

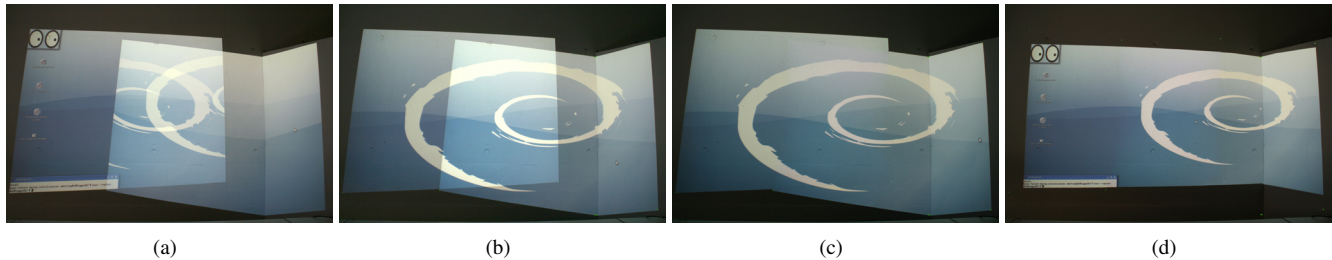
# Multi Projector Displays Using a 3D Compositing Window Manager

Manuela Waldner\*

Christian Pirchheim†

Dieter Schmalstieg‡

Institute for Computer Graphics and Vision  
Graz University of Technology



**Figure 1:** Calibration steps: (a) The unmodified desktop. (b) Warping is applied and the display is rectified as smallest circumscribed rectangle of the projection area. (c) Alpha blending provides approximate uniform image brightness across projection seams. (d) Manual adjustment of the display image.

## Abstract

The usage of projected displays in everyday office environments is still uncommon due to deficiencies in nowadays projection systems. An automatic calibration routine for multi-projector setups should adopt to the existing room geometry while not restricting the users' workflow to specific applications or by modifying their desktop environment.

We present a geometry-adaptive calibration and rendering application for multi-planar surfaces which applies warping and alpha blending to an unmodified X desktop, implemented as plugin for an existing 3D hardware-accelerated compositing window manager. The transformations are performed transparently to X applications and require negligible computational resources.

**CR Categories:** B.4.2 [Input/Output and Data Communications]: Input/Output Devices—Image display I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms H.5.2 [Information Interfaces and Presentation]: User Interfaces—Windowing systems

**Keywords:** multi-projectors, compositing window manager, seamless display, calibration

## 1 Introduction

For team workers, a public projected display wall in the team's office room may provide a convenient walk-up information space for ad-hoc discussions and presentations. Especially in small office environments, the limited wall space often restricts the usage of pro-

jected displays resulting in oblique projection angles or discontinuous overlap regions between adjacent projection areas.

There are numerous solutions to calibrate projector-based displays to compensate for keystone effects and to provide uniform tiled displays automatically. Usually, one or more cameras are employed to detect the display surface using structured light patterns. Based on the camera images, geometric, sometimes photometric, warping is calculated.

A common technique to describe geometric warping for planar tiled displays is to define camera-projector homographies, either for one camera [Raskar et al. 2002; Yang et al. 2001] or for many [Chen et al. 2001]. Overlapping projection areas have to be blended in order to create uniform image brightness. By obtaining a three-dimensional model of the display surface, images can be projected on much more complex geometries without visual distortion [Raskar et al. 1999; Bimber et al. 2005]. Depending on the complexity of the surface, a fairly dense surface mesh is required. Common indoor geometries usually consist of piecewise planar surfaces, such as walls and tables. With a multi-planar representation of the surface, corners can be modeled precisely while dramatically decreasing the density of the polygon mesh. A projection system for multi-planar geometries can be achieved by fitting planes into a point cloud reconstruction of the surface [Quirk et al. 2006] or by describing plane-wise homographies [Ashdown et al. 2004].

If a 3D surface model is available, distortion is usually compensated from a single tracked user's or camera's point of view [Raskar et al. 1999; Bimber et al. 2005; Quirk et al. 2006]. However, this approach is not feasible for multi-user setups. Raskar et al. [2003] proposed the employment of least squares conformal maps (LSCM) texture mapping technique [Lévy et al. 2002] to achieve view-independent projection on arbitrary surfaces. This technique provides a distortion-free parameterization of developable surfaces with no angle-deformation and minimal stretch. For non-developable surfaces, such as a projection spanning over two perpendicular walls and the floor, the algorithm aims at a minimal distortion. Raskar et al. texture-map the single projector's surface model using the obtained LSCM texture coordinates with arbitrary image content and render the model from the projector's point of view. Displays thus appear like a wallpaper attached to the surface and are therefore particularly suitable for multi-user setups.

