Matched-Filter Based Geometric Alignment for Tiled Displays

D.M. Germans\textsuperscript{1*} H.J.W. Spoelder\textsuperscript{1†} T. v.d. Schaaf\textsuperscript{2‡} L. Renambot\textsuperscript{2§}

H.E. Bal\textsuperscript{1,2¶}

\textsuperscript{1}Division of Physics and Astronomy
\textsuperscript{2}Division of Mathematics and Computer Science
Faculty of Sciences – Vrije Universiteit
De Boelelaan 1081 – 1081 HV Amsterdam – The Netherlands

Abstract

This paper describes a geometric calibration procedure for tiled displays. It shows how computer vision techniques can be used to facilitate accurate geometric alignment. Our approach localizes features with sub-pixel accuracy and takes into account inaccuracies due to projector lens distortions. With these enhancements, the results show that common calibration accuracy of several millimeters (on the display surface) can be pushed into the sub-millimeter domain.

1 Introduction

In the field of visualization, PC cluster-driven tiled displays are becoming increasingly popular [Funkhouser and Li 2000]. Tiled displays provide high resolution visualization by tiling together projector outputs on a screen. Each projector is driven by a graphics-oriented PC, interconnected via a fast network (Gigabit Ethernet or Myrinet[Boden et al. 1995]).

Setting up a tiled display introduces important visual calibration issues regarding image intensity, color balance, and most visible, geometric alignment. This paper concentrates on geometric alignment issues. Attempting to align projectors manually is undesirable for a small number or projectors and unrealistic for a large number of them. So there have been several initiatives to research partially or fully automated methods of alignment [Raskar et al. 1999; Chen et al. 2000]. The general idea is to set up the projectors with a certain amount of overlap and then transform each projection in such a way that the overall image is correct. Figure 2 shows two images of an example setup. First, a completely unaligned setup. Notice the higher intensity in the overlap regions, and the misalignment of text and window-elements. Second, a completely calibrated setup. Here, the overlap regions have been corrected and the text and window-elements are aligned.

This paper describes in detail a fully automatic deterministic method to align any number of projectors that project onto a flat surface. Figure 1 shows an overview of the setup. First, a well-defined pattern (stimulus) is projected onto the screen for each projector. These images are captured and processed, and a model is fitted to each pattern to yield a coordinate transformation for each projector. Finally, the transformation is inverted in order to adjust the source image, resulting in a seamless alignment of the tiles.

Our focus lies on achieving high accuracy with off-the-shelf commodity equipment. We show how a sub-pixel-accurate matched-filter convolution can be used to determine the position and orientation of each projector in a tiled-display setup.

Next to this, commodity projectors are often projecting off-axis to allow users with a regular single-projector setup to place the projector on a desktop and still project high enough for their audience. Off-axis projection and the use of low-cost lenses are the main causes for an apparent lens distortion that becomes visible as soon as tiled displays - rather than single-projector setups - are considered. Our approach proposes a simple and effective correction for these lens distortions.

In short, the contributions of this paper are:

- We present a deterministic and fully automatic method to align tiled displays using a consumer digital camera,
- A sub-pixel matched-filter point localization method is applied and its limits are evaluated,
- An approximation is tested to model the projector lens distortions.

Section 2 describes similar approaches, section 3 describes our procedure and the matched filter characteristics, section 4 shows results and comparisons and section 5 gives conclusions and further work.

2 Related Work

Tiled display alignment has undergone development from manual setup to fully automated methods. Several issues of the alignment procedure have been looked at in research.

Model

In order to correctly analyze the patterns that have been captured by the CCD camera, one must use an adequate mathematical model of a projector or projection. A simple model is an (affine) 2D transformation that describes scaling, translation and rotation of the projected image. This is adequate for a limited set of distortions, but if the projectors are placed at more extreme angles, it will fail due to perspective correction problems (like keystoning).
The most complete model for a flat screen is a full 8-term 2D homographic transformation, including perspective correction. This homographic transformation, or homography, is used with success, among others, by [Chen et al. 2001]. However, the plain 8-term homography does not take lens distortions into account. This somewhat limits the achievable accuracy of the model, as commodity projectors usually have an almost visible lens distortion (due to off-axis projection and lens cost reduction).

