Automatic Alignment Of High-Resolution Multi-Projector Displays Using An Un-Calibrated Camera

Yuqun Chen* Douglas W. Clark* Adam Finkelstein* Timothy C. Housel† Kai Li*

Department of Computer Science, Princeton University

Abstract

A scalable, high-resolution display may be constructed by tiling many projected images over a single display surface. One fundamental challenge for such a display is to avoid visible seams due to misalignment among the projectors. Traditional methods for avoiding seams involve sophisticated mechanical devices and expensive CRT projectors, coupled with extensive human effort for fine-tuning the projectors. This paper describes an automatic alignment method that relies on an inexpensive, uncalibrated camera to measure the relative mismatches between neighboring projectors, and then correct the projected imagery to avoid seams without significant human effort.

CR Categories and Subject Descriptors: I.4.0 [Image Processing and Computer Vision]: General - Image Displays; I.4.1 [Image Processing And Computer Vision]: Digitization and Image Capture - Imaging Geometry.

Additional Keywords: seamless tiling, automatic alignment, projective mapping, simulated annealing

1 Introduction

Rapid advances in computer technology have made data visualization widely available and indispensable. However, the resolution of the displays, the critical component in visualization, has been lagging far behind other key enabling technologies. Although flat panel devices have become cheaper and better, it is extremely difficult and expensive to produce really high-resolution, large-scale flat panels. Because of the drive electronics that is required on the sides of a panel display, it is also infeasible to put together multiple flat panels side by side without seams.

Tiling multiple projectors together is a viable way to build a high-resolution, high-brightness, seamless display. But, as such a display system scales beyond three or four projectors, aligning the projectors becomes a challenging issue. A multi-projector display might in principle be perfectly aligned by manual means just once; but in practice physical realities (vibration, lamp-changing, and so on) mean that re-alignment is frequently needed. Aligning the projectors by hand is a time-consuming task that requires both skill and experience. It also requires the use of either sophisticated mechanical devices or electron beam adjustment found only on expensive CRT projectors.

Alternatively one can employ computer vision and image processing techniques to digitally align the projectors. By warping the images, the computer can correct the projected imagery to account for the physical realities of misalignment [4, 16, 17]. In order to apply the correct amount of digital compensation, the computer has to first measure accurately the actual misalignment.

The research challenge here is to devise efficient algorithms without using expensive cameras and instruments. One could in theory use a camera with extremely high resolution to measure the misalignment. Unfortunately, such a device is hard to get and can cost even more than the display itself. Commodity video cameras are inexpensive and available everywhere, but they suffer from low resolution. Although calibrating the cameras overcomes this problem, the calibration process itself involves human labor and could become time-consuming.

We have developed a novel algorithm that uses an inexpensive and uncalibrated camera to align the projectors. Our method avoids camera calibration by taking only relative measurements — for instance, matching a few points and lines between a pair of projectors. Since the camera only has to make "binary" decisions regarding these measurements, it is free to zoom and pan arbitrarily close to the target spot to obtain highly accurate observations. Our alignment algorithm employs the simulated annealing technique to "stitch" the local observations together into a self-consistent global picture, and find a set of projection mappings that are consistent with the observations.

The rest of the paper describes in detail our automatic alignment method and our experimental results.

2 Background and Previous Work

Multi-projector displays have been around for more than two decades [9]. Previous systems employed expensive CRT projectors which have sophisticated mechanisms for adjusting image distortion and color balance. Aligning the projectors was typically done manually by trained technicians; the process often took hours. In recent years, portable presentation projectors based on LCD or Digital Micro-mirror Devices (DMD) have become increasingly cheaper, brighter, and more compact. There is new interest in building high-resolution displays out of these commodity projectors for office and research environments and for entertainment [14, 7, 15, 18, 19]. This has spurred several research projects that study seamless tiling of inexpensive LCD and DMD projectors [16, 4, 8, 5].

