Multi-projectors for arbitrary surfaces without explicit calibration nor reconstruction

Jean-Philippe Tardif Sébastien Roy Martin Trudeau
Département d'informatique et recherche opérationnelle
Université de Montréal
{tardifj, roys, trudeaum}@iro.umontreal.ca

Abstract

This paper presents a new approach allowing one or more projectors to display an undistorted image on a surface of unknown geometry. To achieve this, a single camera is used to capture the viewer's perspective of the projection surface. No explicit camera and projector calibration is required since only their relative geometries are computed using structured light patterns. There is no specific constraint on the position or the orientation of the projectors and the camera with respect to the projection surface, except that the area visible to the camera must be covered by the projectors. The procedure defines a function establishing the correspondence of each pixel of a projector image to a pixel of the camera image. After the mapping of each projector has been carried out, one can display an image corrected in real-time for the point of view of an observer, which takes into account his position, the surface distortion, and the projector position and orientation. This method automatically takes into account any distortion in the projector lenses. Typical applications of this method include projection in small rooms, shadow elimination and wide screen projection using multiple projectors. Intensity blending can be combined to our method to ensure minimal visual artifacts. The implementation has shown convincing results for many configurations.

1. Introduction

Recently, augmented reality has been undergoing a very significant growth. It is envisioned that three dimensions video-conferencing, real-time annotation and simulation will be widely used in the near future [1, 2]. Coupled with increasing computing power, the improvement of electronic frame grabbers and projectors adds even more possibilities. For instance, a virtual world could be displayed through projectors on the walls of a room to give someone a sense of immersion. Projection of an x-ray over a patient's body could help a physician to get more accurate information about a tumor's location, or simple information about the patient's condition could be displayed in his visual field. Many projectors have to be used to cover the whole environment or to prevent occlusions from people or objects. In addition, the projected images have to take into account the observer's position inside the room and the projection surface geometry, as illustrated in figures 1, so the observer's perception is not different from the ideal setup of figure 2. Still, immersing experiences are difficult to implement because image projection in various environment is hard to achieve. This is due to the wide range of screen geometry.

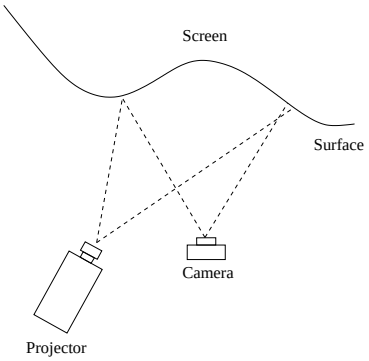


Figure 1. Top view of a single projector setup.

The projection problem can be divided into three main subproblems. First, the image has to be corrected with respect to the screen geometry and the observer's position. Second, multiple projector calibration and synchronization must be achieved to cover the observer's vision field. This includes color correction, intensity blending and occlusion detection. Last, illumination effects from the environment have to be considered and corrected if possible.

Many articles propose methods for solving parts of this problem in different contexts. Systems for projecting over non-flat surfaces already exist. It has been demonstrated that once the projectors are calibrated, texture can be painted over objects whose geometry is known [3]. This result has been confirmed with non-photorealistic 3D ani-

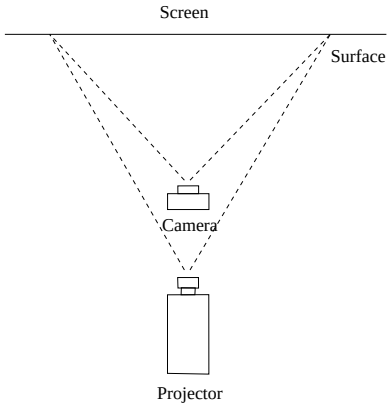


Figure 2. Perception after correction. The camera modeling the observer and the projector are located in the same position and their axis is perpendicular to the screen.

mations projected over objects [4]. Unfortunately, getting projector parameters is not always simple. For example, hemispherical lenses cannot be described with typical matrix formulation. Also, some applications need very fast surface reconstruction which remains today very challenging. Among reconstruction methods are structured light techniques [5, 6, 7] while stereo based systems with landmark projection over the surface offer a simple way to get the 3D geometry of the surface with triangulation. Of course, camera calibration is a prerequisite in each case [8].

When assumptions are made, simpler approaches can be used to correct the images. When the screen is assumed flat, keystoning rectification allows the projector and the observer to be placed at an angle relative to the surface [9, 10]. In this case, a camera and a tilt sensor mounted on the projector, or a device tracking the person is needed. Real-time correction is possible with video card hardware acceleration.

