Automatic Registration of Multiple Projectors on Swept Surfaces

Abstract

In this paper, we present the first method to geometrically register multiple projectors on a swept surface (e.g., a truncated half-dome) using a single uncalibrated camera without using any physical markers on the surface. Our method can even handle non-linear distortion in projectors common in compact setups where a short throw lens is mounted on each projector. Further, when the whole swept surface is not visible from a single camera view, we can register the projectors using multiple pan and tilted views of the same camera. Thus, our method scales well with different size and resolution of the display. Since we recover the exact 3D shape of the display, we can achieve registration that is correct from any arbitrary viewpoint appropriate for head-tracked single-user virtual reality systems. We can also achieve wallpapered registration more appropriately for multi-user collaborative explorations. Our method achieves sub-pixel accuracy and the image correction required to achieve the registration runs in real-time on the GPU.

Swept surfaces are much more immersive than popular display shapes like planes, cylinders and CAVES. Our method opens up the possibility of using such immersive swept surfaces to create more immersive VR systems without compromising the simplicity of having a completely automated registration technique.


Keywords: geometric registration, calibration, multi-projector displays, tiled displays, immersive displays

1 Introduction

A common way to build high-resolution immersive virtual reality systems is to tile multiple projectors on non-planar displays. Automated registration of these immersive non-planar multi-projector displays using a single uncalibrated camera is instrumental for their easy and inexpensive deployment in common environments. Using single uncalibrated camera for registration purpose is much simpler than (a) using calibrated stereo cameras to reconstruct the display shape via structured light patterns [Raskar et al. 1999; Aliaga 2008; Aliaga and Xu 2008; Raskar et al. 2004; Johnson et al. 2007; Cotting et al. 2004]; and/or (b) attaching obtrusive fiducials on the display screen [Harville et al. 2006; Sun et al. 2008]. [Sajadi and Majumder 2009; Sajadi and Majumder 2010a] show that when considering a display surface vertically extruded from a smooth curve via a piecewise linear curve – like ladders and CAVES – automatic registration can be achieved without using a calibrated stereo pair – but just a single uncalibrated camera.

Swept surfaces are formed by sweeping a profile curve along a path curve to create the 3D shape of the display surface (Figure 2). For e.g. a truncated half-dome (Figure 1) is a commonly used swept surface. Unlike vertically extruded surfaces which are curved only in the horizontal direction and not in the vertical direction – swept surfaces are curved in both horizontal and vertical directions. Hence, they can provide greater immersion than vertically extruded surfaces.

Swept surfaces are easy to build and hence their popularity in mechanical design applications. Unlike domes, another non-planar shape that is curved in both horizontal and vertical direction, swept surfaces lend themselves easily to an intuitive 2D parametrization along the parametrization of the path and profile curve. Registering an image from an arbitrary viewpoint, essential in 3D virtual reality systems, can be achieved on any non-planar shape. However, an intuitive 2D parametrization is important for wallpapering images on a display for collaborative multi-user applications. Wall-papering does not provide a perspective correct imagery from any viewpoint. However, since we are used to seeing wallpapered images around us, it provides an acceptable multi-user viewing experience.

Main Contributions: In this paper, we present the first method that can register multiple projectors on most common swept surfaces automatically using a single uncalibrated camera. We use a two-phase non-linear optimization to recover the camera properties using the prior of a swept surface. The first phase is constrained by the corners of the swept surface and the normals of the path and profile curve yielding a crude estimate of the display shape and the camera parameters. In the second phase, this crude estimate is used as an initialization for faster convergence and is additionally constrained by the shape of the path and profile curve resulting in an accurate estimation of the camera parameters. Next, we use the recovered camera properties to extract the 3D shape of the display.

Finally, we project a few patterns from the projector to find the relationship between the projector coordinates and the 3D display coordinates. We represent this using a rational Bezier function for each projector which is then used to warp the part of the image to be displayed by the projector. Since we recover the exact 3D shape of the display, we can create a view correct from any arbitrary viewpoint.
for a single head-tracked user, as is common in 3D walkthrough applications. Alternatively, we can also achieve a wallpapered registration acceptable from multiple view points (Section 3.3).

When the display is large with not much free space around it to prohibit capture the entire display from a single camera viewpoint, we can use multiple pan and tilted camera views to recover the extrinsic camera parameters for each view and also the 2D display parametrization. This allows us to register multiple projectors on a swept surface even when the variation in the tangent of the path and profile curves are greater than 180 degrees and the entire display cannot be covered by a single camera view.

The main advantages of our method are as follows.

1. Using the prior that the surface is a swept surface, we do not need physical markers or calibrated stereo camera pair for 3D display shape recovery. We can recover the display shape and the 2D parametrization thereof using a single uncalibrated camera.