**Scalability**

Scalability for tiled display alignment can be specified in several ways. One can examine the order of computation as a function of the number of tiles, and one can look at the quality of the produced result as a function of the number of tiles. [Chen et al. 2000] designed an iterative method that measures the misalignment of adjacent projected lines and points. Knowing this it adjusts each projector towards a minimum of overall misalignment. In practise, this method is time-consuming (in the order of tens of minutes), and is not deterministic. Also, the individual misalignment between tiles might be reduced, but the global shape of all tiles combined is not guaranteed to be identical every time the system is calibrated. Their newer work, involving the analysis of homography-trees [Chen et al. 2001], is deterministic and much faster. The method analyzes the relationship between 2x2 tile groups, by panning and zooming a camera across the screen surface. Afterwards, the estimated (local) homographies are combined into a tree, and a global alignment is calculated. Even though the method is scalable because local alignment is considered (the camera can pan across any number of tiles with roughly the same accuracy), combining local homographies into a tree introduces an additional estimation with errors.

**CCD Camera**

Depending on the method, the CCD camera that is used might require geometric pre-calibration before the projectors can be successfully aligned. [Raskar et al. 1999] devised a method to apply calibration to arbitrary and complex projection surfaces (curved, room corners, etc.). The method uses two calibrated cameras to find the geometry of the surface and a general mapping from 3D space to projection surface. Even though this method allows for
much more freedom of choice in the projection surface, it requires calibrated cameras.

[Surati, 1999] developed a method where a mapping is retrieved between a camera, the projection screen and each projector. This method relies on a physical grid on the projection screen which is often undesirable in practical setups (for cosmetic reasons, for instance).

Cost
One argument in each approach is cost. The less automatic the method is, the more it will cost to maintain in practical environments. We are particularly interested in educational settings, and here, a fully automatic and low-cost method is preferable. The homography-tree method [Chen et al. 2001] requires a camera that can pan, tilt and zoom to focus on 2x2 tile groups individually. Also, most methods require a camera with a very high resolution, in order to obtain enough accuracy in measuring the patterns. The more the algorithm relies on accurate measurements, the more expensive the camera, the screen and the projectors would become.

Output Performance
Next to the speed of the actual alignment procedure, the modifications that are necessary to generate output are considered. Modern graphics systems allow for several adjustments of the output image. If the model that was used during the calibration is in the order of a homographic transformation, it can be adequately approximated in the output at no cost when graphics generating APIs like OpenGL or Direct3D are used. The homographic transformation translates to a 4x4 transformation matrix without trouble.

Overlap Regions
For the overlap regions between the tiles, it is common practise to either shield off the beam of the projector by adjustable hardware [Hereld et al. 2000], or to multiply the output image of each projector with an alpha mask [Raskar et al. 1998]. Both methods have the effect that the image is projected and gradually fades to black on the sides, allowing the neighboring tile to mix in. The hardware shielding method is more cumbersome to maintain.

3 Procedure
We explore the practical aspects of tiled display calibration, with an emphasis on usability and accuracy. Figure 1 shows our approach.

A checkerboard pattern is projected on each individual tile, and a static CCD camera captures the entire display surface for each projected checkerboard pattern. Now, a sub-pixel-accurate computer vision algorithm recognizes the inner crossings of the checkerboard patterns. An 8-parameter homographic transformation is fitted to the lists of crossings using a non-linear least squares parameter estimation procedure. The result is a set of transformations that describe a relationship between projector-coordinates and coordinates on the display surface. From these transformations the largest displayable rectangle is determined. Finally, the display regions and overlap regions for each tile are calculated. From the overlap information, alpha masks are generated for intensity blending at the overlap regions.

