motion. All such slabs will have the same visibility properties. The vertical axis represents the viewing angle. A dark point in the image represents a cloud point that is not visible.

As expected, all points in the top *XY* grid (red stripe corresponding to $Z = 1$) are visible over all viewing angles, while the deeper *XY* grids are more obstructed (more dark points). Again, at 90 degrees, all points in the cloud are visible, as the bright thin horizontal line that runs through the middle of the image illustrates.

POINT CLOUDS WITH LASER-INDUCED DAMAGE

There are many ways to fabricate a point cloud with passive scatterers. One approach is to embed or etch the scatterers on glass or plastic sheets and then stack the sheets to create a volume. Each sheet serves as a single *XY* grid, and the number of sheets determines the resolution in depth *Z*. Specular balls, diffuse painted patches, or etched patches can serve as the scatterers.

With specular balls and painted patches, the system must project the light rays from the top (opposite to that in Figure 1) since the scatterers are opaque. On the other hand, because etched patches act like diffusers, the system can project from the bottom. We initially intended to use a stack of glass sheets with grids etched on them, but found aligning the sheets cumbersome. In addition, the sheets require strong antireflection coatings to avoid glow due to interreflections between them.

The LID technique uses a focused and pulsed laser beam to produce small cracks in materials such as Plexiglas or crystal glass. LID offers a precise, efficient, and inexpensive way to create a complex point cloud.[3] Each crack then serves as a scatterer that glows when it is lit.

LID has become popular in recent years as a means of creating 2D images and 3D models in ornaments and souvenirs. In our context, LID is attractive because it doesn't impose constraints on the point cloud's nature (regular or random). It can cheaply create a cloud with hundreds of thousands of points in a matter of minutes. Our studies of the radiometric and spectral characteristics of LID scatterers reveal that they have the properties needed to display objects in color and with sufficient brightness to allow viewing within a 120-degree cone that is aligned with the projection direction.[6]

Orthographic projection system

In theory, any type of projection system that generates a 2D set of light rays can illuminate our point clouds. In the case of an off-the-shelf projector, the light rays diverge from a single point. This requires *XY* grids farther from the projector to have lower resolution, which is undesirable from the viewer's perspective. The nonuniform resolution also results in cumbersome alignment and calibration of the point cloud with respect to the projector. For these reasons, we developed an orthographic light engine that produces a parallel set of light rays, as Figure 1 illustrates.

Figure 3a shows the light engine's optical layout. The light source is a Plus Vision U232 DLP projector with a

1,024 × 768 pixel resolution. To reduce the system's size, we used a planar mirror to fold the projection optics.

We used a Fresnel lens (No. 37 from Fresnel Technologies) to convert the DLP projector into an orthographic projector. The Fresnel lens converts the diverging rays from the projector into parallel rays that are focused in the middle of the glass crystal that has the LID point cloud. We used a Fresnel lens instead of a conventional lens because we needed a large projection footprint to illuminate the 200 × 200 × 70 mm glass crystals we use for many of our point clouds.

To choose the projector's zoom setting, the planar mirror's position and tilt, and the Fresnel lens's focal length, we used the commercially available Zemax package to perform optical simulations. We determined that this system can create a fully indexable point cloud with 17,777 points in a 200 × 200 × 70 mm glass cube.[6] Figure 3b shows the complete optical system, including an LID point cloud.

If the light engine is perfectly orthographic, calibrating the display is straightforward.

Since some of our applications only require partially indexable point clouds (as in the case of extruded objects), in such cases, denser point clouds can be used with the current system. Furthermore, the resolution of the current system can be greatly improved by custom designing a Fresnel lens of higher optical quality.

Calibration and rendering

Each LID crack is about 0.18 mm wide and about 0.21 mm long. We measure the length along the direction of the laser beam used to create the crack. The cracks' positions are accurate within 0.05 mm. In most of our point clouds, we used clusters of cracks for each point to increase the brightness of the points.[6] When we use a six-crack cluster, each point has a width of about 0.5 mm and a length of about 0.6 mm.