2. Since we use a rational Bezier patch to relate the projector coordinates with the display coordinates, we can handle distorted projectors, as is common in compact setups where short throw lenses are mounted on the projectors. Further, this allows us to recover the projector to display function even with a sparse sampling of the projector coordinate space. As a result, we can use a low resolution camera to achieve registration of much higher resolution displays.

3. If the display is too large to be seen by a single camera view, we allow registration using multiple pan and tilted camera views. This makes the method scalable to displays of any size and resolution.

4. Our method results in sub-pixel accuracy and yields to real-time image correction on the GPUs.

By providing an easy automatic way to calibrate swept surfaces, our work has the potential to popularize an entirely different class of non-planar shapes – swept surfaces – for immersive virtual reality applications. In the absence of any automatic calibration technique for domes using a single uncalibrated camera until today, layman users may find the use of swept surfaces more attractive since our automatic registration will enable similar sense of immersion as the dome but at a lower maintenance and setup cost.

## 2 Related Work

There has been a large amount of work on registering images on planar multi-projector displays using linear homographies enabled by the planar screen [Raskar 2000; Chen et al. 2002; Raij and Pollefeys 2004; Yang et al. 2001; Yang et al. 2005; Raij et al. 2003; Ashdown et al. 2004], even in the presence of projector non-linearities using rational Bezier patches [Bhasker et al. 2007].

Multi-projector registration on a non-planar display has been achieved by using special fiducials and a large number of structured light patterns for a complete device (camera and projector) calibration and 3D reconstruction of the display surfaces, which are then used to achieve the registration [Raskar et al. 1999]. Aliaga et al. in [Aliaga and Xu 2008; Aliaga 2008] use a similar 3D reconstruction method to achieve registration on complex 3D shapes, but without using any physical fiducials. To constrain the system sufficiently, this method uses completely superimposed projectors and validates the results from photometric and geometric stereo, resulting in a self-calibrating system. Raskar et al. in [Raskar et al. 2004] use a stereo camera pair to reconstruct special non-planar surfaces called quadric surfaces (spheres, cylinders, ellipsoids and paraboloids) and propose conformal mapping and quadric transfer to minimize pixel stretching of the projected images. [Johnson et al. 2007; Cotting et al. 2004] use a stereo camera pair to achieve registration on more general non-planar surfaces like the corner of a room.

More recently, for cylindrical surfaces, [Harville et al. 2006; Sun et al. 2008] do not reconstruct the 3D shape of the display, but finds only a 2D display parametrization in the camera space. This allows a wallpapered registration of multiple projectors on the cylinder by relating a piecewise linear representation of the projector coordinates with a piecewise linear 2D parametrization of the display in the common camera coordinates. However, to find the 2D parametrization, these works need precise correspondences between the physical display and the observing camera. This is achieved by pasting a precisely calibrated physical pattern on the top and bottom rim of the cylinder. Further, the insufficient sampling in the interior of the display surface results in distortions or stretching in those regions. Finally, since the 3D shape of the display is not recovered, view-dependent registrations is not possible.

Our work is closest to a body of work on vertically extruded display surfaces [Sajadi and Majumder 2009; Sajadi and Majumder 2010a] that show that stereo reconstruction is not always necessary when dealing with non-planar surfaces. Using the prior of vertical extrusion and a known aspect ratio it is possible to reconstruct the 3D display shape using a single uncalibrated camera. Consequently, the multiple projectors can be registered on this surface either in a wallpapered fashion [Sajadi and Majumder 2009] or to be correct from any arbitrary viewpoint [Sajadi and Majumder 2010a]. Our method is similar only in essence to [Sajadi and Majumder 2009; Sajadi and Majumder 2010a], but since the class of surfaces we handle is more general and complex, our optimizations use constraints provided by the nature of a swept surface and are entirely different than those provided by vertically extruded surfaces. Further, unlike vertically extruded surfaces, many swept surfaces do not lend themselves to an easy 2D parameterization. We handle this by providing conformal mapping based parameterization for wall-papered registration.
3 Algorithm for Single Camera View

A swept surface is generated by moving a profile curve, \( P(t) \), along the path curve, \( E(s) \). The profile curve can be rotated and scaled in the process. More precisely, for each \( s \) in the domain of the path curve \( E(s) \), the profile curve \( P(t) \) is moved to the point \( E(s) \), possibly with a rotation and scaling. We impose some constraints on this general definition to remove some ambiguities on how \( P(t) \) should be moved along \( E(s) \). We assume that both \( E(s) \) and \( P(t) \) are planar and the plane of the profile curve \( P(t) \) is always perpendicular to the tangent of the path curve \( E(s) \). Finally, we assume that both \( E(s) \) and \( P(t) \) do not have loops. More importantly, these define the more common swept surfaces we see in real life architectures that are more meaningful in the context of displays and do not compromise the applicability of our method.