Existing alignment algorithms usually consist of two stages, camera calibration and geometric registration. In the first stage, one or more cameras are calibrated according to a fixed global coordinate system (either 2-dimensional or 3-dimensional). In the second

*Department of Computer Science, Princeton University, {yuqun,doug,af,li}@cs.princeton.edu

†Cinematographer, Visual Consultant, THousel@aol.com

stage, the calibrated cameras serve as measurement instruments to map pixels from each projector to the points in the global coordinate system.

Surati and Knight developed an algorithm that uses a camera to map the pixels from each projector to the points in a pre-established global screen coordinate system [16]. During the calibration stage, a camera is calibrated against a fine grid affixed to the display surface. The grid is physically drawn by a high-precision plotter. A mapping is established between pixels in the camera field and the physical points on the display surface. In the registration stage, each projector projects a regular grid onto the display surface. A computer vision algorithm accurately locates each projected grid point in the camera's field of view. Using the camera-to-display-surface mapping established previously, the projected grid point (or a pixel in the projector) is mapped to a physical point on the display surface with high precision. This method works well for a small-scale display wall. It can deal with arbitrary distortions of the projectors. However, using an absolute measurement grid to calibrate the camera prevents this method from scaling for a large display wall; it is problematic whether one can generate a physical or project a virtual measurement grid that is large enough but still has fine precision.

Raskar *et al.* attempted to solve a general case in which the display surface can be arbitrarily complex, for example, the corner of a wall, or a curved screen [4]. This requires registration of the 3D surface geometry of the screen surface as well as registration of the projected pixels on the display surface. Their algorithm uses known 3D objects such as painted boxes to calibrate the extrinsic and intrinsic parameters of a few fixed cameras. Two calibrated cameras that overlap in their fields of view can observe the same mesh pattern displayed by a projector. The observations from both cameras are correlated using the stereo vision technique; from this correlation the exact location of a projected pixel on the display surface is derived. The location information is in the 3D space and therefore also reveals the surface contour of the screen. The drawback of this approach is again the requirement of camera calibration.

Our work differs from previous work by the use of an uncalibrated camera to observe misalignment among the projectors. The use of camera calibration implies that the cameras themselves must be fixed and cannot pan and zoom during measurement, for otherwise camera parameters will have to calibrated continuously. It also requires setting up known objects such as as fine-plotted grid and regular 3D objects. Avoiding camera calibration can greatly minimize the amount of human involvement and equipment required in multi-projector alignment.

3 The Automatic Alignment Algorithm

Our automatic alignment algorithm consists of two stages. In the *misalignment measurement* stage, the camera observes geometric relationships – point matches and line matches – between adjacent projectors. In the *alignment computation* stage, we set up a multi-dimensional global minimization problem whose constraints are those observations. The minimization process produces a set of alignment correction functions, one for each projector, that maintain C^0 and C^1 continuities across the projectors. Figure 1 shows a schematic of our alignment system. Our algorithm assumes that projectors are already roughly aligned. This is a reasonable assumption, because coarse alignment can be easily accomplished with an inexpensive projector rack and some manual adjustment. Essentially we assume that the projectors are not so badly misaligned that computational alignment is impossible.

3.1 Projection function

Before getting into the details of our alignment algorithm, let us first review the mathematical representation for a multi-projector

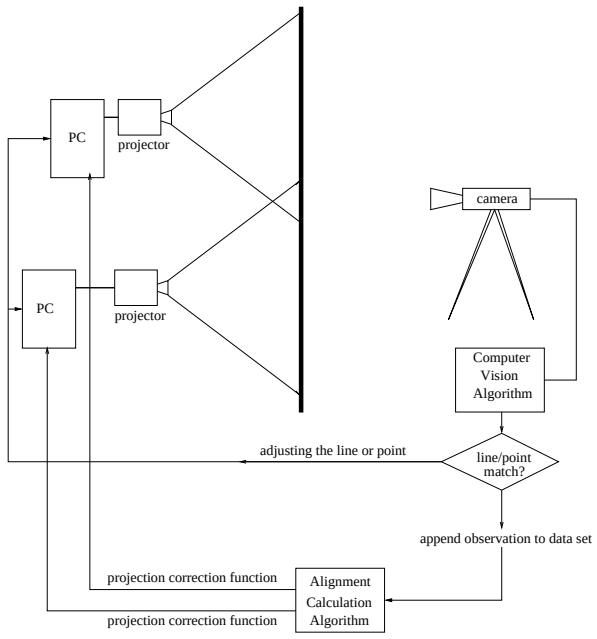

Figure 1: Camera-based Alignment Data Collection

system. Projection can be thought of as a mapping between pixels in projector space (x, y) and the illuminated dots (u, v) on the global display space. This mapping, or the *projection function*, is normally accomplished with a lens system.[1] Figure 2 shows a conceptual diagram of a typical lens system. The projection function can be decomposed into two parts, the projective transformation P and the non-linear distortion D:[2]

