# Vision-based Calibration of Parallax Barrier Displays

Nicola Ranieri, Markus Gross

Computer Graphics Laboratory, ETH Zürich, www.graphics.ethz.ch

## ABSTRACT

Static and dynamic parallax barrier displays became very popular over the past years. Especially for single viewer applications like tablets, phones and other hand-held devices, parallax barriers provide a convenient solution to render stereoscopic content. In our work we present a computer vision based calibration approach to relate image layer and barrier layer of parallax barrier displays with unknown display geometry for static or dynamic viewer positions using homographies. We provide the math and methods to compose the required homographies on the fly and present a way to compute the barrier without the need of any iteration. Our GPU implementation is stable and general and can be used to reduce latency and increase refresh rate of existing and upcoming barrier methods.

**Keywords:** parallax barrier display, calibration, computer vision, hardware accelerated, GPU implementation

## 1. INTRODUCTION

Stereoscopic parallax barrier displays deploy an image layer with two scrambled views and a barrier layer to multiplex the two views to distinct eye positions. A special translucent/opaque pattern is shown on the barrier layer, which exposes pixels of one view to one eye but blocks their sight to the other eye and vice versa. Careful alignment of barrier pattern and scrambled image pattern is required and crucial to avoid crosstalk and aliasing.

Furthermore, computation of the barrier pattern in existing methods often relies on known display geometry. Imperfections and misalignment caused by the manufacturing process can thus lead to Moiré patterns, crosstalk and other artefacts. Also, the iterative nature common to these methods induces a growing latency with increasing display resolution.

Perlin et al. [1] use known display geometry to compute the positions of the barrier slits and the positions of pixels for the left and right view, given the location of an eye pair. They are tracing rays from one eye to barrier pixels to get corresponding pixels on the image layer and vice versa with the other eye. This well-known approach giving an optimal barrier pattern is iterative: The preceding barrier position has to be known to compute the following one.

Sakamoto et al. [2] use the very same procedure to determine corresponding positions on the barrier layer and the image layer. But instead of using a barrier slit pattern acting as pinhole, they use the whole area between two barrier pattern positions as clear or blocking patch.

For both approaches, refresh rate drops and latency increases with higher display resolution due to the iterative nature of the algorithm. In addition, inaccurate knowledge about the display geometry can impose challenges: Tiny aberrations caused by the manufacturing process lower the quality of the displayed stereo content which becomes even more severe for high resolution displays with small pixel size.

A computer vision based approach was used in the work of Annen et al. [3] to calibrate a multi-view parallax barrier setup: They place a camera in front of the setup and capture images to compute for each barrier slit the corresponding pixels for one view point. Interpolation is then used to compute the corresponding pixels for novel views.
The approach is very suitable to compute the scrambling pattern of a multi-view display but would require further adaptations for a free viewpoint stereoscopic display.

