Interpolating Vertical Parallax for an Autostereoscopic 3D Projector Array

Andrew Jones^a, Koki Nagano^a, Jing Liu^{a,b}, Jay Busch^a, Xueming Yu^a,

Mark Bolas^a and Paul Debevec^a

^aUSC Institute for Creative Technologies, Playa Vista, CA, USA;

^bUniversity of Santa Cruz, Santa Cruz, CA, USA

ABSTRACT

We present a technique for achieving tracked vertical parallax for multiple users using a variety of autostereoscopic projector array setups, including front- and rear- projection, and curved display surfaces. This *hybrid parallax* approach allows for immediate horizontal parallax as viewers move left and right and tracked parallax as they move up and down, allowing cues such as 3D perspective and eye contact to be conveyed faithfully. We use a lowcost RGB-depth sensor to simultaneously track multiple viewer head positions in 3D space, and we interactively update the imagery sent to the array so that imagery directed to each viewer appears from a consistent and correct vertical perspective. Unlike previous work, we do not assume that the imagery sent to each projector in the array is rendered from a single vertical perspective This lets us apply hybrid parallax to displays where a single projector forms parts of multiple viewers' imagery. Thus, each individual projected image is rendered with multiple centers of projection, and might show an object from above on the left and from below on the right. We demonstrate this technique using a dense horizontal array of pico-projectors aimed into an anisotropic vertical diffusion screen, yielding 1.5° angular resolution over 110° field of view. To create a seamless viewing experience for multiple viewers, we smoothly interpolate the set of viewer heights and distances on a per-vertex basis across the array's field of view, reducing image distortion, crosstalk, and artifacts from tracking errors.

Keywords: computer graphics, displays, tracking, three dimensions, projector array, autostereoscopic, parallax, multiple centers of projection

Further author information: (Send correspondence to Andrew Jones) E-mail: gl@usc.ict.edu

1. INTRODUCTION

Autostereoscopic displays hold the promise of seamless 3D imagery that can be seen from any viewpoint without the need for special 3D glasses. The fundamental obstacle is finding new ways to redirect pixels in different angular directions in order to be seen by multiple viewers. Unfortunately, full motion parallax comes at a high cost as the total pixel count needed by a display is directly proportional to the number of viewers. Traditionally a 3D display with 10 horizontal views requires 10 times the number of pixels as a similar 2D display, but a display with 10 horizontal and vertical views requires 100 times the pixels. As a result, most autostereoscopic displays are limited to horizontal parallax only (HPO) where the image does not change as the height of the viewer changes. This is a reasonable tradeoff as human movement is dominated by horizontal motion. Yet it is still desirable to handle multiple users with varying physical heights, and occasional changes in vertical parallax as users approach the display, jump, or crouch.

Projector arrays are well suited for 3D displays because of their ability to generate dense and steerable arrangements of pixels. As video projectors continue to shrink in size, power consumption, and cost, it is now possible to closely stack projectors so that their lenses are almost continuous. We present a new HPO display utilizing a single dense row of projectors. A vertically anisotropic screen turns the glow of each lens into a vertical stripe while preserving horizontal angular variation. The viewer's eye perceives several vertical stripes from multiple projectors that combine to form a seamless 3D image. Rendering to such a display requires the generation of multiple center of projection (MCOP) imagery, as different projector pixels diverge to different viewer positions. Previously, Jones et al.^{1,2} proposed a MCOP rendering solution in the context of high-speed video projected onto a spinning anisotropic mirror. A front-mounted projector array may be seen as an unfolded spinning mirror display where each high-speed frame corresponds to a different discrete projector position. In this paper, we extend this framework for use with both front and rear-projection 3D projector arrays.

As every viewer around a HPO display perceives a different 3D image, it is possible to customize each view with a different vertical perspective. Such a setup has the unique advantage that every viewer can have a unique height while experiencing instantaneous horizontal parallax. The problem is create a continuous estimate of viewer height and distance for all potential viewing angles given only a sparse set of tracked viewer positions. This estimate must provide consistent vertical perspective to both tracked and untracked viewers. The set of viewers is dynamic, and it is possible that the tracker will miss a viewer particularly as viewers enter or leave the viewing volume. Previous techniques for rendering MCOP images for autostereoscopic displays^{1,2} assume a constant viewer height and distance for each projector frame. In practice, this limitation can result in visible distortion, image tearing, and crosstalk where a viewer sees slices of multiple projector frames rendered with inconsistent vertical perspective. This is especially visible when two viewers are close together but have different heights. We solve this problem by dynamically interpolating multiple viewer heights and distances within each projector frame as part of a per-vertex MCOP computation. We then compare the performance of different interpolation functions. Our algorithm can handle both flat screens as well as convex mirrored screens that further increase the ray spread from each projector.

