A User-Friendly Method to Geometrically Calibrate Projector-Camera Systems

Samuel Audet and Masatoshi Okutomi
Tokyo Institute of Technology
2-12-1 Ookayama, Meguro-ku, Tokyo, Japan
saudet@ok.ctrl.titech.ac.jp and mxo@ctrl.titech.ac.jp

Abstract

Projector-camera systems drive applications in many fields such as measurement and spatial augmented reality. When needed, we can find their internal and external parameters via geometric calibration. For this process, we have to use both a printed pattern and a projector pattern, but they can easily interfere with each other. Current methods compensate by decoupling their calibrations or by leveraging structured light and color channels, but the required manipulations are not user-friendly. Therefore, we cannot expect normal users to execute the procedure, which can also become a burden for researchers. Although not always required, knowledge of the geometric parameters can often facilitate development of new systems. To make the calibration process easier, we propose a method that uses fiducial markers, from which we can easily derive a prewarp that, once applied to the projector calibration pattern, prevents its interference. Using our method, we confirmed that users can easily calibrate a projector-camera system in less than one minute, which we consider to be user-friendly, while still achieving typical subpixel accuracy.

1. Introduction

Projector-camera systems are changing the way people use computers. Traditionally, projector-camera systems had been limited to surface measurement applications using structured light, but recently research has been progressing in other fields such as spatial augmented reality [3], where we use projectors to complement real world objects by adding art or information to them, for the benefit of human users. Augmented reality has a wide range of potential applications, and to succeed, systems have to be usable by people of any background. At the same time, we need to track and measure precisely physical objects, topics that still challenge many researchers in computer vision. Successful tracking or measurement can more easily be achieved by calibrated systems than uncalibrated ones. Although there exists methods to find the internal and external geometric parameters of both camera and projector, they are cumbersome, impractical, and could not realistically become part of an augmented reality system that anyone could use under a large range of casual conditions, such as the one shown in Figure 1.

Figure 1. Snapshot of our demo video showing the test user calibrating a casually installed projector-camera system.

1.1. Previous Work

Most current methods to geometrically calibrate projector-camera systems, [1, 2, 13, 15, 19, 23] among others, operate in two phases: they first calibrate cameras, then they calibrate projectors. Calibrating cameras in one way and projectors in another automatically doubles the effort. To reduce the amount of work, methods have been proposed to integrate both calibrations together. They either exploit structured light or color channels, or both for one-shot structured light.

In the case of methods using structured light, [6, 11, 16, 17, 23] among others, during the calibration process, the machine may shut the projector off, capture an image, then project structured light, capture more images, and by decoding everything can geometrically relate images together. The ones captured with the projector off are used to calibrate the camera, and the rest, to calibrate the projector. Even though this procedure works, we need to secure phys-

978-1-4244-3993-5/09/$25.00 ©2009 IEEE

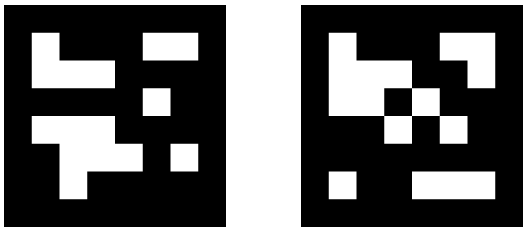
ically the calibration target in place, because within a given set of images the machine assumes that the scene is static, often a requirement of structured light. We cannot simply hold a board in our hands in front of the projector-camera system.

To capture two images simultaneously, we can also exploit color channels, [1, 2, 11, 12, 23] among others, usually the red and blue ones. With properly designed complementary colors for the physical pattern and projected pattern, we can recover them separately, *e.g.*: White and cyan squares show clearly in the red channel of a camera, but should hopefully not appear in the blue channel. While this obviously does not work for grayscale projectors, cameras, and printers, even with color ones, this approach presents two problems. First, depending on the properties of the color filters and inks, the blue channel of the projector might still interfere with the red channel of the camera, and vice versa. For example, results from Chen *et al*. [5] show a 4% overlap between the red channel of their camera and the blue channel of their projector, and the same the other way around. Second, many cameras use only one sensor with an appropriate array of color filters, such that the resolution of the blue and red channel is half of the resolution of the whole sensor. Both of these issues may hurt accuracy, although current results are not conclusive, but more importantly requiring color complicates the execution of calibration.