$$(u, v) = (D_u(P_u(x, y), P_v(x, y)), D_v(P_u(x, y), P_v(x, y)))$$

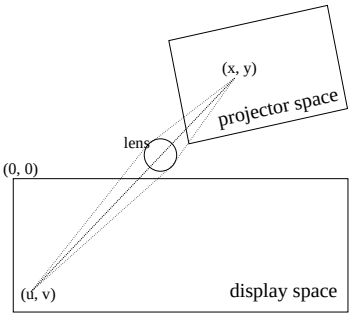

Figure 2: Conceptual diagram of a projection system

The projective transformation can be expressed by a 3x3 projection transformation in homogeneous coordinates using 8 free parameters $(m_{ij}),\ i = 1..3,\ j = 1..3$ [1]:

$$\begin{aligned} P_u(x, y) &= \frac{m_{11} \cdot x + m_{12} \cdot y + m_{13}}{m_{31} \cdot x + m_{32} \cdot y + 1} \\ P_v(x, y) &= \frac{m_{21} \cdot x + m_{22} \cdot y + m_{23}}{m_{31} \cdot x + m_{32} \cdot y + 1} \end{aligned} \quad (1)$$

[1]The algorithm presented here could in principle be applied to curved display surfaces as well, in which case, a 2D parametric coordinate system can be used o the display surface.

[2]This is only an approximation to an actual projection device, as it ignores several distortion effects such as color dispersion.

The radial distortion with respect to an optical center (c_x, c_y) and a distortion parameter ρ is given as

$$\begin{aligned} D_u(u,v) &= u + \rho \cdot (u - c_u) \cdot d^2 \\ D_v(u,v) &= v + \rho \cdot (v - c_v) \cdot d^2 \\ \\ d &= \sqrt{(u-c_u)^2 + (v-c_v)^2} \\ (u,v) &= (P_u(x,y), P_v(x,y)) \\ (c_u, c_v) &= (P_u(c_x, c_y), P_v(c_x, c_y)) \end{aligned} \quad (2)$$

The implementation described in the paper only considers the projective component. However, the algorithm itself can in principle deal with non-linear distortions as well.

3.2 Alignment measurement

The alignment algorithm observes two types of inter-projector relationships: point matches and line matches. A point match simply states that a pixel (x, y) from one projector locates at the same spot on the display surface as another pixel (x', y') from an adjacent projector.[3] A line match means that a projected line from one projector is co-linear with another line from the neighboring projector.

The rationale for using point matches is simple: if there are a lot of point matches between any pair of adjacent projectors, the result of alignment computation will yield a set of projection functions that maintain C^0 continuity across projectors. However, the point matches themselves are insufficient to constrain the system. This is particularly so when the projectors overlap only a small portion of their screens, as in a typical display wall system. The line matches provide further shape constraints, as well as C^1 continuity across the projectors. Given a sufficient number of point and line matches, there is enough information to figure out the relative position and the orientation of the projectors.

The line and point matches can be obtained automatically with a camera attached to a computer. Figure 3 contains a brief sketch of the measurement algorithm that obtains a point match between point P from projector A and a point Q from projector B. It is essentially a negative feedback loop. The feedback parameter ρ in the algorithm is determined before hand eiher manually, or automatically by measuring the distance between two pixels in the camera. The algorithm to obtain line matches works in a similar fashion. Each line match consists of a point match between the inner ends of the two line segments and a match between the slopes of the two line segments.

```
1. pan the camera to roughly center its FOV on pixel P
2. display a cross centered at P on projector A
3. measure P's location L in the camera
4. take a guess of a pixel Q in projector B
5. loop
6.       display a cross centered at Q on projector B
7.       measure Q's location L' in the camera
8.       if ||L, L'|| > ε
9.          Q = Q + ρ · (L − L')
10.      else
11.         return (P, Q)
```

Figure 3: pseudo-code for obtaining a point match

[3] Note here that we use the fact that adjacent projectors overlap by a small amount.

