HOCHSCHULE DER MEDIEN

rise visual effects studios

Automated physically-based defocus and lens aberrations by metadata in a modern visual effects pipeline

by

Philipp Bauer

Student - ID: 30378

In fulfilment of the requirements for the

Degree of Bachelor of Engineering (B.Eng.)

in

Audiovisual Media

at

Media University Stuttgart

RISE | Visual Effects Studios

Berlin, January 20th, 2019

First examiner	***Prof. Katja Schmid***
Second examiner	***Rayk Schroeder***

Confidentiality clause

This bachelor thesis ‘Automated physically-based defocus and lens aberrations by metadata in a modern visual effects pipeline’ contains confidential data of RISE FX Ltd.

This work may only be made available to the first and second reviewers and authorized members of the board of examiners. Any publication and duplication of the bachelor theses – even in part- is prohibited until July, 4th 2019.

An inspection of this work by third parties requires the expressed permission of the author and RISE FX Ltd.

Sperrvermerk

Die vorliegende Bachelorarbeit mit dem Titel „Automated physically-based defocus and lens aberrations by metadata in a modern visual effects pipeline“ beinhaltet vertrauliche Informationen der RISE FX GmbH.

Diese Bachelorarbeit darf nur vom Erst- und Zweitgutachter sowie berechtigten Mitgliedern des Prüfungsausschusses eingesehen werden. Eine Vervielfältigung und Veröffentlichung der Bachelorarbeit ist auch auszugsweise innerhalb der Sperrfrist, 4. Juli 2019, nicht erlaubt.

Dritten darf diese Arbeit nur mit der ausdrücklichen Genehmigung des Verfassers und der RISE FX GmbH zugänglich gemacht werden.

Ehrenwörtliche Erklärung

„Hiermit versichere ich, Philipp Bauer, ehrenwörtlich, dass ich die vorliegende Bachelorarbeit mit dem Titel: „Automated physically-based defocus and lens aberration by metadata in a modern visual effects pipeline“ selbstständig und ohne fremde Hilfe verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen wurden, sind in jedem Fall unter Angabe der Quelle kenntlich gemacht. Die Arbeit ist noch nicht veröffentlicht oder in anderer Form als Prüfungsleistung vorgelegt worden.

Ich habe die Bedeutung der ehrenwörtlichen Versicherung und die prüfungsrechtlichen Folgen (§ 24 Abs. 2 Bachelor-SPO (7 Semester) der HdM) einer unrichtigen oder unvollständigen ehrenwörtlichen Versicherung zur Kenntnis genommen.“

Blaustein, Datum, Unterschrift

Abstract

The use of automated algorithms and tools in the visual effects industry gains in importance. Thereby, already existing data is used, to automate recurring functions, without the need of manual adjustments.

One scope of this automatism is, based on camera and lens-values defined aberrations. Thereby, on the one hand, using a live action plate, these values are already captured by the physical parameters and adjustments of the camera. On the other hand, these values are created, using a virtual camera in a 3D software. These datasets are conveyed through the whole pipeline till the compositing department. Thus, it is possible to simulate, based on real-world lenses, physically-correct depth of field and lens aberrations fully automated during compositing. This thesis covers the complete setup and process of automated camera-based aberrations, including the physical derivation and two-dimensional, synthetic creation and automation of these aberrations.

Kurzfassung

Der Nutzen von automatisierenden Algorithmen und Tools in der Visual Effekts-Branche gewinnt immer mehr an Bedeutung. Dabei werden bereits vorhandene Daten genutzt, um immer wiederkehrende Funktionen ohne manuelle Eingaben automatisieren zu können.

Ein Anwendungsbereich dieser Automatismen stellen durch Kamera- und Objektiv-Werte definierte Aberrationen dar. Dabei sind diese Werte bereits bei realgedrehtem Filmmaterial durch die physikalischen Werte und Einstellungen der Kamera, und in 3D-Softwars durch die virtuell generierte Kamera definiert. Diese Datenwerte werden durch die Pipeline bis hin zum Compositing-Department transportiert. Dort können diese in physikalisch korrekte Schärfentiefe und Linsenaberrationen, umgerechnet werden. Diese Arbeit behandelt den kompletten Aufbau und Ablauf der Automatisierung kamerabasierter Aberrationen, von der physikalischen Erläuterung dieser bis zur zweidimensionalen synthetischen Erzeugung und Automatisierung.

Table of contents

List of Abbreviations

2D	two dimensional
3D	three dimensional
AOV	Arbitrary Output Variable
CG	computer generated
CGI	computer generated imagery
DoF	depth of field
JSON	JavaScript Object Notation
PBR	Physically based rendering
RGB(A)	red/green/blue/(alpha)
TD	Technical director
VFX	Visual Effects

1 Introduction

Due to the rising importance of high-end visual effects for feature films, television and commercials visual effects companies constantly optimize their workflows. While the amount of visual effects is rising, the required quality is increasing simultaneously. Therefore, a well-organized and optimized workflow is necessary in order to keep the costs low and quality high.

Being able to handle fast turnarounds and last-minute changes a lot of tasks are shifted into the compositing department. Thus, mostly all camera aberrations are simulated during compositing, resulting in less render time and faster artistic adjustments. However, all these camera related aberrations are based on certain camera settings, which are already defined in a previous department. Handling these values throughout the pipeline, the aberrations can be set automatically, while keeping the ability of manual overrides.

Besides a theoretical introduction, this thesis aims at the practical use of such systems in a modern visual effects workflow. The work focuses on a camera- and physically-based defocus and aberrations implementation in The Foundry Nuke. The content of this thesis aims at people familiar with the softwares and techniques in a modern visual effects pipeline, especially VFX-, compositing, CG-supervisors and compositing artists and TDs.

2 Real-world photographic particularities

In the real-world cameras and lenses are subject to physical laws. Based on these, specific camera effects are unpreventable, some are often used to create artistic effects and images. On the one hand some of these are directly visible, like depth of field, on the other hand some are more subtle, e.g. chromatic aberration.

2.1 Pinhole camera

The most basic and simple camera setup is the pinhole camera. It consists of an opaque box with a tiny hole in one side of the box. Light falls through the hole and projects a horizontally flipped image of the outside to the opposite side.[1]

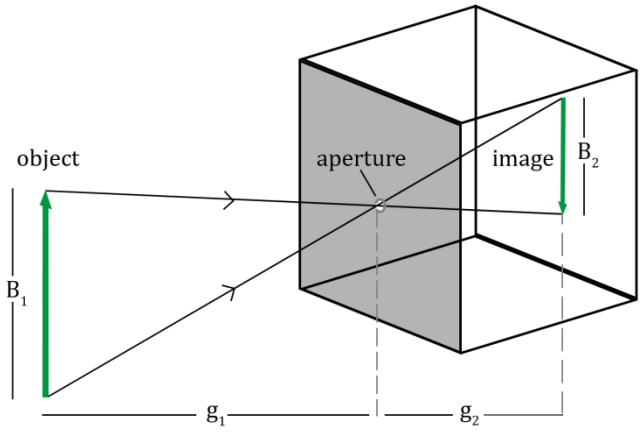


Figure 2-1 | **Pinhole camera principle**

Based on the simplicity of the camera there are only three parameters you can modify, influencing the output image. The image sharpness and brightness are controlled and based on the diameter of the hole, which can be defined as aperture. Furthermore, the output of the camera is based on the object distance and the distance between the pinhole and the photographic film. All these characteristics are adapted from the theorem of intersecting lines.

[1] Cf. Allen et al (2009), p. 6

$$\frac{B_1}{g_1} = \frac{B_2}{g_2} \tag{2.1}$$

Increasing the aperture is causing a brighter image, but as a result $B2$ is getting bigger as well. Thus, a single point in the real world is getting bigger on the projection, which decreases the sharpness of the output image.

Decreasing this effect requires concentrating the light on the film layer of the camera. This can be achieved by using optical lenses in front of the image layer.

2.2 Physical fundamentals of a lens

Optical lenses alter the incoming rays of a light source. These directional changes are subject to physical laws, which will be explained in the following chapter.

2.2.1 Characteristics of light

The definition of light is based on its dualism. On the one hand, light is electromagnetic radiation, a photon. On the other hand, it has a certain portion of the electromagnetic spectrum, and therefore is a wave.

Thereby, the spectrum of visible light of the human visual system is defined by the wavelength of the light in a range of 380-780 nanometers.[2] In geometrical optics, light is treated as particles called photons. Furthermore, the trajectories that these photons follow are termed rays.

[2] cf. Selan, 2012, p 7

2.2.2 Geometrical optics

The science of geometrical optics describes the approximate action of a simple lens, which is ruled by the principles of reflection and refraction.[3] Therefore, it deals with the rectilinear propagation of light in a straight line, without changing direction, encircled by a homogeny media.[4] When the light ray meets a surface, reflection and refraction occur and the light ray bends.[5]

Reflection occurs when a light ray collides with a surface, which does not absorb the energy of the wave and bounces it away from the surface. Thereby the ray of light approaching the surface is called 'incident ray' and the ray leaving the surface is known as 'reflected ray'.

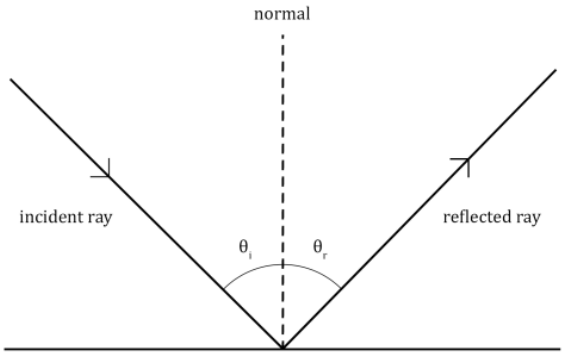


Figure 2-2 I **Law of reflection**

The projected straight line through the intersection of these two lines, vertical to the reflecting surface is the 'normal'. Using these you can measure the angle of incidence, θi and the angle of reflection θr.

Therefore, the angle of incidence equals the angle of reflection, according to the 'Law of Reflection'.

$$\theta_i = \theta_r \qquad (2.2)$$

In addition, the amount of reflection is highly dependent upon the smoothness of the reflecting surface. Almost totally smooth surfaces, which imperfections are smaller than the wavelength of the incident light, reflect virtually almost all the light rays equally. This type of reflection, based on the smooth surface, is

[3] Cf. Sutter et al.
[4] Jenkins et al. (1976), p. 4
[5] Cf. Kohlrausch (1996), p. 459.

called specular reflection. It is visible, using a mirror, which is resulting in a sharp reflection.

Besides the specular reflection, which is quite rare, the diffuse reflection is way more common. Most objects in the real world are convoluted surfaces, which reflect the incident light rays in all directions. As a result, the reflection is way softer or even not directly noticeable as a reflection. If an object would not reflect any of the incidents light you could not see it at all.

Alongside reflection, refraction is bending the ray, as it passes from one medium into another. Crossing the boundary from one transparent medium into another is causing a loss in speed, which as a result bends the ray.