Our primary contributions are:

- 1. An autostereoscopic 3D projector array display built with off-the-shelf components
- 2. A new per-vertex projection algorithm for rendering MCOP imagery on standard graphics hardware
- 3. An interpolation algorithm for computing multiple vertical perspectives for each projector
- 4. An analysis of curved mirrored screens for autostereoscopic projector arrays

2. RELATED WORK

Many different forms of glasses-free displays have been developed over the last century and good surveys³ exist covering emerging 3D display technologies. We will focus on the development of screen materials, projector arrays, and user tracking that pertain to our system.

As early as 1931, Ives⁴ demonstrated that a lenticular screen composed of vertically aligned cylindrical lenses could be used to generate autostereoscopic 3D imagery. Originally a special photograph with alternating left and right eye stripes was mounted directly behind the screen. The multiple stripes behind each vertical cylindrical lens diverge to different horizontal views. Today, due to their relative low cost, LCDs with vertical lenticular lenses remain the most common form of autostereoscopic display. As the number of views is limited by the pixel pitch of the backing image or LCD, Ives showed that increased stripe density could be achieved by focusing 39 film projectors onto a retroreflective or diffuse screen behind the lenticular array. A similar idea was used by Matusik and Pfister⁵ who presented real-time acquisition, transmission, and display using 16 digital projectors and a vertically aligned lenticular screen. In both cases, the light is focused onto a diffuse plane behind the lenticular array then redistributed by each cylindrical lens. As with Ives, the horizontal angular variation still comes from the specific shape and focal length of the lenticular array. Even with additional projector resolution, nearly all vertically aligned lenticular displays have limited horizontal field of view as the lenticular cylinders start to self-occlude.

If the cylindrical lenticular lenses are aligned horizontally then they function as a vertical anisotropic diffuser. Instead of bending light in the horizontal axis, a horizontal cylindrical lens scatters each incoming ray into a vertical plane. This orientation allows for greater angular density as it preserves the angular variation of the original projector spacing. Unlike Ives,⁴ Matusik and Pfister,⁵ the exact focal length of the lenticular screen is not critical as long as the vertical diffusion encompasses all viewers. Based on this principle, the commercial company Holografika has demonstrated large-format projector arrays including the rear-projection HoloVizio 720RC⁶ and the front-projection HoloVizio C80.⁷ Recently, Kawakita et al.⁸ designed a massive 5 meter rear projection display. Both Holografika and Kawakita et al. displays are built using large projector units with wide spacing. Additional optics are added to their screens to refocus and redirect projector rays across a narrow horizontal field of view. Yoshida et al.⁹ have developed an array with 103 micro-LCD projectors as a glasses-free tabletop 3D display. The projectors illuminate a conical anisotropic screen coupled with a holographic diffuser situated below the table surface. The drawback of this form factor is that virtual objects on or above the table surface will always appear blurred as they are outside the depth of field of the display. A more in-depth comparison with these projector arrays can be found in Table 1.

User tracking has long been used for single-user displays with stereo $glasses^{10-12}$ and single-user autosteroscopic displays¹³ in order to update both horizontal and vertical motion parallax. Our system is the first autostereoscopic projector array to incorporate tracking of multiple users for vertical parallax. Our method has the advantage that a large number of users still perceive instantaneous horizontal parallax. Tracking latency is

	Matusik et al. ⁵	HoloVizio 720RC	Kawatika et al. ⁸	Yoshida et al. ⁹	Our system
Angular resolution	1.88°	unknown	0.24°	1.27°	1.66°
Horizontal diffusion	N/A	unknown	0.88°	$0.5-1^{\circ}$	$1-2^{\circ}$
Vertical diffusion	N/A	unknown	35°	60°	60°
Horizontal field of view	30°	40°	13.5°	130°	118°
Horizontal screen size	$1.8\mathrm{m}$	$3\mathrm{m}$	$5\mathrm{m}$	$20 \mathrm{cm}$	$30 \mathrm{cm}$
Screen shape	flat	flat	flat	cone/cylinder	flat/convex
Form-factor	front/rear	rear	rear	tabletop	front/rear
Number of projectors	16	80	57	103	72
Projector distance	unknown	$5.5\mathrm{m}$	$5.5\mathrm{m}$	80cm	$60 \mathrm{cm}$
Number of computers	8	4	unknown	6	1
Vertical parallax	no	no	no	no	yes, with tracking
Horizontal screen size Screen shape Form-factor Number of projectors Projector distance Number of computers Vertical parallax	1.8m flat front/rear 16 unknown 8 no	3m flat rear 80 5.5m 4 no	5m flat rear 57 5.5m unknown no	20cm cone/cylinder tabletop 103 80cm 6 no	30cm flat/convex front/rear 72 60cm 1 yes, with tracking

Table 1. Comparison of our system specifications with other autostereoscopic projector arrays.

less noticeable as it is restricted to the vertical axis where rapid movements are less frequent. None of these existing lenticular or projector arrays generate vertical parallax although our technique could also be implemented for these and other autosteroscopic displays.