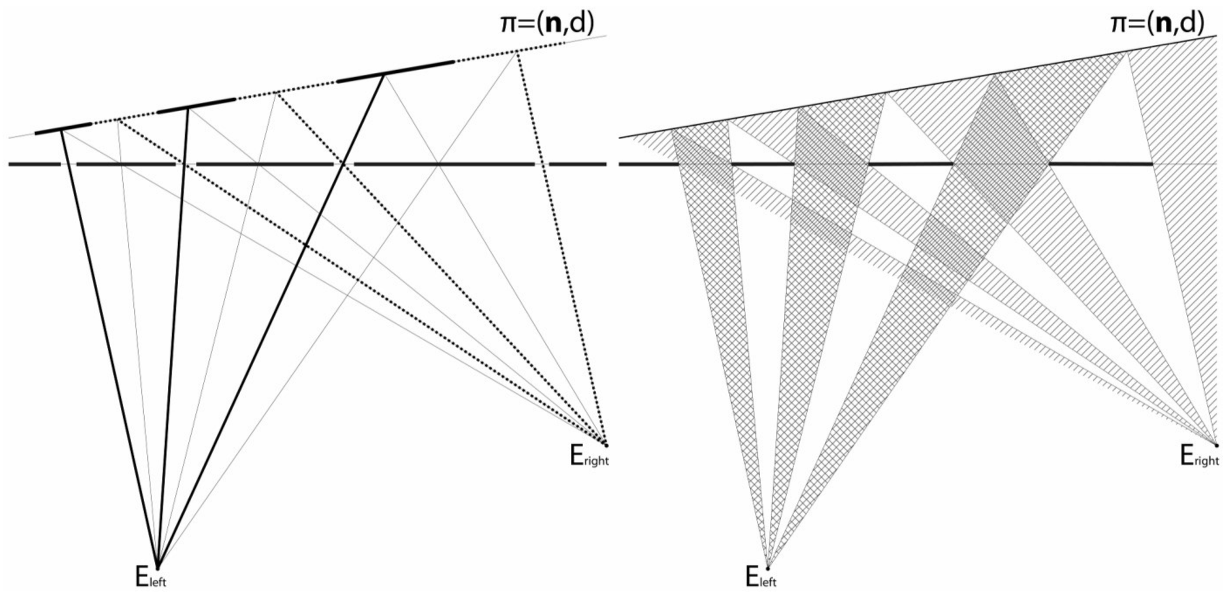

**Figure 1:** Illustration of two optimal barrier patterns. On the left side, barrier positions are used to define slits acting as pinhole, exposing left view's pixels (dashed lines) to the left eye $E_{left}$ and the right view's pixels (solid lines) to the right eye $E_{right}$ only. On the right side, the barrier positions define the border of alternating blocking and translucent barrier patches, providing much more brightness. Latter approach can further be improved by swapping the translucent and opaque patches each alternating frame, providing full spatial resolution.

The advantage of such vision based methods is that they do not require knowledge about the display geometry. They do scale well with the resolution of the camera sensor, which is usually a multiple higher than the resolution of a display. Furthermore, the use of projective transformations allows the image layer to be a projected image with keystone instead of a perfectly rectangular screen.

Therefore, we propose a computer vision based calibration method for static and dynamic parallax barriers. We relate the barrier layer to the image layer using homographies for arbitrary viewing positions. Furthermore, we demonstrate how algorithms for camera calibration can be used to derive the required homography for any given eye position. We provide a method how to use the homographies to compute all barrier positions independently of all others, and thus without the need of any iteration. This method is then used to compute hardware accelerated the whole barrier pattern on GPU, increasing refresh rate and decreasing latency. Proof of concept of our method is given in a physical prototype and we conclude our work with a discussion of our results.

## 2. HOMOGRAPHY-BASED MAPPING

To compute an optimal barrier pattern for a given viewer, a relation between image layer and barrier layer has to be computed for each eye position. The relation must map a barrier pixel to the image pixel which is seen by the eye when looking through the barrier pixel and vice versa.

An optimal barrier slit pattern as computed by Perlin et al. [1] is shown in Figure 1 on the left. The barrier slits are as close together as possible without exposing left eye pixels to the right eye or the other way around. Exactly the same barrier positions can be used to create an optimal stripe pattern as used by Sakamoto et al. [2], which is illustrated in Figure 1 on the right.

Figure 2 on the left hand side shows a geometrical interpretation of the approach proposed by Perlin et al. [1] on how to compute the required positions using ray tracing. To compute a new barrier position, a ray from left eye to the preceding barrier position is intersected with the image layer. Another ray from the intersection to the right eye intersects the barrier layer in the new barrier position. The same relation can also be expressed by homographies, a 3-by-3 matrix defining a mapping between two planes with respect to a center of projection.