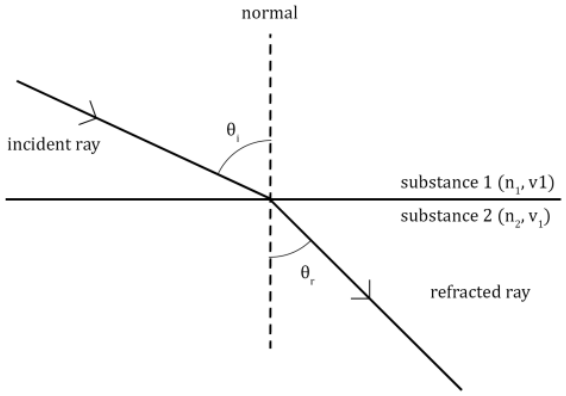


Figure 2-3 I **Snell's law**

This dependence of the media, the ray is travelling through, is made explicit in *Snell's law*. These states that the ratio of the sines of the incident and reflection ray are equal to the ratio of the indices of refraction (n). These indices are constant values for every medium. [6]

$$\frac{\sin(\theta r)}{\sin(\theta i)} = \frac{n_1}{n_2} = \frac{v_2}{v_1} \tag{2.3}$$

2.2.3 Types of lenses

The construction of a simple lens is based on two spherical surfaces. Thereby, lenses can be either convex or concave depending upon whether they cause light rays to converge into a single focal point or diverge outward from the optical axis.[7]

[6] Cf. Halliday (2009), p. 1019f.

[7] Cf**.** Abramowitz et al.

Convex lenses, also known as positive lenses, can be comprised of one or two convex surfaces. Furthermore, common characteristics of this lens type are a thicker, bulging outwards surface in the center than at the edges and the magnification of objects. These types of lenses converge incoming light rays, parallel to the optical axis, and focus them at the focal plane. The distance of the lens to this focal plane is called focal length.

In comparison concave, or negative, lenses diverge, spread parallel incident light rays. This is caused by the surface, which consists of at least one concave surface that is thinner in the center of the lens than at the edges.[8]

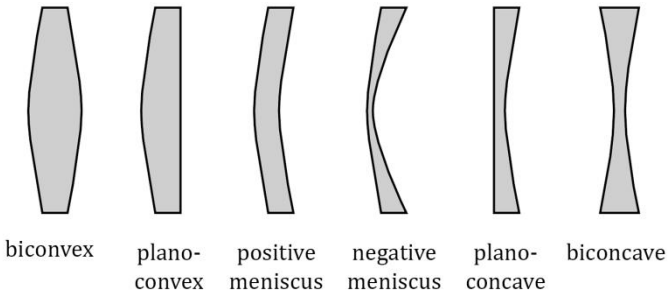


Figure 2-4 I **Types of lenses**

2.2.4 Image formation with lenses

Image formation is mostly based on the refraction of a lens, which is causing the light ray to bend. Thus, a convex lens brings all incoming parallel rays together at a point. The two-dimensional plane of this point is called focal plane and the distance of the camera to this plane is the focal length.

Based on the distance of an object in front of the lens and the focal length an image is projected. This image on the virtual plane is a mirrored representation of the real-world.

$$\frac{1}{S_1} + \frac{1}{S_2} = \frac{1}{f} \tag{2.4}$$

[8] Cf. Hecht (2002), p. 264

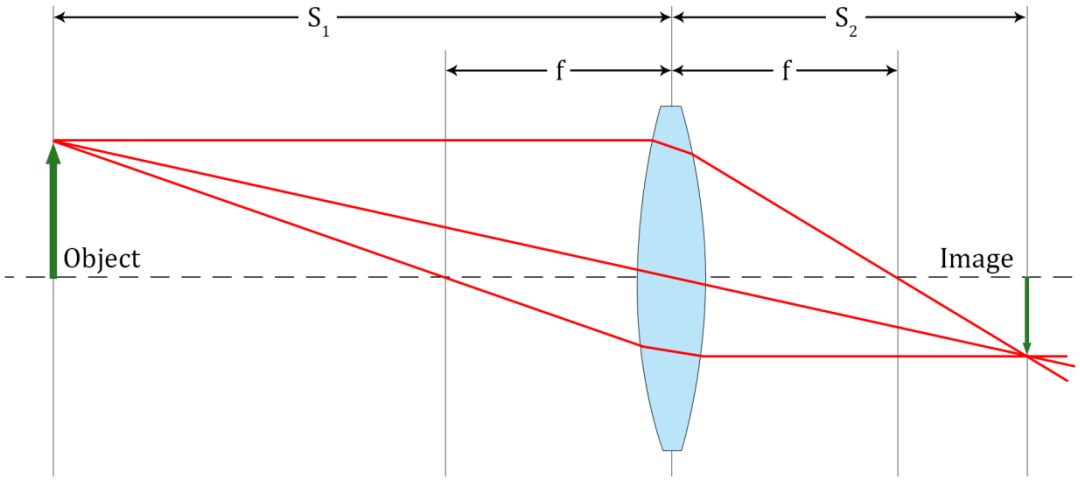


Figure 2-5 | **Image formation with a thin lens**

2.2.5 Depth of field

Every optical imaging system can only precisely focus on one distance at a time. Thus, everything, besides objects directly in the two-dimensional focus plane appears with a loss in sharpness. However, depth of field describes the range in front or behind the point of focus where objects remain acceptably sharp and in focus.

The loss of sharpness occurs, because a lens is unable to reproduce a point source to a single point on the camera sensor. Thus, the image is losing sharpness and appears blurred. The diameter of the defocused point is called circle of confusion.

2.2.6 Circle of confusion

In optics, circle of confusion describes the diameter of a blurred light point, caused by a lens. Thereby, the blurred spot is shaped like the aperture, but in terms of mathematical calculation we assume it is a circle.

Based on the focus distance S_1, the distance of an out of focus point S_2 and the aperture diameter A the diameter C of the blurred point can be calculated.

$$C = A\frac{|S_2 - S_1|}{S_1} \tag{2.5}$$

Using the lens equitation and expressing the magnification in terms of focused distance and focal length the physically-based circle of confusion can be calculated.

$$CoC = \frac{|S_2 - S_1|}{S_1} * \frac{f^2}{N(S_1 - f)} \tag{2.6}$$

In terms of cinematography the term circle of confusion is used to describe the depth of field. Therefore, it specifies the part of an image that is acceptable sharp.

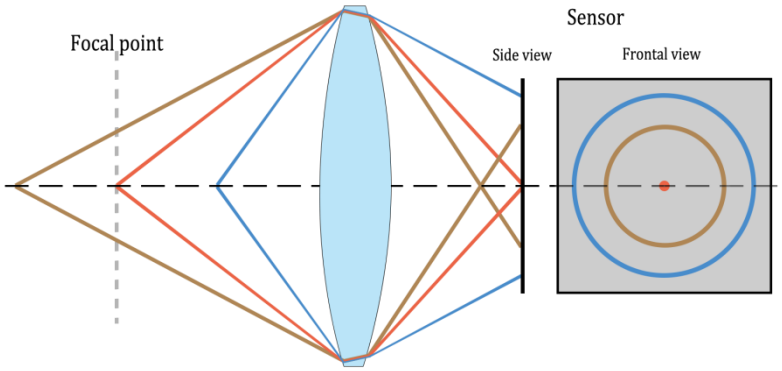


Figure 2-6 | **Circle of confusion**

2.2.7 Factors affecting depth of field

The circle of confusion and therefore, depth of field too, depend on different factors. Besides the variables of the circle of confusion formula, the sensor size of the camera affects the depth of field, too.

2.2.7.1 Aperture

Aperture is the most common factor, which is controlled, changing the depth of field in photography. Adjusting the aperture does not change the composition, instead it influences the depth of field and brightness of the image. Because of the change in brightness and aberrations, adjusting the aperture is rarely used in cinematography.

Based on a closed aperture, a high f-stop number, the light rays are forced to get through a smaller opening. This is narrowing the light beam and increases the distance of the two last points, which are acceptable sharp. Thus, the image has a higher depth of field.

A wide aperture, small f-stop number, results in a smaller depth of field region. Furthermore, the overall blur amount, besides the focal plane is higher.

2.2.7.2 Focal length

The focal length being in the numerator with the exponent two causes a larger circle of confusion, if the focal length is increased. Thus, the depth of field is shallower. Furthermore, the focal length is mostly controlling the range before and behind the focal plane. Long focal lengths produce a more evenly distributed depth of field around the focal plane than short focal lengths.

2.2.7.3 Focal plane

Another denominator of the circle of confusion formula is the focal plane. That is why focusing on a larger distance to the camera is causing a greater depth of field region. Thus, the focal plane does not just define, where the sharp area of the image is, but it is influencing the depth of field amount, too.

2.2.7.4 Sensor size

For a given aperture and effective focal length, a larger sensor is causing a shallower depth of field. Furthermore, the sensor size is influencing the scale of the circle of confusion, too.

2.2.8 Bokeh

The aesthetic quality and shape of the circle of confusion is called bokeh. It describes the look of a blurred point in the out-of-focus part of the image. Thereby, the look is controlled by the shape of the aperture and the overall lens design.

Some lenses for example create different bokeh shapes, depending on the position on the frame. Bokeh close to the edges get stretched, curved and often appear cut.

Furthermore, the shape of the aperture is mostly visible on small highlights, which are totally out of focus.

2.3 Optical aberration

In theory, an optical system converts all rays of a single point in object plane to the exact same point in the image plane. The influences which cause these rays after transmitting through the optical system to converge to different points are called aberrations.

An optical aberration blurs or distorts an image formed by an optical system, compared to the original. Moreover, the distortion/blur depends on the kind of aberrations and lens specific parameters. These optical aberrations are classified as either monochromatic or chromatic aberrations. Thereby, monochromatic aberrations can be specified as distortion, field curvature, astigmatism, coma and spherical aberration, which are also known as Seidel's aberrations.

2.3.1 Chromatic aberrations

Chromatic aberrations describe the divergence of rays of different wavelengths caused by a lens. This artifact occurs because the refractive index (n) of every optical glass formulation varies with wavelength (λ).[9] The refractive index being depended on the wavelength causes a stronger refraction of shorter wavelengths (blue), than of long wavelengths (red). This is causing white light being split into its spectral colors.

$$n = \frac{c}{v}, \qquad with\ v = \frac{c}{T} \tag{2.7}$$

Consequentially, different wavelengths result in a different ray-bending amount and thereby the focal point (F) of each wavelength is varying on the optical lens axis. This inability of an optical lens to bring all wavelengths into a common focus results in a slightly different image size for each predominant color, which

[9] Cf. Yehuda Levi (1968), p. 409

visually leads to colored fringes. These outlines are mostly visible on high-contrast images.

Based on the shape of a lens, it is unable to bring all the wavelengths to the same focal plane towards the lens periphery. That is why chromatic aberration is mostly visible towards the frame of an image.

