Computing and Fabricating Multilayer Models

Michael Holroyd
University of Virginia
Disney Research Zürich

Ilya Baran
Disney Research Zürich

Jason Lawrence
University of Virginia

Wojciech Matusik
MIT CSAIL
Disney Research Zürich

Abstract

We present a method for automatically converting a digital 3D model into a *multilayer model*: a parallel stack of high-resolution 2D images embedded within a semi-transparent medium. Multilayer models can be produced quickly and cheaply and provide a strong sense of an object's 3D shape and texture over a wide range of viewing directions. Our method is designed to minimize visible cracks and other artifacts that can arise when projecting an input model onto a small number of parallel planes, and avoid layer transitions that cut the model along important surface features. We demonstrate multilayer models fabricated with glass and acrylic tiles using commercially available printers.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representation

Keywords: multilayer models, fabrication, volumetric displays

Links: DL PDF WEB

1 Introduction

We describe a method for converting digital 3D models into multilayer models. Our work is inspired by artists like Carol Cohen[1] and Dustin Yellin[2] who reproduce three-dimensional forms by painting on glass or acrylic sheets and then stacking them together (Figure 2). A multilayer model thus consists of a small number of high-resolution 2D images stacked along the 3rd dimension, and creates a natural 3D effect by displaying parts of the object at the appropriate depth over a range of viewing directions.

There is no substitute for the experience of holding and examining a physical 3D object in your hands. Technologies capable of manufacturing 3D objects have seen significant advances in recent years, most notably 3D printing [Dimitrov et al. 2006] and multi-axis milling. However, despite these advances, producing a 3D prototype in full color remains expensive and time-consuming. Furthermore, objects with thin features and disconnected parts (e.g., the tree in Figure 14) cannot be printed at all using existing techniques. In contrast, a multilayer model (as in Figure 1) can be fabricated in minutes and for a fraction of the cost and provides many of the benefits of a physical replica.

[1] carolcohen.com
[2] dustinyellin.com

Figure 1: *A real object next to a multilayer model fabricated in acrylic using our proposed algorithm.*

The process of converting a 3D surface into a multilayer model requires a number of algorithmic advances in order to produce high-quality output. First, we observe that a naïve projection of the input surface onto multiple parallel planes creates artifacts caused by visible seams or cracks and salient features being split between different layers. We propose a novel algorithm that warps each layer based on the way it is occluded by the layers above it in order to avoid these seams while simultaneously seeking cuts along textureless regions. Second, the shadows that each layers casts onto those below it can undermine the intended 3D effect. We describe a fast method for computing a correction factor that compensates for these shadows. Finally, one also needs to consider the contrast loss caused by light absorption in the embedding medium (e.g., glass or acrylic). We propose a simple measurement process to estimate parameters of an analytic absorption model. Based on this model, we restrict the colors printed on each layer to a reduced color gamut that can be achieved throughout the volume.

We show multiple examples of multilayer models produced with our prototype system. Additionally, we include a comparison of our method to simple alternatives that illustrate the importance of properly handling cracks, seams, self shadowing, and volumetric attenuation. We believe multilayer models can serve as useful rapid prototypes, teaching aids, art, and personalized memorabilia.

2 Related Work

Volumetric Glass Models A source of inspiration for this work are artists who use stacks of painted glass or acrylic sheets to create three-dimensional forms (Figure 2). This stunning art makes it apparent that they have overcome similar technical challenges to the ones we address in this work. A related computer-driven technique

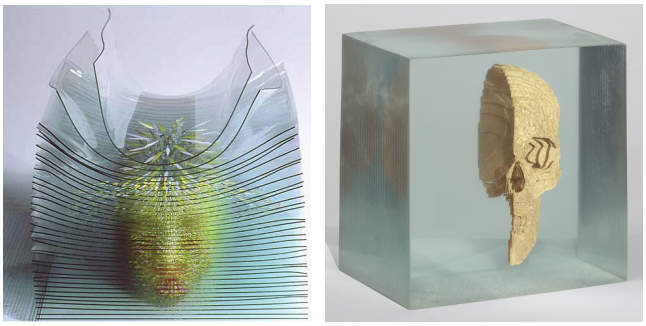


Figure 2: *Our work is inspired by artists who use stacks of painted glass to reproduce three-dimensional forms.* ***Left:*** *Carol Cohen, Head With Halo (1989) Slumped glass, Paint, 12" x 16" x 12".* ***Right:*** *Dustin Yellin, Gold Skull Magnus, No. 1 (2009) Resin, Acrylic, and Ink, 18" x 13" x 12.125". Images reproduced with the permission of the artists.*

is laser-etched glass [Troitski 2005; Wood 2003]: a focused laser beam creates tiny fractures at specific 3D points in a glass cube, causing them to scatter light. This can produce very precise and high-resolution point clouds, but is limited to monochrome representations. Furthermore, the machines used for this type of etching are bulky, expensive, and slow.