Recognizing Checkerboard Crossings
To characterize the position and orientation of the individual projectors, we detect features of a synthetic stimulus generated by this projector. In our case we use a checkerboard pattern of known -
adjutable - size (Figure 3). The features that are being looked for, are the crossings in the checkerboard. These crossings can be detected by performing a convolution of the captured image with a matched filter that has a peak response when the filter matches the crossing. This method has been proposed earlier by [Vuylsteke and Oosterlinck 1990]. In its general form, it is given by:

\[ g(x, y) = I(x, y) \otimes \begin{pmatrix} -1 & \ldots & -1 & 0 & 1 & \ldots & 1 \\ \vdots & & \vdots & & \vdots & & \vdots \\ -1 & \ldots & -1 & 0 & 1 & \ldots & 1 \\ 0 & \ldots & 0 & 0 & 0 & \ldots & 0 \\ 1 & \ldots & 1 & 0 & -1 & \ldots & -1 \\ \vdots & & \vdots & & \vdots & & \vdots \\ 1 & \ldots & 1 & 0 & -1 & \ldots & -1 \end{pmatrix} \] (1)

where \( g(x, y) \) denotes the response of the matched filter at position \((x, y)\) of the image \( I(x, y) \) and the matched filter is a \((2n+1)\) by \((2n+1)\) matrix where \((2n+1)\) is known as the filter size. At intermediate positions, the outcome of the convolution will approximate 0. Figure 4 shows a response of such a convolution. The white and black dots are positive and negative extrema, indicating full match and full anti-match (a checkerboard crossing where white and black are swapped).

When noise is present in the input image \( I \), the danger exists of accepting false positives in the responses. This can be reduced by accepting only those locations where the matched filter responses exceed a preset threshold \( T_0 \).

Note that the size of the matched filter is correlated to the size of the checkerboard blocks. It can be shown that optimal response is achieved when the filter size is half the length of a checkerboard block. In other words, when the checkerboard block size is smaller than twice the filter size, the filter response degrades. Also, when the checkerboard block size is larger than twice the filter size, not more information is obtained from the convolution.

**Sub-pixel Localization**

The localization procedure described so far leads to an estimated position at pixel precision. For the sub-pixel localization we will also follow [Vuylsteke and Oosterlinck 1990]. A computationally inexpensive approach suggested by [Vos 1998] is based on the center-of-mass calculation of the responses to the matched filter. It has been shown that the center of mass is an accurate approximation to the correlation extremum. Again when dealing with noisy input images, one should threshold the responses taken into account for the sub-pixel localization. In general this sub-pixel localization is given by:

\[ \delta u_0 = \frac{\sum \sum r (g(u_0 + x, v_0 + y) x)}{\sum \sum r (g(u_0 + x, v_0 + y))} \] (2)

\[ \delta v_0 = \frac{\sum \sum r (g(u_0 + x, v_0 + y) y)}{\sum \sum r (g(u_0 + x, v_0 + y))} \] (3)

\[ r(r) = \begin{cases} 0 & : r < T_0 \\ r & : r \geq T_0 \end{cases} \] (4)

with \((u_0, v_0)\) the location of the crossing at pixel level and \((\delta u_0, \delta v_0)\) the sub-pixel offsets. The function \( r(r) \) effectively rules out points with a response below the threshold \( T_0 \). The summations are over a symmetric square of \((2n+1)\) by \((2n+1)\) points, centered around \((u_0, v_0)\), with \((2n+1)\) being, again, the filter size. The computation is equal to the number of pixels in the input image. Various highly efficient schemes have been developed for the implementation of such convolutions.

Although in theory, false positives and false negatives can occur, the a-priori knowledge of the input pattern effectively rules out false positives and likewise makes false negatives easy to detect.

**Extracting the Homographic Transformations**

The homographic transformation is the same used by [Chen et al. 2000]. It is essentially a 3x3 homogeneous 2D transformation:

\[ H = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{pmatrix} \] (5)

\[ \begin{pmatrix} x_{\text{hom}} \\ y_{\text{hom}} \\ w_{\text{hom}} \end{pmatrix} = H \begin{pmatrix} x_{\text{in}} \\ y_{\text{in}} \\ 1 \end{pmatrix} \] (6)

Where the 3rd coordinate behaves like a perspective denominator (as in 4x4 homogeneous matrix calculations):

\[ x_{\text{out}} = \frac{x_{\text{hom}}}{w_{\text{hom}}} \] (7)

\[ y_{\text{out}} = \frac{y_{\text{hom}}}{w_{\text{hom}}} \] (8)

Robustness is important, so we fit all points found on the projector surface simultaneously with a non-linear least-squares fitting process. If false positives or false negatives still exist, they can be eliminated at this stage by analyzing the residuals of the fit. If the residual for a specific point is above a relative threshold, the fit is processed again, disregarding this point.