3.3 Alignment computation

We formulate the problem of figuring out a set of projection functions as a global minimization problem whose constraints are derived from point and line matches. It is straightforward to obtain an initial guess for each projector's position shifts, horizontal and vertical, based on the point match data. Since the projectors are already roughly aligned, this guess produces a good starting point that is close to the globally optimal solution for our problem.

The constraints for the global minimization problems are produced as follows. Assuming (P_u^i, P_v^i) is the projection function for projector i, each point match between a pixel $p_1 = (x_1, y_1)$ from projector 1 and a pixel $p_2 = (x_2, y_2)$ from projector 2 results in a Euclidean distance error E^p on the display surface:

$$\begin{aligned} (u_i, v_i) &= (P_u^i(x_i, y_i), P_v^i(x_i, y_i)) \\ E^p(p_1, p_2) &= (u_1 - u_2)^2 + (v_1 - v_2)^2 \end{aligned} \quad (3)$$

Each line match between two line segments $l_i = \overline{p_{11}, p_{12}}$ and $l_i = \overline{p_{21}, p_{22}}$ produces a point-match error and an error based on the angle between two line segments:

$$E^l(l_1, l_2) = \max(E^p(p_{12}, p_{21}), (\angle\overline{p_{11}p_{12}}, \overline{p_{21}p_{22}})^2)$$

Note that the error term $(\angle\overline{p_{11}p_{12}}, \overline{p_{21}p_{22}})^2)$ is also computed in the global display space. The goal is to minimize the maximum of the errors over all line and point matches.

Global minimization over a large number of continuous variables remains a tough problem. However, several effective methods do exist. We chose Simulated Annealing [10, 6, 11] as the minimization method. It has been used successfully in many scientific computations with hundreds and even thousands of continuous variables. This technique is a generalization of a Monte Carlo method for examining the equations of state and frozen states of n-body systems [10]. It mimics the manner in which metals recrystallize in the process of annealing. Among several publicly available implementations, we chose the one provided by *Numerical Recipes in C* [11]. The state evaluation function required by the annealing method is the error function that we just described.

The simulated-annealing method requires representing each projection functions using a vector of continuous variables. During our initial trials, we discovered that an arbitrary projective transformation matrix may not correctly describe a realistic projection device, because a projective matrix allows shear deformation that an actual projector cannot produce. Instead of trying to compute the 8 free parameters in the projection matrix (Equation 1), our minimization routines computes the extrinsic and intrinsic parameters of each projector. A projector can be modeled with 9 parameters, X, Y, Z positions of the projector, its rotations along the three axes, its focal length, and its optical center (c_x, c_y). The projective function can be uniquely derived from these parameters [1]. Given N projectors in the display wall system, alignment computation amounts to minimizing the error function over $9N$ continuous variables (or $10N$ if radial distortion is also included). The total degree of freedom in this problem is quite reasonable for the simulated annealing method, as our experiments will shortly show.

3.4 Computational re-alignment

Having the mapping function for a projector, we can apply it to correct the imagery displayed by that projector, simply by re-sampling the image. Given a projector's mapping function (P_u, P_v), and an image source $I_s(u, v) = (r, g, b)$, we obtain the intensity value I_p for a particular pixel (x, y) using the formula

$$I_p(x, y) = I_s(P_u(x, y), P_v(x, y)) \quad (4)$$

Many studies have been done on efficient re-sampling of an image. One can use the MMX instructions on a Pentium processor to sample multiple pixels at once. Another interesting approach is to leverage the capability of graphics accelerators. Raskar *et al* described a method using the texture-mapping hardware found on most graphics accelerators [13, 12]. Recently ComView Visual Systems has introduced an ASIC solution that provides both geometric correction and color balancing for multi-projector display systems [17].