3D Printers The most common method of rapid prototyping is additive 3D printing [Dimitrov et al. 2006], which constructs objects by depositing and accumulating layers of material. Today there are a wide range of commercially available devices, ranging from high-end machines like ZCorp's ZPrinter 650 that can print full color (but weigh a third of a ton and cost tens of thousands of dollars) to single-material low-end desktop printers. However, producing high-resolution 3D color prototypes is still prohibitively expensive and time consuming for many people. Furthermore, even when cost is not an issue, these printers cannot fabricate certain 3D content including objects with very thin features, non-watertight surfaces, or multiple disconnected components.

3D Displays There is an expanding terrain of 3D displays that fall into two basic categories: stereoscopic displays and volumetric displays. Although 3D displays are able to present dynamic content, which is not possible with our approach, we believe that some of the techniques described in this paper could be adapted to dynamic multilayer displays in the future.

Stereoscopic displays create a 3D effect using only a 2D display by ensuring that each of a viewer's eyes sees a different image and exploiting the binocular cue of stereopsis (disparity). In general, these displays require either special glasses or are optimized for a specific viewing position by using either parallax barriers or lenticular sheets to control the projected light field (autostereoscopic). These systems can produce high quality images, but have limited *angular resolution* (e.g., full "walk arounds") and are often unresponsive to subtle head movements.

In contrast, volumetric displays can support many users simultaneously. They are also fully responsive to head movements and benefit from the additional depth cues of motion parallax, accommodation, and convergence. Despite these advantages, however, constructing accurate high-resolution volumetric displays remains difficult in practice. Current designs rely heavily on multiplexing in time, for example with a high-speed rotating component that sweeps over the volume of interest while emitting or reflecting synchronized light patterns [Jones et al. 2007; Soltan et al. 1992; Favalora 2005].

The 3D displays most closely related to our work consist of multiple parallel high-resolution 2D display surfaces arranged at various depths relative to the viewer. This includes systems that superimpose multiple 2D displays using beamsplitters [Tamura and Tanaka 1982; Akeley et al. 2004], a stack of LCD panels [Gotoda 2010; Wetzstein et al. 2011], or that project light onto thin layers of fog [Lee et al. 2008], parallel sheets of water droplets [Barnum et al. 2010], or shutter planes [Sullivan 2004]. In all of these cases, the density and number of projection barriers is limited by cost or practical considerations such as projector placement and refresh rate. Thus, the challenge is to create a convincing 3D effect while masking the discrete locations of the display surface.

Rendering Imposters Approximating a complex 3D shape with a relatively small number of 2D images is a common technique used to accelerate rendering [Schaufler 1998b; Schaufler 1998a; Decoret et al. 1999]. This can be used to achieve an adjustable level of detail in which distant complex geometry is replaced with these *imposters* to trade rendering quality for speed. Unlike the case considered here of fabricating static multilayer models, these techniques rely on the ability to dynamically update the proxy images based on viewing location and often use a collection of interpenetrating planes at different orientations to minimize artifacts.

3 Computing Multilayer Models

Our goal is to compute a multilayer model that best captures the appearance of an input 3D digital model. We will use the term "pixel" to refer to a discrete location within each layer that is addressable with, for example, a printer. Our goal is to compute a color and opacity at each pixel that best reproduces the appearance of an input model. We assume that the size, position, and resolution of the layers relative to the input model are known in advance.

In the case of volumetric data, this problem can be solved using well understood resampling methods from volume rendering. On the other hand, 3D surfaces present a unique set of challenges that have not been addressed by previous techniques.

3.1 Surfaces

We assume that the position and orientation of the input surface relative to the multilayer domain is defined in advance, so that each pixel on each layer defines a 3D position in a common coordinate system. In addition, we ignore view- and light-dependent aspects of the surface appearance. The desired lighting and material properties should be "baked" into a single RGB texture prior to applying our algorithm. In practice, we render the input model under a constant environment map and use the color as seen from a predetermined camera location. We also assume a limited range of viewing angles for our fabricated model. This is modeled as a cone with cutoff angle $\theta_{\max} < 90^\circ$, centered around a *principal view direction* (the red cone in Figure 3), the front-facing view direction perpendicular to the layer orientation. We typically use $\theta_{\max} = 45^\circ$; the field of view allowed by our approach is several times greater than that of any autostereoscopic display.

3.1.1 Algortihm Overview

We compute a multilayer model from front to back, finalizing each layer before moving on to the next one. First, we compute a projection of the surface onto each layer that avoids visible cracks within the target viewing cone (Section 3.1.2). Second, we refine this assignment of surface locations to pixels in order to avoid splitting the model along salient features (Section 3.1.3). Third, we compute a correction factor that compensates for the shadows that layers cast

onto one another (Section 3.1.4). Finally, we map the output colorspace of each layer into a reduced gamut that accounts for the light attenuation inside the fabrication medium (Section 3.1.5). We then proceed to the next layer and repeat these same steps.