1.2. The User-Friendly Way

If projector-camera systems could easily and accurately be calibrated, more applications could use the information, without fear that users might not be able to install and maintain the system properly. We also believe that researchers could benefit from such a method, since they could concentrate on using the results of the calibration, instead of spending time doing it or developing applications for uncalibrated systems only.

In this paper, we propose a user-friendly method to perform full geometric calibration of a projector-camera system. It is based on fiducial markers typically used for augmented reality applications. Fiala and Shu [9] proposed such a method for camera-only systems. We extend it to projector-camera systems, where half of the markers are physically printed, and half are projected using the projector. Each marker on its own carries information and can easily be identified. As we describe in this paper, this allows the machine to prewarp markers that are projected, in a way that they do not interfere with printed markers. Moreover, markers do not need color, and unlike structured light users can hold the calibration board in their hands. The user simply has to wave the calibration board, ideally at angles of approximately 45° with respect to the image plane as recommended by Zhang [24], and the machine automatically captures and saves about ten good images, with which it computes calibration parameters for both the camera and the projector.



(a) ID = 0. (b) ID = 1.

Figure 2. Two sample fiducial markers made from binary-coded BCH codes with IDs 0 and 1.

Fiala [7] also proposed using fiducial markers in the context of projector-camera systems, but only for multi-projector displays, which do not require full geometric calibration.

In the following sections, we describe how we designed our system, including the detection of fiducial markers, the calibration patterns used, the prewarp of the projector pattern, the practical algorithm, and solutions to some of its issues. Even though we describe a system with only one camera and one projector, it can easily be extended to any number of them.

2. Detection of Fiducial Markers

As fiducial markers, we opted for the BCH codes provided by ARToolKitPlus [21]. Two examples with IDs of 0 and 1 are shown in Figure 2. There are 4096 such unique markers, arranged in squares of 8×8 black or white subsquares composing the black border and 36 interior bits, for only 12 bits of data, thus featuring strong error correction capabilities. Once printed or projected on a sheet of paper and imaged sufficiently large by a camera, ARToolKitPlus can detect these markers. Basically, the software first binarizes the image at an appropriate threshold, then analyzes contours to find shapes close to a quadrangle, estimates the four corners, and warps the shapes into squares, thus removing projective distortion from the camera. Lastly, by decoding the BCH codes inside the squares, it recovers the IDs of the markers. In our case, even if a detected quadrangle is not a marker, since we use a small amount of markers and thanks to error correction, the decoding would most likely simply fail. Fiala [8] provides a more detailed analysis for markers used by ARTag, which are similar to BCH codes used by newer versions of ARToolKitPlus.

While ARToolKitPlus provides most of the desired functionality, we had to add two more features: adaptive thresholding and subpixel corner extraction. The former finds an optimal threshold for binarization even if the level of brightness varies across the image. The latter refines the location of the detected quadrangles, in the original grayscale image.

Adaptive thresholding adjusts the threshold depending on the local neighborhood of each pixel. We adopted Niblack's method [18], which sets the threshold for each pixel (x, y) to

$$t(x, y) = m(x, y) + k \times s(x, y) \ , \tag{1}$$

where k is a negative parameter, usually -0.2, and where $m(x, y)$ and $s(x, y)$ are the local mean and the local standard deviation, which are computed within a neighborhood of (x, y). In our case, we considered an adaptive neighborhood size, instead of a constant one. If the variance is very low, then the intuition is that the window of the neighborhood is too small. We need to make it larger to compute a good threshold. Inversely, if the variance is very high, then it is too large. We are no longer analyzing local statistics and the threshold would not be optimal. For each pixel, we thus need to adapt the window size dynamically. We chose binary search to find the size where the standard deviation $s(x, y)$ drops below a certain threshold. Assuming a grayscale image with pixel values in the range [0, 255], since the maximum standard deviation is a bit less than 128 and the minimum is 0, we found 64 to be an appropriate threshold. We implemented an efficient algorithm based on integral images that runs in $O(N^2 \log N)$ time for an $N \times N$ image.