Calibration method. If the light engine is perfectly orthographic, calibrating the display is straightforward. You can simply use the (X, Y, Z) coordinates of the corner points of the cloud in a coordinate frame attached to the glass crystal and the (i, j) coordinates of pixels in the projector that light up the chosen corner points to determine the projector pixels that light up each and every point in the point cloud. However, due to lens distortions and misalignments in the optical system, such a simple approach is not precise enough.

In our current implementation, we took a semimanual approach to the calibration problem. We separately calibrate each XY grid (layer) of the cloud. The mapping between projector pixels (i, j) and points (X, Y, Z) in a single grid is modeled as $i = f(X, Y, Z)$ and $j = g(X, Y, Z)$, where f and g are low-order (three in our case) polynomials. We developed an interface that permits the user to manually find the projector pixels that light up a small number of points on the given XY grid. These correspondences are used to find the polynomials f and g. Then, we illuminate all the points on the XY grid using the computed mapping. We found that only a few points are not adequately lit by the computed mapping, and the projector pixels corresponding to these points are manually refined using our interface.

We repeat this process for all the XY grids in the cloud. Since our system has chromatic shifts, we must perform this calibration separately for each of the three color channels. For the most complex clouds we used, this process takes about two hours per color channel, or six hours for complete calibration.

Rendering algorithm. We use a simple and efficient algorithm for rendering objects. Given a 3D scene, our goal is to determine the brightness (for each color channel) needed to illuminate each point in the cloud. For this, we use an intermediate representation that is a regular 3D grid aligned with and occupying the same volume as the point cloud, but with higher resolution. That is, if a point cloud P has n^3 points, we choose a regular grid Q that has m^3 points, where $m > n$.

For each point u in the cloud P, we find all its neighboring points $\{v_w | w = 1, 2 \ldots W\}$ in Q that are less than a small distance d from u. We also find the distances of these points from u, $\{r_w | w = 1, 2 \ldots W\}$. For each point in the cloud, we compute and store the lists v_w and r_w *prior to rendering*.

During rendering, we resample each color channel of the input 3D scene to the regular 3D grid Q to obtain the brightness value I_v. Then, for each color channel, we determine the brightness I_u of each cloud point u as a weighted sum of the brightnesses of its precomputed neighboring points in Q:

$$I_u = \frac{\sum_{w=1}^{W} r_w I_{v_w}}{\sum_{w=1}^{W} r_w}$$

As an example, for $n = 64$, we can use $m = 255$ and $d = 10$, all of which are defined in the units of the grid Q. Since we precompute the lists v_w and r_w, the rendering computations are linear in n^3 (the number of cloud points). In our applications, we used clouds with $n < 64$, hence rendering is easily done at frame rate (30 Hz) for a dynamic 3D scene. We implemented the calibration and rendering algorithms on a 2.66 GHz Dell workstation.

APPLICATIONS

Each of the volumetric displays we developed is geared toward a specific class of objects.[6]

