DOI: 10.1111/cgf.14267

COMPUTER GRAPHICS *forum*
Volume 0 (2021), number 0 pp. 1–18

Efficient Rendering of Ocular Wavefront Aberrations using Tiled Point-Spread Function Splatting

István Csoba[1,2] and Roland Kunkli[1]

[1]Faculty of Informatics, University of Debrecen, Debrecen, Hungary
{csoba.istvan, kunkli.roland}@inf.unideb.hu
[2]Doctoral School of Informatics, University of Debrecen, Debrecen, Hungary
kunkli.roland@inf.unideb.hu

Abstract

Visual aberrations are the imperfections in human vision, which play an important role in our everyday lives. Existing algorithms to simulate such conditions are either not suited for low-latency workloads or limit the kinds of supported aberrations. In this paper, we present a new simulation method that supports arbitrary visual aberrations and runs at interactive, near real-time performance on commodity hardware. Furthermore, our method only requires a single set of on-axis phase aberration coefficients as input and handles the dynamic change of pupil size and focus distance at runtime. We first describe a custom parametric eye model and parameter estimation method to find the physical properties of the simulated eye. Next, we talk about our parameter sampling strategy which we use with the estimated eye model to establish a coarse point-spread function (PSF) grid. We also propose a GPU-based interpolation scheme for the kernel grid which we use at runtime to obtain the final vision simulation by extending an existing tile-based convolution approach. We showcase the capabilities of our eye estimation and rendering processes using several different eye conditions and provide the corresponding performance metrics to demonstrate the applicability of our method for interactive environments.

Keywords: human vision simulation, depth of field, ocular wavefront aberrations, point-spread functions

CCS Concepts: • Computing methodologies → **Rasterization**; Image-based rendering; Massively parallel algorithms

1. Introduction

The human visual system provides one of our most important sensing mechanisms. Our eyes allow us to capture a continuous, dense snapshot of our surroundings by absorbing the reflected light waves that bounce off of the objects in our field of view. Without vision, we would lose a large fraction of our ability to perceive our environment, making it substantially harder to perform the vast majority of tasks in our everyday lives.

Just like all of our sensory organs, the human eye is incredibly fragile. Its ability to focus light degrades naturally with age, which is further exacerbated by the various internal and external effects that affect parts or the entirety of the visual system. These imperfections are commonly referred to as *visual aberrations*, or *aberrations* for short. Studies indicate that a large portion of the human population suffers from at least a moderate amount of visual aberration [PGCW01, THBC02, HSH*02, NKN*14], to which ageing is a large contributing factor [GGR*99, MMB01, BTA10]. Besides natural effects, medical operations (such as LASIK and LASEK surgery and intraocular lens implantation) are also significant sources of visual aberrations. While the aim of these procedures is to improve the visual performance of the patients, they often come with significant side-effects [dVWT*11, TFTA16]. Lastly, visual aberrations also heavily depend on factors such as the light wavelength [VDC*15], the pupil size [WZJ*03], and the accommodative state of the eye [CBV*04].

Because of these reasons, visual aberrations affect each person uniquely, making it greatly important to develop methodologies that correctly and plausibly simulate visual impairments. Such techniques could not only help deepen our understanding of these conditions, but they have the potential to improve the effectiveness of such visual impairments as well. More specifically, vision simulations can pave the way for better tailoring the corrective elements to the visual impairments of the individual, which could, for instance,

play a significant role when choosing the proper corrective spectacle lenses and intraocular lens implants.

Besides using the vision-simulated imagery as-is, their interpretation could also be performed using neural networks, via machine learning (ML) approaches. ML-based techniques for data extraction tasks are commonplace today, with the success of the training procedure often depending on the magnitude and quality of the available training data. The problem is often solved by using simulations to artificially generate the training dataset, and as such, developing robust and efficient vision simulation methods is essential for ML-based algorithms to be viable for such workloads.

Researchers have invented a wide array of algorithms that mimic aspects of human vision. Unfortunately, however, no existing algorithm is capable of delivering high fidelity results while supporting arbitrary visual aberrations and achieving the throughput required by interactive applications and data generation purposes. Furthermore, existing algorithms often focus on a small visual angle, the results of which can be highly misleading for the untrained observer. Lastly, the reliance on input data that can only be obtained with expensive medical devices is a fairly common limitation to the wide-range usability of many existing solutions. Overcoming these problems was the primary motivating factor for our work.

