Projection Defocus Correction using Adaptive Kernel Sampling and Geometric Correction in Dual-planar Environments

Shamsuddin Ladha
IITB-Monash Research Academy
Indian Institute of Technology Bombay
sladha@cse.iitb.ac.in

Kate Smith-Miles
School of Mathematical Sciences
Monash University
kate.smith-miles@monash.edu

Sharat Chandran
Department of Computer Science and Engineering
Indian Institute of Technology Bombay
www.cse.iitb.ac.in/~sharat

Abstract

Defocus blur correction for projectors using a camera is useful when the projector is used in ad hoc *environments. However, past literature has not explicitly considered the common situation when the projection surface includes a corner made up of two planar surfaces that abut each other, such as the ubiquitous office cubicle.*

In this paper, we advance the state of the art by demonstrating defocus correction in a non-parametric setting. Our method differs from prior methods in that (a) the luminance and chrominance channels are independently considered, and (b) a sparse sampling of the surface is used to discover the spatially varying defocus kernel.

1. Introduction

Projectors are designed to create large, bright and sharply focused image on a single plane perpendicular to the axis of projection. The large aperture used for maximum energy 'throw' across large distances in projectors comes with the rider that the depth of field (dof) is small. When projectors are deployed in *ad hoc* environments, it is difficult to avoid blurred effects as shown in Fig. 1. The focus of this paper is in developing techniques to combat the effects of blur when the only available physical screen is made up of multiple planar surfaces that offer varying depth to the center of projection of the projector. A space varying, surface savvy, depth compensated deblur correction can be discovered, and computationally applied resulting in (our method) a focused image.

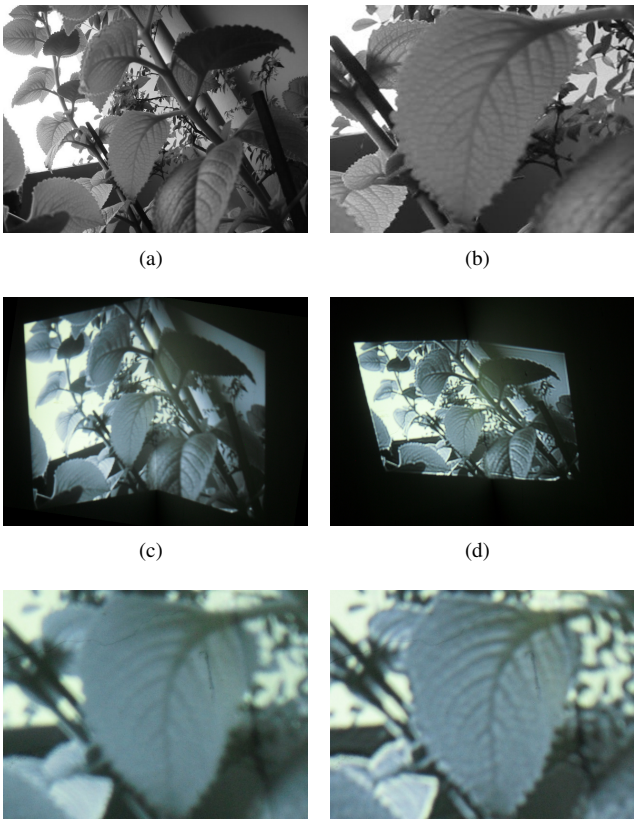
We first note that the question posed here is quite different from the classical deblur problem in image processing; even a well focused input image (seen, say on a laptop display) when sent to a projector can appear blurred due to the physical screen position. Stated equivalently but differently, one may want to see on the screen a deliberately created blurred input image using a projector; however, the display should be as is; neither sharpened nor further blurred.

Hardware focus can at best be set to a single depth. Software solutions to the problem of dealing with deblurs caused by projectors have been considered before [2, 5, 6, 1]. The flavour of most of the solution strategies is briefly described here. The blur at a particular location on the physical screen is modeled by a Gaussian blur of the corresponding pixel location in the input image. The parameters of this Gaussian can be obtained by projecting a test calibration image such as a chessboard, and then observing the blur through a camera. This, in turn, calls for obtaining the homography between the camera and the projector, and color correction. Once the Gaussian parameters are estimated, a suitable deblurring technique, such as Wiener filtering can be applied to prewarp the image.