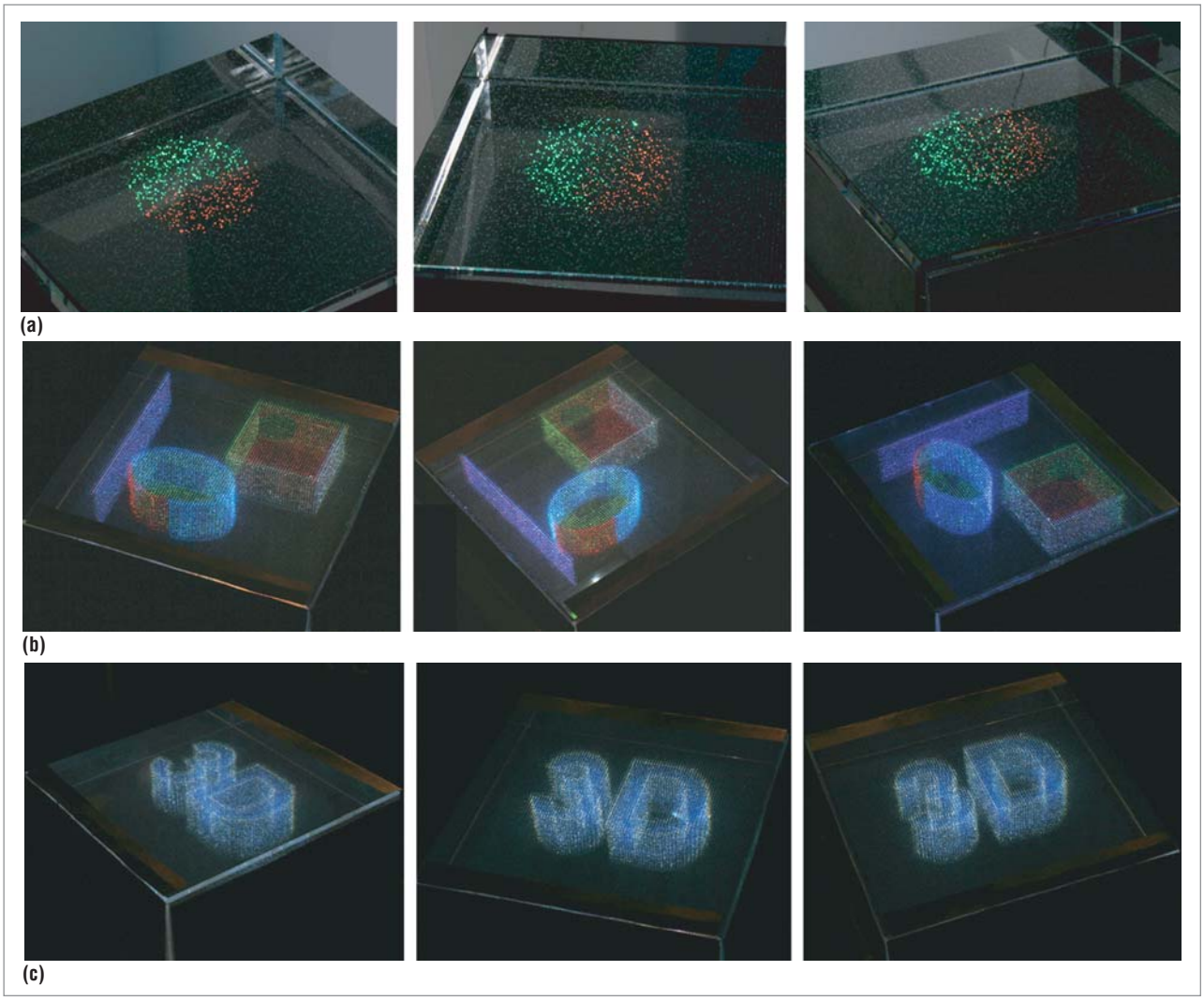


Figure 4. Multiple display views. (a) A fully indexable display with 10,000 XYZ-dithered points developed for rendering simple 3D objects. (b) A display with 190,500 points for rendering extruded objects with arbitrary top-layer textures. (c) A display with 180,500 Z-dithered points for rendering purely extruded objects.

True 3D objects

For the display of simple 3D objects, we implemented a fully indexable *XYZ*-dithered cloud with $25 \times 25 \times 16 = 10{,}000$ points in a $200 \times 200 \times 70$ mm glass cube. When we project all the points onto the same *XY* plane, the distance between adjacent points is 1.9 mm in each of the two dimensions. The distance between adjacent *XY* planes is 4 mm, and the *Z*-dithering is in the range ± 1.5 mm. Figure 4a shows a ball that is half red and half green. We show three different views of the display to convey the 3D nature of the content.

We implemented this display only to demonstrate the feasibility of a fully indexable *XYZ*-dithered cloud. We kept its resolution low to make the calibration easy and because of the light engine's limited optical abilities. With a custom-designed orthographic projector, we believe that the system's optical resolution can be significantly enhanced, enabling the use of denser point clouds.

Extruded objects with top surfaces

We can easily achieve high-resolution renderings for the class of extruded objects with arbitrary top surfaces. These are objects that have translational symmetry in their geometry and texture along the *Z* direction. In addition, the objects can have "lids" with arbitrary textures. For this, we implemented a point cloud with a fully indexable top layer (the *XY* grid that is farthest from the projector) with $100 \times 100 = 10{,}000$ points and a dense extrusion volume with 180,500 points.