In this paper, we present our efficient rendering solution for the simulation of human vision that is affected by visual aberrations. Our algorithm simulates the on-axis image formed by a single human eye, using a single phase-aberration description (via Zernike coefficients) as input. The main benefits of our proposed method include the ability to run at near real-time performance, the minimal input data required, the possibility to simulate arbitrary focus states and pupil sizes from the small input data, and the ability to modify these settings dynamically at runtime. Furthermore, we achieve these goals without having to sacrifice the output quality. To this end, our main contributions are:

- A custom parametric eye model and the corresponding estimation procedure, which we use to reconstruct the physical properties of the simulated eye.
- A coarse PSF parameter sampling approach and a kernel computation scheme for arbitrary PSF parameters.
- A runtime kernel interpolation strategy, which we use to dynamically construct the out-of-focus point-spread functions (PSFs) from a pre-computed coarse kernel grid and approximate the true PSFs with negligible errors.

The rest of the paper is structured as follows: first, we give an overview of related work in Section 2. We then describe our proposed method in Section 3, with Section 3.2 focusing on the pre-computation steps, and Section 3.3 explaining the runtime part. Following this, we demonstrate the quality and performance characteristics of our approach on several different test scenarios and eye conditions in Section 4, and we also evaluate how well our parametric eye model and reconstruction approach simulates visual aberrations. Lastly, we give our closing thoughts and outline potential future research ideas in Section 5.

2. Previous Work

Faithfully replicating the virtual world as seen from the perspective of the main protagonist is a crucial part of most video games in existence today. Although the pinhole camera model that is most often used by these real-time environments is not well suited for reproducing the complex behaviour of light, the more interesting visual effects are typically added on top of the generated raster images via post-process filters. One of the most prominent ingredients for achieving truly immersive simulations is the use of the so-called *depth of field* (DOF) effect, which is not only an essential artistic and storytelling tool but also a crucial part of our everyday lives.

The problem of simulating DOF can be thought of as a special subset of vision simulation. In the more general case, the simple pinhole camera is replaced by a physically-based representation of the human eye. Such an extension facilitates the computation of the various high-level effects that are impossible with a simple model, but at the expense of significantly increased computation times.

Throughout the history of computer graphics, a vast array of algorithms have been developed to solve these problems, ranging in complexity from a simple mean filter to full-blown, physically-based algorithms. In this section, we briefly recall the existing depth of field and vision simulation methods that we consider most relevant to our work.

Ray Tracing. A general solution for simulating the depth of field of an optical system is given by ray-tracing techniques. These methods are often favoured in applications where the importance of output quality significantly outweighs that of the necessary computation times. Since most offline rendering environments already employ some form of ray-tracing, a large variety of algorithms [CPC84, KMH95, FLB*09, HQL*10, LES09, LES10] exist to facilitate depth of field effects to be rendered in these systems.

Plausibly simulating human vision via ray-tracing is typically done by incorporating a detailed model of the human eye. Most algorithms employ either an analytic [MKL97, FM06, DWRW16, LMB*19] or polygonal [WPP14] representation to model the various refracting elements of the eye. Alternatively, wavefront tracing can be used to derive the most important characteristics of the optical system, then fall back to a simplified lens model for the actual ray-tracing process [LSS01, NSG12].

Since the emphasis is on achieving the most realistic simulations possible, such algorithms are typically not well suited for interactive environments. The problem is even more severe for vision simulation, where the complexity of the optical system and the associated aberrations result in significantly increased sample counts and much more costly ray-surface intersection tests. Because of these reasons, algorithms of this kind are inappropriate for low-latency applications.

Multi-view synthesis. Another approach for simulating lens systems is the use of multiple views to reproduce the wave-related properties of the optical system. The pioneering work in multi-view DOF rendering is the accumulation buffer [HA90], which places multiple samples on the camera aperture, then produces the final output by rendering the scene from each different viewpoint and

Figure 1: *Example outputs of our proposed method for simulating human vision, which produces plausible results for arbitrary aberration types with interactive frame rates, on commodity hardware. The two images demonstrate the difference between an* ***(a)*** *ideal (healthy) and* ***(b)*** *aberrated (highly myopic) vision, with the focus point positioned in the background of the scene. The images were rendered at 720p on an NVIDIA TITAN Xp graphics card, with our vision simulation process taking* ***(a)*** *17.17 ms and* ***(b)*** *26.74 ms, respectively.*

accumulating the results in a dedicated buffer. The idea has later been extended to support more complex optical systems, either by directly tracing against the lens geometry [HSS97] or using wavefront tracing to pre-compute a per-vertex displacement map [KTMN07, KTN10].