For known homography $H_{E_{left}}$ with center of projection in the left eye $E_{left}$ and any pixel position $x_i$ on the barrier layer, the corresponding pixel $x'_i$ on the image layer can be computed by

$$x'_i = H_{E_{left}} \cdot x_i \tag{1}$$

as illustrated in Figure 2 on the right side. The iteration presented by Perlin et al. [1] can thus be expressed as

$$x_{i+1} = H^{-1}_{E_{right}} \cdot H_{E_{left}} \cdot x_i \tag{2}$$

using the inverse mapping $H^{-1}_{E_{right}}$ for the right eye. The complete iteration for the *i*-th barrier position is given by

$$x_i = H^i \cdot x_0 \tag{3}$$

with

$$H = H^{-1}_{E_{right}} \cdot H_{E_{left}} \tag{4}$$

and $x_0$ some initial barrier point. Similarly, the image positions can be computed by

$$x'_i = H_{E_{left}} \cdot H^i \cdot x_0 \tag{5}$$

Thus, for each viewer position it is sufficient to know $H_{E_{left}}$ and $H^{-1}_{E_{right}}$ to compute all required barrier and image positions.

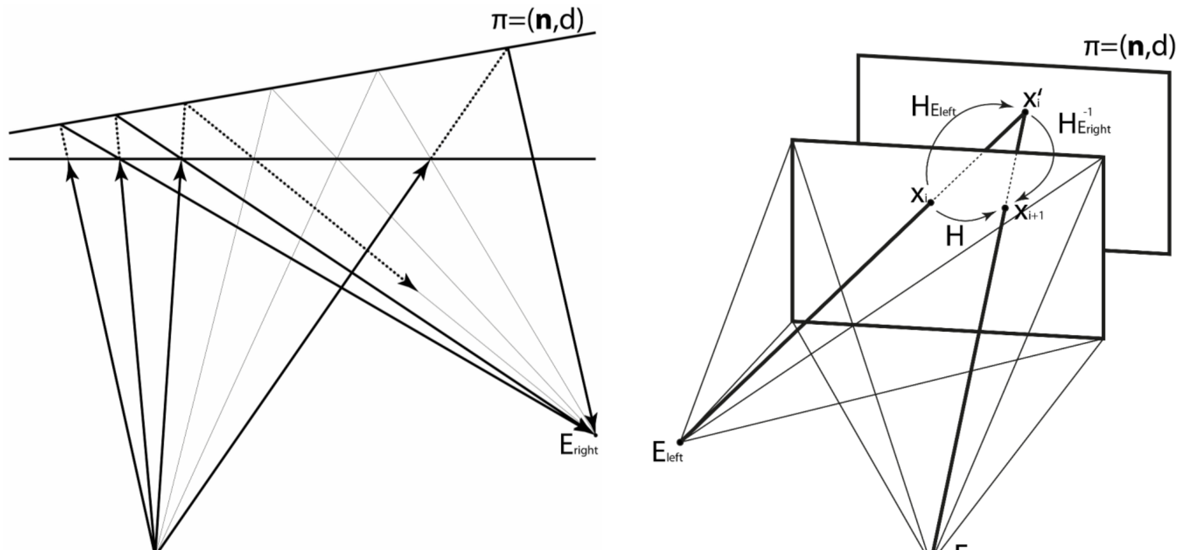

**Figure 2:** Two computation methods for an optimal barrier pattern. On the left, rays are traced from one eye through a barrier layer position onto the image layer and back to the other eye, resulting in the next barrier position. The same can be achieved using one homography for each direction as illustrated on the right. $H_{E_{left}}$ maps any barrier pixel to its corresponding image pixel as seen by the eye $E_{left}$. Similarly, $H^{-1}_{E_{right}}$ maps image pixels to barrier pixels as seen by the eye $E_{right}$. Concatenating both homographies results in a homography $H$, which can be used to determine all barrier positions both at the top and the bottom border of the screen, using Equation (3).