\*e-mail: waldner@icg.tugraz.at

†e-mail: pirchheim@icg.tugraz.at

‡e-mail: schmalstieg@icg.tugraz.at

Projector systems require the desired output image to be warped and blended before being rendered. Most of the examples mentioned above use proprietary software and apply required image transformations directly onto a rendered 3D scene, by using designated pixel shaders, or by employing a two-pass rendering approach. However, if the output image is not generated by the warping application itself, i.e. the desktop image, the usual rendering pipeline has to be intercepted at some point.

Tools like VNC [Richardson et al. 1998] and Microsoft RDP acquire bitmap desktop images stored in an off-screen buffer. The primary purpose of these applications is to send compressed desktop pixels from a server machine to a connected remote computer. Thus, they are commonly used for tiled display setups in combination with PC clusters [Yang et al. 2001; Cotting et al. 2005]. Image data must be transferred from graphics to main memory, where warping and blending operations are applied by the render application. This expensive operation has to be executed whenever the display is modified and a desktop repaint is necessary. Figure 2 (a) shows the required processing workflow.

DeskAlign [Wallace et al. 2004] uses a dual graphics pipeline infrastructure to render the unmodified desktop on the first pipeline and then passes the pixels to the texture memory of the second pipeline where the image is warped. Thus, for running a two-projector display, a quad graphics card is required.

To improve the usage of projected displays in office space, we developed an application-transparent and geometry-adaptive projection system for displaying unmodified X Window System desktop content on multi-planar surfaces using a single workstation.

We make use of Linux 3D compositing window managers to apply the necessary warping and blending functions transparently to the applications. 3D window compositing allows us to transform the desktop using hardware-accelerated graphics in a two-pass rendering technique.

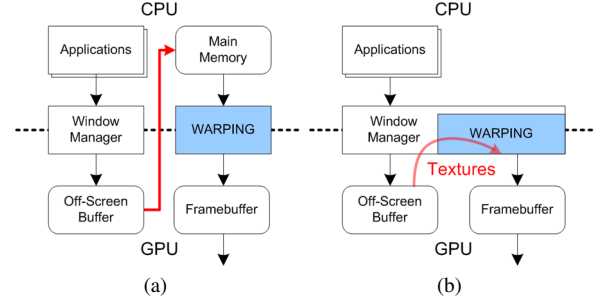
## 2 System Overview

3D accelerated compositing window managers are available on most common platforms, such as Quartz Extreme on Mac OS X, Microsoft Windows Vista’s Aero, and OpenGL-based implementations for Linux OS. They provide compositing effects like window blending or rendering of the desktop onto an interactive 3D cube. The well-known Linux implementations Beryl and Compiz provide a plugin architecture to easily apply new windowing effects. We developed several plugins for the Beryl window manager to enable necessary desktop transformations for multi-projection systems.

The *Composite* extension of the X window system allows to render application windows into an off-screen buffer on the graphics card. The window manager can access these pixmaps as textures using the OpenGL extension `EXT_texture_from_pixmap` – either directly via the NVIDIA driver or indirectly via the Xgl implementation. Thus, 3D compositing window managers make effective use of available graphics hardware and provide efficient ways to manipulate desktop and window geometries, as well as their textures.

In particular, the Beryl window manager enables us to access the final desktop composition in our plugin and to render the desktop image into a frame buffer object. The buffer content is then texture-mapped onto custom geometry and modified according to the calibration results. Figure 2 (b) illustrates the processing steps of our approach.

Our system is divided into two main parts. The offline calibration procedure is controlled by a central master application, while local calibration routines generate and detect structured light patterns.



**Figure 2:** Simplified processing workflow for desktop warping techniques in the single machine case: (a) A dedicated warping application accesses the off-screen buffer content and copies the bitmap content to main memory. (b) The window manager plugin can directly use the off-screen buffer content as texture.

The master application then distributes the calibration results to the local Beryl window manager plugins for online warping and blending. Although our calibration application is designed as distributed system, we will only consider the single machine case in this paper.

## 3 Calibration

A calibration step is required to compensate for distortions caused by oblique projection angles and to provide a uniformly illuminated, rectified display area for overlapping projections. As our approach is designed for multi-user setups in indoor environments, we generate viewpoint-independent geometric warping using the LSCM texture mapping technique for multi-planar surfaces. Plane fitting allows us to reduce the number of vertices to be rendered by the window manager plugin, while corners are properly modeled. For each resulting screen plane a homography warping function is defined. Although a three-dimensional surface model is not necessarily required to describe piecewise planar surfaces [Ashdown et al. 2004], the obtained display reconstruction enables us to describe relationships between spatially discontinuous displays for future work. Edge blending is applied when projection areas overlap. Our calibration system does not compensate for color discontinuities between different projectors yet.