Multi-projector systems are of two types. The first is a large array of projectors which requires calibration and synchronization [11, 12]. As many as 24 projectors can be used together to cover very large screens with high resolution. In all cases, affine matrices are obtained for each projector during a calibration process with cameras. Intensity blending is later used to get uniform illumination over the surface. Effective methods to synchronize a large number of projectors are also presented by the authors. The second type of system does the same in the more general context of augmented reality where the surfaces are not necessarily flat. The process stays essentially the same except that surface reconstruction is needed. Intensity blending can then be used [8].

Real-time algorithms for correcting shades produced by a person or an object placed in front of a projector exist in the literature [13, 14]. The authors rely on other projectors to fix the image on the screen.

Finally, some researchers were interested in the problem of color calibration of multi-projectors [15]. They present a way of correcting the images according to the photometric characteristics of the projectors. A generalization of that method for arbitrary surfaces combined with the geometry correction presented here would yield a perfectly corrected image accounting for different projector characteristics and also surface orientation with respect to the observer.

The next sections introduce a new image correction scheme for projecting images undistorted from the observer's point of view on any given surface. The method exploits structured light to generate a mapping between a projector and a camera. As opposed to the state of the art system [8], **no explicit calibration** of the projector and the camera is needed. Thus, it is a step forward into the simplification of the problem of projecting over non-flat surfaces. Since *calibration* commonly means getting internal and external parameters of the devices, we'll use the word *initialization* for the building the correction function throughout the paper to avoid confusion. Finally, it is shown how this can be used for a multi-projector system.

2. Single projector

In order to project an image on a screen, some assumptions must be made. In general, the screen is considered flat and the projector axis perpendicular to it. Thus, minimal distortions appear to an audience in front of the screen. Notice that the assumptions involve the relative position and orientation of the observer, the screen and the projector (see figure 1). Those constraints cannot be met when an arbitrary projection surface is used. In this case, information about the projector-viewer-screen system has to be determined dynamically to correct projected images to avoid distortion. The approach commonly used starts by finding a function between the observer and the screen, and another one between the projector and the screen. These requires calibrating the camera and projector as

$$\begin{pmatrix} s \\ t \\ 1 \end{pmatrix} \cong F_p \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix} \quad \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \cong F_c \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix}$$

where $\cong$ represents equivalence up to a scale factor. The projective points $(s,t,1)^T$, $(u,v,1)^T$ and $(x_w,y_w,z_w,1)^T$ are the projector image, camera image and world coordinates respectively. 3D world points are related to projector image points by a 3×4 matrix F_p and are related to camera image points by F_c. The world coordinates are known from landmarks located on a calibration object. Then, the

image coordinates are identified by getting the position of those landmarks in the camera image and the projector image. After that, F_p and F_c can be used to reconstruct the screen geometry using a combination of structured light and triangulation [8].

Another approach, limited to a flat projection surface, involves the use of homographies to model the transformations from image and projector planes to the screen. The main advantage of homographies is their representation by 3×3 invertible matrices defined as

$$\begin{pmatrix} s \\ t \\ 1 \end{pmatrix} \cong H_p \begin{pmatrix} x_s \\ y_s \\ 1 \end{pmatrix} \quad \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \cong H_c \begin{pmatrix} x_s \\ y_s \\ 1 \end{pmatrix}$$

where $(x_s, y_s, 1)^T$ is an image point in the screen coordinates system, $(s, t, 1)^T$ is in the projector's and $(u, v, 1)^T$ in the camera's. From H_p and H_c, a relation between the projector to the camera can be established:

$$\begin{pmatrix} s \\ t \\ 1 \end{pmatrix} \cong H_p H_c^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}$$

This mapping is invertible and provides a linear mapping [16]. Although this approach is not directly useful when the screen is non-flat, the interesting idea is that in practice, the function $H_p H_c^{-1}$ is directly found, so that no function is defined w.r.t the screen (H_p and H_c). We propose a generalization of this idea replacing $H_p H_c^{-1}$ by a piecewise linear mapping function R relating the camera and the projector directly. If R is invertible, we can compensate for an arbitrary observer-camera-projector setup. The image correction process is illustrated in figure 3. The next sections explain each step in details.