3.5 Discussion

A salient feature of the alignment algorithm just described is that it avoids camera calibration. No human involvement is required to take misalignment measurement other than placing the camera(s) in front of the display wall. This feature is made possible by taking only relative measurements, *i.e.*, point and line matches. Such observations require only local and "binary" decisions that any inexpensive camera will do.

Measuring only the point and line matches also makes it easy to overcome the resolution limitation of an off-the-shelf camera. One can simply pan and zoom the camera arbitrarily close to the display surface, or place multiple fifty-dollar cameras close to the display surface. Highly accurate measurements, finer than a pixel, are easily obtained this way. Unlike methods based on camera calibration, this new method is insensitive to change of camera parameters during the zoom and pan motions, and can easily employ many cameras at no additional complexity.

Our alignment algorithm is much less sensitive to camera distortions than previous methods. For example, it can tolerate any kind of camera distortion while taking point matches, provided that the camera remains steady during measurement. The only additional requirement for obtaining a line match is that camera's field of view (FOV) is free from non-linear distortions – as long as a straight line on the display surface shows up straight in the camera's FOV.[4] A naive way to meet this requirement is to use the central area of the FOV. A sophisticated solution is to center the FOV on the adjoining ends of the two matching line segments, such that the line segments pass the optical center of the camera's FOV. This makes the entire camera's FOV usable even in the presence of radial distortion, because a straight line passing through the center of a camera FOV is not bent by radial distortion.

The drawback of our algorithm is that it relies on a global minimization technique that gives no convergence guarantee. Although the non-convergence situation has not occurred in our experiments, we remain interested in finding a deterministic and more efficient method to calculate the projection functions.

4 Implementation and Results

In this section we evaluate the effectiveness of our alignment algorithm with empirical results as well as a simulation study.

4.1 Experiment platform

We conducted experiments on our rear-projection display wall. It consists of 8 Proxima 9200 LCD portable projectors in a 2x4 arrangement [7]. Each projector is capable of displaying a true resolution of 1024x768. Adjacent projectors overlap between 40 and 70 pixels. The total resolution that our display wall offers comes out to about 3800 pixels wide and 1500 pixels high. The display surface consists of two pieces of black screens, made by Jenmar, Inc., and is 18 feet wide and 8 feet tall.

The projectors are mounted on mechanical positioning tables that have 6 degrees of freedom. These tables are normally used for time-consuming manual alignment of the projectors. But in our experiments, they provide us with an easy means to "mis-align" the projectors with arbitrary rotations and translations.

Driving each projector is a 450 MHz Pentium-II PC with an Intergraph graphics accelerator. The PCs are interconnected by two networks: 100 Mb fast Ethernet and Myrinet. Communication within this PC cluster can leverage the very high bandwidth and low latency offered by User-level Virtual-Memory-Mapped Communication over the Myrinet system-area network [3, 2].

We place a Canon VC-3 video conferencing camera at an 8-feet distance away in front of the display wall. This camera has motorized pan, tilt, zoom and focus, all controllable through the serial port. We wrote our software in Python and C that controls the camera to gather misalignment observations. Video digitization is done by an Integral Video Grabber card.

4.2 Empirical results

Misalignment measurement: On our 8-projector display wall, it takes 33 minutes to collect the point and line matches over a total of 10 overlapped regions. For each pair of adjacent projectors, 10 point matches and 6 line matches are observed. A large amount of time is spent in panning and tilting the camera to zoom onto a spot. Using multiple cameras, each responsible for a sub-area of the display wall, can reduce the measurement time proportionally. Besides, there is no need to correlate observations from different cameras.

The quality of alignment computation depends critically on the accuracy of the point and line matches. In our experimental setup, the camera can easily distinguish two adjacent pixels from a projector. We use nearest-neighbor fit to match two pixels and two lines. This imples that the worse measurement error for a point match is a half pixel. A more sophisticated algorithm such as weighted average could be used to increase the measurement accuracy.