**Point cloud acquisition.** Structured light patterns are generated and detected by two or more cameras with known intrinsic parameters. Detected pattern points deliver point correspondences between camera images. Two cameras with the most common point correspondences represent the base stereo camera pair. Their relative camera poses are estimated from the recovered epipolar geometry. A three-dimensional point cloud reconstruction is obtained by triangulation of point matches. In an iterative procedure, the absolute poses of the remaining cameras are estimated and the reconstruction is updated.

**Polygonal model creation.** Based on the point cloud, a RANSAC-based plane fitting algorithm is applied to approximate the planar display areas, as proposed by Quirk et al. [2006]. A simple homography thus relates screen coordinates to world plane coordinates. A three-dimensional polygonal model for each projector is created from the infinite fitted planes by projecting the actual screen corners onto each world plane and cutting adjacent planes along the screen edges. Based on the screens’ polygons, overlapping screen portions are detected and combined to displays.

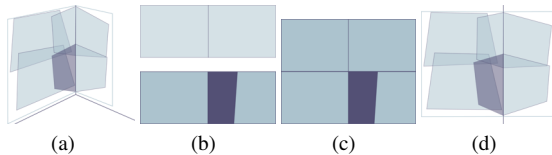
**Surface parameterization.** The simplified display model is then parameterized using the LSCM texture mapping technique. To

retrieve the LSCM texture coordinates, a display triangle mesh, which describes the relationship between overlapping projectors, is required. Due to the multi-projector setup, a 2D triangulation in screen space is not feasible. We therefore label screen polygon vertices according to the world plane on which they are located, irrespective of the projector they belong to. A 2D Delaunay triangulation is applied for each plane separately. The display mesh is combined in 3D world space and is subject to the parameterization routine.

**Display rectification.** The LSCM texture space can be imagined as perfect flattening of the surface and is thus our “ideal” space, where we define the rectification of the display. The parameterized vertices of all screen polygons are circumscribed with a rectangle of the display’s aspect ratio. By circumscribing the display rectangle instead of inscribing the rectangle into the polygons’ boundaries we avoid a loss of resolution and interpolation artifacts caused by the necessary minification of the displayed images to a smaller inscribed rectangle. Instead, our computed image extends the physical projection space and leads to a loss of information at the boundaries of the display (cf. figure 1 (c)). A window manager function allows the user to adjust the display’s size, rotation and position within the projection at runtime to best fit her needs (cf. figure 1 (d)).

## 4 Rendering

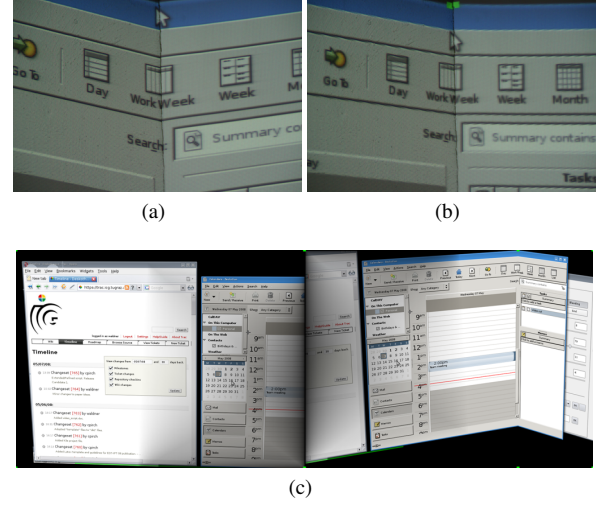
Similar to Raskar et al. [2002], we define homography functions mapping screen pixels  $\mathbf{x}$  from  $X$  display space to undistorted screen space  $\mathbf{x}'$ . Screen pixels are given as homogeneous coordinates in  $\mathbb{R}^2$  in the form  $\mathbf{x} = (x, y, 1)^T$ . In order to deal with the multi-planarity, we apply homography functions not only per projector  $i$ , but for each screen polygon  $j$ , where each projector has at least one polygon. In addition, we introduced the concept of a virtual display, defined by the physical screen topology, which is not necessarily co-incident with the involved  $X$  displays (cf. figure 3 (c)). The first function  $H_{id}$  thus maps  $X$  display coordinates from a projector  $i = 1, \dots, N$  to virtual display coordinates describing the overall resulting display (cf. figure 3 (b) and (c)). Virtual display coordinates are mapped to the rectified display in texture space by the function  $H_{du}$  (cf. figure 3 (c) and (d)). Finally, for each screen polygon  $j = i, \dots, M$ , where  $M \geq N$ , the function  $H_{uj}$  maps polygons from texture space to  $X$  display space (cf. figure 3 (d) and (b)). Thus, for each screen polygon  $j$ , the following warping function is applied:  $H_j = H_{uj}H_{du}H_{id}$ , so  $\mathbf{x}'_j \sim H_j\mathbf{x}_j$ .