Figure 5 shows the visual result of the fit. The lines denote the estimated model and the circles are the actual detected crossings. Note, for each point, the distance between line crossing (the model) and the corresponding point. The average of all distances, the average error distance is used as a metric to assess the accuracy of this method later on.

![Figure 5: The model (lines) fits onto the detected crossings (circles). The average error distance is the average of all distances between model coordinates and corresponding detected crossings.](image-url)
Modeling Lens Distortions

Due to off-axis projection and lens cost-reduction in commodity projectors and camera, the lenses give rise to pin cushion effects and other lens-related distortions. In order to reduce the image errors caused by these effects, we present a model that approximates the effect of lens distortions. For robustness we want to fit all parameters of the entire system simultaneously, so the actual fitting function should incorporate this model next to the homographic transformation. Complete physical lens modeling for cameras as described by [Zhang 2000] in [Intel 2000] introduces many new parameters of which several are not orthogonal to parameters of the homographic transformation that we need to find. This means that the fitting process will be degenerate.

Ideally we look at a combined function that incorporates the homographic transformation and both camera and projector distortions. The most important component in this is obviously the homographic transformation because it accounts for the overall position and orientation of the projected image. Taking a 1x1 tiled display, the lens distortions of the projector and those of the camera can not be distinguished from each other and thus result in a degenerate fit. Taking an nxn tiled display (with n larger than 1), the lens distortions of the camera and the homographic transformation can not be distinguished from each other any more, again resulting in a degenerate fit. In the fitting process, the position and orientation of each individual tile can encompass whatever distortion the camera lens is producing.

Now, taking the above into account we choose only to model the projector lens distortions, and, in order to produce a healthy fit, only the parts of the distortions that are orthogonal to the homographic transformation. Note that neglecting to model the camera distortion is not entirely free of penalty. When the camera is close to the screen (for small displays) the distortion tends to be larger, resulting in a higher average error distance. We will address this in section 4.

To model the projector lens distortions, we take a set of generic radial Taylor-expansions:

\[ r = \sqrt{(x_{\text{out}} - x_c)^2 + (y_{\text{out}} - y_c)^2} \]  
\[ x_{\text{ldm}} = x_{\text{out}} + \alpha_1 r + \alpha_2 r^2 + \alpha_3 r^3 + \ldots + \alpha_{n} r^n \]  
\[ y_{\text{ldm}} = y_{\text{out}} + \beta_1 r + \beta_2 r^2 + \beta_3 r^3 + \ldots + \beta_{n} r^n \]

Here, \( r \) is the radius of the coordinate \( x_{\text{out}}, y_{\text{out}} \) around the center of the distortion \( x_c, y_c \). Note that \( \alpha_i, \beta_i \) are the constant and linear coefficients, are not orthogonal to the homographic transformation (they interfere with the translation and scaling parameters) and should not be considered.

A small investigation shows that higher order terms (in \( r^3 \) and up) have insignificant contribution. This leaves us with the following perturbation:

\[ x_{\text{ldm}} = x_{\text{out}} + \alpha_2 r^2 \]  
\[ y_{\text{ldm}} = y_{\text{out}} + \beta_2 r^2 \]

All 8 components of the homographic transformation, the center of distortion (\( x_c, y_c \)) and the two quadratic coefficients (\( \alpha_2 \) and \( \beta_2 \)) are now fitted simultaneously for every tile to yield an optimal and robust result.

Combining the Projectors

Repeating the above estimation process for each projector results in a series of transformations that transform projector coordinates to coordinates in camera space. This now lets us view the tiled display as a one-projector display with a (virtual) projector at the location of the camera. Knowing the configuration of the tiles, we can determine the largest rectangle that fits inside the projected surfaces (Figure 6).