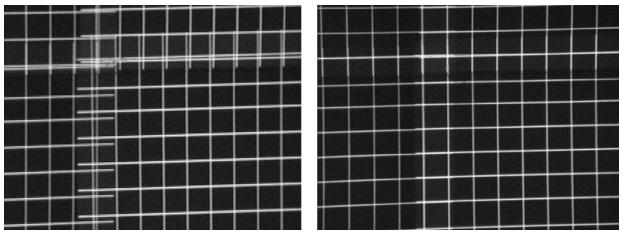

Figure 4: Aligning a grid: before and after pictures

Alignment computation: We run the alignment computation algorithm on an 833 MHz Pentium III PC with Rambus memory. Table 1 shows the time and quality in the alignment computation, as the number of annealing steps increases, for various projector configurations on our display wall. The quality is expressed in terms of the maximum error between two points in a point match (in pixels). The pixel-level error can be largely attributed to the error in the measurement. Figure 4 shows the zoomed-in view of actual alignment result of a grid pattern on our 2x4 multi-projector configuration. The color plate shows more results for the 2x4 projector configuration, all are obtained from the 5,000-step alignment computation.

The quality of simulated annealing depends on the number of steps in the annealing process. The more steps taken, the more gradual the annealing process is and usually the better the alignment result. Figure 5 plots the alignment accuracy as a function of total annealing time for a few projector configurations on our display wall. The improvement of accuracy is very gradual, as the number of annealing steps increases; in a few cases, the accuracy

[4] This is less stringent than requiring the camera FOV be a linear field.

configuration	annealing steps									
	1,000		2,000		5,000		10,000		20,000	
	error (pixels)	time (min)	error (pixels)	time (min)	error (pixels)	time (min)	error (pixels)	time (min)	error (pixels)	time (min)
1x4	0.96	1.2	0.76	2.4	0.77	6.1	0.89	11.9	1.19	24.8
2x2	1.25	1.5	1.38	3.0	1.32	7.8	1.14	15.4	1.10	30.9
2x3	1.27	1.7	1.27	3.5	1.27	8.7	1.42	17.3	1.12	34.6
2x4	1.49	1.8	1.44	3.5	1.32	8.8	1.50	18.1	1.35	35.7

Table 1: Alignment accuracy and time for various configurations

configuration	annealing steps											
	1,000		2,000		5,000		10,000		20,000		50,000	
	error (pixels)	time (min)	error (pixels)	time (min)	error (pixels)	time (min)	error (pixels)	time (min)	error (pixels)	time (min)	error (pixels)	time (min)
1x4	0.47	1.2	0.63	2.3	0.88	5.8	1.23	11.4	0.38	23.3	0.78	59.6
1x8	0.38	1.2	0.43	2.5	0.53	6.2	0.57	12.6	0.74	25.2	0.62	63.3
2x2	0.59	1.5	0.74	2.9	1.07	7.4	0.73	15.2	1.11	32.1	0.95	78.2
2x4	0.55	1.7	0.42	3.4	0.60	8.5	0.52	17.2	0.35	34.5	0.61	86.7
3x3	1.53	1.8	0.52	3.6	0.50	8.9	1.29	17.9	0.43	36.1	0.66	91.2
3x6	3.98	1.8	2.32	3.7	1.35	9.0	1.52	18.5	0.96	37.3	0.99	94.9
4x6	11.95	1.6	11.83	3.6	6.85	9.3	2.91	18.6	1.00	38.0	0.83	95.5
4x8	12.93	1.6	11.76	3.5	9.80	8.9	8.03	17.9	6.99	36.4	3.15	95.7

Table 2: Alignment computation results on simulation data

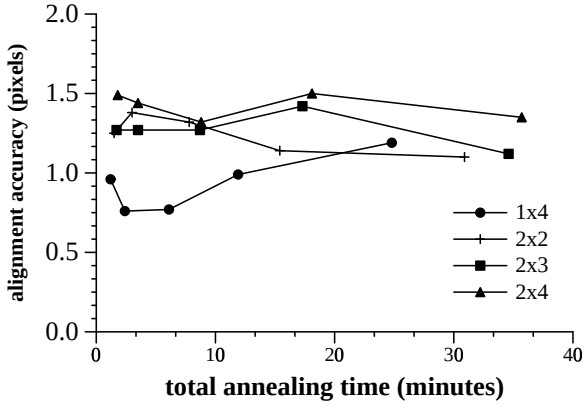

Figure 5: Alignment computation accuracy vs. total annealing time

actually worsens. The very slow improvement could very well be the nature of the simulated annealing technique. The measurement errors and non-linear distortions in the projectors also contribute to the final errors in the alignment computation. Section 4.3 confirms this with error-free measurements from simulated display wall configurations.