These methods are well known for their output quality, as it is possible to generate high fidelity simulations by using a sufficiently large number of viewpoints. On the other hand, it takes an excessive number of samples to avoid artefacts like object ghosting, which makes algorithms of this kind unsuitable for interactive, low-latency applications.

Convolution. An improvement in speed is obtained by using convolution filters to produce the final blurred image. Such algorithms fall into two main categories: gather-based techniques [WBB11, MRD12, NSG12, Sou13, McG14, Gar17], where each pixel collects data from its surroundings, and scattering algorithms [PC81, Shi94, KvB03, KTB09, LH13, MH14, FHSS18], in which case pixel contributions are splat to the neighbours. While conceptually similar, the two approaches differ vastly both in quality and performance, with gather-based methods winning in terms of speed, but kernel splatting yielding substantially more plausible outputs.

A variant of scattering is obtained by not splatting the kernels directly, but using the derivatives instead [Hec86, SBHL99, KHB09, HS17, LSR18]. These methods can achieve fast running times, often with fewer visual artefacts as well. In the majority of cases, however, these benefits come at the expense of a limited subset of supported kernel types.

Convolution-based approaches have been used to simulate human vision as well. Instead of using a uniformly distributed disk to approximate the blur kernel, convolution-based vision simulations typically employ a highly detailed version of the optical system's PSF. While such methods have the advantage of not requiring any knowledge about the physical structure of the eye, they are either limited in the supported aberration types [TX15, BP17, CLB18] or have running times not suitable for interactive applications [GSMM95, Bar11].

Our goal is to preserve the best properties of the various approaches while trying to avoid as many of the associated pitfalls as possible. More specifically, our proposed solution has the generality of the ray-traced approaches, which means that it is not limited to simulating a single focus setting or object distance. Furthermore, our approach works without the need for an elaborate description of the eye structure, and it is fast enough to operate without having to wait several seconds (or more) for a single output. All of this is achieved by reconstructing the eye model and generating a coarse set of PSFs in a pre-computational step, after which our runtime tiled kernel splatting approach produces outputs in an interactive, close to real-time fashion.

3. Our Approach

3.1. Algorithm overview

Conceptually, our algorithm comprises two main steps: a pre-computation of the necessary kernels (Figure 2, a–c), followed by a runtime kernel splatting process (Figure 2, d–g).

Pre-computation. With the on-axis Zernike coefficients as the only input, the purpose of the pre-computation step is to prepare the sparsely evaluated PSF grid. This process necessitates the estimation of the simulated eye's physical structure so that we can compute the missing wavelength- and accommodation-dependent aberration coefficients, which is the first part of this stage.

With a suitable set of structural parameters known, we can proceed with the sampling of the multi-dimensional parameter space, which consists of the object distance, light wavelength, pupil diameter, and focus distance. Given each unique combination of the discrete parameter samples, the next step of the pre-computation stage is the generation and processing of the corresponding PSFs. Closing off this phase, we upload the results to a sufficiently large GPU buffer for the runtime stage of our algorithm.

Runtime. The rendering stage relies on a tiled kernel splatting approach, based on the algorithm of Franke et al. [FHSS18]. The