These extrusion points are *Z*-dithered and lie along projection rays that are different from the rays corresponding to the top layer points. Figure 4b shows several extruded objects displayed using this point cloud. The objects with oval and square cross sections have green and red top lids with holes.

Purely extruded objects

In the case of dynamic text (such as that used in advertisements), displaying the characters as purely extruded objects can give them a 3D appearance. For this we used a dense extrusion volume with 180,500 Z-dithered points in a 200 × 200 × 70 mm glass cube. Figure 4c shows different views of the characters "3D" displayed. The characters are blue in color but have white extruded outlines.

3D *Pac-Man*

Traditionally, users play videogames with 2D displays. An inexpensive approach to volumetric display such as ours opens the possibility of designing 3D games. This not only makes the gaming experience more compelling from a visual standpoint but also challenges the player to reason in 3D. To demonstrate this, we implemented a 3D extension of the popular *Pac-Man* game.

As Figure 5b shows, we etched the point cloud used to implement the game in an 80 × 80 × 80 mm glass cube. Figure 5a shows the paths and targets used in the design; the path segments' four different colors correspond to different depths in the cube. The game is really a maze of path segments that are either horizontal or vertical, and the targets are small patches at the ends of some of the horizontal path segments.

We can view this point cloud as 2.5D, since any given projector pixel can only illuminate a dot on a single horizontal path segment or a full vertical path segment. That is, the paths do not overlap each other in the *Z* dimension.

Figure 5c shows a person playing the game. The system displays the player's location as a green blinking dot, and the player uses a joystick to control this location. The system displays all the path segments in blue and the targets in bright pink. The player can travel in either direction along a horizontal path segment but must initiate a jump when arriving at a vertical path segment.

When a jump takes place, the player's dot automatically sweeps past a vertical path segment, and this motion gives the impression of sliding or climbing. When getting close to a target, the player must initiate an explosion of the target, which is displayed as the fizzling out of the bright target color. Audio clips convey the player's motion along horizontal path segments, the sliding up and down vertical segments, and the explosions of targets.

3D avatar

It is well known that in the case of certain objects, a 3D visual effect can be created by simply projecting a video of the object on a static 3D surface. This is especially true with human faces—video of a speaking person can be projected onto a video-projection surface with the shape of the person's face to create a 3D avatar. Misalignment of facial features in the video and projection surface can occur, but an observer does not easily notice them.