Color plate: On the color plate there are three pairs of images taken of our 2x4 display wall. Each pair consists of a pattern on the misaligned display wall (the "before" shot) and the pattern corrected by our automatic alignment algorithm (the "after" shot). Due to the limited resolution of our digital camera (a Nikon Coolpix 900), we only took pictures at the intersections of four projectors, instead of providing a high-detailed image of the entire display. Figure 1 on the color plate shows the before and after pictures of a grid pattern. Figure 2 shows the zoomed-in views. Figure 3 demonstrates a high-resolution map being corrected aligned by our automatic algorithm. Note that we have purposely disabled edge-blending which is designed to soften the transitions between adjacent projectors. This is to allow the readers to examine alignment and misalignment in detail.

4.3 Simulation results

In order to evaluate the scalability of our alignment algorithm, we wrote a simple simulator that generates misalignment observations for an arbitrarily configured multi-projector display. The simulation assumes that after rough adjustment of the projectors, imperfect position and orientation of a projector (total 6 degrees of freedom) contribute 10 pixels of misalignment, independently. The variation in zoom distance and focal length is 5 %. In other words, our simulation assumes that the projectors are roughly aligned – a quite realistic assumption based on our experience. It is these 10-pixel variations that our automatic alignment algorithm tries to eliminate. Table 2 shows the quality of alignment computation for various simulated configurations.

The simulation results confirm that our algorithm can deal with a variety of projector configurations and misalignment situations, for up to 24 projectors, with satisfactory results. Unlike in the experiments, the measurements in the simulation study are precise. The effect of error-free measurements is manifested as generally higher alignment accuracy for the same projector configuration than in the actual experiments.

The alignment computation time that is required to achieve certain quality, i.e., sub-pixel alignment, generally increases with the number of projectors in the system. Half an hour is sufficient for getting a sub-pixel alignment result on configurations varying from 1x4 to 4x6. However, the alignment computation yields a noticeable error of 3 pixels for the 4x8 configuration even after 96 minutes of computation. For a very large system with 50 or 100+ projectors, the annealing will certainly take hours to achieve alignment at the single pixel level. This is the drawback of our algorithm. A possible solution is to parallelize the computation using the PCs that drive the projectors and the fast network that connects these PCs together. The computational resource in our system scales with the number of projectors. But the challenge is to parallelize the annealing algorithm. We are currently investigating this approach.

5 Conclusions

In this paper, we described an automatic algorithm to align a multi-projector display. Our algorithm uses an uncalibrated camera to ob-

tain the inter-projector misalignment information. It then employs a global minimization technique, simulated annealing, to figure out good correction functions to counter the effect of physical misalignment.

We implemented and experimented our automatic alignment algorithm for an 8-projector display wall. The experimental results show that our algorithm works well in the real setting. We also evaluated our algorithm using data from simulated multi-projector display systems. The simulation results show that our method works for a system with up to 24 projectors. But for systems with more projectors, the simulated annealing process takes a very long time to converge.

Our alignment algorithm takes only relative measurements that require no camera calibration. It solves the inherent tension between the relatively low resolution of a camera and the very high resolution of a scalable display wall, by allowing the camera to freely zoom in on a measurement target. The result is a highly accurate computational alignment among projectors. In addition, since no camera calibration is required, our algorithm can be fully automated. Although our algorithm currently ignores the radial distortion of a projector, the global minimization technique can be adapted to solve for the radial distortion parameters.

There is some room left to improve the speed of alignment data collection and alignment computation, as we have not yet tried to optimize these processes. Two possible ways to improve our algorithm are (1) using multiple uncalibrated cameras to speed up the data collection process, and (2) parallelizing the annealing computation over the PC cluster and the ultra-fast network that together drive the display wall.