When the largest rectangle is found, intersection with each projector surface quad in camera space yields the effective surface. For this, a quad-quad intersection algorithm is required. Our implementation uses an adapted version of the polygon-polygon intersection algorithm by [Holwerda 1998]. Using the inverse of the homographic transformation, the effective surface can be transformed back to projector space.

Knowing the corner points of these quads in camera space is sufficient to generate a static mapping with which, for instance, a desktop application can be spanned geometrically correctly on the entire tiled display.

Extracting the Overlap regions

To generate alpha masks for the overlap gradients, we intersect the effective surface of a projector with its neighboring projectors. This results in small polygons that cover the overlap regions. Simple distance criteria can be used to mark the corner points of these polygons (full (1.0) or black (0.0)). Interpolating between these corner points results in gradients that should seamlessly fit in each overlap region on the display. Our implementation generates an alpha mask for each projector that is multiplied with the projected image in hardware (using texture mapping), resulting in a gradual attenuation of each tile at the overlap regions.

Figure 7 shows the intensity of two adjacent tiles in the overlap region, and the resulting intensity with and without an applied alpha mask. Note that without the attenuation of both tiles at the overlap region, we see an intensity band appearing. Note also that by knowing the position and orientation of each tile we can calculate the overlap region’s geometry (lines A and B in the figure).

Real-time 3D output

The fastest way of adjusting the projector output image is to generate a 4x4 homogeneous 3D transformation matrix that can be processed for free in 3D graphics hardware, and postprocess the final image with the alpha mask, either by hand or by hardware.

Taking OpenGL as an example, the procedure to adjust the matrix goes as follows. We require 3 consecutive transformations to use the homography effectively, \( T_{\text{ac}} \) that transforms from OpenGL unit coordinates in the application to camera coordinates, \( T_{\text{pq}} \) that transforms from camera coordinates to projector coordinates (the inverse of the homographic transformation) and \( T_{\text{po}} \) that transforms from projector coordinates to OpenGL unit coordinates of the individual tile-driving graphics systems. \( T_{\text{ac}} \) can be described as a simple orthographic transformation that maps OpenGL unit coordinates to the display rectangle in camera coordinates (\( x_1, y_1 \) to \( x_2, y_2 \))
Figure 7: Intensity profile at the overlap region of two adjacent tiles for with and without an applied alpha mask. Note that without the attenuation of both tiles at the overlap region, we see an intensity band appearing.

\[ T_{ac} = \begin{pmatrix}
\frac{x_2 - x_1}{2} & 0 & 0 & \frac{x_2 + x_1}{2} \\
0 & \frac{y_2 - y_1}{2} & 0 & \frac{y_2 + y_1}{2} \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix} \] (14)

Where \( x_1, y_1 \) is the upper-left corner of the display in camera coordinates and \( x_2, y_2 \) the lower-right corner. Note that this is the inverse of the standard orthographic projection matrix, because we transform unit coordinates to camera coordinates, and not the other way round.

Now, if we describe each component of \( H^{-1} \) as follows

\[ H^{-1} = \begin{pmatrix}
a_n & b_n & c_n & f_n \\
d_n & e_n & 0 & 0 \\
g_n & h_n & 0 & 0 \\
i_n & 0 & 0 & 0
\end{pmatrix} \] (15)

we can write \( T_{cp} \) as

\[ T_{cp} = \begin{pmatrix}
a_n & b_n & 0 & c_n \\
d_n & e_n & 0 & f_n \\
g_n & h_n & 0 & 0 \\
i_n & 0 & 0 & 0
\end{pmatrix} \] (16)

Note that the adjustment \( 1 - |c_n| - |f_n| \) is an approximation proposed by [Raskar 2000] to reduce z-coordinate distortion.