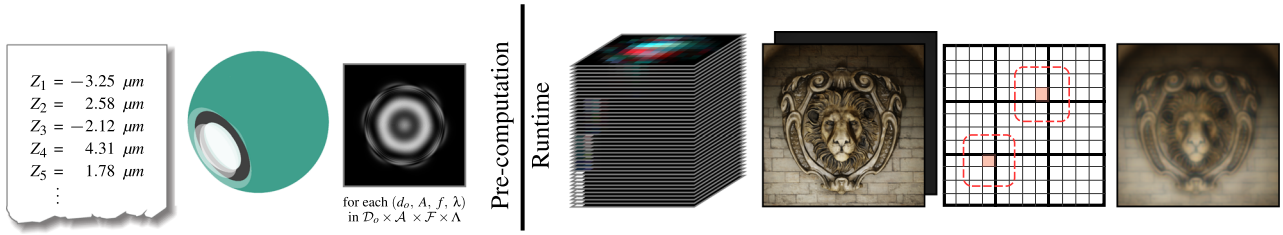


Figure 2: *Overview of our rendering algorithm. During pre-computation,* ***(a)*** *we first take a set of on-axis phase-aberration coefficients. Following this,* ***(b)*** *we estimate the physical structure of the eye. Next,* ***(c)*** *we take a coarse set of parameters (comprised of depth, aperture diameter, focus, and light wavelength) and compute the PSFs for each unique combination using the estimated eye model. At runtime,* ***(d)*** *we first build a 3D texture out of a reduced set of PSFs. Following this,* ***(e)*** *we take the input scene information and* ***(f)*** *populate a set of screen-aligned tiles with it. In the last step of our algorithm,* ***(g)*** *we produce the final result by traversing the tiles for all the overlapping fragments, using the previously generated PSF buffer for the correct and efficient weighting of the samples.*

inputs of this stage are the set of PSFs generated during pre-computation and textures containing scene colour and depth information. We render these before the blurring step, using a traditional pinhole camera model, the parameters of which (field of view, aperture size, clip planes, and resolution) are also taken as input.

First, we interpolate the input PSFs along the aperture diameter and focus distance dimensions to produce a reduced PSF set for the current frame. Next, the pixels of the input textures are combined and transformed into a GPU-consumable format to populate the screen-space bins. Following this, we perform a pixel spreading step to copy the converted information to neighbouring tiles. Next, we sort these tiles by depth. As the last step of this stage of our algorithm, we traverse the fragment bins for each output pixel, taking the weighted sum of the list elements using the depth-dependent interpolation of the previously generated, reduced PSF set.

3.2. Pre-computation: Kernel generation

The pre-computation step is responsible for producing the convolution kernels for the runtime stage. Figure 3 depicts a brief look at the required inputs and flow of data in this stage of the simulation. Here, we provide a detailed description of the various sub-steps involved in the pre-computation stage of our algorithm.

Parametric eye model. As stated earlier, our input comprises the on-axis aberration coefficients, as well as the pupil diameter and wavelength at which these coefficients were measured. To obtain the rest of the needed aberration coefficients, we thus had to devise some method to reconstruct the eye's physical structure.

Our eye estimation process builds on the same idea as the work of Liu and Thibos [LT19]; we also employ a custom eye model and an optimization process to obtain the final physical parameters. On the other hand, we do not possess any physical measurement data prior to optimization; we thus try to reconstruct the entire structure of the simulated eye from the input coefficients. Furthermore, to make our algorithms reproducible in a wide array of programming languages and software environments, we choose to avoid any reliance on special-purpose tools (like OpticStudio).

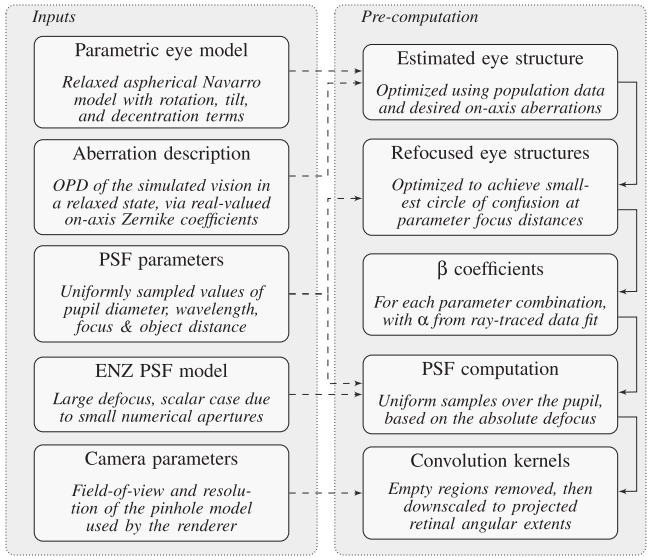
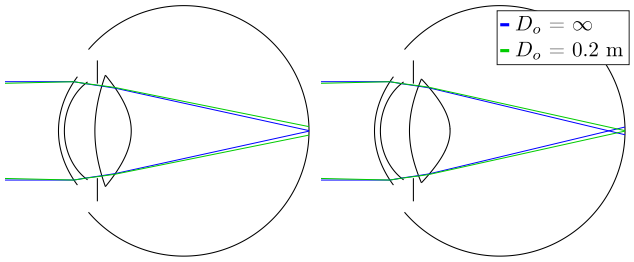