## 3. CLOSED FORM

Computing $H^i$ in Equation (3) and Equation (5) would involve an iteration which is especially unsuited for parallel architectures such as GPUs and would decrease refresh rate and increase latency for parallax barrier updates. Thus we propose to decompose $H$ into

$$H = V \cdot D \cdot V^{-1} \tag{6}$$

using the eigendecomposition. $V$ is a 3-by-3 matrix consisting of the eigenvectors of $H$ and $D$ is a diagonal matrix containing the corresponding three eigenvalues $\lambda_1$, $\lambda_2$ and $\lambda_3$.
The $i$-th power of $H$ can now be rewritten as

$$H^i = \underbrace{\underbrace{V \cdot D \cdot V^{-1}}_{H} \cdot \underbrace{V \cdot D \cdot V^{-1}}_{H} \cdot \ldots \cdot \underbrace{V \cdot D \cdot V^{-1}}_{H}}_{i \times} \tag{7}$$

As $V^{-1} \cdot V$ chancels out, this leads to the more compact form

$$H^i = V \cdot D^i \cdot V^{-1} = V \cdot \begin{bmatrix} \lambda_1^i & 0 & 0 \\ 0 & \lambda_2^i & 0 \\ 0 & 0 & \lambda_2^i \end{bmatrix} \cdot V^{-1} \tag{8}$$

Equation (8) can be computed without the need of costly iterations. In combination with Equation (3) and Equation (5), any barrier or image layer position can be computed independently of other positions and in constant time, both important characteristics for a fast GPU accelerated implementation.

## 4. FREE VIEW TRANSFORM

A specific eye position $E_{ref}$ together with the barrier layer can be seen as a virtual pinhole camera, where the eye is the center of projection and the barrier layer the virtual image plane. Together with any second eye positions $E_{any}$, a virtual stereo camera pair with shared camera image plane is defined, illustrated in Figure 3. As described in [4, p. 325], the display image plane induces a homography $H_{E_{any}E_{ref}}$ between these two virtual cameras. This means that the homography $H_{E_{any}E_{ref}}$ relates two points, one corresponding to each camera, whose rays would share the intersection point on the display image plane, also shown in Figure 3.

Assuming that the homography $H_{E_{ref}}$ relating barrier pixels to display image pixels for some reference eye position is known, the homography $H_{E_{any}}$ for any other eye position can be computed by

$$H_{E_{any}} = H_{E_{ref}} \cdot H_{E_{any}E_{ref}} \tag{9}$$

According to [4], the homography $H_{E_{any}E_{ref}}$ can be computed by

$$H_{E_{any}E_{ref}} = K_{E_{ref}}(R - \boldsymbol{tn}^T/d)K_{E_{any}}^{-1} \tag{10}$$

As the virtual cameras share the same image plane, namely the parallax barrier plane, the relative camera rotation $R$ is the identity matrix. The relative translation can be computed by

$$\boldsymbol{t} = E_{ref} - E_{any} \tag{11}$$

where $E_{any}$ is the given input position of the eye and $E_{ref}$ the position of the reference camera computed during calibration. $K_{E_{ref}}$ and $K_{E_{any}}^{-1}$ denote the intrinsic parameters of the virtual cameras and can be computed by simply putting the eye center into the projection matrix form

$$K_{E_{ref}} = \begin{bmatrix} -E_{ref}.z & 0 & E_{ref}.x \\ 0 & -E_{ref}.z & E_{ref}.y \\ 0 & 0 & 1 \end{bmatrix} \tag{12}$$

Note, the negative sign comes from the positive z-axis being the viewing direction in the used coordinate system. Similarly, the inverse projection matrix can be derived by

$$K_{E_{any}}^{-1} = \begin{bmatrix} -\frac{1}{E_{any}.z} & 0 & \frac{E_{any}.x}{E_{any}.z} \\ 0 & -\frac{1}{E_{any}.z} & \frac{E_{any}.y}{E_{any}.z} \\ 0 & 0 & 1 \end{bmatrix} \tag{13}$$

The remaining unknowns $\boldsymbol{n}$ and $d$ in Equation (10) are the plane parameters of the display image layer

$$\boldsymbol{\pi} = (\boldsymbol{n}^T, d) \tag{14}$$

which will be computed during calibration. As for Equation (10) the eye position $E_{any}$ needs to be at the origin, the plane parameters might need to be shifted accordingly. With this information, all the required relations can be computed by replacing $E_{any}$ by $E_{left}$ and $E_{right}$ for any viewer position.