For subpixel corner extraction, we were able to use directly the code provided in OpenCV [4]. The algorithm works by first considering a small window (*e.g.*: 11×11) around the pixel of interest in the original grayscale image. It then assumes that the following holds for all subpixels $\mathbf{x}$ in the window:

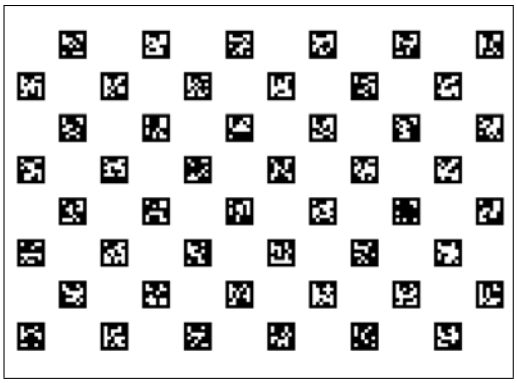
$$(\mathbf{c} - \mathbf{x}) \cdot \nabla_{\mathbf{x}} = 0 \ , \tag{2}$$

where $\mathbf{c}$ is the exact subpixel corner location and $\nabla_{\mathbf{x}}$ is the gradient at $\mathbf{x}$. This is true if either $\nabla_{\mathbf{x}}$ equals $\mathbf{0}$ or is orthogonal to $(\mathbf{c} - \mathbf{x})$, which is valid for a corner (but also for a straight edge, so it cannot be used for corner detection). The algorithm finds the minimum of this function, which is not normally zero because of noise and nonlinear distortions.

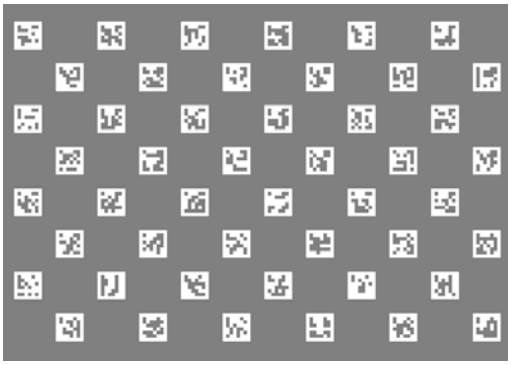
In this manner, we obtain for each marker an ID and the subpixel coordinates of its four corners.

3. Calibration Patterns

Even though it is possible to calibrate using only four corners from only one marker, to obtain best calibration accuracy, we should in fact attempt to cover the widest area possible with a large amount of corners [20]. As calibration pattern, we decided to use a matrix of markers, similar to the one that Fiala and Shu [9] used, but where half of the markers are printed and half of them are from the projector, as shown in Figure 3. As explained in more details below, the projector markers are in inverted color, since it helps to detect them when projected on top of printed markers. Given



(a) Printed pattern.



(b) Projector pattern.

Figure 3. Sample calibration patterns composed of matrices of markers.

the minimum focus distance of our projector, we found that 20×20 mm^2 markers spaced every 30 mm on a B4 size board works well. This gives a total of 12×8 markers, half of which are printed and half from the projector.

If one uses a board larger than the field of views, the system would obviously never be able to detect some markers, but as long as it can detect enough of them, there is no problem. In fact, since corners cover fully the image planes, it should improve accuracy.

In all cases, images captured by the camera contain a

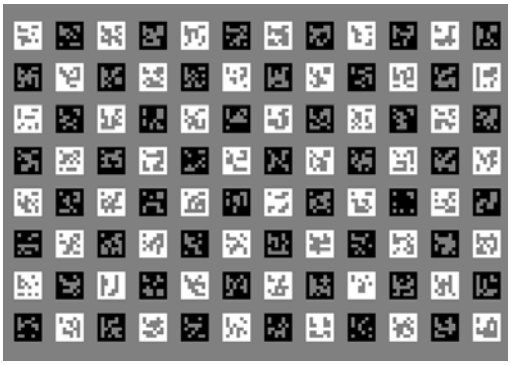


Figure 4. The ideal image that would be created by projecting the pattern from Figure 3(b) over Figure 3(a) under perfect alignment.

mix of both markers. For a given image to be considered appropriate for calibration purposes, the patterns need to be aligned well enough such that the markers from the projector do not interfere with the printed ones, and vice versa, as shown in Figure 4. In the following section, we explain how to accomplish this by prewarping the projector pattern.