1.1. Contributions

We extend the state of the art in software projector defocus correction in the following ways.

- Unlike the method in [2], [5], [6] our blur estimation is non-parametric in nature. We do not need the assumption made earlier of Gaussian-only blur. The non-parametric method is inspired from [9] (see below).
- Unlike the method in [2], we do not assume the existence of any *exemplar* point in focus. In other words, we permit the input image to be displayed to be blurred in the first place, and the goal is to show the image on



(a) (b) (c) (d) (e) (f)

Figure 1. Deblur correction in action. (a) Input image to be shown. Note that many parts of the input image itself are not in focus. (b) A section of the zoomed input image is shown. (c) When projected, the observed image (captured by a camera) is geometrically distorted (horizontal lines are bent). In addition to the luminance mismatch, we also see a blur – see zoomed version in (e). Our corrected output in (d) Although smaller, the proportions are maintained; notice the horizontal line at the bottom is straight. The resultant image is crisp. (f) Zoomed version of our output clearly shows the defocus compensation. Note that the images are captured by a low cost web camera, and are not (intentionally) color compensated.

the displayed screen to have high fidelity to the original input.

Unlike prior methods we demand more from the software solution by making the display surface consisting of dual planar surfaces that abut each other. Such surfaces (*e.g.* Fig. 3) have the property that neither of the two surfaces is orthogonal to the line of projection. Thus, neither surfaces are in focus, and furthermore, at the intersection of the two surfaces, an abrupt change is seen. This work is contrasted with [9] in the following ways:

- While defocus correction is demonstrated in [9], the surfaces on which the compensation is shown, do not abut each other. They are simply at different depths. Our scenario is far more common (*e.g.* office cubicles). The other type of surface demonstrated in [9] is a smoothly varying hemisphere, and definitely useful, but distinct from our application domain.

- While depth estimation is treated comprehensively in [9], it is not used per se in defocus compensation. Our work also does not explicitly compute depth; however, the depth discontinuity is subtly used in developing the defocus kernel. More specifically, the kernel computation are computed far more sparsely than in [9] enabling faster correction. In other words, the apparent scene geometry in the image space detected by our technique aids in selecting very few key points to infer overall defocus blur measurement.

- Instead of compensating for each color channel separately at individual pixel level [9], our algorithm uses a better distance metric based on [8] that recognizes that the human eye is biased w.r.t. to luminance changes as compared to chrominance changes. Specifically the algorithm treats both luminance and chrominance channels together holistically.

In summary, we implement a fast and efficient geometric correction technique [7] and use this to correct defocus in dual-planar environments. The primary point to be noted is that the discovered knowledge of the surface geometry is leveraged to densely measure defocus kernels near regions where naive methods cannot compute correct values (*e.g.* near the intersection of two plane surfaces in a multi-planar environment).

1.2. Scope and Limitations

The work presented in this paper has a few limitations. Fortunately many of these have already been considered in the literature before, and therefore, we believe, do not serve as bottlenecks. Specifically, we do not solve color compensation; this has been addressed in [4]. We do not need to do depth estimation explicitly; this has been done in [9]. As seen in our setup, the exposition in this paper deals only with two surfaces. We do not present the details of geometric correction; this has been considered in [7]. The extension of geometric correction to multiple surfaces has been considered elsewhere. We do assume that the depth from the center of projection in each of the planar surface individually changes smoothly; this follows trivially from the definition of planar; we state this explicitly to indicate that surface deformities such as cavities in the surface are not considered.

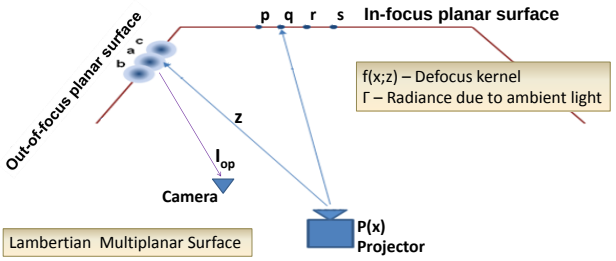


Figure 2. An *ad hoc* environment with piece-wise planar display surfaces depicting in-focus and out-of-focus planar surfaces, with overlapping blur circles on out-of-focus surface

2. Solution

Projectors have large apertures with narrow depths of field, hence traditionally they have been used to created focused image on a single planar surface perpendicular to the axis of projection. When projectors are deployed in environments that have non-planar, or piecewise planar surface attributes they create blurred images. We call such environments *ad hoc* environments.