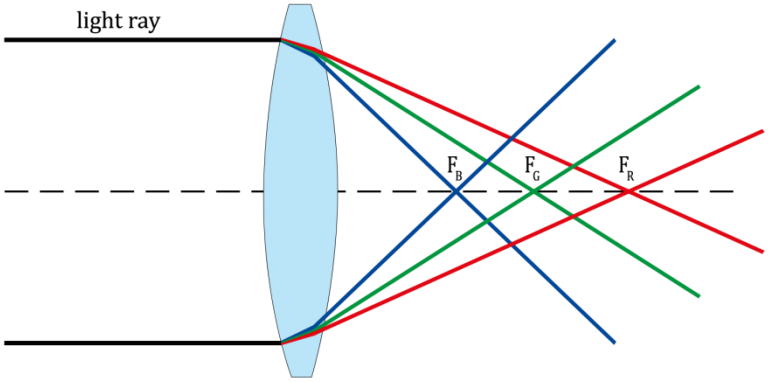


Figure 2-7 | **Chromatic aberration, different F for different wavelengths**

2.3.2 Lens distortion

Unlike most aberrations, lens distortion is manifested by changes in the shape of an image rather than the sharpness or color spectrum. There is a distinction between two opposite cases, pincushion and barrel distortion, which are the most prevalent types.

Assuming the photographic lens is radially symmetric, distortion can be described as a stretching or compression of the image from the optical axis.

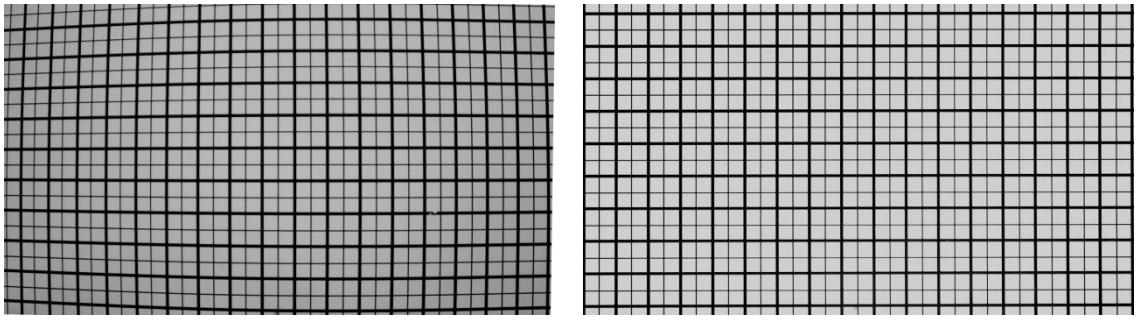
Its origin lies in the fact that the linear magnification, may be a function of the off-axis image distance.[10] Magnification is the process of increasing or decreasing the size of an image by an optical system. Based on the lens shape the magnification amount is changing. A barrel distortion is caused by a higher magnification in the image center

[10] Hecht (2017), p. 277

than at the periphery, which causes the image edges to shrink around the center.

Furthermore, the focal length of the optical lens is influencing the magnification and therefore the distortion, too.

$$M = \frac{f}{f - d_o} \tag{2.8}$$



Lens distortion: WalimexPro T3, 1 14mm ED AS

Lens grid without distortion

Figure 2-8 | **Lens distortion**

2.3.3 Vignetting

Vignetting describes the reduction of brightness or saturation towards the lens edges, compared to the image center. This imaging phenomenal happens with virtually every optical system and can arise for several fundamentally different reasons.

Natural vignetting occurs when light beams reach different locations on the camera sensors at different angles. This variation in angles results in a gradual darkening of the image. On the one hand this effect is manifested by the different distances of a light ray, which has to travel to the edges of the sensor compared to the center. Thereby, the light intensity (I) becomes progressively dimmer with an increasing distance (d), caused by the inverse-square law.

$$\frac{I_1}{I_2} = \frac{d_2^2}{d_1^2} \tag{2.9}$$

On the other hand the lens aperture, seen by the light rays appears more elliptical. Thus, the lens pupil is narrower when viewed at an angle.

Mechanical vignetting is caused by matte boxes, filter rings or other objects physically blocking light in front of the lens. This causes an abrupt vignetting effect, only visible in the corners.[11]

2.3.4 Field curvature

Field curvature is an optical aberration which affects the shape of the image plane on the sensor. When light is focus through a curved lens, the image plane produced by the lens will be a curved surface, too. The resulted image projection shape is known as Petzval surface.

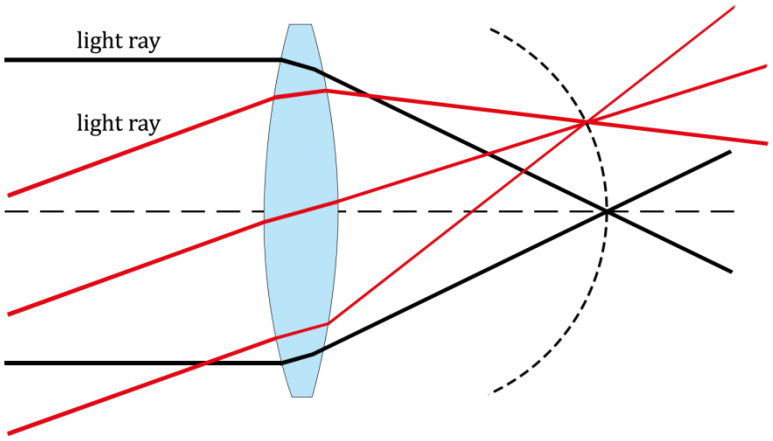


Figure 2-9 | **Field curvature**

This conversion of a flat plane into a Petzval surface, manifested by the lens, results in multiple focal planes on the sensor side. The camera sensor, however being a flat surface, is causing less sharpness towards the lens edges of the image.

The curvature strength can be mathematically described with the Petzval-sum. For an optical system, constructed with multiple thin lenses, with the focal length f_i and refraction index n_i applies:

$$P = \sum_{i=1}^{k} \frac{1}{n_i f_i} \tag{2.10}$$

[11] https://www.red.com/red-101/lens-vignetting

If the Petzval-sum equals null, the Petzval-surface is flat and therefore there is no field curvature effect. Furthermore, if there is also no astigmatism, the image plane will be totally flat and besides depth of field, sharp after the lens, too.

2.3.5 Astigmatism

Astigmatism is caused by the off-axis-image of a specimen point appearing as a line or ellipse instead of a single point. Thus, this aberration highly depends on the oblique angle of the light beam entering the lens. There is a distinction between tangential and sagittal astigmatism, depending on the angle of the off-axis rays. This causes the ideal circular point to blur into a diffuse circle, ellipse or line, increasing from the center.[12]

Oblique astigmatism causes sagittal and tangential rays in the object plane to focus at different distances in the image space. This distortion does not arise from the asymmetry of a lens, but an asymmetry in the nature of the optical paths followed by rays in the tangential and sagittal planes. Furthermore, the sagittal and tangential foci are located on curved planes.

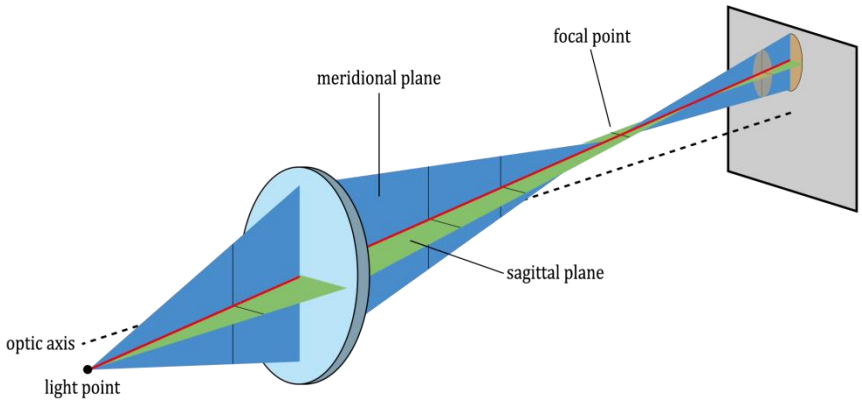


Figure 2-10 | **Astigmatism**

[12] Cf. Keller et al.

2.3.6 Coma

Coma, similar to astigmatism, arises out of an off-axis point of light in the object plane. This aberration focuses the image of a point at sequentially differing heights, which produces a series of asymmetrical spot shapes of increasing size. It derives its name from the cometlike appearance of the image of an object point located just off the lens axis.[13]

This aberration is a result of differences by light rays passing through the various lens zones as the incident angle increases. These differences are caused by the variation of magnification. Thus, a bundle of oblique rays, passing through the edge portions of a lens is imaged at different height than those passing through the center portion. Because of the asymmetric behavior of the comatic aberration, it is often considered as one of the most problematic ones.

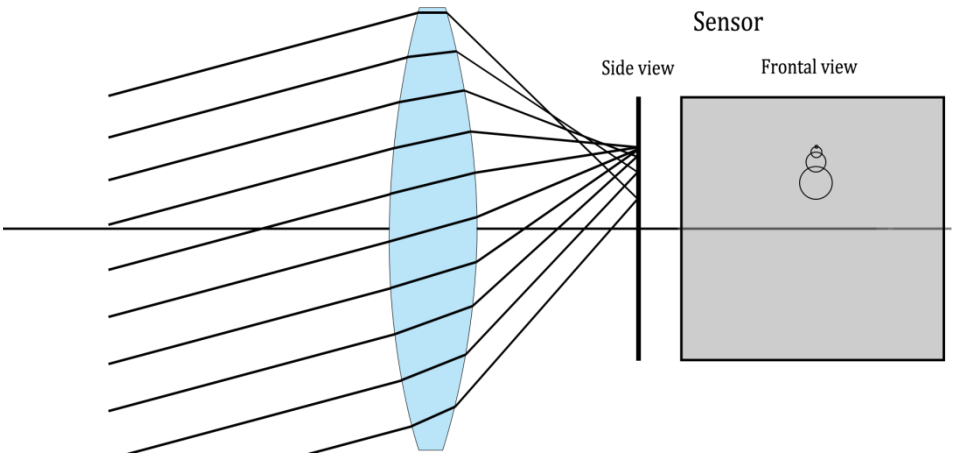


Figure 2-11 | **Comatic aberration**

[13] Jenkins et al. (1976), p. 162

3 Artistic use of camera effects in computer generated imagery

Most real-world lenses try to avoid or reduce the aberration effects to a minimum. However, most aberrations can be minimized but not eliminated. Further, some lenses are even designed to produce a specific aberration more significantly. Thereby, lens aberrations are a key part in the CGI- and VFX-industry, too.

3.1 Visual effects imagery

Visual effects involve the integration of computer-generated imagery into live-action footage. Therefore, the computer simulated and rendered images have to look photorealistic. This requires, adding all the lens specific aberrations, to create the perfect, invisible illusion and integration

Thus, the lens distortion of the live-action footage has to be exactly replicated and added to the CGI. Further, depth of field, including the bokeh shape and all lens aberrations has to be matched exactly. That is why the use of camera aberrations is more technical than artistic in terms of visual effects.