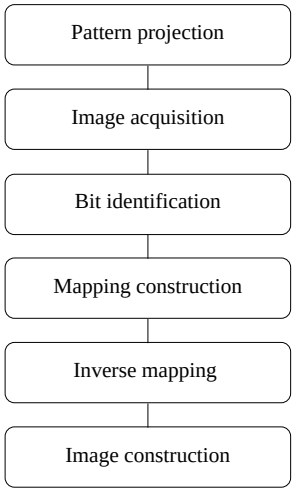


Figure 3. Image construction process corrected for a specific projector-observer-screen configuration.

Figure 4. Projection patterns for bits $b = 4, 3, 2, 1$. (a) patterns used to obtain R_b^s, (b) patterns used to obtain R_b^t. Inverse patterns are not shown.

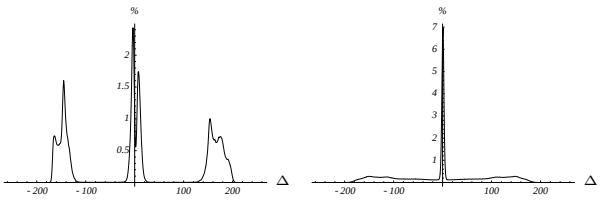
2.1. Pattern projection

Structured light is commonly used in the field of 3D surface reconstruction. Using calibrated devices and a mapping between the camera and the projector, reconstruction can be achieved quite easily [6]. However we will show that even though the camera and projector are not calibrated, a mapping between the camera and the projector is feasible as long as no full 3D reconstruction is needed. Simple alternate black and white stripes are used to build a correspondence from a point of the camera to a coordinate in the projector one bit at a time. For instance, for a n bit coordinate encoding, each bit b ($b \in \{1, ..., n\}$) is processed individually and yields an image of coded stripes, each of width 2^{b-1} pixels. The concatenation of all bits provides the complete coordinates. Figure 4 gives an example of the coded projector images. Many coding schemes are possible. Some try to increase noise resistance (gray code), and some other try to reduce the number of patterns in order to speedup the scanning process (colors, sinus). Since our method is designed to provide high noise robustness and since our implementation is not optimized for speed, we use the simplest possible pattern. As illustrated in figure 4, it consists of two set of horizontal and vertical stripes encoding s and t coordinates.

If a partial knowledge of the projector and camera relative position is available, then a single stripe orientation can be derived from the epipolar geometry. Assuming no such knowledge, we use two orientations to accommodate arbitrary geometries.

2.2. Image acquisition

In order to compute a mapping function R from (u, v) to (s, t), we first decompose it into partial mapping functions R_b^s and R_b^t, mapping the bit b of the s and t coordinates respectively. These mapping functions are built by



(a) Bit 10. (b) Bit 2.

Figure 5. Histograms of Δ values. (a) large stripes yield very good separation, (b) small stripes are hard to differentiate.

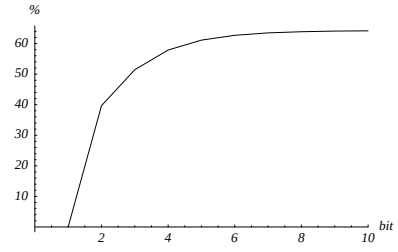


Figure 6. Percentage of usable pixels recovered from different stripe widths. The bit number represents the lowest bit used in the encoding. The number of usable pixel decreases as the low order bits get used, until merely none can be found. The maximum percentage value approximates the camera image coverage by the projector.

observing with the camera the projection of the corresponding stripe image and its inverse. Stripe identification is done with pixel by pixel difference between the image and its inverse, yielding Δ^s and Δ^t values between -255 and 255.

Figure 5 gives examples of histograms of Δ values. From these histograms, we find a rejection threshold τ that is going to be used to define which values are usable or rejected. Although several approaches could be used to select automatically the threshold τ, we use an empirical value of 50. This is possible because the method is designed to tolerate rejected points. From this threshold, values are classified into three groups: 0,1 and rejected (see equation 1). This test preserves only the pixels for which we can tell with confidence that the intensity fluctuates significantly between the inverse images.

Point rejection occurs for two main reasons. First, a camera point might not be visible from the projector. Second, the contrast between the inverse stripes is too small. This occurs when the screen color is too dark or because of the limited resolution of the camera. In that case it causes two stripes of different colors to be projected onto the same camera pixel. This especially happens at borders between stripes. Thus, as the the number of alternate stripes gets higher, the number of rejected pixel increases so much that we have observed that bit 1 and 2 (1 and 2 pixels stripes) are generally useless (see figure 5(b)).