Consider a multi-planar display surface formed by juxtaposition of three planar surface as shown in Fig. 2. We use the modeling in [9]. To summarize, for a point **q**, that is in focus, rays of light emitted from a single projector pixel converge, creating a sharp image. For another point, **a**, that is not in focus, rays of light are distributed in a small area called the *circle-of confusion* (PSF), creating a blurred image.

Therefore, irradiance at point **a**, equals the convolution of its defocus kernel with the light rays from projector. If we assume that the display scene is Lambertian then the radiance I_{op} of **a** along any given outgoing direction can be written as $I_{op} = \alpha f(\mathbf{x}; z) * P(\mathbf{x}) + \Gamma$. where $*$ denotes convolution, α is a factor that depends on surface *albedo* (ρ), Γ is a matrix that represents radiance due to ambient light, $f(\mathbf{x}; z)$ is the defocus kernel, and $P(x)$ is the projector input. The defocus kernel f depends on the depth z of display surface from the projector lens.

The relationship between radius of blur r_b and distance D of a display surface from the camera lens is derived in [3]. The relationship is given by

$$r_b = r_0 v_0 (1/F_l - 1/D) - r_0 \quad (1)$$

where, r_0 is the radius of the aperture, v_0 is the distance of image plane from the lens and F_l is the focal length of the lens. Since a projector can be considered as a dual of camera, a similar relationship holds for projected images.

As the distance of display surface from projector increases, *i.e.*, becomes substantially greater than the focal length of the projector lens , the variation in radius can be assumed to be linear with respect to the distance. In a typical projector camera setup, the distance between display surface and projector is generally such that the linear radius variation assumption can be used. This enables the sparsity feature of our method as described in Sec. 2.4.

In general, projection in *ad hoc* environments result in geometric distortion too. For certain *ad hoc* environments, like the one shown in Fig. 2, that are piece-wise planar, computations done to correct geometric distortion can be leveraged to expedite computations required for defocus blur correction.

In the following sub-sections we first look at the geometric distortion compensation method [7] that is used to in sparsely measuring defocus blur kernels. We then describe the defocus correction algorithm, followed by kernel estimation.

2.1. Geometric compensation

[7] uses uncalibrated structured light technique to segment the unknown projection area into piece-wise planar surfaces. The advantage of using this technique is that it is simple, efficient, and produces accurate results. As a by-product of this computation we get individual homographies for mapping between projector and camera pixels through each of the planar surfaces. We use the planar surfaces geometry and these homographies to efficiently compute defocus kernels at all projector points by measuring defocus kernels at very few points in projector space.

2.2. Defocus correction algorithm

We model the defocus compensation algorithm as defined in [9]. In [9] the problem of computing a compensation image is cast as a constrained optimization problem as shown in Eq. 2.

$$P^* = \arg\min\{d(\alpha f * P + \Gamma, I_{ip}) | \forall x, 0 \leq P(x) \leq 255\}, \quad (2)$$

where, x is the projector pixel coordinate and $d(.,.)$ is an image distance metric. P^* will have all its intensity values within the projector's dynamic range and it will not have any ringing artifacts. The defocus convolution, $\alpha f * P$, is represented as a matrix multiplication, FP. P is generally set to I_{ip} at the start of optimization procedure.

2.3. Distance metric

The distance metric proposed in [9] uses the sum of squares of difference between the observed image and the desired output image for each pixel. It can be written as

$$||FP + \Gamma - I_{op}||^2 \quad \text{or} \quad (FP + \Gamma - I_{op})^T (FP + \Gamma - I_{op}) \quad (3)$$

In the RGB color space, the constrained optimization problem is solved for each color channel independently in [9]. Considering the fact that humans are differently sensitive to luminance changes as compared to chrominance

changes it would be useful to move to a different color space like YCbCr to separate luminance and chrominance.