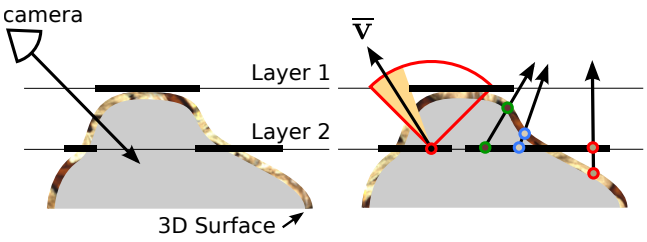


Figure 3: ***Left:*** *Directly projecting each point on the input surface to its closest pixel in a multilayer model results in seams visible along off-axis viewing directions.* ***Right:*** *We compute the color of each pixel by sampling the surface along the mean visible view direction, $\overline{\mathbf{v}}$, computed over the cone of permissible view directions (red). Examples of $\overline{\mathbf{v}}$ at other layer pixels are shown along with the location they intersect the mesh and associated color.*

3.1.2 Geometric Warping

The most straightforward approach for converting a surface to a multilayer model is to project the color of each surface location onto its nearest pixel. However, this creates seams at off-axis viewing directions, as shown in Figure 4. Prior work on using imposters for accelerated rendering addressed this problem by expanding the range of depths projected onto each layer, or "overdrawing", until these cracks disappear [Schaufler 1998b]. However, this can result in duplicated surface features visible at extreme camera angles and distorted silhouettes, also shown in Figure 4. We propose an alternative solution that avoids these problems.

At each pixel in the current layer, we compute the mean visible view direction, $\overline{\mathbf{v}}$, illustrated in Figure 3. This is equal to the average of the vectors that lie within the viewing region and are not occluded by other layers. We then intersect $\overline{\mathbf{v}}$ with the surface to determine the pixel's color. We experimented with alternative ray directions such as the vector centered within the largest cone of unoccluded directions, but $\overline{\mathbf{v}}$ has the advantage of varying smoothly from one pixel to the next. This avoids sharp transitions in the colors printed on each layer (see Figure 7a) that would otherwise produce distracting artifacts in the result. This approach is similar to the use of "bent normals" often performed along with ambient occlusion calculations [Landis 2002]: an environment map is sampled in the average unoccluded direction. In our case, this sampling strategy effectively stretches the content printed on each layer so that there are no seams or cracks visible within the target viewing region. Figure 4 compares our result with several other methods.

Fast Approximation of $\overline{\mathbf{v}}$ A brute-force computation of $\overline{\mathbf{v}}$ requires $O(L^2N^2)$ time for L layers and N pixels per layer (about 2 hours per layer with 1024×1024 resolution). Instead, we introduce an efficiently-computable approximation for multilayer models: we collapse all occluding layers onto a single flattened layer above the current one. This is equivalent to computing the maximum opacity value seen at each pixel along a ray perpendicular to the layer orientation. As Figure 5 shows, using this flattened layer in place of the true geometry provides a good approximation of the occlusion in most cases, especially for roughly convex objects. With a single planar occluder above the current layer, $\overline{\mathbf{v}}$ can be formulated as a convolution and therefore computed efficiently.

Let D be the thickness of a layer, or equivalently, the distance from the current layer to the flattened layer B above. Recall that $\theta_{\max}$ is the maximum angle between the principal view direction and a direction in the viewing cone. We write $\overline{\mathbf{v}}(x, y)$ (unnormalized) as a spherical integral over the viewing cone:

$$\overline{\mathbf{v}}(x,y) = \int_0^{2\pi}\int_0^{\theta_{\max}} V(x,y,\theta,\phi) \begin{pmatrix} \sin\theta\cos\phi \\ \sin\theta\sin\phi \\ \cos\theta \end{pmatrix} \sin\theta \, d\theta \, d\phi,$$

where $V(x, y, \theta, \phi) = B(x + D\tan\theta\cos\phi, y + D\tan\theta\sin\phi)$ is the binary visibility function. We now make the variable substitution $\theta = \cos^{-1}(D/\sqrt{\hat{x}^2 + \hat{y}^2 + D})$ and $\phi = \tan^{-1}(\hat{y}/\hat{x})$. Then the integral becomes:

$$\overline{\mathbf{v}}(x,y) = \iint_{-\infty}^{\infty} B(x+\hat{x}, y+\hat{y}) G(\hat{x},\hat{y}) \, d\hat{x} \, d\hat{y},$$

where

$$G(\hat{x},\hat{y}) = \begin{pmatrix} \sin\theta(\hat{x},\hat{y})\cos\phi(\hat{x},\hat{y}) \\ \sin\theta(\hat{x},\hat{y})\sin\phi(\hat{x},\hat{y}) \\ \cos\theta(\hat{x},\hat{y}) \end{pmatrix} \sin\theta(\hat{x},\hat{y}) \left| \begin{matrix} \frac{d\theta}{d\hat{x}} & \frac{d\theta}{d\hat{y}} \\ \frac{d\phi}{d\hat{x}} & \frac{d\phi}{d\hat{y}} \end{matrix} \right|$$

for $\theta(\hat{x}, \hat{y}) \leq \theta_{\max}$ and $G(\hat{x}, \hat{y}) = 0$ for $\theta(\hat{x}, \hat{y}) > \theta_{\max}$. The (unnormalized) mean unoccluded vector can therefore be written as $B * G$, the convolution of the occluding layer with the three components of the filter function G. We compute these convolutions efficiently using the FFT, taking $O(LN \log N)$ time for all layers. For layers with 1024×1024 spatial resolution, this optimization reduces the cost of computing $\overline{\mathbf{v}}$ at every pixel from hours with a brute force approach to only 3-4 seconds.