4. Prewarp of the Projector Pattern

For camera-only systems, we can easily accommodate the number of corners shown in Figure 4, but this is challenging for projector-camera systems, because we need to image both physical corners and projector corners. Without proper considerations, the pattern from the projector would interfere with the printed pattern. In this section, we show that a prewarp of the projector markers to prevent interference does not change the calibration problem.

The prewarp transformation that we derive is a homography. Although the patterns contain many markers, we actually only need one to estimate a homography with maximum likelihood under the assumption of Gaussian noise [14]. Still, at least a few markers have to be decodable to compute a meaningful estimate of the error. In cases of large errors, the patterns are not well aligned, and we should not use the estimate for calibration purposes, but we can take it to update the projector prewarp, and hope to get a better estimate from the next captured image. In any case, we first need to find the homography between the camera and the board.

When a few printed markers are decoded, we can estimate this homography H_{bc} in

$$\mathbf{x}_{\mathrm{b}} = \mathrm{H}_{\mathrm{bc}}\mathbf{x}_{\mathrm{c}} \ , \tag{3}$$

where $\mathbf{x}_{\mathrm{b}}$ is a point on the board, $\mathbf{x}_{\mathrm{c}}$ the corresponding point on the camera image plane. H_{bc} is also the initialization information required to calibrate a camera using Zhang's method [24]. For the projector, we have the similar relationship

$$\mathbf{y}_{\mathrm{b}} = \mathrm{H}_{\mathrm{bc}}\mathrm{H}_{\mathrm{cp}}\mathbf{y}_{\mathrm{p}} \ , \tag{4}$$

where $\mathbf{y}_{\mathrm{b}}$ is a point on the board, $\mathbf{y}_{\mathrm{p}}$ the corresponding point on the projector image plane, the homography H_{bc} as defined above, and H_{cp}, the homography transforming points from the image plane of the projector to the one of the camera, which we find using detected projector markers in the camera image. With the combined homography $\mathrm{H}_{\mathrm{bc}}\mathrm{H}_{\mathrm{cp}}$, we can calibrate a projector in the same way as a camera. This is how all current projector calibration methods work. However, we can equivalently write

$$\mathbf{y}'_{\mathrm{b}} = \mathrm{H}_{\mathrm{bc}}\mathrm{H}_{\mathrm{cp}}\mathbf{y}'_{\mathrm{p}} \ , \tag{5}$$

where $\mathbf{y}'_{\mathrm{p}} = \mathrm{H}_{\mathrm{p}}\mathbf{y}_{\mathrm{p}}$ and H_{p} is the homography that prewarps our projector image. In other words, we can choose H_{p}, such that a transformed point $\mathbf{y}'_{\mathrm{p}}$ on the image plane of the projector appears where we want it on the board, $\mathbf{y}'_{\mathrm{b}}$, such that it does not interfere with the printed pattern. As previously, H_{cp} simply becomes the homography computed using detected projector markers in the camera image, and we can say that the calibration method is equivalent.

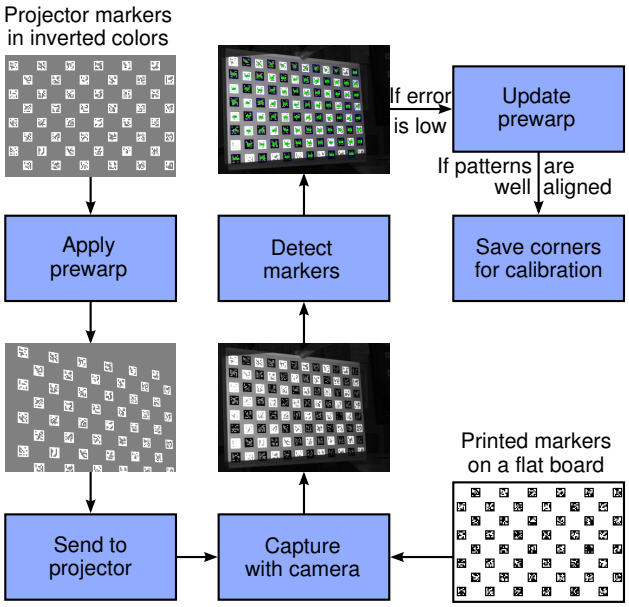


Figure 5. Flowchart illustrating one iteration of the practical algorithm.