3. APPARATUS

Our projector array system consists of 72 Texas Instruments DLP Pico projectors each of which has 480x320 resolution. At the time of publication, these projectors are the smallest commercially available projector as they do not include additional battery or processing hardware. Other LCoS or MEMS-based projector technologies could also be produced in a similar size. Even though the projectors use low-power LED light source, additional fans were added to provide active cooling. Our projectors are evenly spaced along a 124 cm circular curve with a radius of 60 cm. This setup provides an angular resolution of 1.66° between views. At the center of the circle, we place a 30 cm x 30 cm vertically anisotropic screen (see Figure 1. The ideal screen material should have a wide-vertical scattering profile so the 3D image can be seen from multiple heights, and a narrow horizontal scattering profile that maintains the divergence of different projector pixels to varying horizontal views. When a projected image passes through the vertically anisotropic screen, it forms a series of vertical stripes. Without any horizontal diffusion, the stripe width is equivalent to the width of the projector lens.

Each projector is 1.42cm wide with a 4mm lens. In order to eliminate any gap between stripes, this would require stacking several hundred projectors in overlapping vertical rows.¹⁴ We found that acceptable image quality could be achieved with a single row of projectors if we use a holographic diffuser to generate $1-2^{\circ}$ of additional horizontal diffusion and stack the projectors with a 2 mm gap (see Figure 2 and Video 1 at 1:00). The



Figure 1. (left) A 3D face is shown on our autosteroscopic 3D projector array. The display combines both autosteroscopic horizontal parallax with vertical tracking to generate different horizontal and vertical views simultaneously over a full 110° field of view. (center) Diagram showing the dimensions of the display.(right) 3D stereo photographs of a human face. The stereo pairs are left-right reversed for cross-fused stereo viewing.



Figure 2. The anisotropic screen forms a series of vertical lines, each corresponding to a projector lens. A $1-2^{\circ}$ horizontal diffuser is used to blend the lines and create a continuous image. The top rows shows stripes and imagery reflected on a flat anisotropic screen. The bottom row shows imagery reflected on a convex anistropic screen. By varying the curvature of a mirrored anisotropic screen, we can decrease the pitch between reflected projector stripes. This increases the spatial resolution at the screen depth at the expense of overall angular resolution and depth of field.

holographic diffuser generates a slight angular blur to the projector rays, smoothly filling in the gaps between the projectors with adjacent pixels. Ideally, the width of the diffusion lobe should be equal to the angle between projectors lens, though in practice we are limited to commercially available scattering profiles which typically come in 1° increments.

For rear-projection setups, we use a horizontally aligned lenticular sheet layered with an anisotropic holographic diffuser with a horizontal 2° and vertical 60° scattering profile (see Figure 3) and Video 1 at 0:14) Alternatively for front-projection, we reflect off a fine lenticular screen behind the holographic diffuser (see Figure 1 and Video 1 at 0:20). The lenticular screen is painted matte black on the reverse flat side to reduce ambient light and improve black levels. As the light passes twice through the diffuser, only 1° of additional horizontal diffusion is required. All our screen components are currently available off-the-shelf. The holographic diffusers are from Luiminit Co. The lenticular material is a 60lpi 3D60 plastic screen from Microlens Technology. We also explored other anisotropic screen materials. For example, finely brushed aluminum or steel can be highly anisotropic with similar vertical and horizontal scattering suitable for a reflective front-projection screen.² Brushed metal is widely available and inexpensive. However in an experimental comparison, we found that brushed metal generally produces a lower contrast ratio than a front-surface lenticular reflection.

The amount of horizontal holographic diffusion also influences the perceived resolution of the display. It is preferable to stack projectors as closely as possible to increase the horizontal spatial and angular resolution. The image seen be each viewer is composed of vertical stripes from multiple projectors. Smaller gaps between stripes means higher horizontal spatial resolution. As the amount horizontal diffusion increases, the gaps are filled with blurred pixel information from adjacent angular rays. This effectively increases the horizontal resolution at the expense of angular resolution.

We drive our projector array using a single standard Windows computer with no special memory or CPU. The only requirement is that the motherboard can accommodate four graphics cards. We use four ATI Eyefinity graphics cards with a total of 24 video outputs. The CPU is primarily used for user-tracking; all other operations are performed on the GPU. The animations shown in the video used around 5000 triangles and ran at 100-200 fps on an ATI Eyefinity 7800. To achieve maximum performance, rendering is distributed across all four graphics cards. As not all graphics libraries provide this low level control, we explicitly create a different application and render context per GPU, with the different instances synchronized through shared memory. The main bottleneck is Front Side Bus data transfer as textures and geometry need to be uploaded to all GPUs. We then split each video signal using 24 Matrox TripleHeadToGo video splitters, each of which supports three HDMI outputs. This allows for a maximum of 72 tiled video signals. We use DisplayFusion desktop manager software to handle the multi-monitor setup.