Noticeable is, that the homography $H_{E_{ref}}$ can consist of any combination of other homographies, including e.g. the projection and keystone of a projected image. This is useful for projector-based parallax barrier displays as the one given by Sakamoto et al. [2].

Hence, if the plane parameters $\boldsymbol{\pi}$ and the homography $H_{E_{ref}}$ can be calibrated for one reference position $E_{ref}$, then the homographies for any new eye position can be easily computed using Equation (9) and Equation (10). These derived homographies for left and right eye can then be used in Equation (4) to compose the required homography for the barrier computation.

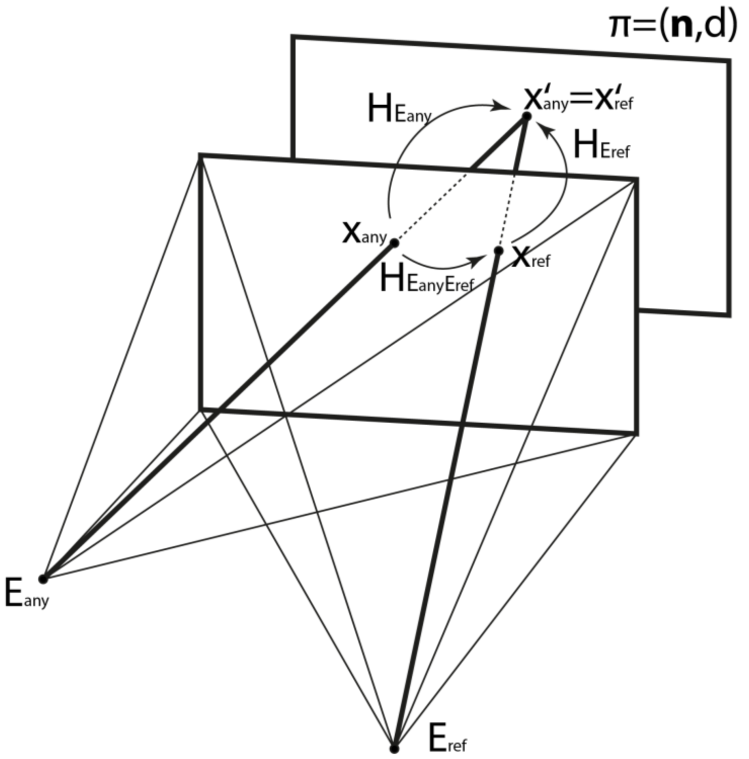

**Figure 3:** The mapping $H_{E_{any}}$ from barrier layer pixel to image layer pixel (or vice versa) can be computed for any eye position $E_{any}$ using the mapping $H_{E_{any}E_{ref}}$ defined in Equation (10), given a known reference eye position $E_{ref}$, known plane parameters of the image layer $\boldsymbol{\pi} = (\boldsymbol{n}^T, d)$ and known mapping for the reference eye position $H_{E_{ref}}$.

## 5. CALIBRATION

In standard camera calibration procedures, images of a planar calibration pattern with known pattern coordinates are taken from different camera positions [5]. For each image, a homography relating the coordinates in the calibration pattern to image coordinates of the camera sensor is computed. Based on this, intrinsic and extrinsic camera parameters can be computed. Intrinsic parameters contain camera characteristics as focal length and principal point and extrinsic parameters describe the position and orientation of the camera for each image.

To calibrate a parallax barrier display, the same algorithms can be used. One calibration pattern is shown on the barrier layer and one on the image layer, both being visible from different viewing positions at the same time. Example of such a calibration pattern as used in this work is shown in Figure 4 on the top right. Similar to camera calibration, pictures from different view positions are taken and the pattern on the barrier layer is used to compute the extrinsics of the camera.
This gives for each image the corresponding center of projection, expressed in pixel coordinates relative to the pixel positions of the barrier layer. These positions can be interpreted as candidates for the reference eye position $E_{ref}$ used in Section 4.

Also, using the known calibration pattern corner coordinates and their corresponding coordinates detected in the camera image, one homography that maps barrier layer pixels to camera sensor pixels and one homography that maps camera sensor pixels to display image layer pixels can be computed, using the standard direct linear transform method [4, p. 90]. Multiplying both matrices results in the homography that maps directly barrier pixels to display image pixels for each camera position. Hence, each such combined homography is candidate for $H_{E_{ref}}$ required in Section 4.