The bit mapping R^s_b can now be defined as:

$$R(u,v)^s_b = \begin{cases} 0 & \text{if } \Delta^s(u,v) < -\tau \\ 1 & \text{if } \Delta^s(u,v) > \tau \\ rejected & \text{otherwise} \end{cases} \quad (1)$$

where b the bit number and Δ^s values are the inverse vertical stripes difference. Exactly the same process using horizontal stripes defines $R(u,v)^t_b$ from Δ^t values. When a pixel is rejected, it will not be used anymore for the rest of the algorithm.

2.3. Mapping construction

To obtain a complete mapping R from camera to projector, we concatenate the bit functions R^s_b and R^t_b to get

$$R(u,v) = \begin{cases} \overbrace{R(u,v)^s_n ... R(u,v)^s_1}^{s}, \overbrace{R(u,v)^t_n ... R(u,v)^t_1}^{t} \\ \quad \text{if} \quad R(u,v)^s_b \in \{0,1\} \; \forall \, b \in \{1,..,n\} \\ \quad \text{and } R(u,v)^t_b \in \{0,1\} \; \forall \, b \in \{1,..,n\} \\ rejected \\ \quad \text{otherwise} \end{cases} \quad (2)$$

As mentioned before, acquiring partial functions $R(u,v)^s_b$ and $R(u,v)^t_b$ for low order bits b is generally impossible and equation 2 rejects points for almost all (u,v) coordinates. Starting from the highest order bit, we observe in figure 6 that the percentage of usable pixels drops as lower order bits are used. Unfortunately, at some point, the percentage drops significantly (in figure 6: below bit three). Consequently, we will use a mapping on a number of bits n' sufficiently small so the number of usable points is not too small compared to the highest percentage. The $n - n'$ unused bits are set to 0 yielding a new mapping R' define as

$$R' = (R^s_n ... R^s_{n-n'+1} 0...0, R^t_n ... R^t_{n-n'+1} 0...0)$$

The following section explains how to rebuild the function R^{-1} from R'.

2.4. Inverse mapping

For all pairs of coordinates (s_0, t_0) with bits from one to $n - n'$ set to zero, we define a set of camera pixels $S_c(s_0,t_0) = \{(u,v) \mid R'(u,v) = (s_0,t_0)\}$. This is a contained region of the camera image as the thresholding eliminates possible outliers. We also define a set

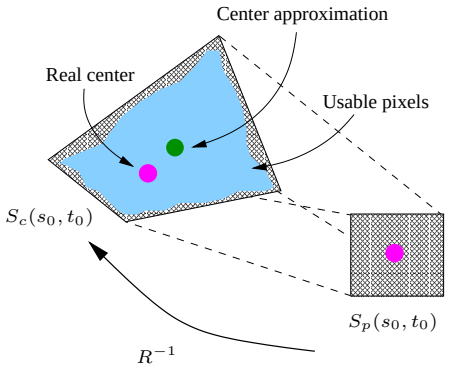


Figure 7. The projection of the center of $S_p(s_0,t_0)$ is approximated by averaging the pixel positions that were not rejected, i.e. from $S_c(s_0,t_0)$. The mapping of the center of $S_p(s_0,t_0)$ onto the approximation of the center of $S_c(s_0,t_0)$ is a sample of R^{-1}.

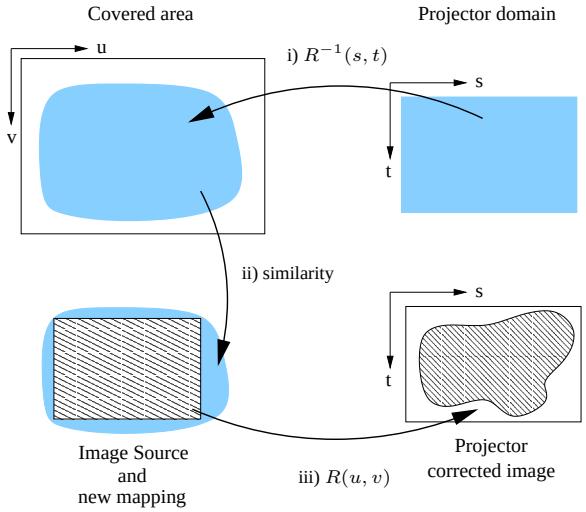


Figure 8. Image construction process.