Acknowledgments

This project is part of the Princeton Scalable Display Wall project which is supported in part by Department of Energy under grant ANI-9906704 and grant DE-FC02-99ER25387, by an Intel Research Council and Intel Technology 2000 equipment grant.

We would like to thank Kenneth Steiglitz who discussed with us the alignment algorithm, and Liming Wang who proof-read the final version of the paper. We also would like thank the reviewers who provided much useful feedback on improving this paper.

References

[1] Yuqun Chen, Douglas Clark, Adam Finkelstein, and Kai Li. A method to align high-resolution multi-projector displays using an uncalibrated camera. Technical report, Princeton University, Computer Science Department, March 2000.

[2] Yuqun Chen, Stefanos N. Damianakis, Sanjeev Kumar, Xiang Yu, and Kai Li. Porting a user-level communication architecture to nt: Experience and performance. In *3rd Usenix Windows NT Symposium*, July 1999.

[3] Yuqun Chen, Czarek Dubnicki, Stefanos N. Damianakis, Angelos Bilas, and Kai Li. Utlb: A mechanism for translations on network interface. In *The 8th International Conference on Architectural Support for Porgramming Languages and Operating Systems (ASPLOS-VIII)*, October 1998.

[4] Raskar et al. Multi-projector displays using camera-based registration. In *IEEE Visualization '99*, 10 1999.

[5] Volodymyr Kindratenko. Compute vision guided cross-projector color alignment on multi-projector displays. In *The Fourth International Immersive Projection Technology Workshop*, June 2000.

[6] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[7] Kai Li, Han Chen, Yuqun Chen, Douglas W. Clark, Perry Cook, Stefanos Damianakis, Georg Essl, Adam Finkelstein, Thomas Funkhouser, Allison Klein, Zhiyan Liu, Emil Praun, Rudrajit Samanta, Ben Shedd, Jaswinder Pal Singh, George Tzanetakis, and Jiannan Zheng. Building and using a scalable display wall system. *Computer Graphics and Applications*, 20:29–37, 2000.

[8] Aditi Majumder, He Zhu, Herman Towles, and Greg Welch. Color matching of projectors for multi-projector displays. In *EuroGraphics 2000*, volume 19, August 2000.

[9] Theo Mayer. New options and considerations for creating enhanced viewing experiences. In *Computer Graphics*, pages 32–34, May 1997.

[10] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[11] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*, chapter 10, pages 444–455. Cambridge University Press, 2 edition, 1992.

[12] R. Raskar. Immersive planar display using roughly aligned projectors. In *IEEE Virtual Reality 2000*, March 2000.

[13] R. Raskar, M. Cutts, G. Welch, and W. Stuerzlinger. Efficient image generation for multiprojector or multisurface displays. In *9th EuroGraphics Rendering Workshop*, 1998.

[14] Ramesh Raskar, Greg Welch, Matt Cutts, Adam Lake, Lev Stesin, and Henry Fuchs. The office of the future: A unified approach to image-based modeling. In *SIGGRAPH 98*, pages 179–188, July 1998.

[15] Daniel R. Schikore, Richard A. Fischer, Randall Frank, Ross Gaunt, John Hobson, and Brad Whitlock. High-resolution multiprojector display walls. *Computer Graphics and Applications*, 20:38–44, 2000.

[16] Rajeev Surati. *A Scalable Self-Calibrating Technology for Large Scale Displays*. PhD thesis, MIT, 1999.

[17] ComView Visual Systems. Comview viewscreen and viewmaestro product sheets. *http://www.comview-vs.com*.

[18] Bin Wei, Claudio Silva, Eleftherios Koutsofios, Shankar Krishnan, and Stephen North. Visualization research with large displays. *Computer Graphics and Applications*, 20:50–54, 2000.

[19] Greg Welch, Henry Fucks, Ramesh Raskar, Herman Towles, and Michael S. Brown. Projected imagery in your office of the future. *Computer Graphics and Applications*, 20:62–67, 2000.