5. Practical Algorithm and Issues

The sections above fully cover the theoretical aspects, but the machine executes in practice an algorithm. One of its iteration is depicted in Figure 5. First, it applies the prewarp of Section 4 to the projector pattern and sends it to the projector. We assume the user holds at all times the flat board with printed markers in front of the projector-camera system. In this case, projector markers appear along side the printed ones. The system can then continue and attempt to detect all printed and projector markers, one by one as explained in Section 2. If it considers that the error on the detected corners is low enough (details below) it uses them to update the prewarp homography. Further, if the patterns are aligned well enough (details below) it saves their marker corners. Finally, it loops and expects the user to move the board, unless enough corners have been saved, at which point they are used for calibration.

Although a conceptually simple algorithm, a few thorny technical issues remain. First, if we want the results of the subpixel detector to mean anything, we must make sure the prewarped projector image contains as little aliasing as possible. Next, when projector and printed markers overlap, it is not usually possible to detect them. However, using inverted colors makes them more easily detectable, even when

they interfere with each other. Third, even though we can derive a useful update for the prewarp from a rough homography estimation, outliers do not help. Finally, the machine must somehow judge when both patterns are aligned well enough and automatically select images for calibration. In the following subsections we explain in further details and provide practical solutions.

5.1. Antialiased Projector Images

Typical projector calibration methods do not warp the projector image and do not have to worry about aliasing. In our case, arbitrary warps produce subpixel motion, and aliasing issues must be managed. As antialiasing measure, we opted for simple $4\times$ supersampling [10], which gives good results and executes fast enough in software. Basically, we first create an image buffer that has four times the resolution of the projector. For example, a 4096×3072 image for a typical 1024×768 projector. Next, we draw the markers in the high-resolution buffer using well know drawing and filling algorithms [10] that work on a per-pixel basis. Since we make no attempt at drawing at the subpixel level, this high resolution image contains a lot of aliasing artifacts. However, when smoothed with simple averaging and decimated by a factor of four, the resulting image is subpixel sharp and free from problematic aliasing artifacts. Graphics processing units (GPUs) can also usually perform this operation quickly and efficiently via OpenGL or Direct3D, but the proprietary nature of the actual algorithms they use does not amend itself well to reproducible results. In either case, the system can then send the resulting image to the projector.

5.2. Detection of Overlapping Markers

Afterward, if the user holds the calibration board in front of the projector and camera, printed and projected markers usually overlap up to some degree. Because of the limited dynamic range of a typical camera, if we use black markers over a white background only, both black regions from the projector and regions printed in black appear in the same shade of black. Any overlapping markers do not appear as quadrangles anymore, and detection fails. To remedy, we observed that using projector markers with inverted colors (*i.e.*: white markers over a black background) overlapping markers are more robustly detected. This can be explained by the fact that with markers designed this way, the camera automatically images as gray the empty regions of the board, while printed black regions appear black, and white projected regions appear white, naturally providing orthogonal intensity ranges for both types of markers. For this reason, we use inverted colors for projector markers. In addition, the machine can easily adjust the brightness of the markers to match the ambient light and fall within the dynamic range of the camera or alternatively, in dark environments, use the projector as an adjustable source of light, by setting the minimum intensity of pixels.

5.3. Rough Homography Estimation

Even if overlapping markers are detected, the accuracy of their corners might obviously suffer, and consequently any homography we might estimate from them. In extreme conditions, markers may also be incorrectly detected. To compensate, we found that it works well to reject estimated homographies with large errors (*e.g.*: with an RMSE, root mean squared error, greater than 10 pixels, depending on the nonlinear distortions). One marker, which has only four corners, always fits a homography perfectly, so this also implies that the minimum number of detected markers must be two. While an RMSE of 10 pixels is still unacceptable for calibration, the hope is that updating the prewarp with this rough estimate moves the markers to a better position in the next captured image.