3.2 Fully computer-generated imagery

Besides visual effect imagery, fully computer-generated imagery for animation movies is a big part of the film industry, too. These images are fully generated by using 3D software and a render engine, which is converting the virtual 3D scene into a two-dimensional image. However, these software packages do not reproduce physically accurate lens aberrations by default.

Nevertheless, these artifacts are often desired and therefore reproduced. *The Lego Movie* (2014) is an example of a full CG movie, using lens aberrations and defocus heavily. The cinematography philosophy was, it should look like it's all been made in someone's basement, perfectly lit and photographed.[14] Thus, the

[14] Cf. Welsh (2014)

small size of the Lego figures caused a shallow depth of field. Furthermore, the dirty look of real-world-lenses was implemented, in order to avoid a clean look, which would directly indicate the CG-camera.

Another full CG movie, using camera artifacts as a storytelling instrument is *Inside Out* (2015). The two worlds, inside the thoughts of the girl and the girls world, are additionally differentiated by the camera language. The outside world is based on real world locations, thus the camera produces artifacts, too. Therefore, lens distortion, depth of field, chromatic aberration and a focus-pull offset were implemented. Besides, the inside, mind world is totally imaginary and so the camera is virtual and perfect.[15]

[15] Cf. Barraclough (2015)

4 Camera effects emulation for computer generated imagery

Nowadays most render engines are based on physically based rendering (PBR) or ray tracing, which convert the 3D scene into a 2D image, supporting accurate physical simulations of lighting, shadows and shading. Nevertheless, the camera is often based on a (modified) pinhole camera, which does not replicate real lens functionality. The resulting image shows a perfectly in focus output at every camera distance. Furthermore, a CG-camera is not based on optics, consisting of lenses, which are causing reflection and refraction. Therefore, a perfect image without any kind of aberration is the output of the render engine by default.

4.1 Depth of field processing while rendering

Using PBR or raytracing render engines, it is possible to render depth of field directly. To overcome the limitations of the pinhole camera a thin lens-based simulation takes place. Thus, the camera has further adjustable attributes, like the focal length, sensor size, f-stop and focus distance.

There are different techniques and algorithms that approximate depth of field. The most accurate method is the ray-traced depth of field. Thereby, rays are casting from across the lens, rather than from a single point. Using appropriate statistical distribution across the lens calculates accurate depth of field. Nevertheless, this method highly depends on the sample count. In order to create an acceptable amount of noise the samples and render time are quite high.[16] Based on the high render time and the fact, that adjusting the depth of field requires rendering the image again, the simulation of this effect is mostly shifted into compositing.

[16] Cf. Cook et al. (1984), p. 139f

4.2 Depth buffering

In order to convert a virtual 3D scene into a two-dimensional array of pixels, the render engine has to take the depth values of each object into account. Therefore, a Z-buffer algorithm iterates over each pixel of the frame. If more than one object is located on the pixel, only the object with the smallest distance to the camera will be rendered to its *rgb* values. Thus, all the depth values are already available during render-time and just have to be stored to a separate file. The resulting image represents the depth value of the rendered scene.

4.3 Linear blur based on depth pass

Besides the final image, the render engine is able to produce secondary images, too. These are called Arbitrary Output Variables (AOVs) and there can be any number of them generated simultaneously and each one may got a different file or use different pixel filter settings.[17]

Using one of these AOVs the depth pass can be stored directly with the beauty image. Furthermore, this channel can be used with the industry standard application for node-based compositing *The Foundry Nuke*. Thereby, it is important, that this channel is rendered without anti-aliasing, so the depth information is correct on the edges.

The most basic implementation of a post processing depth-blur is reading the depth-channels color value of each pixel, which represents the camera distance and applies it as the corresponding blur size. Therefore, simple iterations over both image passes, the beauty and depth are necessary. This can be achieved, using the Nuke internal BlinkScript. The Foundry's Blink is a C++-based image processing framework designed to allow complex algorithms to be implemented in a device-independent manner.[18]

Nevertheless, this implementation is not usable, because the artist has no control at all over the look of the depth blur. That is why the depth values have to

[17] Pixar Animation Studios (2018)

[18] The Foundry (2013)

get normalized and an in- and decreasing ramp towards an adjustable focal plane is necessary. These functions can be added by simple subtracting the focus distance from the pixel depth values

This results in an adjustable, depth based post processing blur, which is already implemented in Nuke, the *ZBlur* node. Nevertheless, a real lens is not producing a blurred/soft image, but a lens shape based bokeh.

```
1.  void process(int2 pos){
2.    const float pixDepth = depth(pos.x, pos.y, 0);
3.    const float focusDepth = bilinear(depth, focus[0], focus[1], 0);
4.    float z = fabs(pixDepth - focusDepth);
5.  //use the depth to set the blur size for the current pixel
6.    const int blurSize = clamp((int)(z * size + 0.5f), 0, size);
7.    SampleType(src) sum = 0.0f;
8.    for(int j = -blurSize; j <= blurSize; j++)
9.      for(int i = -blurSize; i <= blurSize; i++){
10.       sum += src(i, j);
11.     }
12.   //normalise and set dst image
13.   const int blurDiameter = 2 * blurSize + 1;
14.   dst() = sum / (blurDiameter * blurDiameter);
15. }
```

Code 4-1 I **ZBlur Blink script process method implementation**

4.4 Convolution Kernel based on aperture shape

The term convolution describes the treatment of a matrix by another one, namely convolution matrix or kernel. Thereby, each pixel of the image gets multiplied with and kernel and the results are added up. This means that for each pixel of the input matrix the new value is calculated by centering the kernel image on the pixel, examining its neighbors, multiplying each pixel value by the corresponding pixel values in the filter image and then adding the results together. The sum of products is the new output image.[19]

In mathematical terms a one-dimensional Convolution can be described with the following formula. The variable f represents the input image, g the kernel and m the kernel size.

[19] The Foundry 2016, p. 319

$$(f * g)(i) = \sum_{j=!}^{m} g(j) * f\left(i - j + \frac{m}{2}\right) \tag{4.1}$$

Based on this a convolution kernel can be implemented inside Nuke. Using the kernel matrix to blur the image, it creates a defocus, which blurs the input based on the kernel shape. This is reproducing the bokeh effect of a real camera, but it is a uniform blur for the whole image.

4.5 Physically based defocus implementation

Creating a defocus, simulating a real-world camera lens, requires the combination of the linear blur, which is based on a arrangement of the depth pass and the convolution kernel. This is already implemented in Nuke, namely the *ZDefocus*. Using this node, the artist can specify the focal distance and blur size. Nevertheless, the resulting defocus still is completely camera and lens independent. In order to have a non-linear, physically- based defocus which is based and controlled by real camera settings, the circle of confusion will be calculated.

4.5.1 Circle of confusion calculation

Based on the depth channel, which stores the distance of the scene's objects to the camera, and the camera focal length, f-stop, horizontal sensor size and focal distance it is possible to calculate the circle of confusion for every pixel.

Writing the results of each pixel into a 2D-array, with the exact same resolution like the input image, results in an image, storing the physically correct and camera based defocus value of each pixel. The defocus, iteration over each pixel, reads the corresponding pixel circle of confusion value and sets the convolution kernel size.

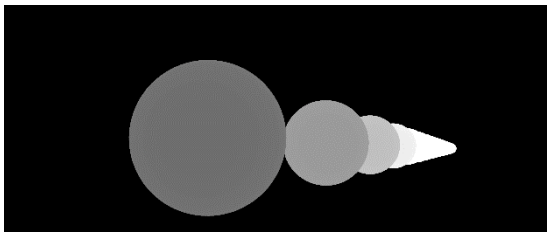
$$CoCPixel = \frac{\frac{|S_2 - S_1|}{S_1} * \frac{f^2}{N(S_1 - f)}}{sensor.width} * \frac{img.width}{2} \tag{4.2}$$

Besides the Nuke internal *BlinkScript* it is possible to calculate these pixel-based values with an *Expression-node*, too. An *Expression* allows the artist to apply complex mathematical formulas to a channel's values using C-like syntax expressions.[20] Furthermore, the processing time is tremendously faster, compared to the *BlinkScript*.

```
1.  void process(){
2.      SampleType(src) input = src();
3.      // get depth value
4.      float3 srcPixel(input.x, input.y, input.z);
5.      // calculate circle of coufusion
6.      float focusDist = focus * unit;
7.      float CoC = ((focusDist-srcPixel.x)/srcPixel.x)
8.                  *(pow(focal,2)/(fstop*focusDist-focal));
9.      float CoCpix = CoC / sensor * src.bounds / 2;
10.     // write coc to output image
11.     dst() = float4(CoCpix, CoCpix, CoCpix, input.w);
12. }
```

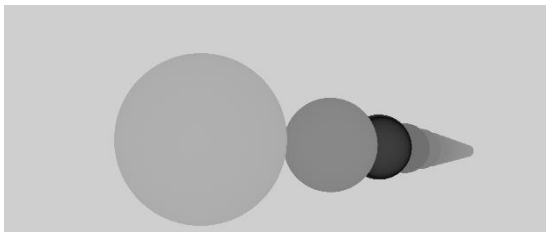
Code 4-2 I **Circle of confusion Blink script process method**

In addition to the blur/defocus values a node, applying these to a convolution kernel is necessary. Therefore, Nuke provides the *ZDefocus*, which convolutes a kernel shape over an image, based on the depth value. Setting the convolution math type to *direct* sets the blur value to the exact value, the circle of confusion calculation stored.

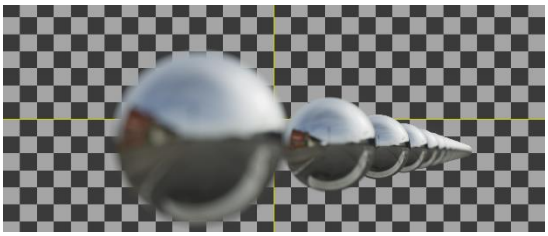


a) Depth pass rendering (Mantra)

b) Rendering (Mantra)



c) Circle of confusion based depth pass



d) Physical-based defocused image in Nuke

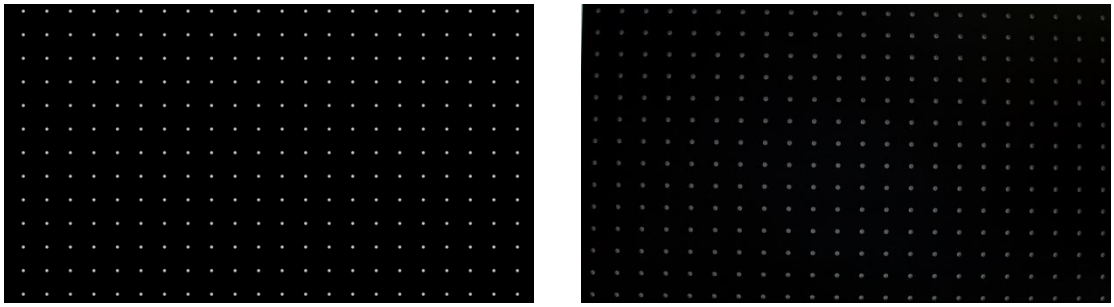
Figure 4-1 I **Physically-based defocus creation**

[20] The Foundry 2016, p. 240

4.6 Lens and camera dependent optical aberration

Besides the physically correct calculation of the defocus all the optical aberrations are visual approximations, based on real cameras and lenses. Thereby the effects are better adjustable, and the calculation time is minimized. Because real world lenses try to avoid these effects, they result in subtle and almost not visible aberrations. Using a non-physically correct calculation of these effects, therefore gives the artist more control and flexibility.