3.1.3 Preserving Salient Surface Features

Discontinuities produced by a multilayer model occur at layer boundaries. When these "seams" intersect salient surface features, they can lead to visible artifacts (Figure 6). We address this problem by lifting the requirement that each point on the input surface be projected onto its closest pixel. Instead, we allow surface features to remain intact on a single layer at the expense of increasing the distance between their location and the pixel onto which they are projected. We achieve this by reformulating the assignment of surface points to layer pixels as the solution to a graph-cut problem.

We model each pixel in the current layer as a node in a graph with edges that connect it to its adjacent pixels and to a source and sink node. After the cut is computed, any pixels that are connected to the source will remain on the current layer whereas those connected to the sink will be discarded (to be printed on a subsequent layer).

So that surface points prefer being printed on their closest layer, the weights along the edges connecting each node to the sink and

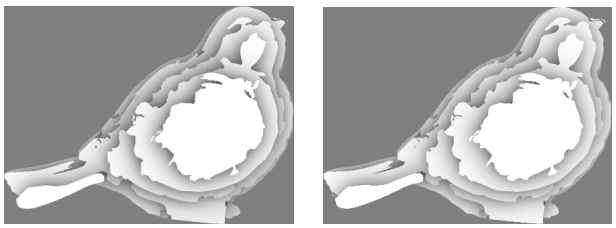
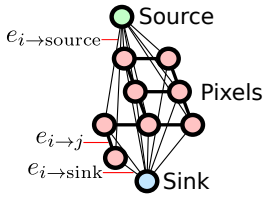


Figure 5: *A comparison between a reference ambient occlusion solution of a simple multilayer model (left) and our approximation that involves collapsing the occluding layers and using the FFT to expedite the computation (right). These solutions typically show very close agreement, especially for roughly convex objects.*

Figure 4: *This figure compares the results of several projection strategies, rendered at an oblique angle. From the principal view direction, all five renderings look the same. From left to right: input model, projection to the nearest plane, alpha-weighted projection to the nearest two planes, overdraw (1.5×) [Schaufler 1998b], mean visible view direction (our method).*

source are set based on the distance between the pixel and its intersection with the surface along the direction $\overline{\mathbf{v}}$. To avoid cutting across important features, the edge weights between adjacent pixels are based on a saliency map S computed using the output from the previous step (Figure 7).

Let t be the signed distance along the ray that connects the center of each pixel i to its nearest surface point along $\overline{\mathbf{v}}$. Note that $t > 0$ if this nearest point is *above* the current layer and $t < 0$ if it is *below*. We compute a saliency map S by convolving the current layer with a difference of Gaussians ($\sigma = 5, 1$), but more complex saliency measures are possible including those computed in 3D over the input surface. Alternatively, S can be "painted" by the user to protect surface features not easily captured by automatic methods. Edge weights are assigned as follows:



$$e_{i\to\text{sink}} = \begin{cases} |t| & \text{if } t < 0 \\ 0 & \text{otherwise} \end{cases}$$

$$e_{i\to\text{source}} = \begin{cases} 0 & \text{if } t < 0 \\ t & \text{otherwise} \end{cases}$$

$$e_{i\to j} = S_i + S_j$$

We compute the minimum graph cut using the library provided by Boykov and Kolmogorov [2001]. Pixels connected to the sink are discarded from the current layer (made transparent) and will appear in a lower layer. Figure 7 shows an example input layer, saliency map, signed distance field, and the resulting minimum cut. Note that important surface features such as the eyes remain intact, whereas surface points far from the current layer are removed.

3.1.4 Accounting for Shadowing Between Layers

Each opaque pixel potentially casts shadows onto the layers below. As illustrated in Figure 8, these shadows can reveal the discrete nature of a multilayer model and undermine the desired 3D illusion. If we assume that the lighting environment is known *a priori*, we

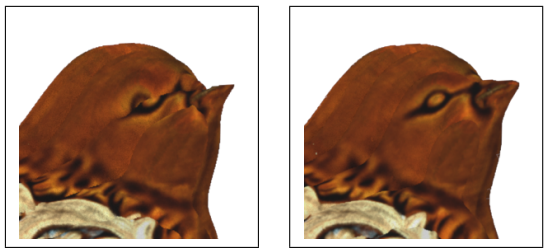


Figure 6: *(left) Artifacts occur when salient image features, such as the bird's eye, are split between neighboring layers. (right) We solve a graph-cut problem that refines the projection of the surface onto each layer in order to keep features intact on a single layer.*