Finally, \( T_{po} \) can be written as another orthographic transformation

\[ T_{po} = \begin{pmatrix}
2 & 0 & 0 & -1 \\
0 & -2 & 0 & 1 \\
0 & 0 & -1 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix} \] (17)

Combining \( T_{ac} \), \( T_{cp} \) and \( T_{po} \) on the OpenGL matrix stack gives the following piece of code:

```c
void glCalibTile(float* invhom) {
    float m[16];
    // apply Tp0:
    glOrtho(0,0,1,1,-1,1);
    // apply Tcp:
    memset(m,0,16 * sizeof(float));
    m[0] = invhom.a; m[1] = invhom.d; m[3] = invhom.g;
    m[10] = 1.0f - fabs(invhom.g) - fabs(invhom.h);
    m[12] = invhom.c; m[13] = invhom.f;
    m[15] = 1.0f;
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glMultMatrixf(m);
}
```

A typical display function for a certain tile would then look like:

```c
void display(int tile) {
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    // apply the tile transformation:
    glCalibTile(invhom[tile]);
    // application-defined projection:
    glPerspective(60,2,1,1000);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    // perform drawing here
    glFlush();
}
```

Adding the lens distortion model to the equation makes it harder to approximate by a 4x4 matrix, so a full per-pixel warping procedure needs to be considered. This procedure may be implemented using pixel shader technology, available on most modern 3D hardware.

The above procedure is implemented transparently in the Aura parallel scene graph library [Germans et al. 2001; van der Schaaf et al. 2002]. This library provides scene graph access for a cluster of PCs driving a tiled display. Any 3D application designed with Aura is automatically and transparently corrected for tiled-display alignment.

4 Results

To show the effect of sub-pixel localization and lens distortion modeling, a series of measurements have been done. The measurements...
were done at the ICWall classroom site [ICWall 2002]. This is a 9-node PC cluster, driving 8 Philips Ugo-X DLP(DMD) projectors. The projectors project on the rear of a Stewart Film acrylic back-projection screen into a slightly darkened classroom, as shown in Figure 8. The screen surface is 6m by 2m. The camera used is a Canon S40 PowerShot with a 2272x1704 pixel color CCD. The camera was set up perpendicular to the screen, capturing the entire width of the screen on the width of the CCD (so 6 meters corresponds to 2272 pixels).

![Figure 8: ICWall site: lecture on protein dynamics. Calibration for this display is performed as described in this paper. The procedure is completely automatic, which makes it attractive for an educational environment.](image)

As described earlier, the metric used is the average error distance. This is the average of all distances between measured crossings and model results. The metric is expressed in millimeters on the surface of the screen. These values can be calculated by comparing camera distances with a calibration distance of one meter on the screen.

The rest of this section will give an analysis of the noise and parameter estimation health, as well as results for various tile configurations.

**Noise and Block Size**

Figure 9 shows a typical distribution of grayscale values in a capture of a projected checkerboard. From the figure we clearly see a large separation between low (black) and high (white) values. This indicates that the capture is relatively noise-free. The spread in high values in Figure 9 is due to an overall gradient over the captured image, caused by light sources other than the projector (sunlight or room lighting).

Table 1 and accompanying Figure 10 show the average error distance for a 1x1 tile setup, using different block sizes and a fixed filter size of 9. A larger filter size would not add much information, considering the low amount of noise in the captured image. Table 1 suggests that the most optimal (lowest error) are large blocks (6x6, 325 pixels). Expectable as this might seem, it is important to realize that using less crossings for the fitting process decreases the precision of the parameters. Taking a look at Table 2 shows that for the parameter h of the homographic transformation, the actual values are comparable for different block sizes, but the error is much lower for 18x18 blocks than it is for 6x6 blocks. The reason for this is that for an 18x18 pattern, there is more information available to make an estimation, allowing for more precise parameters. This implies that the final choice for pattern size needs to be a tradeoff between filter accuracy and parameter precision. For our further measurements, we choose an 8x8 pattern.

![Figure 9: Histogram of the grayscale values in a typical captured checkerboard pattern. Note a large separation between the dark peak and the light area. The spread in light values is due to an overall gradient over the image caused by, for instance, sunlight.](image)