A distance metric proposed in [8] for spatially augmented reality (SAR) environment is useful. Here the authors aim to make the physical environment appear similar to the virtual environment. Their physical and virtual environments are modeled as consisting of small planar patches. In our formulation we model the input image as the virtual environment and the observed image as the real environment with pixels in each image taking the roles of patches in SAR. The new distance metric aims to minimize

- Absolute error for luminance and chrominance per pixel
- Spatial error for luminance and chrominance discontinuities in the neighborhood of each pixel

The distance function can be written as,

$$d(.\ ,\ .) = w_1 f_{\mathrm{l}} + w_2 f_{\mathrm{c}} + w_3 f_{\mathrm{sl}} + w_4 f_{\mathrm{sc}} \qquad (4)$$

where, $f_{\mathrm{l}} = \sum_i (Y^i_{ip} - Y^i_{op})^2$ is the absolute luminance error,
$f_{\mathrm{c}} = \sum_i [(Cb^i_{ip} - Cb^i_{op})^2 + (Cr^i_{ip} - Cr^i_{op})^2]$ is the absolute chrominance error,
$f_{\mathrm{sl}} = \sum_{(i,j)\in nbd} [(Y^i_{ip} - Y^j_{ip}) + (Y^i_{op} - Y^j_{op})]^2$ is the spatial luminance discontinuities error,
$f_{\mathrm{sc}} = \sum_{(i,j)\in nbd} [(Cb^i_{ip} - Cb^j_{ip}) + (Cb^i_{op} - Cb^j_{op})]^2 + [(Cr^i_{ip} - Cr^j_{ip}) + (Cr^i_{op} - Cr^j_{op})]^2$ is the spatial chrominance discontinuities error, and w_1, w_2, w_3, and w_4 are non-negative weighting parameters.

2.4. Kernel estimation

We now describe how to obtain F for piece-wise planar surfaces, given the planar surface geometry available in the camera space, and, with the homographies that map projector pixels to camera pixels, the projector space.

For each planar region in projector space, four distinct points are selected (no three of which are collinear) that are significantly far apart in the plane (ideally four corner points of the plane in projector space can be selected). We project a dot at each of these four points, and the PSFs for these points is captured by the camera. For the rest of the pixels in the same projected plane the PSFs are computed by bilinear interpolation of the four measured PSFs.

However, the defocus kernels can vary considerably near the intersection of two planar surfaces. To account for this, two additional points on the surface edge are illuminated. This is possible because we have the edge in the camera space, and with the homographies, we can turn on corresponding edge pixels in the projector space. The PSFs for edge pixels are observed in the camera. These PSFs for edge pixels along with the previously computed PSFs on either side of the edge are used to compute PSF for points in the immediate vicinity of the edge using bilinear interpolation. Each PSF represents one column of the F matrix and the rows of the matrix represent the defocus kernel of the corresponding pixel modulated by its albedo *i.e.* $\alpha f(\mathbf{x}; z)$.

3. Experiments and Results

We have implemented our software for two planar surfaces. The implementation can be extended to cater to more than two planar surfaces. For our experiments we use off-the-shelf hardware components. One of the goals is to make the system deployable in real world environments. We used a Logitech Quick Cam Pro webcam and a Sharp XR-1X Multimedia Projector, brightness 1100 ANSI Lumens and native resolution 800×600 as shown in Fig. 3.

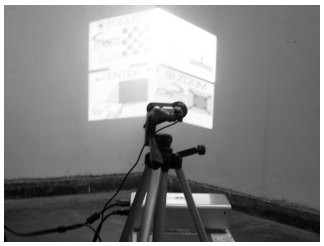
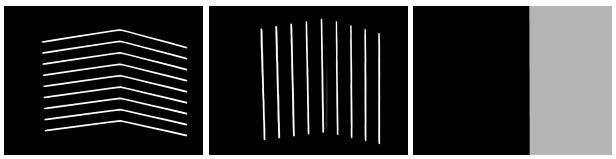


Figure 3. A typical experimental setup.



(a) (b) (c)

Figure 4. (a) & (b) Structured light patterns, (c) Computed left and right rectified planar surfaces in camera space.

3.1. Geometric correction

As per [7], we detect individual planar surfaces in camera space by finding the intersection line of these surfaces. To find the intersection line, a set of structured light patterns (in two directions) is projected onto the display surface and captured using camera as shown in Fig. 4 (a & b). If a projected straight line is detected with a bend in the camera space then that point lies on the intersection line. Identifying several such points we fit the intersection line through these kink points using least squares. The intersection line divides the camera space into two planar regions

Fig. 4(c) corresponding to the two planar surfaces shown in Fig. 3. Using the intersection points of the projected horizontal and vertical structured lines, forward and inverse homographies for each plane is computed. The projector space is accordingly partitioned into two planar surfaces and the corresponding image segments are warped using respective homographies and stitched together to form a composite warped image, which when projected compensates for geometric distortion. The above procedure is done in an offline mode and computed homographies and intersection line are stored. At the time of projecting an image only inverse homography needs to be applied along with stitching of the two partial images.