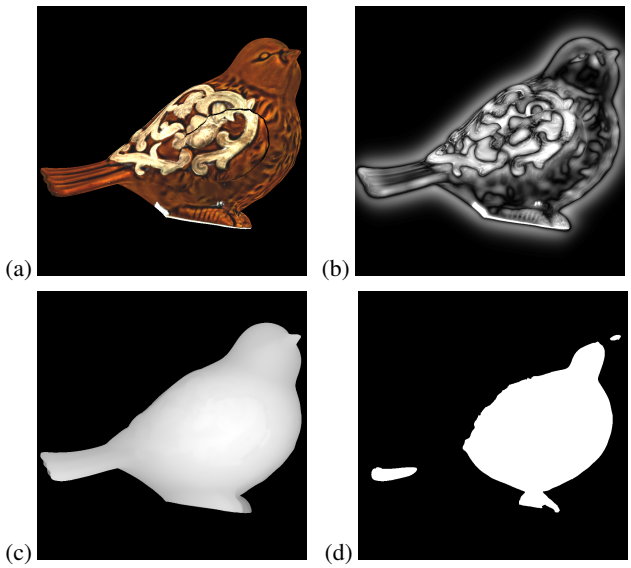


Figure 7: ***(a)*** *Starting from the mean view direction projection of the surface onto the current layer, we compute the minimum cut of a graph designed to avoid splitting features between different layers.* ***(b)*** *Image saliency, computed from the 2D projection, determines the edge weights between adjacent pixels.* ***(c)*** *Signed distance from each pixel to the nearest surface location along $\overline{\mathbf{v}}$ determines the source/sink edge weights.* ***(d)*** *The minimum cut discards pixels that are sampled from distant surface locations while avoiding cuts across salient features such as the eyes and mouth.*

can compensate for the shadowing at each pixel by adjusting its brightness. Under a constant environment, this correction factor is equivalent to dividing by the *ambient occlusion* at each pixel.

We efficiently approximate the ambient occlusion using the same convolution strategy described in Section 3.1.2, but replace the filter G with a scalar cosine-weighted filter that accounts for the $(n \cdot l)$ falloff. Figure 8 shows a global illumination rendering[3] of a multilayer model before and after applying this correction factor based on our approximation of a constant diffuse environment. Figure 9 illustrates that correcting using ambient occlusion is acceptable for other low-frequency lighting environments.

3.1.5 Accounting for Absorption in the Fabrication Medium

Another important consideration is the way light is absorbed and scattered by the fabrication medium itself (e.g., acrylic or glass). This can reduce the contrast and brightness of layers near the back of the model and lead to an uneven appearance. Inspired by "airlight" models used for atmospheric correction [Nayar and

[3] Jakob, Wenzel. `mitsuba-renderer.org`

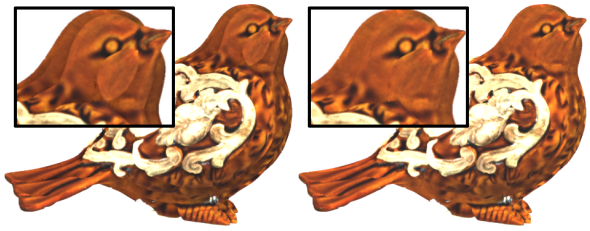


Figure 8: *Global illumination renderings of the multilayer bird model. (left) The dark bands around each layer are caused by shadowing and can disrupt the intended 3D effect. (right) We compensate for this shadowing by increasing the brightness at each pixel according to the ambient occlusion.*

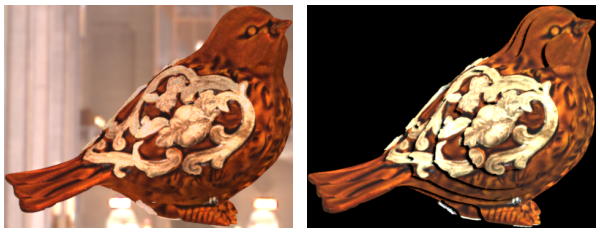


Figure 9: *(left) The bird model after our shadow correction in the Grace Cathedral environment. (right) The same model illuminated by a small area lightsource that casts hard shadows. Our method for shadow compensation breaks down under these types of high-frequency lighting conditions.*

Narasimhan 1999], we chose to fit the parameters of an analytic function that predicts color desaturation as a function of the distance a ray of light travels through the medium. Our model accounts for both the absorption of the medium as well as multiple scattering. Specifically, we assume that an opaque pixel with color K, when observed through our print medium with depth d, will produce a color $K' = Ke^{-\sigma d} + A(1 - e^{-\beta d})$, where A is known as the "airlight" color and refers to the color of the medium itself due to multiple scattering. We estimate the airlight color A, the scattering coefficient σ, and the isotropic in-scattering coefficient β by recording measurements of a printed color checker chart below an increasing number of layers of our print medium.

The results we obtained for our materials are shown in Figure 10. When using non-flatbed printers, we print onto thin acetate sheets that are then attached to the surface of either acrylic or glass tiles (Section 4.1). These graphs plot measurements and the resulting fits with and without these acetate sheets. Because these functions flatten out rather quickly, we found that reducing the gamut of just the first few layers to compensate for the contrast loss of lower layers was sufficient and avoids unnecessarily reducing the entire model's contrast to that of the back-most layer. We perform simple linear rescaling to compress the gamut of the first three layers so that they do not exceed the range of colors visible on the fourth layer. Within this reduced gamut, we then solve for K, the color that we ultimately print, in order to achieve an observed color K'. Figure 11 shows a printed model with and without this correction applied.