The same homographies are also used to compute the plane parameters $(\boldsymbol{n}^T, d)$ of the display image layer. For each camera center the corresponding mapping from image layer pixels to barrier layer pixels is applied to a single image layer pixel. This gives a number of barrier layer pixels which define together with their corresponding camera center a set of rays. These rays share a common intersection point on the physical image layer which can be found using known least squares methods. For three or more of such points, the plane parameters $\boldsymbol{\pi} = (\boldsymbol{n}^T, d)$ expressed in pixel coordinates relative to the barrier layer can be fitted.

As the homographies are only used to find corresponding pixels on the barrier layer, the image of the display image layer does not have to be rectangular and can e.g. be a projection with keystone or warped by any other homography. Also, the intrinsics derived during camera calibration are not used at all. Only the radial distortion parameters are used to undistort the images before any step of our algorithms.

The camera center closest to the preferred viewing position can be used as reference eye position. $E_{ref}$, $H_{E_{ref}}$ and $\boldsymbol{\pi} = (\boldsymbol{n}^T, d)$ can be further improved with a non-linear optimization, minimizing the re-projection error of Equation (10) when applied to all the other cameras and detected calibration pattern corners.

## 6. IMPLEMENTATION

For the implementation of our calibration algorithm we use the libraries ARToolKitPlus [6] and OpenCV [7]. ARToolKitPlus provides a convenient way to automatically detect virtual markers and extract their corners in our captured images. These initial guesses are then refined using sub-pixel accurate corner detection algorithms provided by OpenCV. Also, the inbuilt camera calibration and homography computation functions of OpenCV were used to get the camera centers and corresponding pixel mapping homographies.

We further use Matlab to fit the plane parameters and for a non-linear optimization of the reference camera parameters.
As reference eye position we choose a virtual one positioned at the centroid of the viewing volume. We first compute its homography based on the proposed method using the closest camera position. We then use a non-linear, gradient descent method to further optimize this initial guess of $H_{E_{ref}}$. For this purpose, we compute for each captured camera image the mapping homography based on our reference camera using Equation (10). Then, some barrier pixels are mapped to image pixels, once using this homography and once using the homography derived during calibration. The summed

distance between corresponding image pixels is used as error metric in the gradient descent method, as it optimally should be zero.

Our real-time barrier renderer is based on DirectX and also uses OpenCV. Each barrier stripe is represented as quad, with the $y$-coordinate as screen coordinate and $x$-coordinate as barrier index. The vertex buffer for the barrier has to be uploaded to GPU only once as it remains constant. In each frame, the input eye coordinates are used to compute the homography $H$ as described in Section 2 and Section 4. OpenCV is then used for the eigendecomposition of $H$. We use the pseudo-eigendecomposition to avoid complex eigenvalues which will save GPU operations. In each frame, $H_{E_{right}}^{-1}$, $V$, $V^{-1}$ and $D$ are uploaded to the GPU shader. In the shader, these informations and the index of each vertex found in the $x$ coordinate are used with Equation (8) to compute $H^{i}$ and thus the new barrier position or, together with $H_{E_{right}}^{-1}$, the image position.