**Figure 3:** (a) A tiled display in a corner is driven by (b) two  $X$  displays. (c) The virtual display topology is retrieved from the physical relationship of projected images. (d) In 2D parameterization space, the display rectification is obtained (circumscribed rectangle). It is clearly visible in (a) and (d) that the virtual display extends the physical projection area. The single steps are illustrated for a high-lighted screen polygon.

To preserve approximate uniform image brightness across single projectors and overlap regions, an alpha blending function is calculated as proposed by Raskar et al. [2002] and saved in the alpha channel of a texture. For this purpose, the relationship between adjacent projection screens on common virtual displays is required.

The function mapping screen pixels from one screen space to the other via texture space is dependent of the polygons  $j$  and  $k$  the pixels are associated with:  $\mathbf{x}'_k \sim (H_{uk}H_{uj}^{-1})\mathbf{x}_j$ .



**Figure 4:** (a) A corner with unmodified projected data: mind the different scales caused by the projection angle. (b) Plane-wise warping provides approximate uniform pixel size across corners and compensates for angle deformations. Warping causes interpolation artifacts which are especially disturbing when small text is displayed. (c) A screenshot of what is actually rendered by the 3D compositing window manager: the individual screen polygon corners are marked as green dots, the overlap region between the two projectors is blended.

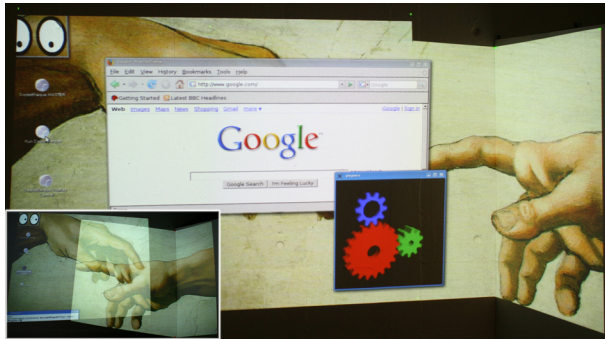
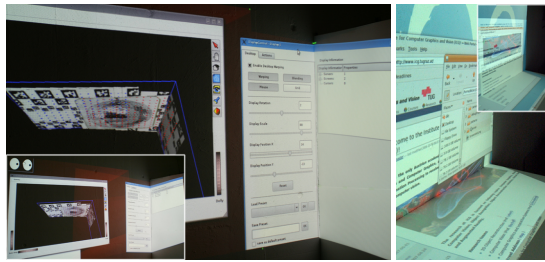
Our plugins for the Beryl window manager perform display warping, blending, and user-defined display transformations, as well as rendering of structured light. A network plugin serves as communication interface for data exchange with the projection system’s master application.

Our projection plugin is responsible for warping and alpha blending in a two-pass rendering technique. The projection matrix is set to an ortho matrix co-incident with  $X$  display coordinates. The plugin receives the  $X$  display coordinates of each projector’s polygons, as well as the associated warping functions. The unmodified  $X$  desktop is first rendered into a frame buffer object. The polygons are individually texture-mapped by the off-screen buffer content and warped with the corresponding homography matrices. For each screen, the alpha blending texture is rendered fullscreen on top of the warped polygons using an appropriate OpenGL blending function. Figure 1 shows the rendering of different calibration states. Figure 4 (c) shows a screenshot of a warped and blended two-projector  $X$  display.

As the mouse pointer is directly rendered in hardware by default, the cursor remains unaffected by the desktop warping. This leads to an offset between rendered windows and cursor interaction space. Our mouse pointer plugin renders the cursor representation, which is then added to the frame buffer object texture, while the hardware cursor is hidden.

A GUI front-end allows the user to toggle warping and blending, as well as to apply custom display transformations to place the display rectangle within the available projection area.