This step is a one-time effort and takes about 1 minute (on a low end 2007 vintage laptop) to detect the kink and compute homographies. If the system setup is changed then this step should be done again.

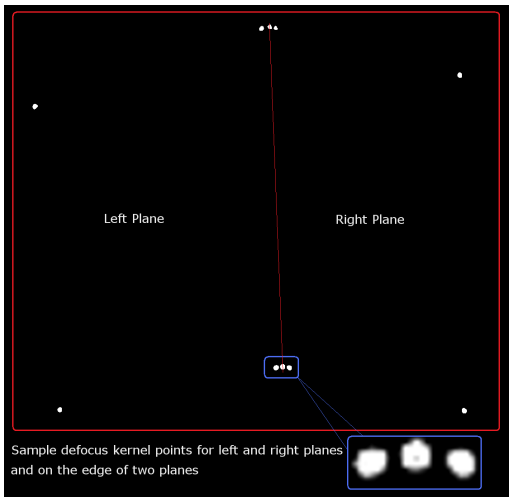


Figure 5. Patterns for defocus kernel measurement for planar surfaces.

3.2. Defocus correction

Defocus correction is done in the following two steps.

1. Computation of F

We now know the geometry of the display surface. For purposes of brevity, we measure PSF at four points on each planar surface and at other points along the kink as shown in 5. Using these PSFs we estimate PSF for all other points on that plane using bilinear interpolation. We warp these kernels from camera domain to projector domain (and normalize them) so that the defocused compensation image $FP+\beta$ can be compared conveniently with the input image I_{ip}. Note that we have to do computations in the projector domain,since we don't know the ideal camera output image.

This is also a one-time step that takes approximately 2 mins to measure defocus kernels at various points and perform bilinear interpolation. If the setup is changed then we need to run this step again.

2. Computation of compensation image

We have applied an iterative, constrained, steepest-descent algorithm. The defocus convolution, $\alpha f * P$, is represented as a matrix multiplication, FP, where each row of F is the defocus kernel of the corresponding pixel modulated by its albedo. The algorithm starts with $P_{j=0} = I_{ip}$ and iterates to compute P_{j+1}. Output at each iteration is clamped so that it lies within the dynamic range of projector (0:255).

After loading the defocus kernel matrix and input images in memory; per iteration of the optimization algorithm takes about 10-12 seconds to compute the compensation image. This optimization code runs on a single core machine with 2.8GB RAM.

3.3. Observations

As shown in Fig. 6, we tested our system on a range of input images. We observe that the output produced by our method compensates for both geometric distortion and defocus blur. The defocus blur compensation is spatially varying. Table 1 shows a count of the number of edge pixels in the warped camera output image against the defocus corrected camera output image. We see that there is a significant increase in number of pixels in defocus corrected input and output images. Clearly, defocus blur corrected images are more sharper than only geometrically corrected images.