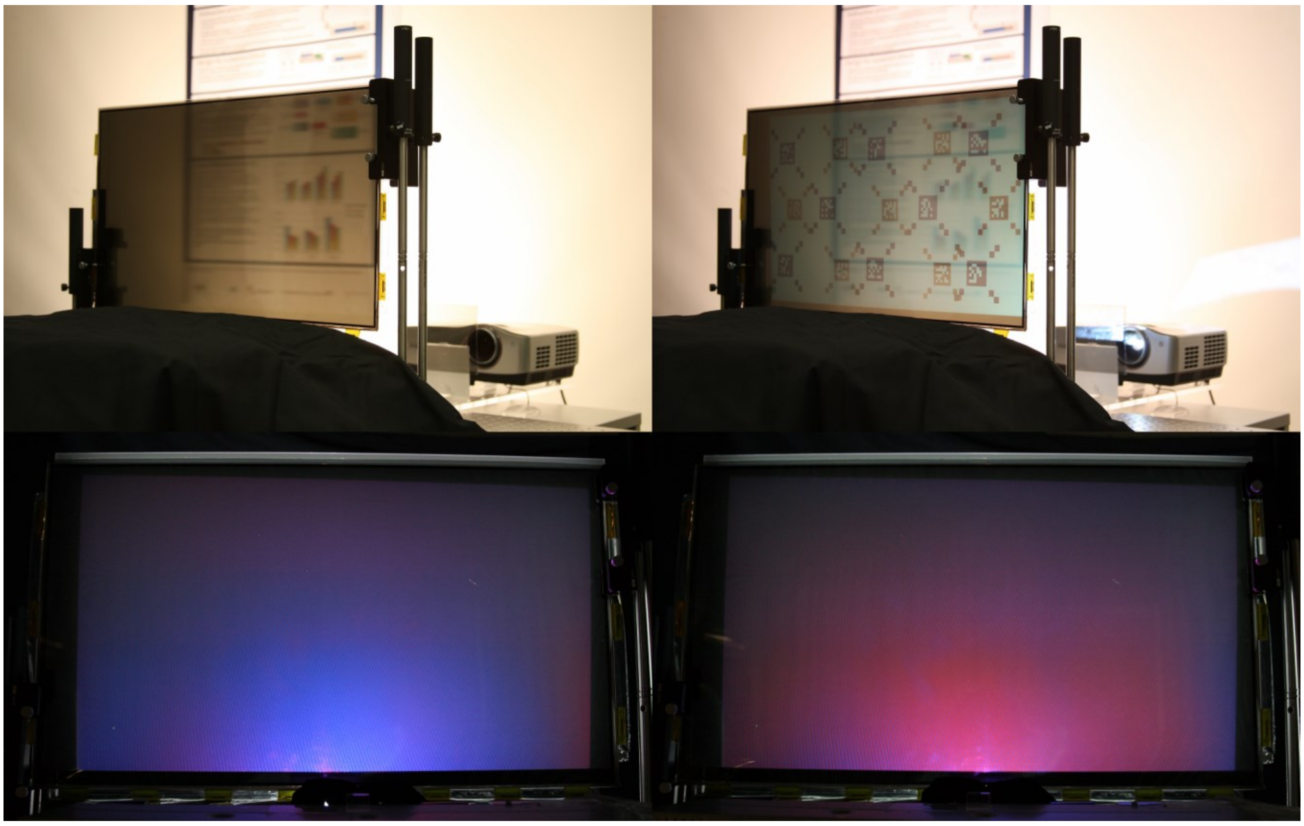
**Figure 4:** The top row shows our transparent stereoscopic display prototype once without displaying anything to illustrate transparency (top left) and once showing the calibration pattern that was used in our calibration algorithms (top right). A RGB liquid crystal display is used as barrier layer loosing much transparency in the embedded color filters. Using a gray-scale liquid crystal display as deployed in e.g. medical screens would improve transparency significantly. The bottom row shows results captured on our prototype using the barrier pattern illustrated in Figure 1 on the right and calibrated with the proposed method. Left eye's view (bottom left) and right eye's view (bottom right) show the image separation. Time-multiplexing for alternating transparent/opaque patches was applied to regain full spatial resolution. Crosstalk at the right border of the display is due to the slightly non-planar projection surface. The strong radial fall-off comes from the projector and is not subject of our work.