**Figure 5:** Different setups, each driven by two projectors. The small sub-images show the unwarped configurations.

## 5 Results

We built several prototype setups, all including at least one corner. Our hardware configuration consisted of a workstation running Linux Ubuntu Feisty Fawn (7.04) with an NVIDIA GeForce 8800 graphics card, one or two LCD projectors with XGA resolution, and two or three XGA gigabit ethernet grayscale cameras. The cameras are intrinsically calibrated and the camera image is filtered for radial and tangential lens distortion. We do not consider radial distortions caused by the projector lenses so far. The overall calibration procedure usually takes no longer than one minute. During run-time the framerate is hardly affected by the warping operations and far above the physical hardware limits. With our plugins enabled, Beryl's CPU and memory consumption increase is negligible.

Figure 5 shows different setups running custom applications, including our GUI configuration tool and the reconstruction visualization of the display surface. The modeling of corners is illustrated in close-up figure 4 (a) and (b).

## 6 Conclusion

We presented an application-transparent approach for bringing projector-camera systems one step further to people's everyday working environment. Our calibration routine adopts to most common indoor environments while the integration into an existing 3D compositing window manager allows the user to run her preferred applications on an unmodified desktop environment.

In the future, we envision a system that is not restricted to a single machine setup. To scale up the tiled display, X displays have to be stitched together. Cursor re-routing between involved machines, as well as window duplication and migration is required in order to create seamless display interaction. Our distributed calibration framework and the retrieved 3D model of display surfaces allow us to virtually combine existing displays, such as single and tiled projection walls, as well as private monitors, to a large distributed desktop with seamless interaction for multiple users.

## Acknowledgements

This project was funded in part by the Austrian Science Fund FWF under contracts Y193, and W1209-N15, and FIT-IT 813398. We would like to thank Ralph Wozelka for his parameterization interface, Werner Trobin for computer vision support, and Albert Walzer for video production.

## References

- ASHDOWN, M., FLAGG, M., SUKTHANKAR, R., AND REHG, J. 2004. A Flexible Projector-Camera System for Multi-Planar Displays. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 165–172.
- BIMBER, O., WETZSTEIN, G., EMMERLING, A., AND NITSCHKE, C. 2005. Enabling View-Dependent Stereoscopic Projection in Real Environments. In *Proceedings of the 4th IEEE/ACM International Symposium on Mixed and Augmented Reality*, 14–23.
- CHEN, H., SUKTHANKAR, R., WALLACE, G., AND CHAM, T. 2001. Calibrating scalable multi-projector displays using camera homography trees. In *Proceedings of Computer Vision and Pattern Recognition*.
- COTTING, D., FUCHS, H., ZIEGLER, R., AND GROSS, M. H. 2005. Adaptive instant displays: Continuously calibrated projections using per-pixel light control. *Comput. Graph. Forum* 24, 3, 705–714.
- LÉVY, B., PETITJEAN, S., RAY, N., AND MAILLOT, J. 2002. Least squares conformal maps for automatic texture atlas generation. In *Proceedings of ACM SIGGRAPH 2002*, ACM Press, 362–371.
- QUIRK, P., JOHNSON, T., SKARBEZ, R., TOWLES, H., GYARFAS, F., AND FUCHS, H. 2006. RANSAC-Assisted Display Model Reconstruction for Projective Display. In *Proceedings of the IEEE Virtual Reality Conference*.
- RASKAR, R., BROWN, M. S., YANG, R., CHEN, W.-C., WELCH, G., TOWLES, H., SEALES, B., AND FUCHS, H. 1999. Multi-Projector Displays Using Camera-Based Registration. In *Proceedings of the 10th IEEE Visualization Conference*.
- RASKAR, R., VAN BAAR, J., AND CHAI, J. 2002. A Low-Cost Projector Mosaic with Fast Registration. In *Proceedings of Asian Conference on Computer Vision*.
- RASKAR, R., VAN BAAR, J., BEARDSLEY, P., WILLWACHER, T., RAO, S., AND FORLINES, C. 2003. iLamps: geometrically aware and self-configuring projectors. *ACM Transactions on Graphics* 22, 3, 809–818.
- RICHARDSON, T., STAFFORD-FRASER, Q., WOOD, K. R., AND HOPPER, A. 1998. Virtual Network Computing. *IEEE Internet Computing* 02, 1, 33–38.
- WALLACE, G., CHENY, H., AND LI, K. 2004. Automatic Alignment of Tiled Displays for a Desktop Environment. *Journal of Software* 15, 12 (December), 1776–1786.
- YANG, R., GOTZ, D., HENSLEY, J., TOWLES, H., AND BROWN, M. S. 2001. PixelFlex: a reconfigurable multi-projector display system. In *Proceedings of the conference on Visualization*, 167–174.