Since we consider an open swept surface, the path curve \( E(s) \) lies on the XZ plane and is flanked by two planar profile curves on the two sides. We call the left one \( P_{L}(t) \) and the right one \( P_{R}(t) \). \( P_{L}(t) \) lies on the XY plane. However, the plane on which \( P_{R}(t) \) lies, denoted by \( L \), need not be parallel to the XY plane, as is the case in most common swept surfaces. \( L \) and \( E(s) \) are given by the rotation of the XY plane about the Y axis and a translation. Let the points at the two ends of \( P_{R}(t) \) be \( A \) and \( B \). Similarly, let the points at the two ends of \( P_{L}(t) \) be \( D \) and \( C \) respectively. Hence, \( A \) and \( D \) are the two end points of the path curve \( E(s) \). Let the tangent of \( E(s) \) at \( A \) and \( D \) be \( T_{R} \) and \( T_{L} \) respectively. Finally, the tangent of \( P_{R}(t) \) at \( A \) and \( B \) are \( T_{A} \) and \( T_{B} \) respectively and the tangent of \( P_{L}(t) \) at \( C \) and \( D \) are \( T_{C} \) and \( T_{D} \) respectively. We define the origin to be at \( A \) and the vertical distance between \( A \) and \( B \) (and \( C \) and \( D \)) is \( 1 \). We assume that \( E(s) \) lies on the XZ plane and \( P_{R}(t) \) lies on the XY plane. Note that though \( P_{L}(t) \) and \( P_{R}(t) \) have the same shape, they can lie on two planes which are not parallel to each other. Since the tangent of \( E(s) \) at any point is perpendicular to the plane of the profile curves, \( T_{R} \) coincide with \( Z \) axis (Figure 3).

We assume that \( N \) projectors are casually arranged to project on the swept surface \( S \). We denote the 3D display coordinates by \((x_{s}, y_{s}, z_{s})\), the projector coordinates with \((x, y)\), and the camera coordinates with \((u, v)\). We assume that the camera is a linear device with no radial distortion. However, our projectors need not be linear devices. Finally, we assume that the user provides a reasonable estimate of the angle between the XY and L planes.

Our method has three steps (Figure 4). First we use a single image of the display (Figure 5) from the uncalibrated camera to recover the camera properties (intrinsic and extrinsic parameters) using a non-linear optimization. Using the estimated camera parameters, we next recover the path and profile curves in 3D, which is then used to recover the 3D shape of the display. After calibrating the camera and reconstructing the display, we capture an image of a blob-based pattern from each projector (Figure 5) and use these to find samples of the mapping from the projector \((x, y)\) to the display coordinates \((X, Y, Z)\). Once we find this mapping we can register the projectors on the display surface either in a wallpapered fashion or for any arbitrary viewpoint. The next sections provide details of each of these steps.

### 3.1 Recovering Camera Properties

We first use a single image of the display surface (Figure 5) to recover the intrinsic and extrinsic parameters of the observing uncalibrated camera using a non-linear optimization. In most cameras today, it is common to have the principal center at the center of the image, no skew between the image axes and square pixels. As in [Snavely et al. 2006; Sajadi and Majumder 2009], using these assumptions, we express the intrinsic parameter matrix of a camera, \( K_{c} \), as

\[
K_{c} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix}
\]

The camera calibration matrix that relates the 3D coordinates with the 2D camera image coordinates \((u, v)\) is given by \( M = K_{c}[R|T] \) where \( R \) and \( T \) are the rotation and translation of the camera with respect to the world coordinate system. In this step, we use the initial estimate of \( f \), the user provided estimate of the angle between the XY and L planes, and the geometric constraints of a swept surface to setup a non-linear optimization that can estimate seven parameters of the camera calibration matrix. These include the focal length \( f \), the three rotations that comprise \( R \) and the three coordinates of the center of projection of the camera \( T \).

Our non-linear optimization has two phases. In the first phase, endpoint and tangent constrained optimization (Section 3.1.1), the seven camera parameters are estimated using just the projection of the endpoints and the tangent vectors of the path and profile curves on the camera image. These estimates are used to initialize the path and profile curve constrained optimization (Section 3.1.2) with a more expensive error function that uses constraints on the entire path and profile curves to refine the camera parameters.

#### 3.1.1 Endpoint and Tangent Constrained Optimization

In this first phase, we use the constraints on the endpoints of the profile curves \((A, B, C \) and \( D)\) and the tangents of the path and profile curves \((T_{L}, T_{R}, T_{A}, T_{B}, T_{C} \) and \( T_{D})\) to setup the error function which we minimize. To solve for the seven unknown camera parameters, we need at least seven constraints.