Figure 3: *Overview of the pre-computation stage of our algorithm.*

One crucial consideration when building our eye model was the overall computational complexity of measuring the aberrations of the system. Since we plan on using this eye model in an expensive optimization process, we want it to be simple enough to efficiently support the required computations, but maintain the ability to generate arbitrary visual aberrations.

With these factors in mind, we started from Navarro's unaccommodated aspherical eye model [NSB85]. This model comprises the retina, pupil, and four refracting surfaces; two surfaces for the cornea (anterior and posterior cornea) and another two for the crystalline lens (anterior and posterior lens). All surfaces (except the pupil) are represented using quadrics of revolution, which can be described with the following formula [PGO08]:

$$z(x,y) = \frac{x^2+y^2}{R_s \cdot \left(1+\sqrt{1-(1+k)\cdot\frac{x^2+y^2}{R_s^2}}\right)}, \tag{1}$$



(a) $d_f = \infty$ (b) $d_f = 0.2$ m

Figure 4: *Cross-sectional view of our parametric model simulating a healthy eye. The output of our estimation process is a relaxed model (**a**), which in the case of a healthy eye, properly focuses paraxial rays* (blue) *on the retina, but fails to focus rays emanating at an object distance of 0.2 meters* (green)*. On the other hand, when the lens diameter is altered to focus the model at 0.2 meters (**b**), the model correctly focuses the rays that originate from such an object distance but fails at focusing the paraxial bundle.*

where R_s is the radius of curvature, and k is the conic constant (asphericity factor) of the surface. The conic constant identifies the type of the surface, which is either a hyperboloid ($k < -1$), paraboloid ($k = -1$), prolate spheroid ($-1 < k < 0$), sphere ($k = 0$), or oblate spheroid ($k > 0$).

Since the Navarro model was constructed to represent a healthy eye, we had to extend it with several parameters to simulate visual aberrations. To this end, we added an astigmatism parameter to both surfaces of the cornea ($R_s = (R_x,\ R_y)$) and enhanced the anterior cornea with an offset function (using Zernike-polynomials [BW54]), which we use to describe the fine-scale corneal irregularities. Lastly, we also added axial rotation to the cornea and a set of tilt and decentration parameters to the crystalline lens.

To support focusing at arbitrary distances, we model the crystalline lens similar to how the real eye works. That is, we used the lens volume, equatorial diameter, asphericity, and the constant ratio between the anterior and posterior thicknesses [HPD*09] to define the overall shape of the lens, which allows us to compute the corresponding radii of curvature using an efficient, closed-form expression. An example output of this refocusing process is demonstrated in Figure 4 on the model of a healthy eye.

One limitation of using symmetric crystalline lens surfaces is that we cannot model lenticular astigmatism with it. We decided to make this simplification, as we can still reproduce astigmatism through the cornea, and the parameter space of our model is already fairly extensive. If handling such conditions is necessary, it is possible to extend the lens surfaces with the necessary astigmatism parameters, at the expense of a costlier reconstruction step.

Eye estimation. Given our parametric eye model and the target Zernike aberration coefficients, we can proceed with the optimization process. Our goal is to find a set of model parameters such that the induced aberrations of the resulting eye are sufficiently close to the input, while also preserving the structural integrity of the model eye. The result of this process is an estimation of the physical structure of the simulated eye in an unaccommodated state, which forms the basis of the Zernike coefficient computation procedure.