5.4. Automatic Image Selection

Eventually, the machine should be able to detect a lot of the projector and printed markers (*e.g.*: 50% of each). Also, if the corners of the calibration patterns have not moved a lot from the last captured image (*e.g.*: less than 5 pixels on average, depending on the slight movements from the user or oscillations when updating and applying the prewrap), then the system assumes that the patterns are aligned well enough, and it saves the detected corners for later calibration. Next, if the user does not move the board a lot, it would keep saving corners, but similar images are not useful for calibration. For this reason, we added another criterion. The system only selects an image if, since the last saved one, the corners of the calibration patterns moved a lot (*e.g.*: more than 50 pixels on average). When enough images have been selected (*e.g.*: 10 images), then all the saved corners go to the final stage: camera and projector calibration.

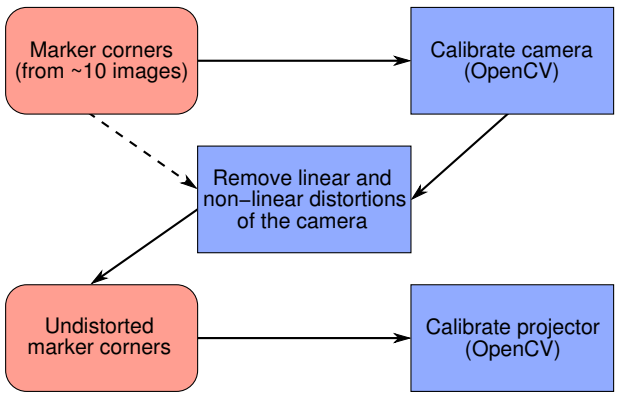


Figure 6. Dataflow diagram for calibrating the camera and the projector.

6. Camera and Projector Calibration

Once a sufficient amount of accurate corners have been accumulated, the calibration algorithm can process them directly. Using only the corners from printed markers $\mathbf{x}_b$ and their detected corners $\mathbf{x}_c$, OpenCV [4] can calibrate the camera with no difficulties using Zhang's method [24]. For the projector, on the other hand, we must first remove the linear (projective) and nonlinear (radial, etc.) distortions of the camera from the projector corners. Note that removing projective distortion is equivalent to applying the homography H_{bc} of Equation 5 to the imaged corners. Using such undistorted corner points $\mathbf{y}'_b$ on the board and corresponding $\mathbf{y}'_p$ on the image plane of the projector, we can apply the same method to calibrate the projector. Figure 6 illustrates the dataflow involved.

Although we have been writing about corners all along, we can also calibrate using the *center* of each marker. A center is defined as the average of a marker's four corners. When nonlinear distortions are small, as is usually the case of projectors, Fiala and Shu [9] showed that we can obtain better results using centers than corners. This happens because corners that are not perfectly in focus are not accurately extracted. This is especially relevant in the case of projectors, since they usually have a narrow depth of field. To illustrate the problem, Figure 7 shows the lower left area of the detected markers in Figure 5: The edges of the defocused projector markers make them appear larger than they really are. Taking the average of the four corners lowers this bias.

7. Results

The description of our method is now complete, and we present here some results. We built an application in Java, which integrates ARToolKitPlus, OpenCV, and PGR FlyCapture as appropriate, and that implements the procedure described in this paper. Our test hardware consisted of a NEC LT157 (1024×768 color LCD) projector, and a PGR Grasshopper GRAS-14S5M/C (1280×960 grayscale CCD) camera attached to a Fujinon HF16SA-1 (16 mm) lens, both connected to a Dell Vostro 400 computer with an Intel Core 2 Quad Q6600 2.4 GHz CPU. For the calibration target, we used the calibration patterns of Figure 3, printed the first one on a B4 size sheet of paper and pasted it on a (mostly) flat foam board.

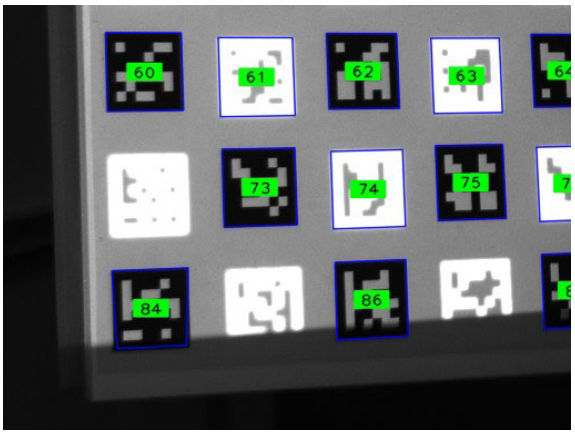


Figure 7. Lower left area of the detected markers from Figure 5. We can observe severe defocusing of projector markers.