To track viewer positions, a Microsoft Kinect camera is mounted directly above the screen. The depth camera is ideal as it provides both viewer height and distance. The Kinect device does have some built-in latency around 100 ms. To get a more robust position, we track both the body's center of mass as facial features with Kinect API. We implement an additional Kalman filter to further smooth the positional data and estimate tracking confidence. The output of the Kinect tracker along with corresponding interpolated viewer heights can be seen in Video 2. The current Kinect API is limited to tracking 6 active users. However, our interpolation method would easily work with other 3D tracking methods.



Figure 3. (left) Photograph of our rear-mounted projector array setup. (right) Photograph of the calibration setup for front-mounted projector array. To compute relative projector alignment, we sequentially correspond a virtual AR marker generated by each projector with a printed AR marker.

4. CALIBRATION

Even with a machined projector mount, there is still noticeable geometric and radiometic variation between projectors. We automate the geometric calibration of the projectors using a 2D rectification process to align projectors images to the plane of the screen. We first place a diffuse surface in front of the screen, then sequentially project and photograph an AR marker from each projector¹⁵ (see Figure 3). We then compute a 2D homography matrix that maps the detected marker corners in each projector to a physical printed marker also attached to the screen. At the same time, we measure the average intensity of light from each projector on the diffuse surface and compute a per-projector scale factor to achieve consistent brightness and color temperature across the entire projector array. While there is some variation between projectors, the LED light source for each projector is relatively stable over time.

5. VIEWER INTERPOLATION

As described above, anisotropic screens do not preserve a one-to-one mapping between projectors and views as projector rays diverge to multiple viea wers at potentially different angles, heights, and distances. To generate an image that can be viewed with a single perspective, we must render MCOP images that combine multiple viewing positions. A brute force solution would be to pre-render out a large number of views with regular perspective and resample these images based on the rays generated by the device as done by Rademacher et al.¹⁶ A variant of this technique was implemented by Jones et al.¹ for existing photographic datasets. However, resampling introduces a loss-of-resolution.

For scenes with known geometry, an alternative approach is to replace the standard perspective projection matrix and directly render MCOP images. We define this MCOP operation so that it maps each 3D vertex point to 2D projector coordinates based on a different viewing transformation. This per-vertex geometric warp can capture smooth distortion effects and be efficiently implemented in a standard GPU vertex shader. Our method is based on a similar approach was used by Jones et al.¹ to generate MCOP renders.

The first step is to project each 3D vertex onto the screen. For each vertex (Q), we trace a ray from the current projector position (P) through the current vertex (Q). We intersect this first ray (PQ) with the screen surface to find a screen point (S_P) . The second step is to compute a viewing position associated with this vertex. The set of all possible viewers can be represented as a continuous manifold spanning multiple heights and distances. In the general case, the intersection of the ray (PQ) with this manifold defines the current corresponding viewer (V). Finally, we trace a second ray from the viewer (VQ) back to the current vertex (Q) and intersect with the screen a second time (S_V) . The actual screen position uses the horizontal position of S_P and the vertical position of S_V . This entire process can be implemented in a single GPU vertex shader. In essence, the horizontal screen position is determined by projecting a ray from the projector position, while the vertical screen position is based on casting a ray from the viewer position. The difference in behavior is due to the fact that the anisotropic screen acts as a diffuse surface in the vertical axis. We multiply the final screen position by the current calibrated projector homography to find the corresponding projector pixel position for the current vertex.

In practice it is not easy to define the manifold of viewer positions as we only know a few sparse tracked positions. We propose a closed-form method for approximating this manifold as a series of concentric rings. In 2007, Jones et al.¹ assumed that the viewing height and distance was constant for all projectors. This arrangement corresponds to a single circle of potential viewers. As the viewers are restricted to lie on this circle, the viewpoints represented by each rendered MCOP image vary only in their horizontal angle, with no variation in height and distance. One trivial extension would be to interpolate viewer height and distance once per projector frame and adjust the radius of the viewing circle. In our comparison images we refer to this method as *per-projector* interpolation. However when tracking multiple viewers close together, it is possible for the viewing height to change rapidly within the width of a single MCOP projector frame.

We solve this issue by interpolating the viewer height and distance *per-vertex*. We pass the nearest two tracked viewer positions to the vertex shader. Each viewer height and distance defines a cylinder with a different viewing radius. We then intersect the current projector-vertex ray (PQ) with both cylinders. To determine the final viewing position for this ray, we compute a weighted average of the two viewer heights and distances. The interpolation weights of each tracked viewer position is a function of the angle (θ_1, θ_2) between the cylinder intersection and the original tracked point. A top-down view of these intersections is shown on the left side of Figure 4. When computing the weighted average, we also add a third default value with very low weight. This value corresponds to the average user height and distance appropriate for untracked viewers. As the viewer's angle from each tracked point increases, the influence of the point decays and the viewer returns to the default height and distance (right side of Figure 4). We implemented two different falloff functions centered around each track point - a normalized gaussian and a constant step hat function. The gaussian function smoothly decays as you move away from a tracked viewer position, while the hat function has a sharp cutoff. A comparison of the two interpolation functions is included in the results section. The weight function can be further modulated by the confidence of the given tracked viewer position. This decay makes the system smoothly adjust to viewers missed by the tracking system or new viewers that suddenly enter the viewing volume. Pseudocode for computing per-vertex projection and viewer interpolation can be found in Table 2. The final rendered frames appear warped as different parts of each frame smoothly blend between multiple horizontal and vertical perspectives. Figure 5 shows a subset of these MCOP frames before they are are sent to the projector.