In order to visualize the aberrations a dot grid was designed, which can be captured on set with a real camera, too. Thus, it is possible to create real lens world based presets.



a) Nuke lens grid without aberrations

b) Dot-grid: WalimexPro T3, 1 14mm ED AS

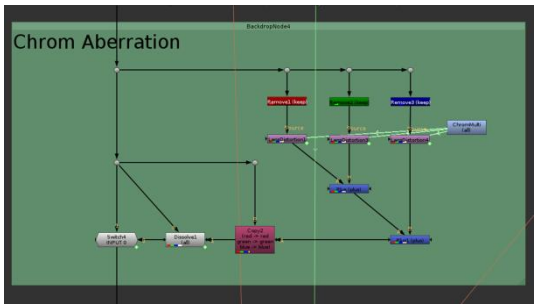
Figure 4-2 I **Dot-grid to capture/visualize optical aberrations**

4.6.1 Chromatic aberration

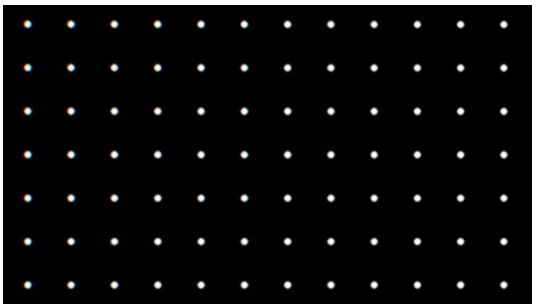
Based on the shape of a lens, chromatic aberration is getting stronger towards the edges of the image. To recreate this effect, splitting the image into its red, green and blue channels is necessary.

Using the theorem of additive color science, the addition of these three single channels will reproduce the exact same image. In the real-world chromatic aberrations are caused by different offsets of different wavelengths. By splitting the image into its *rgb* channels and applying a different offset/transformation for each of these will reproduce this effect. Furthermore, this aberration is getting stronger towards the edges of the frame. Therefore, the offset of each pixel can be controlled with the *LensDistortion* node in Nuke, which automatically increases the offset, based on the increasing distance of a pixel to the image cen-

ter. Keeping these offsets rather small will make the effect visible on the edges of an object, but the object itself remains, nearly, the same. The calculation and implementation of this distortion effect will be discussed in more detail in the following chapter.



a) Nuke node setup



b) Top-left corner output

Figure 4-3 I **Nuke chromatic aberration recreation**

4.6.2 Lens distortion

Using mathematical terms, a lens distortion is simply stretching an image to or from the center of itself. Thus, the distance of every pixel to the center is calculated and multiplied by the direction vector. Adding the result of this to the center position will calculate the new position of the current pixel. This formula is iterating over every pixel and results in a distorted image.

Using the Nuke internal BlinkScript, converting these theoretical formulas into a distorted image is possible. Dependent on the '*distStrength*' value the amount of the distortion is defined. Therefore, this is totally camera independent and not accurate enough.

```
1.  void process( int2 pos ) {
2.    // Calculate distance to center
3.    float2 posPix = float2( pos.x, pos.y );
4.    float2 distToCenter = center - posPix;
5.    // Calculate distortion strength
6.    float dist = length(distToCenter);
7.    float normDist = dist / max_dist;
8.    float targetDist = dist * ( 1 - distStrength * pow( normDist, 2) );
9.    //Apply distortion to output image
10.   float2 distPos = center - normalize(distToCenter) * targetDist;
11.   dst() = bilinear( src, distPpos.x, distPos.y );
12. }
```

Code 4-3 I **Blink script distortion process method script**

Besides this generic calculation of the distortion, a real camera-based lens distortion can be calculated. Capturing a lens grid, which basically is a representation of straight lines, can be used, to generate a lens specific distortion. Simplifying the calculation, the offset of each pixel to the straight line is calculated.

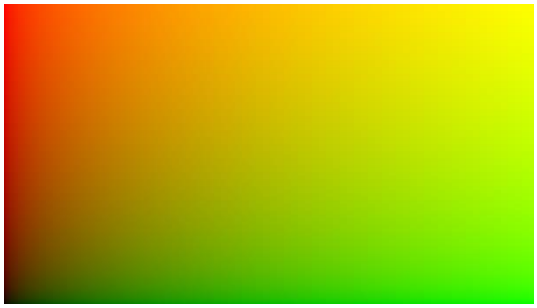
These offsets can be stored using an *UVMap*, which has a unique color ID for each pixel. An *UVMap* is based on two linear ramps and stores the x and y offset information of each pixel. Therefore, the red channel is representing the distortion in x-direction, the green in y-direction.

$$r = \frac{(x + 0.)}{width}, g = \frac{(y + 0.)}{height} \tag{4.3}$$

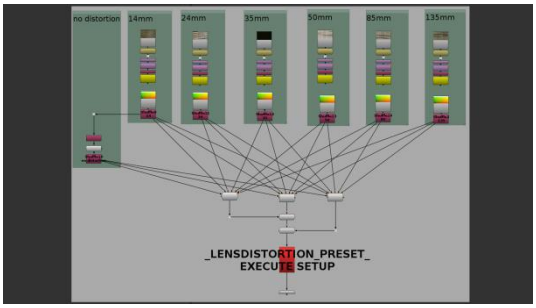
Applying the before calculated offset values of each pixel on this standard UVMap is visualizing the distortion. Using a STMap inside Nuke applies this visualization to the image and results in a distorted image.

This distortion has to be added to every CG rendering, because unlike a real-world camera lens, a CG-camera does not create these artifacts and all lines are straight. Applying the STMap is recreating this effect but pushing pixels towards the center after the sensor-level is producing areas with no image information on the corners and edges. Thus, the image has to be rendered with more image information on the edges, namely overscan. This overscan mostly adds extra information of about 10% of the final image size. Applying this distortion on this format and cropping the overscan after the distortion is creating the desired effect, without black pixels around the frame.

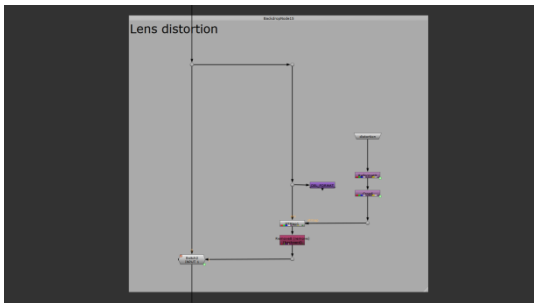
Every lens having a unique lens distortion causes, that this process has to be done for every lens by itself.



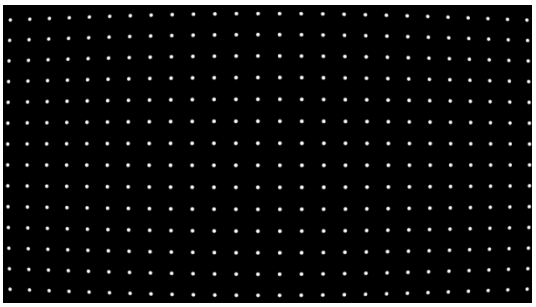
a) UVMap



b) STMap preset creation



c) Nuke node setup



d) Dot-grid with distortion

Figure 4-4 | **Nuke lens distortion recreation**

4.6.3 Vignetting

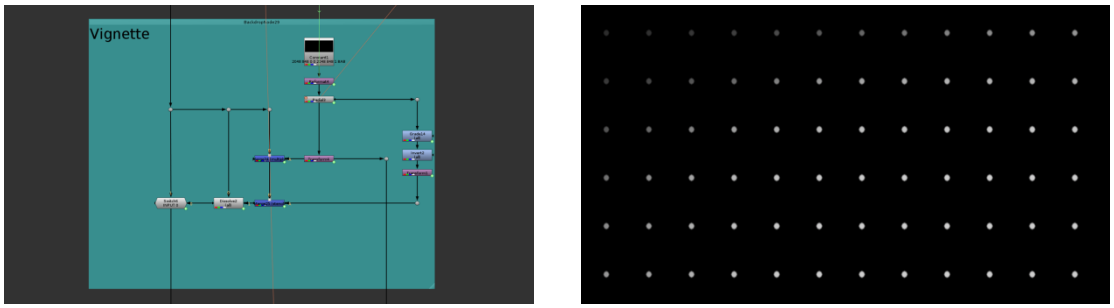
Like most lens related aberrations, vignetting occurs mainly on the image edges. Having a circular lens shape, this aberration effect has a circular or elliptical shape, too. Hence a radial falloff, from the center of the image, is necessary, recreating this effect.

Rest on the *Pythagorean Theorem*, $\sqrt{x^2 + y^2}$ is producing a radial gradient, with its center point at$(0,0)$. Subtracting this from the scale is inverting the gradient, which now has to get normalized, by dividing it by the scale.

The center point of the gradient can be defined by subtracting its x-coordinate from the current pixel x-value and correspondent for the y-value. Further control of the radial falloff shape can be added, by multiplying the square of the pixel distance to the center with a specific squeeze factor for each axis.

$$radial = \frac{scale - \sqrt{(scaleX(x - posX)^2 + scaleY(y - posY)^2)}}{scale} \quad (4.4)$$

This radial shape can be used as a mask, to create a brightness falloff on the image edges. Inside Nuke, there is already an implementation of this, the *Radial*-node.



a) Nuke node setup b) Top-left corner output

Figure 4-5 I **Nuke vignetting recreation**

4.6.4 Field curvature

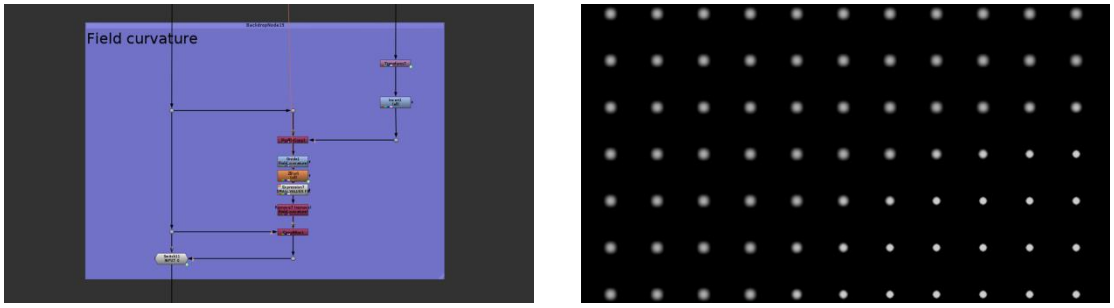
In a simple field curvature scenario, the light rays are only perfectly in focus in the center of the image. Based on the curved lens, the area of a sharp image plane is curved, too. Camera sensors though are straight, which is causing the sharpness of the image to drop, moving from the center of the frame. This is resulting in less resolution in the mid-frame and even less in the corners of the image.