$S_p(s_0,t_0) = \{(u,v) \mid (u,v) \equiv (u_0,v_0)\ mod\ 2^{n-n'}\}$ which is a $2^{n-n'} \times 2^{n-n'}$ square of pixels in the projector image. It is generally hard to establish the exact correspondence between points of $S_c(s_0,t_0)$ and $S_p(s_0,t_0)$. However one can estimate the projection center of $S_p(s_0,t_0)$ by taking the average of the points of $S_c(s_0,t_0)$ (see figure 7). Now, we can make the assumption that the latter is mapped through R onto the center of $S_p(s_0,t_0)$. Applying this process for all non-empty S_c for all (s_0,t_0) defines an under-sampling of the function R, and thus R^{-1} as well.

2.5. Image construction

When building the images, there are two aspects to consider: geometrical correctness and efficiency. The first idea is to get R^{-1} from its under-sampling with an interpolation scheme. Bilinear interpolation is simple but non-intuitive geometrically. Another possibility is to define a homography for each corresponding square, but this is computation intensive. In practice we use the under-sampling of R^{-1} as a square mesh, each point of which defines coordinates in the camera image domain. For a 1024×768 image rebuilt with seven bits, this grid of points has dimensions 128×96, representing 12065 squares. Implicitly, this is a texture mapping of the camera image domain. One possibility to correct an image is to copy and scale it into the region covered by the projector in the camera image. But a better solution is to modify the mapping itself to use any image directly as the texture. Achieving this is rather simple: i) we find the biggest rectangle that fits inside the covered area of the projector into the camera image; ii) using a rectangle with the same ratio as the target image, we can define a similarity transformation S between the rectangle and the target image. This transformation is actually a translation to the upper-left corner and a scale. Applying S to each value of R^{-1} modifies the mapping to fit for the target image directly; ii) finally, to build the image itself, an openGL texture mapping over the new mapping is used so that it can be achieved in real-time. Thus, the interpolation scheme and the image construction are done in one very fast step. The whole process is summarized in figure 8.

3. Multi-projectors

Addition of more projectors to cover a larger screen is rather simple. The method described above supports an arbitrary number of simultaneous projectors, (see figure 9). The only difference is that the initialization must be carried out one projector at a time and that the similarity transformation is defined using the covered area of all projectors simultaneously. The same transformation is applied to each projector function. A scheme for intensity blending should be used for an arbitrary number of projectors [8]. Ideally, every point of the projection surface visible by the camera should be illuminated by at least one projector. To expect good results, a higher resolution camera is required when each projector only covers a small part of the camera image. As an alternative, the number of used bits n' can adjusted accordingly.

The functions R^{-1} are recovered for one projector at a time. To provide a corrected image without holes, the projector images must overlap, resulting in unwanted intensity fluctuations. These could be effectively corrected by intensity blending algorithms.

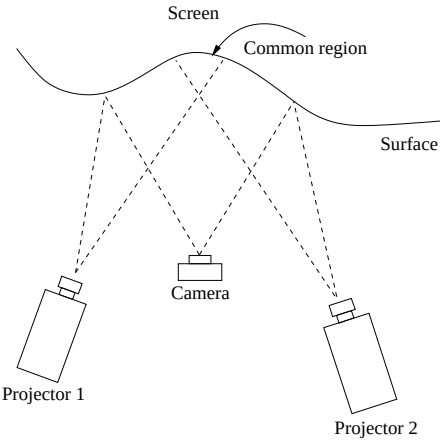


Figure 9. Top view of a multiple projector setup. Notice how the combined projectors cover the camera's view.

4. Experimental setup

Even if the implementation does not depend on the projector or camera resolution, the quality of the results increases with the resolution of each device. In the following experiments, we used a Sony Digital Handycam DCR-VX2000 (720×480 pixel resolution) and a Kodak DC-290 (1792×1200 pixel resolution). In most cases, acquisition time is proportional to the resolution of the camera. Initialization time of a each projector using the video camera was below two minutes and about 20 minutes for the Kodak digital camera. Two DLP projectors were used for multi-projector setup: a Projectiondesign F1 SXGA (1280×1024) and a Compaq iPAQ MP4800 XGA (1024×768). Like every system using structured light, the optical characteristics of each device itself limit the possible screen shape that can be reconstructed. For instance, the depth of field of both camera and projector restricts the geometry and size of the screen. Once the initialization is done, the image correction can be done in real-time on current video hardware technology.