A related MCOP rendering algorithm was also proposed by Jones et al. in 2009.² In this later work, the entire per-vertex projection operator from 3D vertices to 2D projector positions was precomputed as a 6D lookup table based on 3D vertex position, mirror angle, and viewer height and distance. The lookup table was designed to handle more complex conical anisotropic reflections that occur when projector rays are no longer perpendicular to the mirror's vertical anisotropic axis. For projector arrays, we found that rays scattered by the screen remained mostly planar with very little conical curvature. Furthermore, the lookup table's height and distance was indexed based on a single reflection angle per projector frame. This assumption is analogous to our *per-projector* interpolation examples. Instead of modeling different heights and distances for each projector, Jones et al. used a concave anisotropic mirror to optically refocus the projector rays back towards a single viewer. Such an approach would not work for a rear-mounted projector array where there is no mirror element. Our software solution is more general and allows for a wider range of screen shapes.

```
// project 3D vertex positions to 2D screen coordinates using MCOP warp
void warpVertex(
 float3 Q : POSITION,
                               // input 3D vertex position
 uniform float3x3 Homography // 2D projector homography
 uniform float4 P,
                               // current projector position
 uniform float3 T[],
                               // tracked viewers positions
 uniform float C[],
                               // tracked viewer confidences
 out float3 oQ : POSITION) // output 2D screen coordinate
Ł
 P = computeProjectorPostion(P);
                                       // use reflected projector position if front-projection screen
 V = interpolateViewer(Q, P, V, T, C); // find viewer that sees current vertex from current projector
 float3 VQ = V - Q;
                                       // define ray from viewer position V to vertex Q
 float3 S = intersectRayWithScreen(VQ) // intersect with planar or cylindrical curved screen
 oQ = mul( Homography, S );
                                       // apply projector homography to screen coordinate
}
// interpolate tracked viewer positions per-vertex
float3 interpolateViewer(float3 Q, float3 P, float3 V, float3[] T, float[] C) {
 float sum_of_weights;
 float current_viewer;
 float3 PQ = Q - P;
                                       // define ray from reflected projector position P to vertex Q
 for each tracked viewer t
   float3 I = intersectRayWithCylinder(PQ, radius(t)); // radius of cylinder is tracked viewer distance
   float angle = computeAngle(T, I);
                                                       // compute angle between intersection and tracked viewer
   float weight = falloffFunction(angle);
                                                       // falloff function could be Gaussian or Hat function
   weight *= C[t];
                                                       // also weight by tracking confidence
   current_viewer += t * weight;
   sumWeight += weight;
 }
 current_viewer += default_weight * default_viewer;
                                                       // add in default viewer
 sumWeight += default_weight;
                                                       // with low default weight
                                                        // compute weighted average of tracked viewer positions
 return currentViewer / sumWeight;
}
```

Table 2. Vertex shader pseudocode that projects a 3D scene vertex into 2D screen coordinates as described in Sections 5. The code interpolates between multiple tracked viewer positions per vertex. It assumes helper functions are defined for basic geometric intersection operations.



Figure 4. (left) Diagram showing how we compute the corresponding viewing position for each vertex by tracing a ray from the projector position P through the vertex Q. We intersect the ray with the screen to find the horizontal screen position S. We intersect the ray with viewing circles defined by the nearest two tracked viewers (V_1, V_2) . We interpolate between the tracked viewer positions based on their angular distance from the intersection points. (right) Diagram showing the continuous curve formed by the interpolated viewer positions. The curve returns to a default viewer height and distance away from tracked viewer positions (V_1, V_2) .



Flat screen

Convex screen

Figure 5. Our algorithm renders out a different MCOP image for each of 72 projectors. This is a sampling of the generated images using per-vertex vertical parallax blending with a gaussian falloff function. Each image smoothly blends between multiple horizontal and vertical viewer positions which gives the appearance of an unwrapped object. Flat front and rear projection screens produce almost identical imagery. Convex screens have additional warping as each projector spans a wider set of viewers.

6. CONVEX SCREEN PROJECTION

For a front-mounted array, the pitch between reflected stripes can also be reduced by using a convex reflective anisotropic screen. The convex shape magnifies the reflected projector positions so they are closer to the screen, with narrower spacing and a wider field of view (Figure 2). As less horizontal diffusion is required to blend between stripes, objects at the depth of the screen have greater spatial resolution. This improved spatial resolution comes at the cost of angular resolution and lower spatial resolution further from the screen. Zwicker et al.¹⁷ provide a framework for defining the depth of field and display bandwidth of anisotropic displays. For a given initial spatial and angular resolution, the effective spatial resolution is halved every time the distance from the screen doubles. In Figure 6 we plot the tradeoff between spatial resolution and depth of field given the initial specifications of our projector prototype. A curved mirror is also preferable for 360° applications where an anisotropic cylinder can be used to reflect in all directions without any visible seams.