Therefore, it is necessary to create a blur, which is increasing to the image edges in Nuke. Using a radial falloff, increasing from the center of the image can store the blur size. These values are visualizing the blur size of every pixel. Reading out these values and interpreting them as the blur size can be achieved with the following code.

```
1.  void process (int2 pos) {
2.      //Get blur size at current pixel
3.      const float blurSize = blurVal(pos.x, pos.y, 0);
4.
5.      SampleType(src) sum = 0.0f;
6.      for(int j = - blurSize ; j <= blurSize ; j ++ )
7.        for(int i = - blurSize ; i <= blurSize ; i ++ ) {
8.          sum + = src(i, j);
9.        }
10.     //Normalize blurred value and set into dst image
11.     const int blurDiameter = 2 * blurSize + 1;
12.     dst() = sum / (blurDiameter * blurDiameter);
13. }
```

Code 4-4 I **Input-based blur Blink script process method script**

This simple implementation is already implemented in Nuke, with a few more features, the *ZBlur* node.



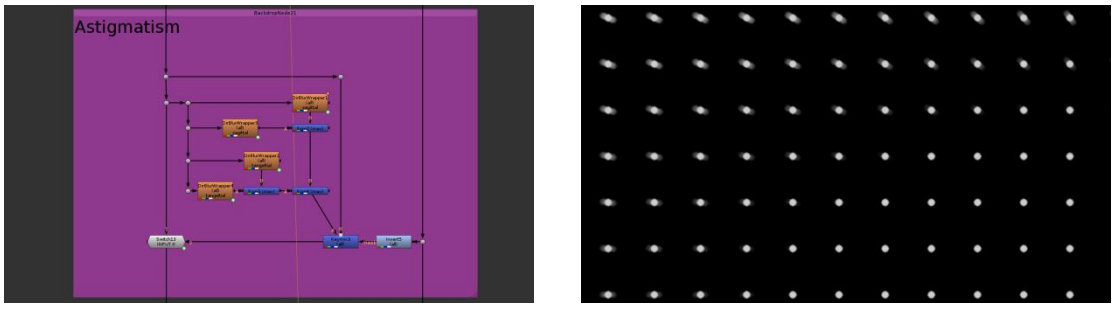
a) Nuke node setup b) Top-left corner output

Figure 4-6 I **Nuke field curvature recreation**

4.6.5 Astigmatism

Astigmatism blurs the ideal circular point into a diffuse circle, ellipse or line, increasing from the center. Furthermore, the strength of this aberration highly depends on the off-axis angle. Thus, a vector-based blur is necessary, so both sagittal and tangential astigmatism can be simulated. A directional based blur node is already implemented inside Nuke, the *DirBlur*.

Thereby, different blur-direction types can be simulated. A zoom blur distorts the image on the meridional plane, recreating sagittal astigmatism. The perpendicular plane, tangential astigmatism, is simulated with the *DirBlur* set to radial, which distorts the image vertically compared to the sagittal blur.



a) Nuke node setup b) Top-left corner output

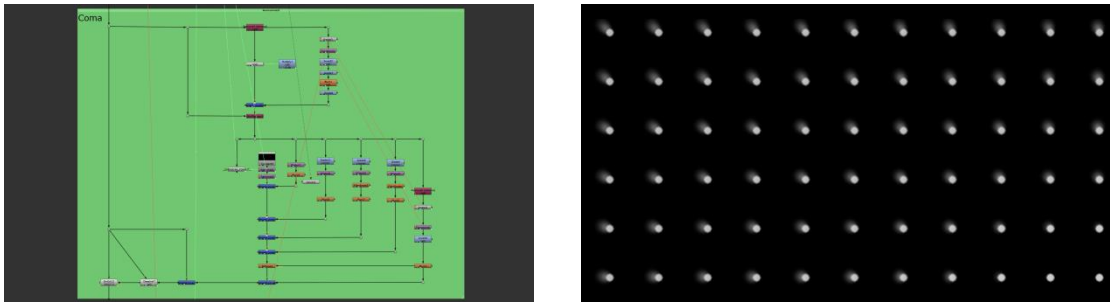
Figure 4-7 I **Nuke astigmatism recreation**

4.6.6 Coma

Comatic aberration focuses the image of a point at sequentially differing heights, which produces a series of asymmetrical spot shapes of increasing size. The different heights can be simulated using an UVMap. Therefore, the UVMap gets offsetted, so the origin of the map is in the center of the frame ($r = 0, g = 0$). This map can be applied onto the image using the *IDistort* node in Nuke. This node wraps the input image based on the image's UV channels. The U and V values are offsets for where a pixel will come from.[21]

Merging multiply layers, with increasing U and V values over each other creates the differing heights of the image point. Furthermore, with the increasing map values, the image gets blurred and eroded, in order to simulate the comet like effect. The advantage of using the *IDistord* instead of a normal transform node to scale the image is that a non-linear scale on one layer is possible.

Again, this aberration is masked by a radial fall off, because it mainly occurs on the lens edges.



a) Nuke node setup b) Top-left corner output

Figure 4-8 I **Comatic aberration implementation in Nuke**

[21] The Foundry 2016, p. 492

5 Automated camera-effects-through metadata

All lens aberrations are based on certain camera values. Thus, it is possible to automatically set the values of these aberrations, based on the camera settings.

Therefore, all relevant settings of the camera, created in a 3D application or by a real camera, have to be stored with the image data. Using certain image file formats, it is possible to store this data directly with the image information, consulting metadata. Metadata is data, which describes other sort of date, like creation date, timecode, etc. and can be modified by the user.

5.1 Houdini/Mantra rendering setup

This setup is based on the in-house RISE pipeline, which is using *SideFX Houdini* and its internal renderer *Mantra* as their main 3D software. Nevertheless, the output of specific metadata is possible using other 3D applications and renderers, too.

The Mantra render node is already providing an additional metadata section. Thereby *EXR Attributes* can be added, which can create multiple parameters and their values and save them as separate metadata values. Therefore, a simple python expression, with a Houdini specific syntax is necessary.

```
"{'lens/focal':"+pythonexprs("hou.parm(hou.parm('camera').eval()
+'/lens/focal').eval()")+"}
```

Code 5-1| **EXR Attributes camera expression syntax**

First of all, the name of the metadata parameter is defined, which in this example is '*focal*'. The corresponding value is added by calling the selected render camera of the Mantra node and its parameter. This has to be done for all camera and lens related values, namely focal length, f-stop, focus, horizontal sensor size and the overscan amount.

```
"{'lens/focal':"+pythonexprs("hou.parm(hou.parm('camera').eval()+'/focal').eval()")+",
'lens/fstop':"+pythonexprs("hou.parm(hou.parm('camera').eval()+'/fstop').eval()")+",
'lens/focus':"+pythonexprs("hou.parm(hou.parm('camera').eval()+'/focus').eval()")+",
'lens/aperture':"+pythonexprs("hou.parm(hou.parm('camera').eval()+'/aperture').eval()")+",
'lens/overscan':"+pythonexprs("hou.parm(hou.parm('camera').eval()+'/overscan').eval()")+"}"
```

Code 5-2 | **Mantra EXR Attributes camera expression**

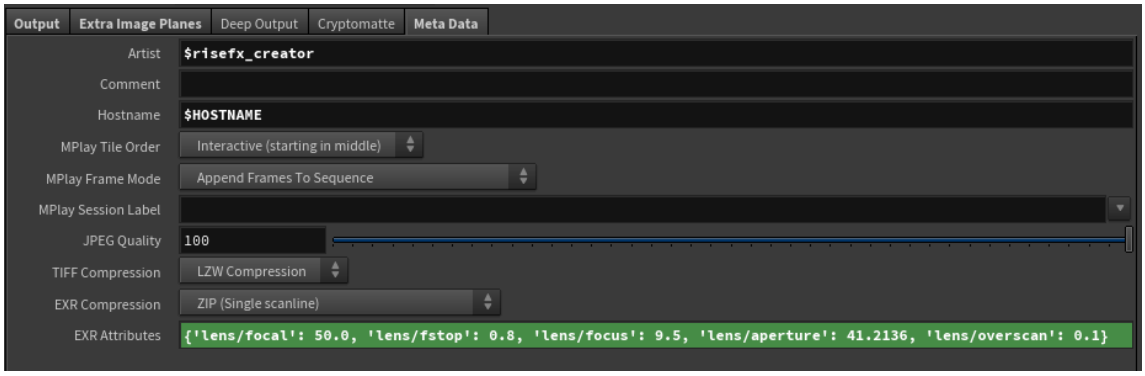


Figure 5-1 I **Houdini Mantra render node EXR Attribures set-up**

5.2 Metadata exchange with OpenEXRs

“OpenEXR is a high dynamic-range image (HDR) file format developed by Industrial Light & Magic for use in computer imaging applications.”[22] Besides the ability of being able to save up to 32-bit, lossless HDR images, using OpenEXR can generate Multi-part image files. Thus, EXR files can contain any number of separate, but related, images in one file. Furthermore, each file can be accessed independently of the others.

Another important feature is the ability of adding new or changing existing image attributes, e.g. strings, vectors, integers and floats, namely metadata. These attributes can be added during render time by a frame basis. In addition, it is possible to read and change these values in the compositing software Nuke.

All these features and some additional like the ability of storing deep image data, lead OpenEXR to the industry standard image format for CGI.

[22] Cf. http://www.openexr.com/ (2018-11-28)

5.3 Automation in The Foundry Nuke using metadata

The Foundry Nuke directly reads in the metadata embedded in an image when loading an image file. These metadata values are passed through the node tree, so the artist has access to them at any point in his script.

Using the python function *metadata()*, the user can edit, read or delete existing metadata or add new attributes. Thus, the camera values can be copied to the Expression, which is calculating the circle of confusion. Besides this, all aberration settings can be controlled based on the metadata camera values. This will be discussed in more detail in the next chapter.

a) Testscene raw beauty pass rendering

a) LensDefocus applied, based on metadata (focal length: 85mm, f-stop: 0.8)

Figure 5-2 | **LensDefocus Testscene**

6 LensDefocus documentation

This defocus is based on the physics of a camera lens. All camera settings can be set by the metadata automatically or adjusted by the user itself. Additionally, it is possible to control multiple aberrations and lens distortions and automate their values by the metadata, too.

6.1 Inputs

The LensDefocus group has three inputs and one output. The image input is the main one, which always has to be connected with the input image. Distortion, filter, camera and edgeMask are additional inputs, which can be used, but don't have to be connected. Nevertheless, it is recommended to use the distortion input, too.

Table 6-1 I **LensDefocus node inputs**

Image	This input can include any number of channels and layers, but to work probably, it should contain a z-depth and rgba channel. Defocus, aberration and distortion will be applied on this input image.
Distortion	Connect _LENS_DISTORTION_ group, which stores the distortion information.
Filter	Optional input to use own filter image.
Camera	Optional camera input, which can be used to bake the Nuke-camera settings of the connected camera to the defocus.
edgeMask	Input sets mask for object(s), whose depth-pass will be edge extended in order to reduce hard edge artifacts. Using Crypromatte is highly recommended.