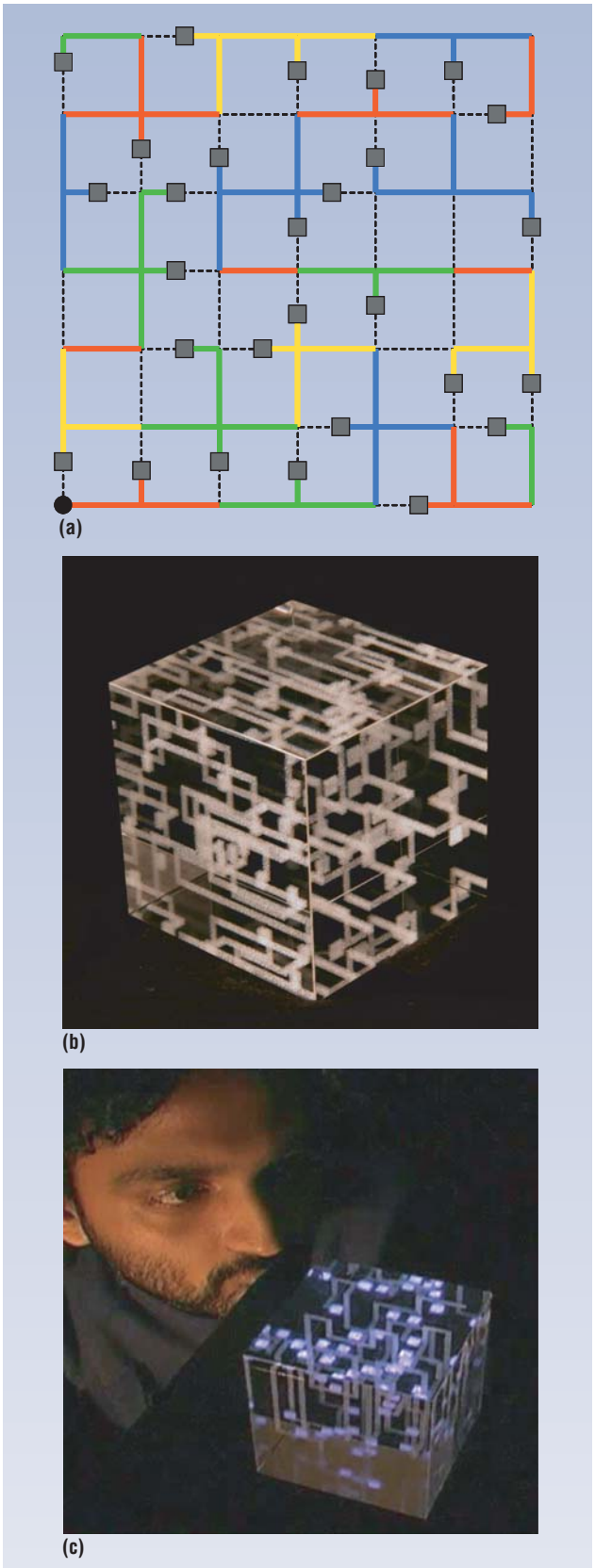


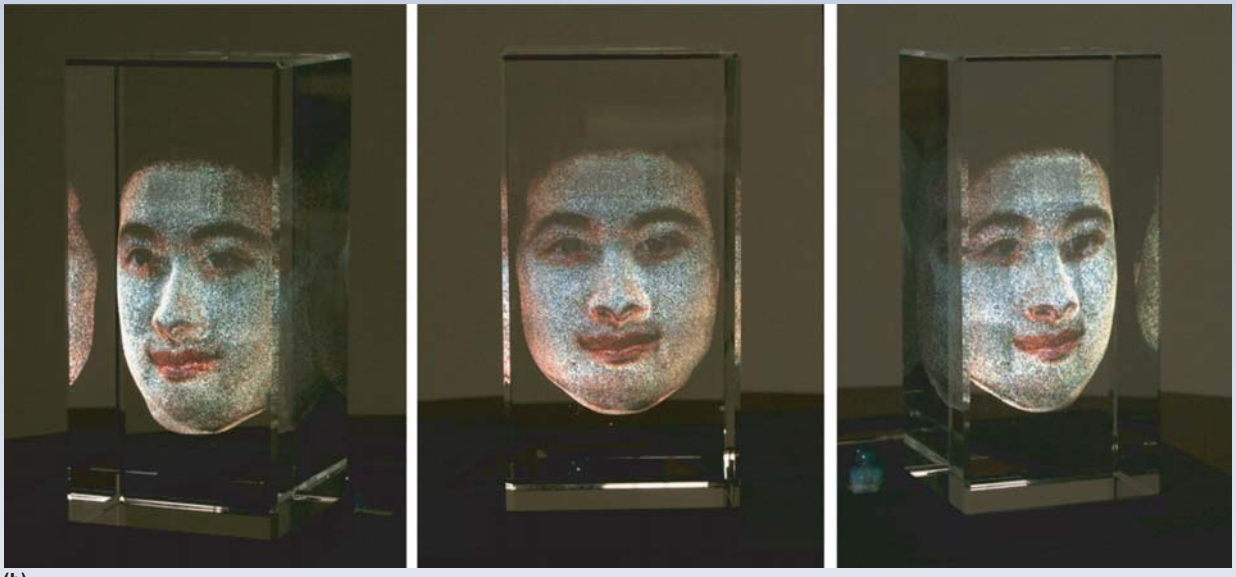
***Figure 5. 3D* Pac-Man.** *(a) The path segments' different colors correspond to different depths. (b) The LID point cloud used in the game. (c) A person playing the game.*

This idea, referred to as *relief projection*, has been around since Walt Disney World introduced talking heads at its Haunted Mansion in 1980. Previous systems used the tedious process of creating a physical bust to make the face model. Our system captures the face model using a depth sensor and etches it into a glass or plastic cube using LID in a matter of minutes.