For this, we first detect the image of the three curves – \( E(s), P_{R}(t) \) and \( P_{L}(t) \) – in the single image of the display surface seen by the camera using standard image segmentation and edge detection techniques. Let us denote these by \( I(E), I(P_{R}) \) and \( I(P_{L}) \), where \( I \) stands for the 2D image of a 3D entity. Then we detect the images of the four endpoints \((A, B, C \) and \( D)\) in the 2D camera image, denoted by \( I(A), I(B), I(C) \) and \( I(D) \) respectively. We also find the images of the tangent vectors \( T_{L}, T_{R}, T_{A}, T_{B}, T_{C} \) and \( T_{D} \), denoted by \( I(T_{L}), I(T_{R}), I(T_{A}), I(T_{B}), I(T_{C}) \) and \( I(T_{D}) \) respectively, by finding the tangents to the detected path and profile curve – \( I(E), I(P_{R}) \) and \( I(P_{L}) \) – in the 2D camera image.

Our next step is to back-project in 3D these points detected in the 2D camera image using the camera calibration matrix \( M \). We back-project the image of the endpoints and the tangents at the endpoints
of $P_k(t)$, given by $I(A)$, $I(B)$, $I(T_k)$, and $I(T_b)$ respectively, on the XY plane. We denote this back-projection by $R_{XY}$ where $R$ denotes the back-projection function and the subscript provides the plane on which it is back-projected. Thus, the back-projected 3D points are given by $R_{XY}(I(A))$, $R_{XY}(I(B))$, $R_{XY}(I(T_k))$, and $R_{XY}(I(T_b))$ respectively. Similarly, we find the back-projection of the endpoints and the tangents at the endpoints of $I(E)$ on the XZ plane, given by $R_{XZ}(I(A))$, $R_{XZ}(I(D))$, $R_{XZ}(I(T_k))$, and $R_{XZ}(I(T_b))$ respectively. Next we find the plane $L$ by rotating the XZ plane about the Y-axis by the angle between $R_{XZ}(I(T_k))$ and $R_{XZ}(I(T_b))$, the back-projected tangents at the endpoints of $I(E)$; and then translating it by the distance between $R_{XZ}(I(A))$ and $R_{XZ}(I(D))$, the back-projected endpoints of $I(E)$. Then we back-project the endpoints and the tangents at the endpoints of $I(P_k)$ on this plane $L$ to get the points $R_L(I(D))$, $R_L(I(C))$, $R_L(I(T_b))$, and $R_L(I(T_c))$.

We seek to set the error metric such that after convergence the following properties are satisfied.

1. The distance between the Y-coordinates of $R_{XY}(I(A))$ and $R_{XY}(I(B))$, the back-projected endpoints of the $I(P_k)$, should be 1. Hence, the difference of this distance from 1, denoted by $e_1$, should be zero.

2. Similarly, the distance between the Y-coordinates of $R_L(I(D))$ and $R_L(I(C))$, the back-projected endpoints of $I(P_k)$, should be 1. Hence, the difference of this distance from 1, denoted by $e_2$, should be zero.

3 & 4. $R_{XY}(I(A))$ and the origin $(0,0,0)$ should coincide. Thus, the distance of $R_{XY}(I(A))$ from origin along the X direction denoted by $e_3$, and along the Y direction denoted by $e_4$, should be zero.

5. The euclidian distance between $R_{XZ}(I(D))$ and $R_{XZ}(I(C))$ should be equal to the euclidian distance between $R_{XY}(I(D))$ and $R_{XY}(I(C))$. Hence, the difference between these two distances, denoted by $e_5$, should be zero.

6. The angle between the back-projection of the tangent $T_R$ $R_{XZ}(I(T_k))$, and X-axis, denoted by $e_6$, should be zero.

7. Finally, the angle between $R_{XZ}(I(T_k))$ and $R_{XY}(I(T_b))$ should be equal to the angle between $R_L(I(T_b))$ and $R_L(I(T_c))$. Hence, the difference between them, denoted by $e_7$, should be zero.

The above provide us with seven constraints, that is sufficient to solve for the seven unknown camera parameters. Each of the above provide different types of constraints. $e_1$ and $e_2$ constraint the size of the display, and hence serve as scale constraints. $e_3$ and $e_4$ are positional constraints and $e_5$, $e_6$, and $e_7$ serve as shape constraints. To keep the scale of the distances and angles similar, we express the angles in radians. Our error metric in this first phase of the optimization, denoted by $E_f$ is the root mean squares of the weighted error function $e_j$, $1 \leq i \leq 7$. Formally, we seek to minimize $E_f = \sqrt{\sum_{i=1}^{7}(w_i e_i)^2}$. Note that usually the scale constraints are much more important than shape constraints to guide the solution towards the correct size of the display. Also, the shape and positional constraints are equally important since deviation from any one would not preserve the shape and position of the display. Using these guidelines, we design our weights such that $w_3 = w_4 = w_5 = w_6 = w_7 = 1$ and $w_1 = w_2 = 4$.