Table 1: *Optimization parameters of our custom eye model. The main types of parameters: thickness (T), diameter (D), volume (V), radius of curvature (R), conic constant (k), Zernike elevation coefficients of radial order n (Z_n), translation (Δ), tilt (α), and rotation about the optical axis (Φ). The upper index is used to differentiate between the anterior ($\cdot^A$) and posterior ($\cdot^P$) surfaces, with multi-axis parameters following an* (h, v) *pattern.*

		Unit	$\bar{x}_l$	σ_l	Limit	a_l
Eye	T	mm	23.82	0.81	[21.82, 25.82]	2.0
Cornea	T	mm	0.55	0.03	[0.4, 0.7]	32.0
	R^A	mm	(7.81, 7.81)	(0.25, 0.25)	[6.5, 9.81]	1.0
	R^P	mm	(6.44, 6.44)	(0.23, 0.23)	[5.5, 8.44]	1.0
	k^A		−0.29	0.09	[−2.29, 1.71]	1.0
	k^P		−0.34	0.24	[−2.34, 1.66]	1.0
	Φ^A	deg	0.0	0.0	[−45.0, 45.0]	0.1
	Φ^P	deg	0.0	0.0	[−45.0, 45.0]	0.1
	Z^A_{1-4}	mm	0.0	0.0	[−0.10, 0.10]	1.0
	Z^A_{5-6}	mm	0.0	0.0	[−0.05, 0.05]	1.0
Aqueous	T	mm	2.90	0.39	[2.2, 3.5]	1.0
Lens	D	mm	11.1	0.3	[10.6, 11.6]	1.0
	V	mm^3	160.1	2.5	[153.1, 167.1]	0.1
	k^A		−4.4	1.6	[−10.4, −1.0]	2.0
	k^P		−4.0	2.0	[−10.0, −1.0]	2.0
	Δ	mm	(0.0, 0.0)	(0.0, 0.0)	[−0.8, 0.8]	8.0
	α	deg	(0.0, 0.0)	(0.0, 0.0)	[−7.0, 7.0]	1.0

To ensure that the eye models generated by our reconstruction procedure are physically plausible, we rely on population data to control the weighting of the anatomical parameters. We extracted these values from several different sources to obtain the average values, standard deviations, and bounds of the parameters, for the cornea [RAT11, RRNT16, DSVdH06, CMDlCSNM*16, PAA*10], the crystalline lens [DVdH01, RAT11, RRNT16, RAK*12, KPV*18, HPD*09], and the axial lengths of the various elements [GD07, Lar79, KMNN10]. We also made sure to cover several different age groups and eye conditions in this process, in order to have the necessary representative power to support as many different visual aberrations as possible. The full list of optimization parameters and their associated statistical properties are listed in Table 1. We would like to point out that the range of our lens diameters is slightly larger than that of a real human eye, which is the result of modelling both lens surfaces with a single aspherical surface.

Next is the computation of the aberration coefficients, which we implement via ray-tracing. We start the process by tracing a parallel ray through the centre of the cornea to determine the starting retinal location for the reverse ray-tracing process. Following this, we shoot a dense ray grid through the posterior lens surface and obtain the exit pupil size by finding the minimal volume enclosing ellipsoid around the expected corneal ray positions. To obtain the final Zernike coefficients, we take the slope information of the outgoing ray bundle and use it to perform a least-squares fitting process [Her80], using scaling [LU07] to handle the slightly elliptical pupil projections.

After having established the bounds and statistical properties of the model parameters, we attempt to find a set of values with which our eye model approximates the target aberrations suitably well. To this end, we perform a constrained minimum search with the following objective function:

$$L = \sum_k (w_f \cdot f_k |\hat{\alpha}_k - \alpha_k|)^2 + \sum_l (w_a \cdot a_l \cdot \max(|\bar{x}_l - x_l| - \sigma_l,\ 0))^2, \quad (2)$$

where α_k and $\hat{\alpha}_k$ are the kth expected and generated Zernike coefficients, f_k is the weight of the kth coefficient, x_l and a_l are the lth model parameter and its associated weight, and $\bar{x}_l$ and σ_l are the mean and standard deviations of the lth parameter.

The user-defined parameters w_f and w_a allow for an overall weighting of the functional and anatomical properties of the resulting eye model, controlling which aspect of the model the optimizer should favour. Based on the magnitudes of the associated variables, our default weights are $w_a = 0.1$ and $w_f = 200$.

As for the optimizer algorithm, our experience is that gradient-based optimizers are prone to get stuck early in a local minimum due to the highly complex loss landscape that results from the anterior corneal surface of our eye model. Based on this observation, we decided to use a generalized pattern search (GPS) algorithm for all of our test cases, which we found capable of providing adequate solutions to our proposed optimization problem.