As a convex mirror effectively increases each projector's field of view, each projector frame must represent a wider range of viewer positions. It becomes more critical to compute multiple viewer heights per projector frame. To directly render MCOP frames we use the flat screen projection algorithm from the previous section with two minor modifications. For front-mounted projection setups that use a mirrored anisotropic screen, we first unfold the optical path to find reflected projector positions. The rays reflected off a convex mirror do not always reconverge at a single reflected projector position (Figure 7 (right)). A first order approximation is to sample multiple reflected rays and average the resulting intersection points. (Figure 7 (center)) This can still result in distortion for extreme viewing angles. A more accurate solution is to compute a different projector position per-vertex. In the per-vertex variant, we use the average reflected projector to compute an initial intersection point with the screen. We then interpolate a second more accurate reflected projector position based on the local curvature around this screen point (Figure 7 (right)). This process could be iterated to further refine the reflected projector position, though in our tests the reflected rays converged in a single iteration. Secondly, we discard rays that reflect off the convex mirror near grazing angles as these regions are extremely distorted and are very sensitive to small errors. In comparison to a flat screen, a convex screen requires rendered frames covering a wider variation of views and greater per-vertex warping (Figure 5).

7. RESULTS

Our display can generate stereo and motion parallax over a wide field of view. While it is difficult to communicate the full 3D effect with 2D figures, Figure 1 shows several stereo photographs printed for cross-fuse stereo viewing. The motion parallax can also be seen throughout the accompanying video.



Figure 6. Graph showing tradeoff between spatial magnification using convex mirrored screens and depth of field. Greater mirror curvature increases density of projector stripes and spatial resolution at the screen, however spatial resolution falls off rapidly away from the screen.



Figure 7. (left) For a convex mirrored screen, reflected projector rays no longer converge on a single projector position. (center) We sample multiple points on the mirror surface and compute an average reflected projector position for each real projector. (right) We then iteratively refine by reflected position per-vertex by tracing a ray from the average position through the vertex to the convex screen. We then compute a more accurate reflected position based on the local neighborhood of the screen point.

To evaluate our view interpolation algorithms. we render multiple geometric models from a variety of heights and distances (Figure 8). We use a checkered cube to identify any changes in perspective or warping and a spherical model of the Earth to illustrate correct aspect ratio. We also show two high-resolution face models as examples of organic shapes. We then photograph the display from three views: a lower left view, a center view, and a high right view. We measure the real camera positions and render matching virtual views to serve as ground-truth validation (Figure 8, row 1). With no vertical tracking, the display only provides horizontal parallax and all viewers will see a foreshortened image as they move above or below the default height (row 2). Note that the top and bottom faces of the cube are never visible and the eye gaze for the face is inconsistent. If we enable tracking for the left and right cameras (rows 3 through 6), then it is possible to look up and down at the objects.

Also in Figure 8, we compare our new interpolation algorithm for handling multiple different viewers. For rows 2 and 3, we compute a per-vertex viewer height and distance as described in Section 5. Per-vertex vertical parallax interpolation produces plausible and consistent perspective across the entire photographed view. In contrast, rows 4 and 5 demonstrate interpolation that uses a constant viewer height and distance per projector. Each projector still interpolates the nearest two tracked viewers positions, however the interpolation weights are uniform across all vertices. Per-projector interpolation generates significant distortion for all three views where the vertical perspective on the left side of the image does not match the perspective on the right side. These errors also depend on the shape of the weight falloff function. Using per-projector gaussian weights (row 5) makes straight lines curved while a per-projector step hat function (row 6) causes image tearing as the view abruptly changes from one vertical height to another. The distortion is less visible on organic objects such as a face, although the left and right eyes are no longer looking in the same direction. Additional results at the start of Video 3 show how this distortion ripples across the geometry as the camera moves back and forth between two tracked projector positions.

Another advantage of per-vertex interpolation is that it reduces errors for untracked viewers. Untracked viewers see the 3D scene based on a default height and distance that should not be affected by the vertical movement of nearby tracked viewers. Despite the fact that each projector frame may be seen by multiple viewers, by computing multiple vertical perspectives within each projector frame (per-vertex) we can isolate each viewer (Figure 8, rows 5 and 6). In contrast, the center view of the face appears distorted when interpolating a single height per-projector as this untracked view shares some projectors with the nearby lower left camera (rows 3 and 4). The same effect is shown for a dynamic user in the accompanying video. Without per-vertex interpolation (Video 3, time 0:48), the untracked viewer is clearly distorted whenever the tracked viewer is nearby. At time

0:34 in Video 3, this extraneous crosstalk is considerably reduced with per-vertex interpolation.