3.2 Volumes

For completeness, we describe a method for producing multilayer models from volumetric datasets, such as those produced by a CT machine, MRI machine, or density estimation simulation. For volumetric data, assigning color and opacity values to the pixels in each layer can be viewed as a classic resampling problem that has

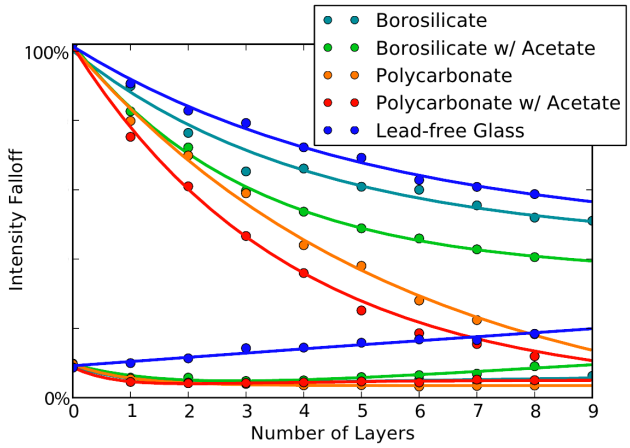


Figure 10: *Measured falloff in available gamut (circles) and our airlight model fit (lines) as a function of the number of layers. Upper and lower lines represent the maximum and minimum apparent color K' that can be observed (red channel; other channels are similar).*

Figure 11: ***Left:*** *A multilayer model printed using polycarbonate and acetate sheets without accounting for attenuation and multiple scattering.* ***Right:*** *The same model printed after applying these corrections. This model is composed of seven layers of polycarbonate with acetate sheets. Note how the brightness and contrast of the front-most layers match the back layers in the corrected model.*

been well studied in the past [Marschner and Lobb 1994]. We use a simple linear filter that is anisotropic along the sparsely sampled z-axis as shown in Figure 12. To improve performance, rather than integrating over the filter's domain at each pixel we instead forward-project in parallel every voxel of the input onto its set of overlapping pixels.

Our printouts are designed to be subtractive – the fabricated volumes are backlit and the ink deposited at each pixel absorbs some amount of light. All light passes through the same number of layers, thus we do not need to correct for attenuation through the print medium. Fabricating emissive layers would require applying the type of attenuation correction we use for surfaces (Section 3.1.5).

4 Results

We used three commercially available printers and a variety of transparent media to fabricate multilayer models. Note that a maximum possible viewing angle can be determined from the index of refraction of the embedding medium using Snell's law (42° in the case of acrylic, which has an index of refraction of $\eta = 1.49$). This provides a natural way of determining the cut-off value of the viewing region $\theta_{\max}$. We ignored refraction otherwise because the

Figure 13: ***Left:*** *Input model.* ***Right:*** *Four views of the corresponding 9-layer physical replicas constructed from lead-free glass.*

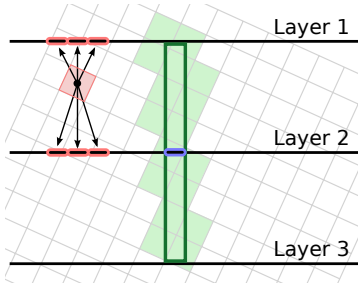


Figure 12: *We resample an input volume (light gray) using an anisotropic linear filter (dark green). Instead of iterating over pixels in each layer, we iterate over voxels in the input and project their values onto their surrounding layers (red).*

effect is essentially equivalent to shifting the viewer position.

4.1 Surfaces

Fabricating multilayer surface models requires the ability to apply opaque ink to a semi-transparent medium. Our initial prototypes used an ALPS MD-5500 thermal printer to apply an opaque white base coat followed by the desired layer colors in a second pass. This particular printer cannot print directly onto glass or acrylic and were therefore required to use thin flexible transparencies that in our experience are of lower optical quality than glass or acrylic tiles alone. We found that .005" thick acetate film transparencies are sturdy enough for printing and sufficiently clear to produce good results. We set the printed film between glass or acrylic tiles to produce the final model.

We experimented with two different types of tiles. Borosilicate glass tiles of size 4" $\times$ 4" $\times$ .125" required about 30 seconds of printing time and cost $8 per tile. Polycarbonate acrylic came in 6" $\times$ 6" $\times$ .236" tiles and required roughly 2 minutes of printing time and cost $2.60 per tile. We also had multilayer models produced using a Durst flatbed UV printer, which can deposit ink directly onto glass tiles. We used high quality lead-free glass to improve the optical quality of our final models. In this case, 6" $\times$ 6" tiles required 3-4 minutes per layer and cost $18 per tile.

Converting a 3D surface to a multilayer model using our system takes between 5 to 45 seconds per layer, the majority of which is spent calculating ray/geometry intersections.

Figure 1 shows a 7-layer polycarbonate model computed from a 3D surface scanned from a real object [Holroyd et al. 2010]. Figure 14 shows a 10-layer polycarbonate model of a tree that has many thin features and disconnected components, created using image-based tree modeling [Tan et al. 2007]. This type of object would be impossible to fabricate using an additive 3D printer or a milling machine. Figure 13 shows several 9-layer models produced using lead-free glass, along with renderings of the original models.

We chose the number of layers for each model manually. While increasing the number of layers improves the quality of the final model, it also increases the cost and printing time. Figure 15 shows the bird model with 5, 9, and 17 layers, which corresponds to .118, .236, and .472 inch thickness tiles respectively.