As in [Snavely et al. 2006], we use the focal length obtained from the EXIF tags of the captured image to initialize the intrinsic parameter matrix in our non-linear optimization. For a reasonable initialization of the camera position and orientation, the user needs to provide an estimate of the angle $\alpha$ between the $XY$ and $L$ planes of the two profile curves. We initialize the camera position to have a Y coordinate which is roughly at the center of the height of the screen (the camera is), we use some simple image processing to find the width $W$ and height $H$ of the display in the camera image. Note that an estimate of the field of view covered by the screen in the vertical direction is given by $\frac{H}{2}$. From this we can find how much in front of the screen the camera should be placed to achieve this for a unit height screen by $-\frac{1}{2}\cot \frac{H}{2}$. Finally, to initialize the X coordinate to be in the middle of the length of the screen, we use $\frac{W}{2}$. Thus, the initialization for the camera position is $\left(\frac{W}{2}, -\frac{1}{2}\cot \frac{H}{2}\right)$. The orientation of the camera is computed by rotating the Z-axis by $\frac{\pi}{2}$ about the Y axis.

3.1.2 Path and Profile Curves Constrained Optimization

The seven estimated camera parameters in the previous step are used to initialize the path and profile curve constrained optimization step that attempts to refine these parameters further. For this, we add another parameter $e_s$ to the error metric $e_f$ in the previous step and use a non-linear optimization method that minimizes the error $e = e_f + w_e e_s$. The error metric $e_s$ provides an estimate of the difference in the similarity in the shape of the left and right profile curves of the display in 3D, and $w_e$ is the associated weight.

The swept surface display is constrained by the fact that the points on the left profile $P_L(t)$ when translated by the distance between the endpoints of $E(s)$ and rotated by the angle between the tangent vector at the endpoints of $E(s)$ should coincide with the right profile curve $P_R(t)$. We use the deviation from this constraint to define the error $e_s$. Please refer to Figure 6 for the following explanation. We first sample the curve $I(P_k)$ in the camera space and fit a parametric curve $\mathcal{C}(I(P_k))$ (green curve) to the samples on $I(P_l)$. Next we sample the $I(P_k)$ in the camera image (dark blue points) and back-project them on the XY plane in 3D to get samples on $R_{XY}(I(P_k))$ (cyan points). Next, we use the current estimate of $M$ to back-project the end points $I(A)$ and $I(D)$ of the path curve to provide $R_{XZ}(I(A))$ and $R_{XZ}(I(D))$ respectively. Similarly, we back-project the tangents at the endpoints, $T_B$ and $T_L$, on the XZ plane to provide $R_{XZ}(I(T_k))$ and $R_{XZ}(I(T_b))$. Next we rotate the samples on $R_{XY}(I(P_k))$ (cyan points) by the angle between $R_{XZ}(I(T_k))$ and $R_{XZ}(I(T_b))$ and translate them by the distance between $R_{XZ}(I(A))$ and $R_{XZ}(I(D))$. These rotated and translated samples (orange points) are then reprojected on the camera image plane (red points). The distance of these reprojected points from $\mathcal{C}(I(P_k))$ (green curve) provides us the error function $e_s$ which we minimize. At convergence, these points should lie on $\mathcal{C}(I(P_k))$.

To solve both optimizations, we use standard gradient descent methods. To assure faster convergence we (a) apply a preconditioning to the variables to normalize the range of the values assigned to them; and (b) use decaying step size.
3.2 Recovering Display Properties

The path and profile curves constrained optimization provides us with a robust estimate of the camera calibration matrix \( M \). In this step, we use the estimated \( M \) to find the 3D shape of the swept surface. We represent the 3D swept surface by a dense set of 3D point samples lying on the surface. To generate a sampling of the 3D swept surface, we first find the 3D samples on the back-projected path curve, \( R_{xz}(I(E)) \). We generate an estimate of the local tangent at each of these samples by considering their immediate neighborhood. Then, we generate 3D samples on the back-projected path curve, \( R_{xz}(I(E)) \) given by \( R_{xy}(I(P_s)) \). We translate this set of 3D samples to place it at every sampled 3D point on \( R_{xz}(I(E)) \) and then rotate it by the amount the tangent changes from one sample on \( R_{xz}(I(E)) \) to the adjacent one. Thus, we generate a dense sampling of the 3D swept surface using a method similar to its construction – by sweeping the sampled profile curve along the sampled path curve. Note that since we sample the image of \( P_L \) uniformly in the camera space and back-project it in 3D, the samples are not uniformly placed on the 3D curves. Hence, the 3D points generated to sample the display are dense but non-uniform. We call this set of 3D points as \( S_C \) where \( C \) stands for cartesian coordinates.