Figure 6a shows the 3D avatar we developed. We etched the face model, which has 127,223 points, in a $100 \times 100 \times 200$ mm glass cube. In this case, we used a conventional (perspective) projector and folded its throw distance using two planar mirrors to make the system compact.

The entire projection system resides within the black box beneath the glass cube and projects the images up at the face model at a 30-degree angle. Since video of the face may have been captured from a different viewpoint than that of the projection system, we use a simple calibration procedure to warp each frame of the video so it is better aligned with the face model. To perform the calibration, we click on points—prominent features such as eye and lip corners—in a single video frame and the projector pixels that illuminate the corresponding points on the face model. We use these correspondences to compute a piecewise linear mapping between the video frames and the face model. Figure 6b shows three different views of the 3D avatar.

(a)



(b)

Figure 6. 3D avatar. (a) A face model with 127,223 points and the projection system used to develop the 3D avatar. (b) Three different views of the avatar.

Implementing volumetric displays offers an easy and inexpensive way to effectively convey certain types of 3D information. Displays with different resolution properties are geared toward specific classes of objects and contents. Due to the inherent tradeoff between resolutions in the three dimensions, the displays' main limitation is resolution. However, there is substantial room for improvement in the resolution of our fully indexable display.

In future work, we plan to custom design the components of our light engine to achieve significantly greater optical performance. Once this is done, we believe we will be able to implement a fully indexable display with about 250,000 points using a projector with 1 million pixels. We can further enhance this resolution with an array of projectors rather than a single projector.

A second limitation is the size of the glass or plastic cube in which the point cloud is etched. Current LID machines can create cubes that are only as large as $300 \times 300 \times 300$ mm. One way to overcome the size limitation is to construct a cube by tiling together a large number of component cubes. By attaching adjacent cubes using an adhesive with a refractive index matching the material of the cubes, it is possible to avoid undesirable optical effects at the interfaces between the cubes.

Acknowledgments

This work was supported by the Columbia University Initiative in Science and Engineering. Li Zhang provided the face model used in the 3D avatar, and Sergey Trubko did the Zemax optimization of the display optics. Further details and videos related to volumetric displays can be found at www.cs.columbia.edu/CAVE.

References

1. T. Okoshi, *Three-Dimensional Imaging Techniques*, Academic Press, 1976.
2. M. Halle, "Autostereoscopic Displays and Computer Graphics," *Computer Graphics*, vol. 31, no. 2, 1997, pp. 58-62; http://web.media.mit.edu/~halazar/autostereo/autostereo.html.
3. R.M. Wood, *Laser-Induced Damage of Optical Materials*, Institute of Physics, 2003.
4. I.N. Troitski, "Laser-Induced Image Technology (Yesterday, Today, and Tomorrow)," *Proc. SPIE*, vol. 5664, SPIE, 2005, pp. 293-301.
5. K.R. Boff, L. Kaufman, and J.P. Thomas, eds., *Handbook of Perception and Human Performance, vol. 1: Sensory Processes and Perception*, John Wiley & Sons, 1986.
6. S.K. Nayar and V.N. Anand, *Projection Volumetric Display Using Passive Optical Scatterers*, tech. report CUCS-030-06, Dept. Computer Science, Columbia Univ., 2006.

Shree K. Nayar *is the T.C. Chang Chaired Professor in the Department of Computer Science at Columbia University. His research interests include digital imaging, computer vision, computer graphics, human-computer interfaces, and robotics. He received a PhD in electrical and computer engineering from the Robotics Institute at Carnegie Mellon University. Contact him at nayar@cs.columbia.edu.*

Vijay N. Anand *is a 3D graphics programmer at Prana Studios in Mumbai, India, where he develops shaders and artist tools. Anand received an MS in computer science from Columbia University and a BE in computer science and engineering from Regional Engineering College, Tiruchirappalli, India. Contact him at vn2117@columbia.edu.*