6.2 Defocus

Based on the depth value and camera settings the circle of confusion is calculated for each pixel. This directly stores and sets the blur size. Choosing *Circle of Confusion* in the output dropdown menu visualizes this directly.

All camera settings can be set automatically by using *CG*, if the plate was rendered in Mantra*, ALEXA*, if an ARRI Alexa plate is selected and a CSV-file, including all the metadata values is stored, or *Nuke-camera*, if a Nuke camera is connected with the camera input of the node. Furthermore, the *Shotmachine*[23] button is doing exactly the same, just with some *Shotmachine* specific functions. To overwrite these values, simply delete the existing keyframes and set a new value. Checking use focal point overwrites the focus distance, created by the metadata and a new one can be picked.

To speed up the defocus, decrease the maximum setting. In addition, you can - analyze the depth values, which automatically will set the highest distance as the maximal depth value.

Using the filter tab will give you full control of the shape and look of the kernel filter. Furthermore, you have the ability, putting in your own image as a filter.

[23] *Shotmachine* is a RISE internal tool to automate specific compositing shots/tasks.

Table 6-2 I **LensDefocus defocus settings and knobs**

Channels	The effects are only applied to these channels.
GPU	Enable to use GPU for the defocus node.
Output	**result:** defocused image with aberrations and distortion **circle of confusion:** visualization of the actual blur size **aberration dots:** grid of dots for better aberration visualization **filter:** preview of the bokeh shape
Depth unit	Scale of world units. Should be the same scale, like in the CG-set-up.
Depth channel	Select desired depth channel.
Focal length	Focal length of the camera.
F-stop	F-stop of the camera. Does not influence the brightness
Sensor size	Horizontal sensor size of the camera [mm].
Focus distance	Distance of the camera sensor to the plane in focus.
CG	Bakes camera relevant values of the metadata into camera settings. Based on focal length specific aberration and lens distortion values are set.
ALEXA	Bakes camera relevant values of the Arri Alexa plate metadata into camera settings. Therefore, a CSV-file, created with the Arri MetaConverter is necessary. Based on focal length specific aberration and lens distortion values are set.
Nuke-camera	Bakes camera relevant values of the connected camera into camera settings. Based on focal length aberration and lens distortion values are set. If you are using a scanline renderer, you have to check *invert depth* and manually set the *overscan*.
Shotmachine	Sets an expression of the camera metadata values to the camera settings of the defocus.
Maximum	Maximal defocus value. Check *use analyze depth* to automatically set the highest depth value.
Analyze depth	Calculates the highest depth/blur value of the Circle of confusion and sets it as the maximum.
Edge extend depth objects	Based on the maskEdge input, the masked object(s) depth pass will be edge extended.
Depth layers	Control the number of depth layers that will be used for the blur.
Filter	Full control over the shape and look of the filter.

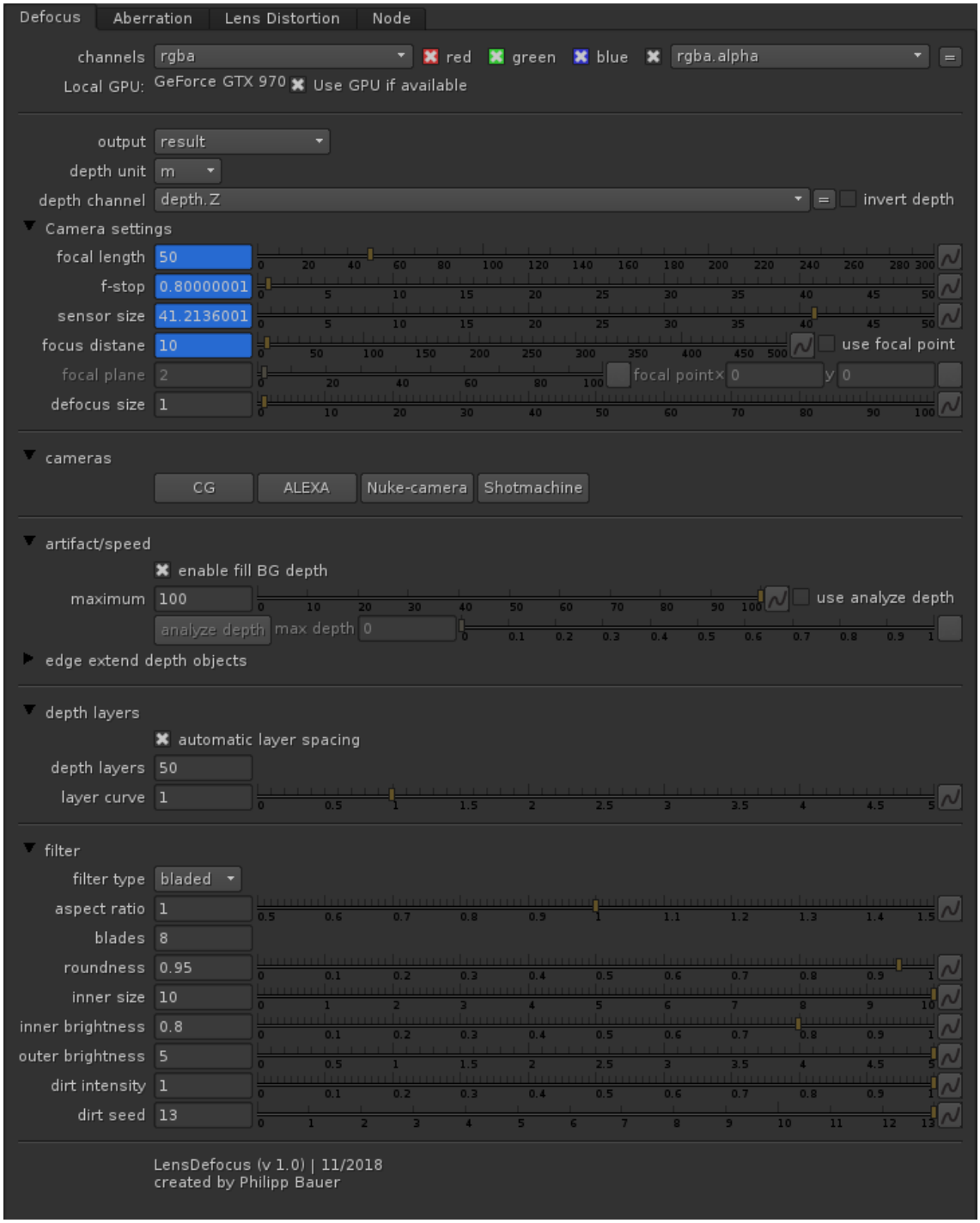


Figure 6-1 I **LensDefocus defocus user interface**

6.3 Aberration

In the Aberration-tab the artist has full control over multiple, typical lens aberrations. Furthermore, it is possible to set a white balance value in Kelvin.

By default, all values and knobs are locked, so every artist hast the same look for the same lens by default, which the supervisor provided. Nevertheless, the artist is able to change these values by checking the *edit* check box. To completely disable an aberration the artist can simply check individually *disable* for every effect.

Besides manually setting these knob values, you can set all knobs inside the *Aberration-tab* to lens specific values, by using the *use lens preset* button. Saving a new default aberration preset for a lens is possible by using *save as lens preset*. These presets are saved locally inside the nuke script.

Besides these locally stored presets, it is also possible to export the aberration presets of all lenses at once to a JSON (JavaScript Object Notation) file. JSON is a lightweight data-interchange language and format, which is completely language independent. Furthermore, it is on the one hand easy for artists to read and write, on the other hand it is easy for machines to parse and generate. These properties make JSON an ideal data-interchange language.[24]

Before exporting the user can specify the hierarchy level, the JSON file should be saved to (project, sequence, shot). Based on the selection the preset file will be exported and imported.

[24] https://www.json.org/

Table 6-3: **LensDefocus aberration settings and knobs**

Chromatic Aberration	Control each channel (rgb) by itself or regulate the overall intensity.
Bloom	Define the threshold, gain and size of the defocus bloom, which will automatically increase with the distance to the focal plane.
Glint	Full control of the glint settings.
ProMist	Filter, softening the highlights.
Diffusion	Filter, softening the whole image.
Field curvature	Increasing blur towards the image edges.
Astigmatism	Simulates sagittal and tangential focus aberration.
Coma	Increasing blur and distortion towards the lens edges.
Ghosting	Ghosting effect of the highlights.
White balance	Controls the white balance in Kelvin.
Use lens preset	Sets all aberration knobs to its lens specific, predefined preset.
Save as lens preset	Saves the current values of all aberration knobs as the lens specific default.
Save presets to JSON	Incrementally saves the current lens presets to a JSON file on the server, which every artist will reference to, using the load latest JSON preset.
Load latest JSON file	Loads the highest version of the JSON preset file, if manual file path is empty, otherwise it will use the file of the manual file path.
File path	File path of the current JSON file.
Manual file path	Set manual file path instead of latest JSON version.

Figure 6-2 | **LensDefocus aberration user interface**

6.4 Lens distortion

Because lens distortions are different for every project, they are stored in a separate group. Thus, the compositing supervisor only has to change the UVMaps inside a simpler group setup.

Furthermore, you can choose between prime lenses, or create a zoom lens by adjusting the zoom dissolve slider. 0 represents no distortion, 1 the first focal length of the pull-down choice and so on. Lens breathing is working similar, just

with a focus factor. This will only be calculated, if you are using the *metadata/Nuke camera*.

If you use the lens distortion input, you have to specify the overscan amount of your rendering, which will be set automatically if you use the metadata camera. In addition, you have to reformat the input to the working resolution before this node (resize type: none, preserve bounding box).

Table 6-4 I **LensDefocus distortion settings and knobs**

Lens distortion node	Put in the name of the lens distortion node (_LENS_DISTORTION_)
Focal length	Drop down menu of the available lenses/distortion maps
Zoom dissolve	Dissolves between the lenses (zoom lens simulation)
Lens breathing	Dissolves between all lenses with a focus factor (only working with metadata camera)
Overscan	Overscan amount of the rendering in percentage.