5. Results

Single projector setup: Figure 10 illustrates how the image of one projector is corrected for a two-plane surface consisting of two circular screens. The camera and the projector were placed together so the angles to each screen was about 50 and 70 degrees. On the discontinuity between the two surfaces in the projector image, the distance along projection rays from one plane to the other was up to 15 centimeters (figure 10(a)). The Kodak camera was used so the R function could be constructed on eight bits out of 10. This allowed high precision corrected images (figures 10(c) and 10(d)). Two types of artifacts are still noticeable. The first is some empty little black squares where no stripes could be well acquired. The second appears in the same area, where the interpolation yielded wrong values (figure 10(e)). Both results from big gaps between the two screens.

Multiple projector setup: The second test demonstrates how a multi-projector setup can correct occlusions on a very peculiar surface geometry. Here, projection was done on a dodecahedron in front of a flat screen (figure 11(a)). Occlusions occur from both projectors, but very little from both simultaneously (figures 11(b-d)). The Sony video camera was used and seven bits could be identified to compute R^{-1} for both projectors. Results are shown in figures figures 11(d-g). Note that even though the distortions and occlusions were large, the corrected images (figures 11(f,g)) feature very few artifacts.

6. Conclusion

We presented a new image projection method that allows arbitrary observer-projector-screen geometries. Relying on a robust structured light approach, the method is simple and accurate and can readily be adapted to multi-projector configurations that can automatically eliminate shadows.

Several directions for future research come to mind. Algorithmic determination of the rejection threshold τ and of stripe width would automate the whole process. It would also make it possible to have these parameters adapt across different regions of the screen resulting in better reconstruction. Acquisition time could be decreased using improved patterns. Furthermore, hardware acceleration of video cards could be used to boost the speed of the construction of function R^{-1}. This could allow real-time applications where slides or movies are projected over moving surfaces.

References

[1] Ronald Azuma, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, and Blair MacIntyre. Recent advances in augmented reality. *IEEE Computer Graphics and Applications*, 21(6):34–47, Nov/Dec 2001.

[2] Ronald Azuma. A survey of augmented reality. *Presence:Tele- operators and Virtual Environments*, 6(4):355–385, Aug. 1997.

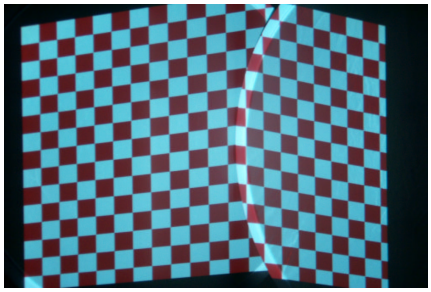
[3] Ramesh Raskar, Kok-Lim Low, and Greg Welch. Shader lamps: Animating real objects with image-based illumination. Technical Report TR00-027, 06 2000.

[4] R. Raskar, R. Ziegler, and T. Willwacher. Cartoon dioramas in motion. In *International Symposium on Non-Photorealistic Animation and Rendering (NPAR)*, June 2002.

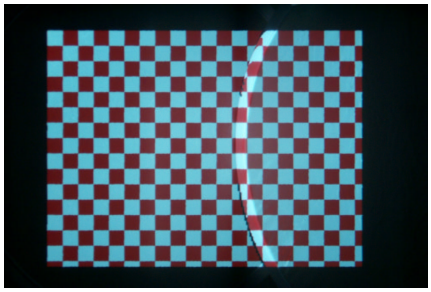
[5] C. Rocchini, P. Cignoni, C. Montani, P. Pingi, and R. Scopigno. A low cost 3D scanner based on structured light. *Computer Graphics Forum (Eurographics 2001 Conf. Issue)*, 20(3):299–308, 2001.

[6] Szymon Rusinkiewicz, Olaf Hall-Holt, and Marc Levoy. Real-time 3d model acquisition. In *ACM Transactions on Graphics*, volume 21, pages 438–446, 2002.

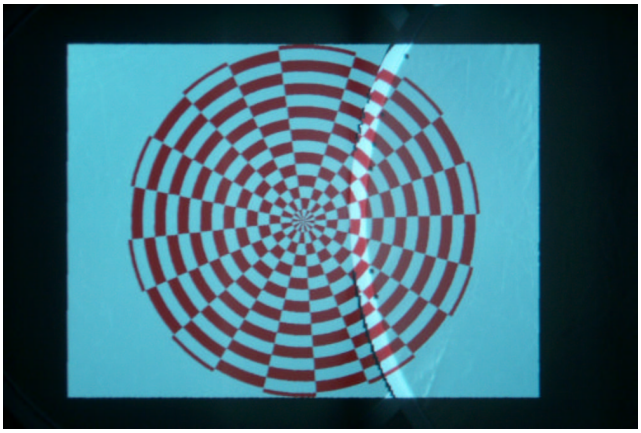
(a) Side view of the two planes of the screen at about 60 degree of angle.