PSF model. Classic depth of field rendering algorithms typically ignore wavefront aberration theory altogether and rely solely on the *circle of confusion* of the optical system. The reason why such an approximation works is that the on-axis aberrations of traditional cameras comprise mainly defocus and spherical aberrations, which can be well approximated by an aperture-shaped kernel. Omitting high-frequency details is often an acceptable trade-off, given the sampling ambiguity caused by the finite resolution of camera sensors. On the other hand, wavefronts with non-spherical aberrations undergo considerable changes through the focal region, which necessitates a more general out-of-focus PSF model.

In the field of ophthalmology, the point-spread function (the projection of an infinitely small point source on the retina) of a human eye is typically computed using Fraunhofer's far-field diffraction formula [Goo17, Wat15]. Since aberrations are often measured in an unaccommodated eye state, such a representation is generally valid and enables the simple and fast evaluation of the PSFs. For a more elaborate analysis of the eye – such as computing its out-of-focus diffraction patterns – a different formulation is desirable.

An efficient means of obtaining the point-spread function at arbitrary defocus is given by the Extended Nijboer-Zernike (ENZ) theory of diffraction integrals [Jan02, VH10]. The ENZ diffraction theory offers a wide selection of analytical expressions for the Debye diffraction integrals, leading to the following PSF formulation:

$$U(x, y) = 2 \sum_{n,m} \beta_n^m i^m V_n^m(r, f) \exp(im\phi), \quad (3)$$

where $U(x, y)$ is the point-spread function, $x,\ y$ and $r,\ \phi$ are, respectively, Cartesian and polar coordinates in the image plane, and f is the defocus parameter, which equals zero at best focus and is related

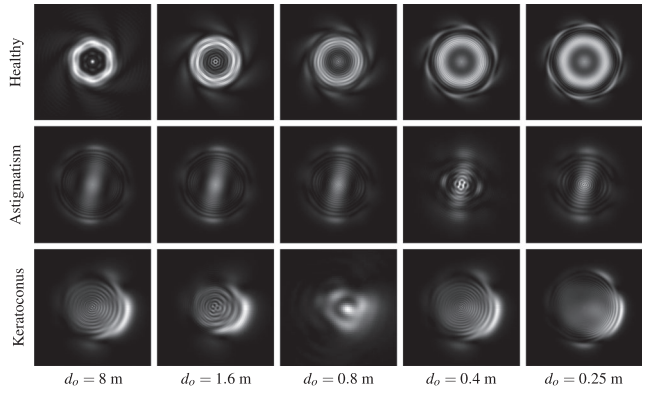


Figure 5: *Example monochromatic point-spread functions computed using Equation (3) for three different eye conditions, at five object distances. The images use a varying scale, with each image focusing on the interesting parts of the PSFs; the physical extents thus differ on a per-PSF basis. Note that the repeating ringing pattern (commonly referred to as* Fresnel-rings*) is a natural result of diffraction. The number of Fresnel-rings increases with the defocus parameter, which allows the comparison of the true physical PSF extents.*

to the free-space focus shift z as $f = \frac{-2\pi u_0}{\lambda} z$, with $u_0 = 1 - \sqrt{1 - s_0^2}$ and s_0 denoting the image-side numerical aperture of the optical system. The V_n^m functions are the linearized products of Zernike polynomials, and the β_n^m terms are complex values that can be derived from the real α_n^m phase-aberration coefficients. We used the Bessel-Bessel series expression for the V_n^m terms [JBD04] and the fitting approach described in [AV15] with a cosine sampling strategy [JvHJ*08] to compute the β_n^m coefficients. Figure 5 shows a few PSFs computed using this formulation.

There are several benefits of such a PSF representation. First, in terms of accuracy, the ENZ diffraction theory is known to have a clear edge over FFT-based approaches [BvHJD08], allowing the PSFs to be computed with only negligible errors. Second, the linear representation of the point-spread functions means we only need to evaluate the V_n^m terms once for each unique combination of n and m; the results can be stored and reused in every PSF computation. Finally, the use of a dedicated defocusing term allows us to handle the object distance in a relatively simple manner, omitting the full computation of the α and β coefficients.