3.3 Geometric Registration

In the geometric registration step, we first define for each projector, a function \( M_{D_{3-p}} \) that maps the projector coordinates \((x,y)\) to the 3D display coordinates \((X,Y,Z)\) via the camera coordinates \((u,v)\). Mathematically, \( (X,Y,Z) = M_{D_{3-p}}(x,y) \). We use the rational Bezier patches to represent \( M_{D_{3-p}} \). Hence, \( (X,Y,Z) = (B_X(x,y), B_Y(x,y), B_Z(x,y)) \).

(2)

To find \( B_X, B_Y \) and \( B_Z \), we find correspondences between \((x,y)\) and \((X,Y,Z)\). We project a number of blobs from each projector and capture them using the camera. Back-projecting the corresponding camera blob centers using the estimated \( M \) and intersecting with the surface \( S \) would provide us the corresponding 3D coordinate. We find this intersection from the set of 3D samples \( S_C \), representing the display, as follows.

First, we represent every point in \( S_C \) using angular coordinates centered at the center of projection (COP) of the recovered camera.

Thus, each 3D point in \( S_C \) has an alternate representation using \((\theta, \phi, d)\) where \((\theta, \phi)\) provides the angular coordinates and \( d \) is the distance at which the ray from the COP of the camera at an angle \((\theta, \phi)\) meets the display surface.

To find the 3D display coordinates corresponding to each blob center \((x,y)\) in the projector space, we first find the angular representation \((\theta, \phi)\) of its corresponding point in the camera space. We interpolate the value of \( d \) for this coordinate by choosing the \( k \)-nearest neighbors of this ray in \( S_C \) and then fitting a smooth interpolating surface through these points in \((\theta, \phi)\) space. Finally we convert back the interpolated points to cartesian coordinates to find the corresponding 3D points in the \((X,Y,Z)\) space.

To find the Bezier patches \( B_X, B_Y \) and \( B_Z \), we fit a rational Bezier patch to these correspondences using a non-linear least squares fitting solved efficiently by the Levenberg-Marquardt gradient descent optimization technique. Rational Bezier is perspective invariant and can model non-linearity. Hence, it can adequately represent the combination of the non-linear distortion due to the swept surface and perspective projection due to the projector.

Registering for an Arbitrary View Point: \( M_{D_{3-p}} \) allows us to correspond every projector pixel to a 3D display coordinate. However, when putting up an image on the display, we need to define how an image coordinates \((s_1, t_1)\) is associated to the 3D display. Essentially, how is the image mapped on the display. This is application dependent. For example, for a single user head-tracked VR application, one would like to render an image of a 3D scene for a virtual arbitrary viewpoint (completely unrelated to the location of the camera used for calibration) and then projectively texture the image for this virtual arbitrary viewpoint on the display surface. As the user moves, the arbitrary viewpoint will change and so will the projective texture on the 3D display surface. In this case, to find the image coordinate \((s_1, t_1)\) associated with each projector pixel \((x,y)\), we first use Equation 2 to find the corresponding 3D point \((X,Y,Z)\) on the display. Then we project this 3D point on a virtual camera at an arbitrary viewpoint to find the image coordinates \((s_1, t_1)\) on the image plane of this camera. Now, for every projector pixel, we can pick the color from the corresponding \((s_1, t_1)\) to create the image to be projected from this projector to create a registered display.

Registering a Wallpapered Image: The function \( M_{D_{3-p}} \) provides us a mapping between the projector pixel \((x,y)\) and the 3D display coordinates \((X,Y,Z)\). For wallpapering, we need to associate 2D coordinates obtained by the 2D display parametrization at this 3D point \((X,Y,Z)\). Instead of repeating this process for every projector pixel, we sample the projector pixels and define \((s_1, t_1)\) at the corresponding 3D display coordinates. Then we use Bezier functions, similar to Equation 2, to interpolate the \((s_1, t_1)\) coordinates at the other projector pixels. Hence, the equation for this case will be

\[ (s,t) = (B_x(x,y), B_y(x,y)) \] (3)

The challenge in wallpapering lies in parametrizing the display to associate a \((s,t)\) parameter with any 3D point \((X,Y,Z)\). A general way to achieve wallpapering, is to use conformal mapping of the image on to the 3D display surface. We use [Springborn et al. 2008] to map an image in an angle-preserving (i.e. conformal) manner on the display mesh. This method considers a 3D mesh of the display surface and associates an image coordinate at every vertex of the mesh. Conformal mapping trims the image to wallpaper it on the surface while preserving the angles. Therefore there is no guarantee that the final trimmed image is rectangular. Alternatively, if the surface is simple with no self-intersections, we can use the inherent 2D parametrization provided by the path and profile curves.