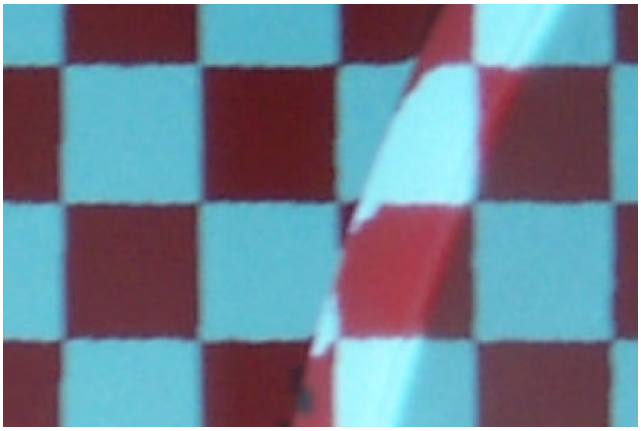
(b) Source image (a checkerboard) directly copied into the projector image. This results into a distorted image from the camera's point of view.



(c) The corrected image displayed by the projector, and observed by the camera. The result is an image identical to the target image with some reconstruction errors.



(d) Another source (polar coordinate checkerboard) displayed as the corrected image.



(e) Zoom on errors in the corrected image observed by the camera. The errors are the differences in color between this image and the target image. Black squares (middle bottom) result from holes in the mapping. Distortions are due to interpolation at borders featuring large discontinuities. Intensity variations occur because of varying screen color.

Figure 10. Single projector test for an non-flat screen.

[7] Li Zhang, Brian Curless, and Steven M. Seitz. Rapid shape acquisition using color structured light and multi-pass dynamic programming. In *1st international symposium on 3D data processing, visualization, and transmission*, Padova, Italy, June 2002.

[8] Ramesh Raskar, Michael S. Brown, Ruigang Yang, Wei-Chao Chen, Greg Welch, Herman Towles, Brent Seales, and Henry Fuchs. Multi-projector displays using camera-based registration. In *IEEE Visualization '99*, pages 161–168, San Francisco, California, October 1999.

[9] Ramesh Raskar and Paul Beardsley. A self correcting projector. In *IEEE Computer Vision and Pattern Recognition (CVPR) 2001*, Hawaii, December 2001.

[10] Ramesh Raskar. Immersive planar display using roughly aligned projectors. In *IEEE VR*, New Brunswick, NJ, USA, MARCH 2000.

[11] Ruigang Yang, D. Gotz, J. Hensley, H. Towles, and M.S. Brown. Pixelflex: a reconfigurable multi-projector display system. In *IEEE Visualization 2001*, pages 167–174, October 2001.

[12] Rahul Sukhthankar. Calibrating scalable multi-projector displays using camera homography trees. In *Computer Vision and Pattern Recognition*, 2001.

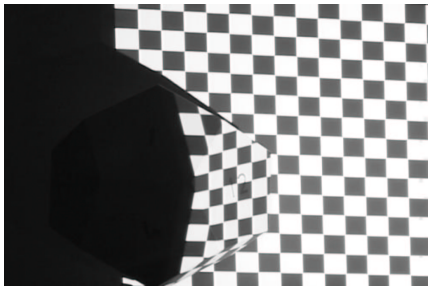
[13] R. Sukthankar, T. Cham, and G. Sukthankar. Dynamic shadow elimination for multi-projector displays. In *CVPR, Projector IR Camera IR Light*, 2001.

[14] C. Jaynes, S. Webb, R. M. Steele, M. Brown, and W. B. Seales. Dynamic shadow removal from front projection displays. In *IEEE Visualization 2001*, pages 175–182, October 2001.

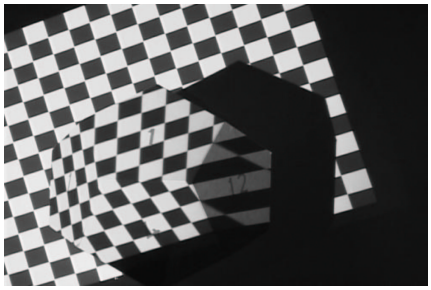
[15] A. Majumder, Zhu He, H. Towles, and G. Welch. Achieving color uniformity across multi-projector displays. In *IEEE Visualization 2000*, pages 117–124, October 2000.