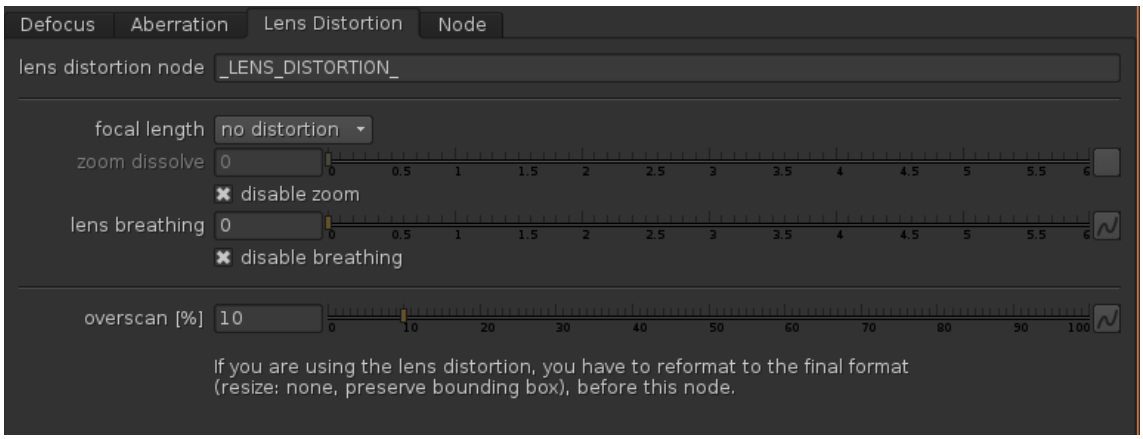


Figure 6-3 | **LensDefocus lens distortion user interface**

6.5 Project specific set-up

In order to set up this tool for a new project, the UVMaps inside the lens distortion node have to be replaced. On the one hand the distortion map can be calculated directly inside nuke, if a distortion grid was captured on set. On the other hand a lens specific UVMap is calculated during the matchmove, which can

be exported directly as an UVMap. Irrespective of the generation method, the map must be in the exact same format as the CG rendering with overscan.

Furthermore, the naming convention is a key part of the setup. Once, the artist or compositing supervisor has to specify the focal length of the corresponding UVMap. If this is done correctly, the script of the _LENSDISTORTION_PRESET_ node inside the lens distortion group can be executed. This will automatically create all focal length-based dropdown menus and the lens aberration preset nodes.

Finally, the aberration values have to be set. Therefore, it is recommended to capture an *aberration dots grid*. This can be created by using a white, flat and homogeny light source. This is masked by a black, light absorbing surface, which has multiple tiny holes in a regular pattern. Capturing this with the camera and changing the f-stop and/or focus distance visualizes the characteristics of the lens. Especially chromatic aberration, vignetting, glint, fied curvature, astigmatism, coma and the bokeh shape are visualized and can be matched in Nuke.

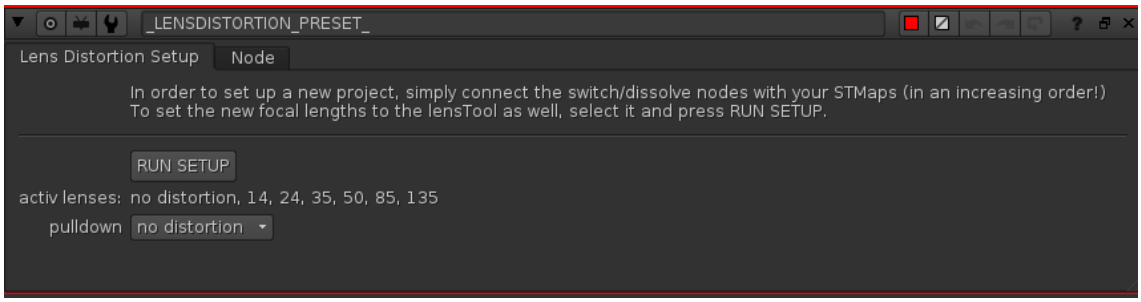


Figure 6-4 | **LensDefocus preset user interface**

6.6 Workflow and test scenes

This tool has a variety of customizable automations, effects and override functions. Thus, a specific workflow is highly recommended.

Table 6-5: **LensDefocus workflow example**

1	Connect the LensDefocus with the CG-plate (depth-channel and metadata included) and the _LENS_DISTORTION_ node.
2	Remove the overscan of your CG-plate right before the LensDefocus (preserve bbox).
3	Execute *camera*s script (creates all camera settings, aberrations and lens distortion).
4	*CG* if connected plate was rendered in Mantra with camera metadata.
5	*ALEXA*, select the Alexa plate, CSV file has to be generated before.
6	*Nuke camera*, connect your nuke camera with the camera input.
7	*Shotmachine*, only use this function in combination with the Shotmachine.
8	Adjust camera settings if necessary. Simply delete the keyframes and set new values. New focus distance can be set by checking *use focal point* and picking a new value on the viewer.
9	Reduce *maximum* value in the *artifact/speed* group for faster processing (check *use analyze depth* and run *analyze depth* to set max. depth automatically).
10	Use *edgeExtend depth objects* if object(s) close to the camera and out of focus have hard edges.
11	Set depth layers.
12	Adjust aberrations.
13	Disable lens breathing (automatically set if focus is changing) and/or zoom dissolve (automatically set if focal length is changing) if effect isn't desired.

The LensDefocus was implemented and tested for using the RISE internal pipeline. Therefore, some automation scripts are based on the given folder and project structure. This includes the ALEXA-CSV automation and the automatic JSON- in- and export. Further, the camera metadata have to match the implemented syntax and the *Shotmachine* knob is not necessary using another pipeline. Besides this, the LensDefocus-tool is fully functional in every pipeline.

The appended test scene and shots include multiple, different camera parameter animations. Furthermore, the 3D-scene was designed with detailed elements in different depths and highlights, in order to test and visualize the aberrations correctly.

7 Conclusion and Outlook

The LensDefocus is a fast computing tool, which can fully automate multiple lens aberrations and physically correct depth of field. Furthermore, it is easy to use and gives the artist the ability to override all automated settings. Thus, the compositing artist has full control of the lens artifacts.

Nevertheless, any 2D based ZDefocus is producing artifacts, because there is no information about what is behind the defocused foreground objects. These edge artifacts can be improved by using deep image data. Deep image data is able to store multiple depth and color values per pixel. Using this information for the ZDefocus reduces the artifacts a lot. Nevertheless, deep image data files are tremendously larger than normal EXR files. Thus, it is slowing down the compositing workflow a lot and needs a lot of memory on the server.

Furthermore, the Nuke internal ZDefocus is quite limited and real depth of field and bokeh are way more complex. One missing factor is that bokeh look differently depending on their position on the image. Bokeh close to the edges get stretched, curved and often appear cut. This effect is clearly visible on a Petzval lens but appears on a lot of lenses.

Another missing bokeh based artifact is bokeh caustics. Caustics occur, if light rays get reflected or refracted by a curved surface or object. Thus, bokeh caustics arise when a ray of a bright light source hits a water drop, which is directly on the lens. This results in an organic looking bokeh shape, directly in front of the lens, similar to a lens flare.

Besides these bokeh based artifacts, a fully automated and physically based lens flare creation is missing, too. Lens flares are caused by internal reflections and scattering of light rays inside the lens. They also occur if the light source is outside of the frame and therefore highly depend on the incoming ray angle. Because of that, lens flares highly depend on the lens design and the incoming light angle.

To sum it up and make a point, there has to be a fast and versatile algorithm capable of producing physically-based depth of field with non-uniform bokeh shapes.

Besides these technical challenges, the automation of all these camera- and lens- based aberrations has to improve, too. Standard high-end film cameras like the Arri Alexa are capable of recording most camera and lens settings, using metadata. However, these values can only be extracted from the original ProRes or raw files, using a specific program. The visual effects vendor only gets the edited plates after the conforming process. Thus, these important metadata values are mostly not available for the visual effects artists.

However, the usage of metadata is increasing a lot. Besides camera settings like f-stop, focal length, focus distance etc. lens-based aberration can be captured, too. “The ZEISS eXtended Data is a unique technology which is based on the /i* Technology and provides information about the lens’ distortion and shading characteristics in real time.”[25] This dataset would speed-up and simplify the workflow on set and in post-production tremendously.

Furthermore, the technology of lens design is still improving and therefore the lens artifacts are reduced even more. Nevertheless, a cinematographer often chooses an old and dirty lens, because of its unique, aesthetic look.

In conclusion, the usage of automated, physically based camera effects will and has to increase in the next few years. Thus, the artist can focus more on the creative aspect of computer-generated imagery. Furthermore, the visual effects vendor can save time and money.

[25] Zeiss (2017)

8 Appendix

List of sources

1 Abramowitz Mortimer, Sprint Kenneth R., Davidson Michael W.
Introduction to Lens and Gemoetrical Optics
https://www.olympus-lifescience.com/de/microscope-resource/primer/lightandcolor/lensesintro/ (accessed November 22, 2018)

2 Allen Elizabeth, Trianphillidou Sophie, Attridge Geoffrey, Bilissing Efthimia, Jenkin Robin, Sidney Ray
The Manual of Photography and Digital Imaging 10th edition
Burlington: Focal Press, 2009

3 Barraclough Leo
Camerimage: Pixar's Patrick Lin on the Cinematography of 'Inside Out' (2015)
https://variety.com/2015/film/global/pixars-patrick-lin-cinematography-inside-out-1201646039/ (accessed January 03, 2019)

4 Cook, Robert L., Thomas Porter, Loren Carpenter
***Distributed ray tracing*, ACM SIGGRAPH Computer Graphics**
New York: July 1984: 137-145

5 Halliday David, Resnick Robert, Walter Jearl, Halliday
Physik 2. Auflage
Weinheim: WILEY_VCH GmbH & Co. KGaA, 2009

6 Kohlrausch
Praktische Physik – Volume 3
23. Auflage, Vieweg & Teubner, 1996

7 Hecht Eugene
Optik 4. Auflage
Oldenbourg Verlag München Wien, 2002

8 Hecht Eugene
Optics Fifth Edition
London, Pearson Education, 2017

9 Jenkins Francis A., White Harvey E.
Fundamentals of Optics Fourth Edition
The McGraw-Hill Companies, 1976

10 Keller H. Ernst, Spring Kenneth R., Long John C., Davidson Michael W.
Astigmatism Aberrations
https://www.olympus-lifescience.com/de/microscope-resource/primer/java/aberrations/astigmatism/ (accessed Dezember 05, 2018)

11 Pixar Animation Studios
RenderMan 22 Documentation
Emeryville, 2018

12 Selan Jeremy
Cinematic Color – From Your Monitor to the Big Screen
VES Technology Committee, 2012

https://github.com/jeremyselan/cinematiccolor/blob/master/ves/Cinematic_Color_VES.pdf (accessed January 4, 2019)

13 Sutter Robert T., Fellers Thomas J., Davidson Michael W.
Refraction of Light

https://www.olympus-lifescience.com/de/microscope-resource/primer/java/refraction/refractionangles/ (accessed November 17, 2018)

14 The Foundry.
Nuke User Guide Version 10.0V5
London, 11 11, 2016

15 The Foundry
Blink API
London, 2013

https://learn.foundry.com/nuke/developers/80/Blink/index.html

16 Welsh Craig
The Cinematography of 'The Lego Movie' (2014)

https://www.expandedcinematography.com/the-cinematography-of-the-lego-movie.html (accessed January 03, 2019)

17 Yehuda, Levi
Applied Optics – A Guide to Optical System Design |Volume 1
New York: John Wiley & Sons, 1968

18 Zeiss
ZEISS Compact Prime CP.3 and CP.3 XD Lenses

https://www.zeiss.de/camera-lenses/cinematografie/products/compact-prime-cp3-lenses.html (accessed Dezember 18, 2018)

Image Sources