Handling More Complex Swept Surfaces: Depending on the rel-
5 Using Multiple Camera Views: The algorithm described so far assumes that the uncalibrated camera sees the entire display surface. This is often impossible for large displays. So, we design a method that can use multiple views of parts of the screen from the same uncalibrated camera to register the multiple projectors. For this we adapt the method presented in [Sajadi and Majumder 2010b] that does the same for vertically extruded displays. As in [Sajadi and Majumder 2010b], we assume that the camera, placed on a pan-tilt unit, is panned and titled (but not translated) to capture \( Q \) views of the display, in each of which only a small part of the display is visible. These are denoted by \( V_i \), \( 1 \leq i \leq Q \). The zoom of the camera is not changed across these views. This assures that only the extrinsic parameters of the camera change across the views, but the intrinsic parameter matrix \( K_0 \) remains constant. We consider \( V_1 \) as the reference view and the extrinsic matrix of \( V_1 \) to be \( C_1 \). Since the camera is only rotated and not translated, the extrinsic matrix \( C_i \) of view \( V_i \), \( i \neq 1 \), is related to \( C_1 \) by a rotation matrix \( R_i \), i.e \( C_i = R_i C_1 \). We assume considerable overlap between adjacent views.

When handling multiple views, for each camera view a similar set of images as in Section 3 is captured but only for the projectors which are fully or partially visible in that view. As in [Sajadi and Majumder 2010b], we first recover the common intrinsic parameter matrix \( K \) using angular constraints on pairs of correspondences across multiple views, followed by the relative rotation matrix \( R \), which relates every \( C_i \) to \( C_1 \) by finding the minimal spanning tree in a homography graph formed by the homographies relating adjacent overlapping camera views. Finally, the camera calibration matrix \( C_i \) for the reference view is extracted as follows.

To recover the pose and orientation of the reference view with respect to the 3D display, \( C_1 \), we extend the method presented in Sections 3.1.1 and 3.1.2, using only the images which do not have any projectors turned on. The key difference from the single view case is that our boundaries do not appear in a single camera image, but multiple ones. As in [Sajadi and Majumder 2010b], both the back projection and the reprojection happen in the camera views in which the boundary feature or curve is detected. So, in the endpoint and tangent constrained optimization, we do all the back-projections from the respective camera views where they are detected using the matrix \( K R_i C_1 \). Error metric is similar to the one in Section 3.1.1 but is summed across the multiple relevant views and minimized to provide a rough estimate of \( C_1 \). In the path and profile curve constrained optimization stage, first, all the images where the right profile curve is detected are used. Let one such camera image be \( V_k \). The right profile curve is sampled in \( V_k \) and the corresponding 3D points are found by back-projecting from the view \( V_k \) using the matrix \( K R_k C_1 \) on the XY plane. To find the proper rotation and translation of the right profile curve that would give the left profile curve, we find the 3D location of the end points of the path curve and the tangents at these points by identifying appropriate camera views that see these entities. Using back-projection on the XZ plane in the neighborhood of these ends in these camera views, we decipher the rotation and translation of the back-projected 3D points (cyan points in Figure 6) on the right profile curve that yields 3D points close to the left profile curve (orange points in Figure 6).

Now, unlike Section 3.1.2 which reprojected these 3D points of the estimated left profile curve to the single camera view (red points in Figure 6), we reproject these points to all the camera views where a part of the left profile curve is detected. Each reprojected points may appear within the field-of-view (FOV) of multiple overlapping views. The average of the distance of the reprojected point from the detected 2D left profile curve across all these views define our error for each point. We sum the errors across all the points to provide a reprojection error \( e_i \). We seek to minimize the sum of \( e_i \) across all the camera views where the left profile curve is detected to find \( C_1 \).

4 Implementation and Results

We have implemented our method using Matlab on two real displays (Figure 1). The first one is a truncated half-dome, 30° in ra-

---

**Figure 8:** Results on the real truncated half-dome (30° radius, 26° high, subtending 180 degrees horizontal angle). Top: The reconstructed 3D shape of the truncated quadrant of the sphere along with the five camera views used for this purpose. Since the five views are just panned from the single position, the homography graph and tree are both a straight line. Bottom: The display made of six projectors is wallpapered using the natural curve based parameterization of the surface. The overlays are not blended to show the different projectors.
The offline registration takes about five minutes. The image correction grid of $32 \times 8$ near the corners and hence require a degree 7 rational Bezier and a cicient. However, the bowl display shows sharp changes in curvature correspondences [Raskar et al. 1999; Yang et al. 2001] (Figure 5).