Table 1. Reprojection RMSE, root mean squared errors, in pixels after calibrating using our method with either corners or centers.

	Reprojection RMSE (pixels)	
	Camera	Projector
Corners	0.43	0.66
Centers	0.33	0.20

Camera capture via software trigger took on average 158 ms, the display delay was about 64 ms, and the processing ran on only one core in approximately 350 ms, for a total of 572 ms per iteration. The test user could fully calibrate both camera and projector within about 30 seconds of manipulation. Figure 1 shows him in action on the casually installed system. The full sequence can be downloaded from `http://www.ok.ctrl.titech.ac.jp/%7Esaudet/procams2009.mp4`.

The reprojection errors, as listed in Table 1, also indicate that we were able to obtain an accurate subpixel calibration from this test session. Moreover, we could easily reproduce such results at will. The errors also show that, as predicted, because the projector has a narrower depth of field than the camera, it benefits more from the use of marker centers.

For quick comparison, we summarize in Table 2 other reprojection errors reported in the literature. For publications with more than one set of numbers, we took only the best.

Table 2. Reprojection RMSE, root mean squared errors, in pixels of other methods. For tidiness, we cite only the first author.

Authors	Reprojection RMSE (pixels)	
	Camera	Projector
Ashdown [1]	0.25	0.47
Audet [2]	0.23	0.52
Drouin [6]	N/A	0.27
Gao [11]	0.63	0.43
Gao [12]	0.15	0.24
Griesser [13]	< 0.4	< 1.5
Kimura [15]	≈ 0.3	≈ 0.4
Legarda-Sáenz [16]	0.60	0.79
Li [17]	0.094	0.15
Zhang [23]	≈ 0.4	≈ 0.6
Average	0.34	0.54

8. Discussion and Conclusion

Judging from our results and comparing them to data available from previous methods, little of which include information about usability, we conclude that our method is the most user-friendly. A calibration session that typically requires only a light board, a printed pattern, and less than one minute justifies in our eyes this designation. Furthermore, the subpixel accuracy achieved compares favorably with all previous methods. Assuming the noise characteristics of projectors are similar to cameras, since we based our method solely on existing theory of camera calibration, published results for those [20, 24], with regards to the actual physical 3D accuracy, should translate similarly. However, this remains to be proven. For this paper, our main goal is to demonstrate the ease of use.

We hope that this will help research on projector-camera systems in two ways. First, from the point of view of a user, since the calibration procedure is simple and does not require any special hardware, it could realistically become part of standard installation instructions targeted at end users. Second, researchers may save time when (re)calibrating their systems, instead spending hopefully more time actually using the calibration information.

Nevertheless, the method does have a few quirks. It can fail when all markers completely overlap, but we confirmed that a human user can easily detect such conditions and slightly move the board to help the machine. Also, to obtain the most accurate calibration, the casual user might not be aware that the calibration board should be placed at angles of approximately 45° with regards to image planes of the projector and the camera, and that it should also cover the largest area possible. A future implementation might be able to solve this by properly decomposing the homographies [22] and by providing appropriate feedback to users, *i.e.*: Point them in the right direction for the next ideal sweet spot. Despite all this, prewarps that use homographies alone might still fail for projectors that exhibit strong nonlinear distortions, such as omnidirectional projectors.

In any case, we believe that the method as it stands currently could already be useful to other researchers. For this reason, releasing the source code as free software is our next priority. We will soon post more information at `http://www.ok.ctrl.titech.ac.jp/%7Esaudet/research/` . Other future work includes implementing improvements, such as support for multiple projectors and cameras, and researching applications of projector-camera systems that could benefit from full geometric calibration.

Acknowledgments

This work was supported by a scholarship from the Ministry of Education, Culture, Sports, Science and Technology (MEXT) of the Japanese Government.