## 7. PHYSICAL PROTOTYPE

To demonstrate the potential of our algorithms we implemented a simplified version of the back-projected transparent stereoscopic screen suggested by Sakamoto et al. [2]. The isotropic back-projection screen ST-Professional-Trans from Screen-Tech® was used in combination with a BenQ SH910 Projector to create a transparent image layer which preserves polarization. As barrier layer we use an Acer HN274H with removed diffusing polarizer. The barrier layer has a theoretical transparency of only *16%* as only a third of the light passes the color filters and additionally half the light is lost in the polarizer. As the barrier layer does not require colors, the system's transparency could further be improved by using a gray-scale screen as available e.g. in medical high-contrast displays. Instead of using two projectors with two polarizations as proposed in [2] we only use one projector and time-multiplexing to swap the barrier pattern in each alternating frame.

A polarizer in front of the projector is used to create two polarized scrambled views on the image layer. The light then passes the twisted nematic liquid crystal barrier layer, and is, depending on the rotating state of the liquid crystal, either blocked by a polarizer in front of the setup or transmitted to the viewer.

Both image and barrier layer provide FullHD resolution and deploy an average spacing of *2.5cm* in between. The average barrier width at a viewing distance of one meter is 5 pixels. The back-projection screen is not co-planar to the barrier layer, the projected image is slightly rotated and suffers from keystone distortion. All these geometric unknowns are addressed by our algorithms. The complete setup is shown in Figure 4 on the top left.

## 8. RESULTS AND LIMITATIONS

To assess quality of our calibration we use the pixel distance between calibrated homography and homography computed by our algorithms as described in Section 6. Sub-pixel accuracy in display pixel space has been achieved for completely planar barrier and image layer. We have observed an increasing error for slightly bent or non-planar layers as they do not fit the assumption of planar surfaces in our algorithm.

Our implementation is able to run at *120Hz* for FullHD on a state of the art desktop graphics card, tested with an opaque parallax barrier display. The *60Hz* refresh rate (limited by the projector) in our presented prototype could be achieved without any problems. The crucial eigendecomposition can be computed in less than a millisecond due to the small and fixed size of the homography. Barrier computation is as fast as rendering at most *1920* quads on GPU for full HD.

Results captured on our prototype are shown in Figure 4. The two bottom images show left and right eye's view with clear color separation. Slight crosstalk is visible at the right border of the screen, coming from the slightly non-planar back-projection screen.

## 9. CONCLUSION

We have proposed a sub-pixel accurate calibration method for parallax barrier displays using known methods from computer vision. Without prior knowledge about display geometry, we are able to relate image layer and barrier layer with respect to arbitrary viewer positions. We provide a stable, scalable and fast GPU implementation to compute the barrier and scrambled image pattern in real-time.

Proof of concept of our method is given by a transparent, back projected stereoscopic display. Though currently driven by a desktop computer, we believe that our algorithms can meet the requirement of mobile devices as low power consumption and low processing power. Especially for mobile devices, the calibration procedure could be implemented fully automatically without the need of manual interaction, which could then be included into a production pipeline.

## ACKNOWLEDGMENTS

This research, which is carried out at BeingThere Centre, is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

## REFERENCES

[1] Perlin, K., Paxia, S. and Kollin, J. S., "An autostereoscopic display", Proc. of SIGGRAPH '00, 319-326 (2000).

[2] Sakamoto, K., Kimura, R. and Takaki, M., "Parallax polarizer barrier stereoscopic 3D display systems", Proc. of the 2005 International Conference on Active Media Technology, 2005. (AMT 2005), 469-474 (2005).

[3] Annen, T., Matusik, W., Zwicker, M., Pfister, H. and Seidel, H., "Distributed Rendering for Multiview Parallax Displays", Proc. of Stereoscopic Displays and Virtual Reality Systems XIII, 231-240 (2006).

[4] Hartley, R. I. and Zisserman, A., "Multiple View Geometry in Computer Vision", Cambridge University Press, ISBN: 0521623049 (2000).

[5] Zhang, Z., "Flexible Camera Calibration By Viewing a Plane From Unknown Orientations", The Proc. of the Seventh IEEE International Conference on Computer Vision(1), 666-673 (1999).

[6] Wagner, D. and Schmalstieg, D., "ARToolKitPlus for Pose Tracking on Mobile Devices", Proc. of the 12th Computer Vision Winter Workshop, (2007).

[7] http://www.opencv.org (accessed 01.01.2014)