4. Summary

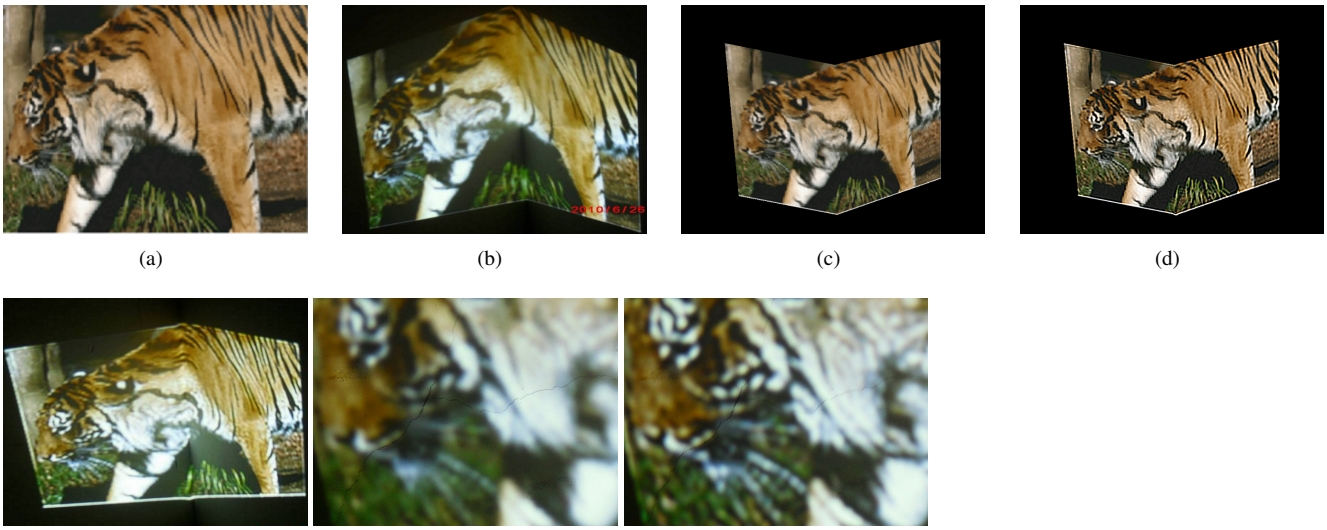
We have presented a novel technique to correct blur induced due to projection in dual planar environments. Our technique leverages the surface geometry information and homographies computed for geometric compensation to accurately compute defocus kernels across the planar regions in the projector space. These kernels are then used to compute the corrected input image by solving the minimization equation Eq. 2. Our results demonstrate that projection of the corrected image essentially reduces the effects of defocus blur, thereby extending the depth of field of the projector. While the presentation has been for dual planar surfaces, there is no fundamental limitation in the approach to consider more than two planar surfaces.

References

[1] O. Bimber and A. Emmerling. Multifocal projection: A multiprojector technique for increasing focal depth. *IEEE Transactions on Visualization and Computer Graphics*, 2006. 1

	Input Images			Output Images		
Image name	Geometry corrected	Defocus corrected	Percent increase	Geometry corrected	Defocus corrected	Percent increase
Cat	181081	240577	32.86	167383	218688	30.65
Leaf	176973	212434	20.04	180990	322382	78.12
Plant	109595	182400	66.43	92347	118577	28.40
Dog	68119	115329	69.31	67512	78905	16.88

Table 1. Comparison of number of edge pixels detected in geometry corrected versus defocus corrected input and output images.



(a) (b) (c) (d)

(e) (f) (g)

Figure 6. (a) Original input image that needs to be shown to an audience. (b) Output as seen by the audience if (a) is projected. (c) Input image corrected for geometry. (d) Input image corrected for both geometry and defocus. (e) Output (f) Zoomed output as seen by camera with only geometric correction (g) Zoomed output as seen by camera from the algorithm. Additional results are available in the supplementary material & at www.cse.iitb.ac.in/graphics/~sladha/publications.

[2] M. S. Brown, P. Song, and T.-J. Cham. Image pre-conditioning for out-of-focus projector blur. In *Computer Vision and Pattern Recognition*, 2006. 1

[3] S. Chaudhuri and A. Rajagopalan. *Depth from defocus: a real aperture imaging approach*. Springer, 1999. 3

[4] C. Jaynes, S. Webb, and R. M. Steele. Camera-based detection and removal of shadows from interactive multiprojector displays. *IEEE Transactions on Visualization and Computer Graphics*, 2004. 2

[5] Y. Oyamada and H. Saito. Focal pre-correction of projected image for deblurring screen image. In *Projector Camera Systems*, 2007. 1

[6] Y. Oyamada and H. Saito. Blind deconvolution based projector defocus removing with uncalibrated projector-camera pair. In *Projector Camera Systems*, 2009. 1

[7] K. Paidimarri and S. Chandran. *Computer Vision, Graphics and Image Processing*, pages 289–298. Springer Berlin / Heidelberg, 2006. 2, 3, 4

[8] Y. Sheng, T. Yapo, and B. Cutler. Global illumination compensation for spatially augmented reality. In *European Computer Graphics Conference*, 2010. 2, 4

[9] L. Zhang and S. Nayar. Projection defocus analysis for scene capture and image display. *ACM Transactions on Graphics*, 2006. 1, 2, 3