[16] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge, 2000.

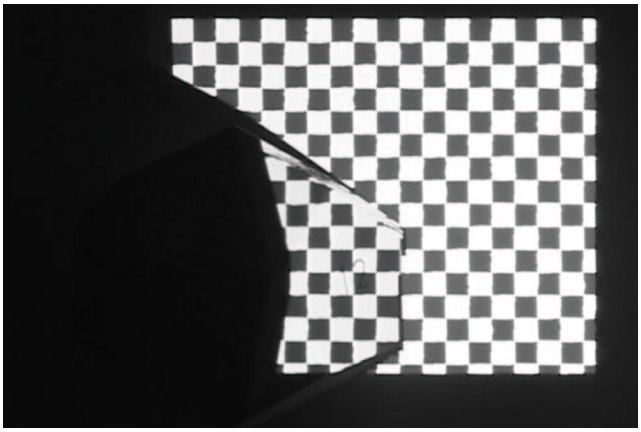
(a) Side view of the screen showing the covered region of the projectors.



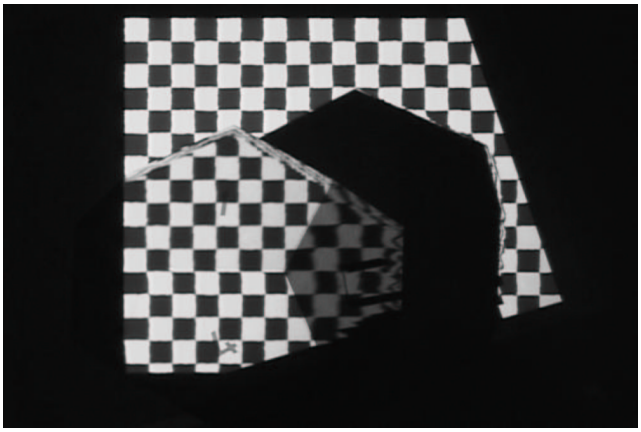
(b) Source image directly copied into the first projector image.



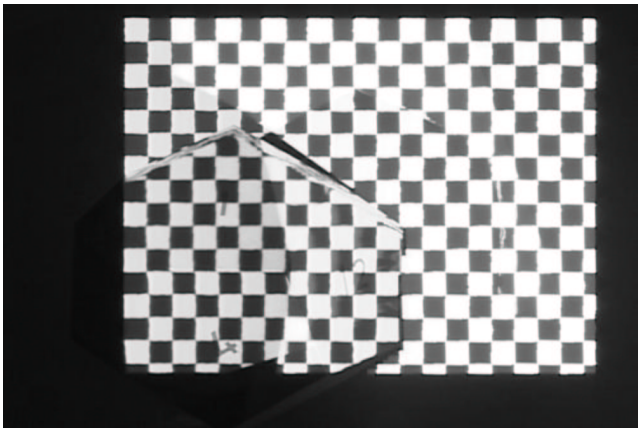
(c) Source image directly copied into the second projector image.



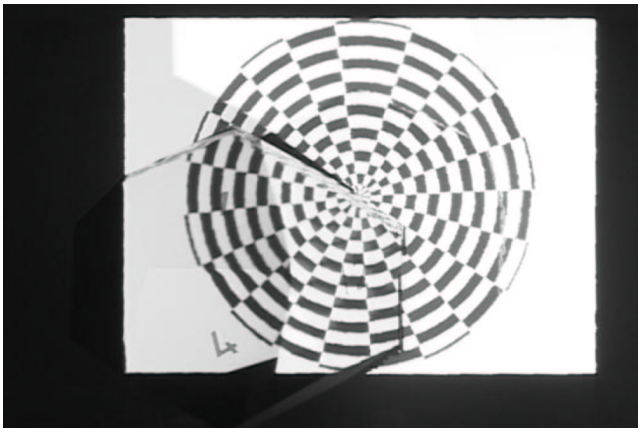
(d) View from the camera of the corrected image displayed by the first projector.



(e) View from the camera of the corrected image displayed by the second projector.



(f) Combined images of the two projectors. Notice the lack of geometric distortion and few artifacts.



(g) Other checkerboard pattern.

Figure 11. Multi-projector setup.