There is no limit to how many untracked people can view the display simultaneously as it generates dense horizontal autostereoscopic imagery. However, when viewers start to occlude each other, we can not completely isolate each viewer's vertical perspective. If two tracked viewers completely overlap or stand right above each other, then their average height and distance is used. An alternative solution to resolving conflicts would be to give higher confidence to nearer viewers so their heights have precedence. For all the results shown in the paper, the width of the gaussian and hat interpolation functions was 10 degrees which approximates the width of a viewer's shoulders. A wider interpolation falloff provides a horizontal buffer with similar vertical perspective. This is particularly useful if the tracking algorithm has high latency and can not keep up with sudden horizontal head movements. In contrast, a narrower interpolation falloff further reduces interference when a tracked user is looking over another tracked user's shoulder. As the number of users increases, the tracking system also produces lower confidence values. The current Kinect device is limited to tracking up to 6 active viewers. In the worst case, the system reverts back to a standard autostereoscopic display with only horizontal parallax.

We test our projection algorithm on a curved mirror with a 10 degree curvature and a magnification factor of 1.43. Figure 9 shows imagery on the curved mirror with and without per-vertex vertical parallax interpolation. In the later case, perspective distortion increases significantly. This distortion could be reduced by using a wider gaussian function so that nearby frames would be forced to have similar heights, but this would have the negative effect of more crosstalk with nearby viewers. To validate our projection model for a wider range of screen shapes, we implemented a projector array simulator that can model arbitrary screen curvature and diffusion materials. The simulator uses the same render engine but projects onto a screen with a simulated anisotropic BRDF. As shown in the accompanying video (Video 4), we can maintain a stable image with correct perspective as a mirror shape changes significantly. We can also determine the ideal horizontal diffusion width by simulating different anisotropic reflectance lobes.

8. FUTURE WORK

In our work, we treat the tracking algorithm as a black box. Future advances in facial tracking will continue to reduce latency and be able to track more viewers over a larger view volume. Knowledge of viewer positions



Figure 8. This figure shows three virtual objects viewed by an untracked center camera and two tracked left and right cameras. As a ground-truth comparison, we calibrate the positions of three cameras and render out equivalent virtual views of the objects (1st row), while the remaining rows show photographs of the actual display prototype. If a single constant viewer height and distance is used then the viewer sees significant foreshortening from high and low views (2nd row). We also compare different viewer interpolation functions for interpolating multiple viewer heights and distances. The tracked view positions are interpolated either with a constant height/distance per-projector (3rd and 4th rows) or with different height/distance per-vertex (5th and 6th rows), or. Photographs taken with per-vertex interpolation show less distortion with consistent vertical perspective across the entire image, and the untracked center view is not affected by the nearby left viewer. Photographs with per-projector interpolation exhibit multiple incorrect perspectives with warped lines and image tearing, and the untracked center view is distorted by the nearby left viewer. The local weight falloff of each tracked position is implemented as either a normalized gaussian (3rd and 5th rows) or sharp step hat function (4th and 6th rows). Gaussian interpolation errors appear as incorrect curved lines while errors using a sharp hat falloff result in a image tearing.

has other applications beyond correcting for vertical parallax. For example, viewing positions can be used to optimize various forms of computational displays or take advantage of stereo perceptual metrics. We are looking to find ways to accelerate the rendering of large numbers of views - for example large parts of the rendering pipeline do not need to repeated for every view. Even on modern hardware, rendering 70+ simultaneous views limits scene complexity. An alternative solution would be to distribute rendering across multiple devices. In the



Figure 9. Comparison of different viewer interpolation functions for a convex mirror. The left set uses per-vertex viewer height and distance with a gaussian falloff. The right set uses constant height and distance per-projector with a gaussian falloff. Photographs taken with per-vertex interpolation show less distortion with consistent vertical perspective. In contrast, straight lines appear curved in photographs using constant per-projector interpolation.

extreme case, each projector could have a dedicated low-cost single-board computer such as Raspberry Pi. This could decrease costs, but makes it more difficult to synchronize frames and share resources.

Finally we plan to adapt our tracking interpolation to other autosteroscopic displays including larger-scale projector arrays. As our display system contains mo moving parts, there is no inherent limit to the size of the display. The lenticular and holographic diffuser materials are available in large sheets or rolls. As the display size increases, the projectors will need to be brighter with higher pixel resolution to maintain image quality. It is also possible to scale the field of view. For example, if additional projectors were mounted in a full circle around a cylindrical anisotropic screen, then the display could provide a complete 360° viewing experience.