References

[1] M. Ashdown and Y. Sato. Steerable Projector Calibration. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05) - Workshops (Procams 2005)*, volume 3, page 98. IEEE Computer Society, 2005.

[2] S. Audet and J. R. Cooperstock. Shadow Removal in Front Projection Environments Using Object Tracking. In *Proceedings of the 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '07) - Workshops (Procams 2007)*. IEEE Computer Society, 2007.

[3] O. Bimber and R. Raskar. *Spatial Augmented Reality: Merging Real and Virtual Worlds*. A. K. Peters, Ltd., Natick, MA, USA, 2005.

[4] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly, 2008.

[5] X. Chen, X. Yang, S. Xiao, and M. Li. Color Mixing Property of a Projector-Camera System. In *Proceedings of the 5th ACM/IEEE International Workshop on Projector-Camera Systems (Procams 2008)*, pages 1–6. ACM, 2008.

[6] M.-A. Drouin, G. Godin, and S. Roy. An Energy Formulation for Establishing the Correspondence Used in Projector Calibration. In *Proceedings of the Fourth International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, June 18–20, 2008.

[7] M. Fiala. Automatic Projector Calibration Using Self-Identifying Patterns. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05) - Workshops (Procams 2005)*, volume 3, page 113. IEEE Computer Society, 2005.

[8] M. Fiala. Comparing ARTag and ARToolkit Plus Fiducial Marker Systems. In *Proceedings of the IEEE International Workshop on Haptic Audio Visual Environments and their Applications (HAVE 2005)*, page 6. IEEE Computer Society, October 1–2, 2005.

[9] M. Fiala and C. Shu. Self-identifying patterns for plane-based camera calibration. *Machine Vision and Applications*, 19(4):209–216, July 2008.

[10] J. Foley, A. van Dam, S. Feiner, and J. Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley Professional, Second edition, 1990.

[11] W. Gao, L. Wang, and Z.-Y. Hu. Flexible Calibration of a Portable Structured Light System through Surface Plane. *Acta Automatica Sinica*, 34(11):1358–1362, November 2008.

[12] W. Gao, L. Wang, and Z.-Y. Hu. Flexible Method for Structured Light System Calibration. *Optical Engineering*, 47(8):083602, August 2008.

[13] A. Griesser and L. V. Gool. Automatic Interactive Calibration of Multi-Projector-Camera Systems. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshop (CVPRW '06), Procams 2006*, page 8. IEEE Computer Society, 2006.

[14] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Second edition, 2004.

[15] M. Kimura, M. Mochimaru, and T. Kanade. Projector Calibration using Arbitrary Planes and Calibrated Camera. In *Proceedings of the 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '07) - Workshops (Procams 2007)*, page 2. IEEE Computer Society, 2007.

[16] R. Legarda-Sáenz, T. Bothe, and W. P. Jüptner. Accurate procedure for the calibration of a structured light system. *Optical Engineering*, 43(2):464, March 3, 2004.

[17] Z. Li, Y. Shi, C. Wang, and Y. Wang. Accurate calibration method for a structured light system. *Optical Engineering*, 47(5):053604, May 29, 2008.

[18] W. Niblack. *An Introduction to Image Processing*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1986.

[19] R. Raskar and P. Beardsley. A Self-Correcting Projector. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '01)*, volume 2, pages 504–508. IEEE Computer Society, 2001.

[20] W. Sun and J. R. Cooperstock. An empirical evaluation of factors influencing camera calibration accuracy using three publicly available techniques. *Machine Vision Application*, 17(1):51–67, 2006.

[21] D. Wagner and D. Schmalstieg. ARToolKitPlus for Pose Tracking on Mobile Devices. In *Proceedings of the 12th Computer Vision Winter Workshop 2007 (CVWW'07)*. Graz University of Technology, St. Lambrecht, Austria, February 6–8, 2007.

[22] T. Wong, P. Kovesi, and A. Datta. Projective Transformations for Image Transition Animations. In *Proceedings of the 14th International Conference on Image Analysis and Processing (ICIAP '07)*, pages 493–500. IEEE Computer Society, 2007.

[23] S. Zhang and P. S. Huang. Novel method for structured light system calibration. *Optical Engineering*, 45(8):083601, August 2006.

[24] Z. Zhang. A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.