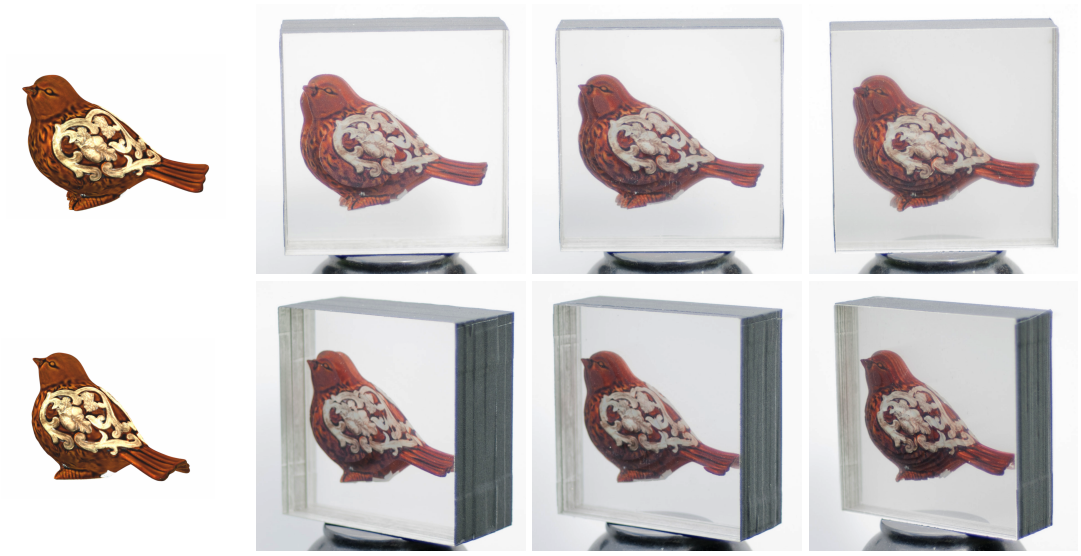


Figure 15: *The input bird model (left) and the 5, 9, and 17 layer multilayer models (right) for two different views.*

Figure 14: *A 10-layer model of a tree constructed from polycarbonate and acetate sheets, viewed from approximately 20° above the principal view direction. Objects with very thin features such as this tree cannot be printed using additive 3D printers.*

4.2 Volumes

For volumetric multilayer models, we used a DCS Direct Jet 1309 flatbed printer,[4] which can print semi-transparent ink directly onto $100 \times 100 \times 3$mm acrylic tiles. For 16-20 layer models, resampling the original volume takes only a few seconds, and the entire printing process requires approximately 20 minutes. We estimate the cost of materials (acrylic and ink) to be $2.80 per tile.

Figure 16 shows an MRI volume dataset comprised of 170 slices next to a 17-layer model fabricated with our system. We believe multilayer models of volumetric datasets could serve as useful visualization aids and instructional tools.

[4] directcolorsystems.com

4.3 Limitations and Discussion

Our method produces compelling results in most of the cases we tested, but has difficulty in others. The best results we obtained were for primarily convex objects with varying surface texture, which helps make seams less conspicuous. Long thin features that cross multiple layers may cause cracks from certain views regardless of the warp applied (e.g., the lion's rear right leg in Figure 13). Additionally, hard directional lighting runs counter to our ambient occlusion assumption as illustrated in Figure 9. Our method is also limited to diffuse surfaces. An interesting area of future work is to reproduce directionally-dependent appearance in multilayer models.

An alternative approach to our layer-by-layer algorithm would be to solve a global optimization over all layers simultaneously. The challenge in that case is defining an objective function that captures our desired visual properties and can be computed in a feasible amount of time. One possible approach is to compute optimal images in a least-squares sense over a discrete set of viewpoints. This approach is taken by Wetzstein et al. [2011] who use a tomographic solver to match a target lightfield. Although this method can achieve excellent results for small fields of view, it can lead to overblurring when computed over larger fields of view due to an insufficient number of degrees of freedom. Our method avoids this problem by relaxing the requirement that multilayer models match the 3D model in the least squares sense; it instead allows the introduction of non-linear spatial distortions to accomodate the objective.

5 Conclusion and Future Work

We have described a set of algorithms for converting 3D surfaces and volumes into multilayer models. Our algorithm avoids visible cracks between layers, splitting surface features between layers, and compensates for inter-layer shadows and light absorption inside the fabrication medium. We demonstrated a prototype system for fabricating multilayer models that uses commercially available printers with glass or acrylic tiles and demonstrated that these models are fast and inexpensive to construct. Our approach is approximate but fast, allowing the user to interactively adjust the orientation of the object along with the number of layers before printing.