9. CONCLUSION

In this work, we have shown a new hybrid approach that can display 3D objects with correct horizontal and compelling perspective from any view. Our new 3D projector array generates autostereoscopic horizontal parallax with multi-user tracking for vertical parallax. Our display produces full color, no horizontal latency, and a wide field of view that can accommodate a large number of viewers. Our system is reproducible with off-the-shelf projectors, screen materials, graphics cards, and video splitters. Unlike previous 3D display techniques, we directly render MCOP images with varying horizontal and vertical parallax for every projector. Even though each user sees slices of multiple projectors, the perceived 3D image is consistent and smooth from any vantage point. Without this method, users experience significant cross-talk and geometric distortion particularly when multiple viewers are in close proximity. This rendering framework frees us to explore different projector configurations including front and rear-mounted projector arrays and and non-flat screens. We envisage that this will enable a range of new multi-user applications from live teleconferencing to immersive virtual scenes.

ACKNOWLEDGMENTS

We thank the following people for their support and assistance: Bill Swartout, Randall Hill, and Randolph Hall. This work was sponsored by the University of Southern California Office of the Provost, U.S. Air Force DURIP, and U.S. Army Research, Development, and Engineering Command (RDECOM). The content of the information does not necessarily reflect the position or the policy of the US Government, and no official endorsement should be inferred.

REFERENCES

- Jones, A., McDowall, I., Yamada, H., Bolas, M., and Debevec, P., "Rendering for an interactive 360 light field display," ACM Transactions on Graphics (TOG) 26(3), 40 (2007).
- [2] Jones, A., Lang, M., Fyffe, G., Yu, X., Busch, J., McDowall, I., Bolas, M., and Debevec, P., "Achieving eye contact in a one-to-many 3d video teleconferencing system," in [ACM Tr ansactions on Graphics (TOG)], 28(3), 64, ACM (2009).
- [3] Urey, H., Chellappan, K., Erden, E., and Surman, P., "State of the art in stereoscopic and autostereoscopic displays," *Proceedings of the IEEE* 99(4), 540–555 (2011).
- [4] Ives, H., "The projection of parallax panoramagrams," Journal of the Optical Society of America 21(7), 397–409 (1931).
- [5] Matusik, W. and Pfister, H., "3d tv: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes," in [ACM Transactions on Graphics (TOG)], 23(3), 814–824, ACM (2004).
- [6] Rodriguez, T., de Leon, A. C., Uzzan, B., Livet, N., Boyer, E., Geffray, F., Balogh, T., Megyesi, Z., and Barsi, A., "Holographic and action capture techniques," in [ACM SIGGRAPH 2007 emerging technologies], SIGGRAPH '07, ACM, New York, NY, USA (2007).
- [7] Kovacs, P. T. and Zilly, F., "3d capturing using multi-camera rigs, real-time depth estimation and depthbased content creation for multi-view and light-field auto-stereoscopic displays," in [ACM SIGGRAPH 2012 Emerging Technologies], SIGGRAPH '12, 1:1–1:1, ACM, New York, NY, USA (2012).

- [8] Kawakita, M., Iwasawa, S., Sakai, M., Haino, Y., Sato, M., and Inoue, N., "3d image quality of 200-inch glasses-free 3d display system," SPIE Stereoscopic Displays and Applications XXIII 8288 (2012).
- [9] Yoshida, S., Kawakita, M., and Ando, H., "Light-field generation by several screen types for glasses-free tabletop 3d display," in [3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON), 2011], 1–4, IEEE (2011).
- [10] Cruz-Neira, C., Sandin, D. J., and DeFanti, T. A., "Surround-screen projection-based virtual reality: the design and implementation of the cave," in [*Proceedings of the 20th annual conference on Computer graphics* and interactive techniques], SIGGRAPH '93, 135–142, ACM, New York, NY, USA (1993).
- [11] Krger, W., Bohn, C.-A., Frhlich, B., Schth, H., Strauss, W., and Wesche, G., "The responsive workbench," *IEEE Computer Graphics and Applications* 14, 12–15 (1994).
- [12] Ware, C., Arthur, K., and Booth, K. S., "Fish tank virtual reality," in [Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems], CHI '93, 37–42, ACM, New York, NY, USA (1993).
- [13] Perlin, K., Paxia, S., and Kollin, J. S., "An autostereoscopic display," in [Proceedings of the 27th annual conference on Computer graphics and interactive techniques], SIGGRAPH '00, 319–326, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (2000).
- [14] Jurik, J., Jones, A., Bolas, M., and Debevec, P., "Prototyping a light field display involving direct observation of a video projector array," in [Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on], 15–20, IEEE (2011).
- [15] Woods, E., Mason, P., and Billinghurst, M., "Magicmouse: an inexpensive 6-degree-of-freedom mouse," in [GRAPHITE 2003], 285–286 (2003).
- [16] Rademacher, P. and Bishop, G., "Multiple-center-of-projection images," in [Proceedings of the 25th annual conference on Computer graphics and interactive techniques], SIGGRAPH '98, 199–206, ACM, New York, NY, USA (1998).
- [17] Zwicker, M., Matusik, W., Durand, F., Pfister, H., and Forlines, C., "Antialiasing for automultiscopic 3d displays," in [ACM SIGGRAPH 2006 Sketches], 107, ACM (2006).