We use a Canon Rebel xSi camera (around $800) for calibrating respectively. We illustrate these shapes with panorama images using spherical correspondences; we capture a rectangular grid of Gaussian blobs with known projector coordinates displayed by the projector. We binary-encode the blobs and project them in a time sequential manner to recover the exact IDs of the detected blobs and find the correspondences [Raskar et al. 1999; Yang et al. 2001] (Figure 5).

The degree of the rational Bezier patches $B_x$, $B_y$ and $B_z$ and the number of blobs used depend on the amount of non-linearities present due to the screen curvature and the distortions in the projectors. Our truncated sphere quadrant shows smooth variation in curvature and hence a rational Bezier of degree 5 in both dimensions and a grid of $16 \times 8 = 128$ blobs for each projector was sufficient. However, the bowl display shows sharp changes in curvature near the corners and hence require a degree 7 rational Bezier and a denser grid of $32 \times 16 = 512$ blobs for each projector.

The offline registration takes about five minutes. The image correction can be implemented in real-time, as in [Sajadi and Majumder 2009], using GPUs through Chromium - an open-source distributed rendering engine for PC clusters [Humphreys et al. 2002]. The coordinate-mappings of all pixels of the projector should be first precomputed. This per-pixel projector to screen lookup table can then be used by a fragment shader to map pixels from the projector coordinate space to the screen coordinate space during rendering.

For the truncated quadrant of the sphere and the bowl display, we use five and six camera views respectively. The camera views, reconstructed 3D display shape and the registered projectors for these two displays are shown in Figures 8 and 9 respectively.

To demonstrate our results for displays with more projectors, we show two complex shapes in simulation. The first is a truncated ellipsoid subverting an angle of 300 and 100 degrees in the horizontal and vertical direction (similar to the surface in the bottom right of Figure 2) lighted by a $2 \times 8$ array of sixteen projectors captured by six panned camera views (Figure 10). The second is a large panoramic shape similar to the swept surface in the bottom left of Figure 2 lighted by a $4 \times 10$ array of forty projectors captured by seven panned and tilted camera views (Figure 11).

In Figure 9 we show a view-dependent registration from an arbitrary viewpoint, as is common in 3D VR environments. The accompanying video illustrates better the distortions that appear when the viewer deviates from the ideal viewpoint. We also register the display in a wallpapered manner. Since this is a self-intersecting surface, we use a conformal-map-based wall papering (Figures 1 and 9). Note that when using conformal mapping, the constraints imposed do not allow us to map the entire image on the display, but clip off some regions near the boundary. In Figure 8 we show a wallpapering based on the natural curve-based parametrization. In Figure 10 and 11 we show wallpapering registration using the natural curve based parametrization of the display. Finally, we demonstrate the ability of our method to register images in the face of non-linear distortions for the complex bowl display in Figure 9. Our projectors have relatively large throw-ratios and hence little lens distortions. So, we chose to simulate the distortion digitally by distorting the input images to the projectors. Note that our results use particularly challenging contents like lines and texts to demonstrate accuracy. For better zoom in and fly-through experience of VR applications on our immersive swept surfaces we strongly encourage the readers to see the video.

5 Discussion

Camera Placement: There is a set of camera positions that will lead to degenerate cases for one or both phases of our non-linear optimization. First, the camera should not be placed at a location from which any of the path or profile curves would be projected as a line. This will happen if the normal to the image plane lies on the XZ plane ($E$ will be projected as a line), XY plane ($P_y$ will be
It also opens up the possibility of the use of smooth pleasing swept surfaces (as in Figure 11) for applications like digital signage and aesthetic projections in malls, airports and other public places.

### References


### Table 1: Percentage Errors of the estimated camera and display parameters over a large number of simulations with different device and display configurations.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Max</th>
<th>Mean</th>
<th>Std</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera Orientation (deg)</td>
<td>0.637</td>
<td>0.231</td>
<td>0.195</td>
</tr>
<tr>
<td>Camera Position (%)</td>
<td>0.592</td>
<td>0.213</td>
<td>0.189</td>
</tr>
<tr>
<td>Focal Length (%)</td>
<td>3.571</td>
<td>1.932</td>
<td>0.891</td>
</tr>
<tr>
<td>Path Curve (%)</td>
<td>0.680</td>
<td>0.254</td>
<td>0.203</td>
</tr>
<tr>
<td>Profile Curve (%)</td>
<td>0.713</td>
<td>0.269</td>
<td>0.191</td>
</tr>
</tbody>
</table>

### Figure 11: Results from simulation: Top: The 3D reconstructed display and the seven camera views for the wavy panoramic display lighted with a × 1 array of 40 projectors. Bottom: An image registered in a wallpapered fashion using the natural curve based parametrization of the display. Please zoom in to see results.