<table>
<thead>
<tr>
<th>Pattern</th>
<th>horz. block size</th>
<th>avg. err. dist.</th>
</tr>
</thead>
<tbody>
<tr>
<td>6x6</td>
<td>325 pix.</td>
<td>0.9mm</td>
</tr>
<tr>
<td>8x8</td>
<td>245 pix.</td>
<td>1.1mm</td>
</tr>
<tr>
<td>10x10</td>
<td>195 pix.</td>
<td>1.1mm</td>
</tr>
<tr>
<td>12x12</td>
<td>162 pix.</td>
<td>1.4mm</td>
</tr>
<tr>
<td>14x14</td>
<td>139 pix.</td>
<td>1.5mm</td>
</tr>
<tr>
<td>16x16</td>
<td>122 pix.</td>
<td>1.5mm</td>
</tr>
<tr>
<td>18x18</td>
<td>108 pix.</td>
<td>1.6mm</td>
</tr>
</tbody>
</table>

Table 1: Average error distance as function of checkerboard block size. The larger the number of blocks, the smaller each block (see also Figure 10).

Pattern | parameter h | error in h |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6x6</td>
<td>0.02014</td>
<td>±0.0056</td>
</tr>
<tr>
<td>12x12</td>
<td>0.01811</td>
<td>±0.0015</td>
</tr>
<tr>
<td>18x18</td>
<td>0.01695</td>
<td>±0.0007</td>
</tr>
</tbody>
</table>

Table 2: Estimation error for parameter h of the homographic transformation, deduced from the covariance matrix of the fit. Note that the suggestive 6x6 (lowest distance in Table 1) has the highest estimation error (lowest precision).

**Fitting Results**

Table 3 and accompanying Figure 11 show the average error distances for various tile configurations. The topmost line (in Table 3, the column marked as plain) shows the error distances for a situation where sub-pixel localization and lens distortion modeling are not used. Here we see a steady increase in error, as the camera gets further away from the screen (larger number of tiles with each tile covering less camera pixels). The center line (in Table 3, the column marked as subp) shows the error distances for a situation where only sub-pixel localization is used, but no lens distortions are modeled. After an initial setup peak at 2x2 we see the same steady increase for larger setups. The bottom line (in Table 3, the column marked as LDM) indicates the error distances for a situation where both sub-pixel localization and lens distortion modeling are applied. Here we see that the short distance between camera and screen forces the extra parameters from equation 13 to fit to the
camera lens, instead of the projector lens, resulting in a slight error increase.

<table>
<thead>
<tr>
<th>Configuration</th>
<th>plain</th>
<th>subp.</th>
<th>LDM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1x1</td>
<td>0.9mm</td>
<td>0.8mm</td>
<td>0.6mm</td>
</tr>
<tr>
<td>2x2</td>
<td>1.2mm</td>
<td>1.2mm</td>
<td>0.5mm</td>
</tr>
<tr>
<td>3x2</td>
<td>1.2mm</td>
<td>1.0mm</td>
<td>0.5mm</td>
</tr>
<tr>
<td>4x2</td>
<td>1.4mm</td>
<td>1.0mm</td>
<td>0.6mm</td>
</tr>
</tbody>
</table>

Table 3: Average error distance for plain, with sub-pixel and with lens distortion modeling for an 8x8 checkerboard pattern on various tile configurations (see also Figure 11).

The result of applying sub-pixel localization and lens distortion modeling to the 8-projector ICWall tiled display can be seen in Figure 2 and Figure 8. For a typical viewer distance in the order of several meters, errors in alignment are unnoticeable. Furthermore, automatic, hassle-free and fast calibration is a definite plus in an educational environment.

5 Conclusions and Future Work

We show, in detail, an approach to aligning tiled displays using off-the-shelf components and tools. The method is deterministic and completely automatic. Looking at the absolute results and the trends indicated by Figure 11, we see that for the current range of used tiled display setups, this method reaches sub-millimeter accuracy.

For the current educational setup, this is accurate enough, but future plans for this site include conversion of the display to passive stereo by overlapping two polarized projector outputs on the same tile. For this, the average error distance directly influences the stereo separation (the maximum optical ‘depth’ that can be achieved with a stereo display).

Even though this approach yields very accurate results over a promising range of tile configurations, the effects of the camera lens distortion becomes apparent at small tile setups (1x1, 2x2). For a better overview of the effect of camera lens distortion, a more in-depth analysis is required. Also, for robustness, a simultaneous fit of all tiles at once can open possibilities to use a more complex model where camera distortions can be expressed.

References