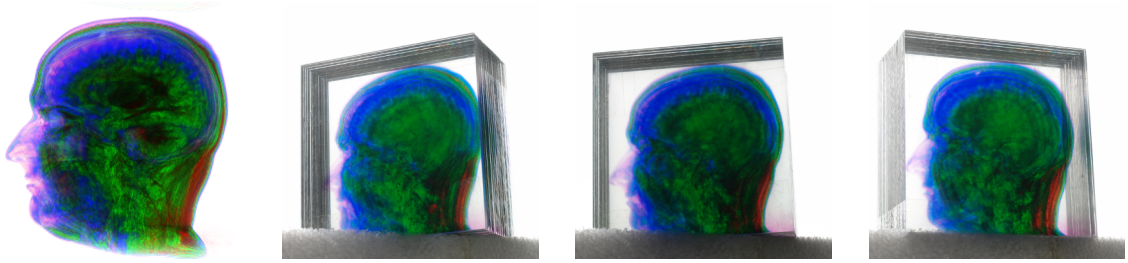


Figure 16: *A MRI dataset fabricated using our volumetric resampling algorithm.* ***Left:*** *Traditional 170 slice rendering.* ***Right:*** *three views of a printout formed from a stack of 17 acrylic tiles.*

We plan to adapt these core algorithms to active multilayer displays, which could be achieved by assembling a series of parallel transparent LCDs. The key issues we expect to face are how to ensure temporal coherency between frames and how to achieve interactive framerates. We also intend to study using flatbed printer technology with a wider range of inks to create more compelling and realistic printouts. Another consideration is the possibility of modifying the surface geometry of individual tiles, for example by milling the acrylic surface, to enable multilayer models with more complex surface shading.

Acknowledgements

We wish to thank Dustin Yellin and Carol Cohen for granting us permission to reproduce images of their art. We'd also like to thank the CAVGRAPH and SIGGRAPH reviewers for their helpful and constructive feedback.

References

AKELEY, K., WATT, S. J., GIRSHICK, A. R., AND BANKS, M. S. 2004. A stereo display prototype with multiple focal distances. *ACM Transactions on Graphics 23*, 3 (Aug.), 804–813.

BARNUM, P. C., NARASIMHAN, S. G., AND KANADE, T. 2010. A multi-layered display with water drops. *ACM Transactions on Graphics 29*, 4 (July), 1.

BOYKOV, Y. Y., AND KOLMOGOROV, V. 2001. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. In *EMMCVPR*, 359–374.

DECORET, X., SILLION, F., SCHAUFLER, G., AND DORSEY, J. 1999. Multi-layered impostors for accelerated rendering. *Computer Graphics Forum 18*, 3 (Sept.), 61–73.

DIMITROV, D., SCHREVE, K., AND DE BEER, N. 2006. Advances in three dimensional printing state of the art and future perspectives. *Rapid Prototyping Journal 12*, 136–147.

FAVALORA, G. 2005. Volumetric 3D displays and application infrastructure. *Computer 38*, 8 (Aug.), 37–44.

GOTODA, H. 2010. A multilayer liquid crystal display for autostereoscopic 3D viewing. In *Proc. SPIE*, vol. 7524.

HOLROYD, M., LAWRENCE, J., AND ZICKLER, T. 2010. A coaxial optical scanner for synchronous acquisition of 3D geometry and surface reflectance. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2010)*.

JONES, A., MCDOWALL, I., YAMADA, H., BOLAS, M., AND DEBEVEC, P. 2007. Rendering for an interactive 360 degree light field display. *ACM Trans. Graph. 26* (July).

LANDIS, H., 2002. Production-ready global illumination. Course 16 notes, SIGGRAPH 2002. http://www.spherevfx.co.uk/downloads/ProductionReadyGI.pdf.

LEE, C., DIVERDI, S., AND HÖLLERER, T. 2008. Depth-fused 3D imagery on an immaterial display. *IEEE transactions on visualization and 15*, 1, 20–33.

MARSCHNER, S., AND LOBB, R. 1994. An evaluation of reconstruction filters for volume rendering. *Proceedings Visualization '94*, 100–107,.

NAYAR, S. K., AND NARASIMHAN, S. G. 1999. Vision in bad weather. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 820–827.

SCHAUFLER, G. 1998. Image-based object representation by layered impostors. *Proceedings of the ACM symposium on Virtual reality software and technology 1998 - VRST '98*, 99–104.

SCHAUFLER, G. 1998. Per-object image warping with layered impostors. *Rendering Techniques 1*, 145–156.

SOLTAN, P., TRIAS, J., ROBINSON, W., AND DAHLKE, W. 1992. Laser Based 3-D Volumetric Display System (First Generation). *SPIE-The International Society for Optical*, May, 9–14.

SULLIVAN, A. 2004. DepthCube solid-state 3D volumetric display. In *Stereoscopic Displays and Virtual Reality Systems XI*, SPIE, San Jose, CA, USA, A. J. Woods, J. O. Merritt, S. A. Benton, and M. T. Bolas, Eds., vol. 5291, 279–284.

TAMURA, S., AND TANAKA, K. 1982. Multilayer 3-d display by multidirectional beam splitter. *Applied Optics 21*, 3659–3663.

TAN, P., ZENG, G., WANG, J., KANG, S. B., AND QUAN, L. 2007. Image-based tree modeling. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007) 27*.

TROITSKI, I. 2005. Laser-induced image technology (yesterday, today, and tomorrow). In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, vol. 5664, 293–301.

WETZSTEIN, G., LANMAN, D., HEIDRICH, W., AND RASKAR, R. 2011. Layered 3D: Tomographic image synthesis for attenuation-based light field and high dynamic range displays. *ACM Trans. Graph. 30*, 4.

WOOD, R. 2003. *Laser-induced damage of optical materials*. Taylor & Francis.