Parameter sampling. Since our PSF parameter space is four-dimensional (aperture diameter, focus & object distance, and light wavelength), evaluating the PSF at an arbitrary point poses some difficulties if we want to keep the rendering efficiency. We overcome this issue by evaluating the point-spread function only at the intersection points of a coarsely sampled parameter grid. Due to the sampling ambiguity caused by the finite number of input pixels, we can fill in the gaps in the PSF grid via the linear interpolation of the evaluated kernels without severe visual artefacts.

Following Barsky's insights [Bar11], we also sample the focus and the object distances using dioptres. The reason for this is that

the PSFs scale linearly with dioptres, but not with meters. Thus, uniformly sampling a diopter-based representation of the distance parameters significantly reduces the number of required parameter evaluations. Our typical configuration uses $N_o = 41$ and $N_f = 5$ samples for the object and focus distance parameters.

As for the pupil diameter, we employ a simple linear sampling strategy to obtain the final coarse parameter set for PSF evaluation. Our goal is to cover the typical range of human pupil sizes; based on the unified formula of Watson et al. [WY12], we found $N_A = 4$ unique samples in the range of 2–7 mm to be sufficient for covering the vast majority of realistic scenarios. Lastly, we use $N_\lambda = 3$ distinct samples for the light wavelength parameter, with one sample corresponding to each channel of the RGB image.

To proceed with PSF computation, we need to obtain the aberration coefficients for each unique parameter combination. To this end, we first take the previously estimated eye model and compute the lens diameter for the current pupil size and focus distance. We consider a ray bundle that originates from the on-axis object point with the desired focus distance, then find the lens diameter with which the system best focuses these rays on the retina. Given this refocused eye model, we obtain the final aberration coefficients using the ray-traced least-squares approach that we explained earlier.

As stated previously, we can handle object distances differently: we rely on the defocus parameter of the ENZ model to obtain the through-focus PSFs. Here we recall that it is defined to be zero at the best focus; we thus have to work out the distance of this plane first. In our case, we can rely on the estimated eye model, but we note that other approaches, such as Strehl ratio minimization [JvHBD07] and quadratic approximation [BvHJD08], also exist.

Based on these observations, we compute the defocus parameter from the corresponding focal shift using the defocus-focal shift relationship. To this end, we first take the focal shifts resulting from the aberrated eye condition (z_α) and the distance between the focus and object planes (z_d), which we compute in the following way:

$$z_\alpha = d_{pr} - f, \qquad z_d = f - \frac{1}{\frac{1}{f} + \frac{1}{d_o}}, \tag{4}$$

where f is the focal length of the focused eye, d_{pr} is the pupil-retina distance, and d_o is the target object depth. We then obtain the final combined focal shift z_o and defocus parameter f_o as follows:

$$z_o = z_\alpha + z_d = d_{pr} - \frac{1}{\frac{1}{f} + \frac{1}{d_o}}, \qquad f_o = z_o \cdot \frac{-2\pi \cdot u_0}{\lambda}, \tag{5}$$

where $u_0 = 1 - \sqrt{1 - s_0^2}$, s_0 is the image-side numerical aperture, and λ is the light wavelength. Figure 6 demonstrates the different types of focal shifts using our schematic eye model.

Kernel computation. Having obtained all the necessary parameters, we compute the PSFs for each unique parameter combination using Equation (3). To this end, we place a grid on the exit pupil and evaluate U at the intersection points. The samples have a uniform size (μ_s), with N total samples allocated for each 2π increment of the defocus. We typically use $\mu_s = 4\ \mu m$ and $N = 4$ to generate our convolution kernels.

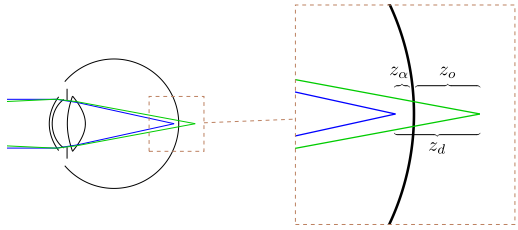


Figure 6: *Overview of the main types of focal shifts used in the computation of the defocus parameter. The total focal shift z_o is the sum of the aberration-induced focal shift z_α, and the out-of-focus focal shift z_d, both of which are signed. The ray colors follow the same scheme as we used in Figure 4.*

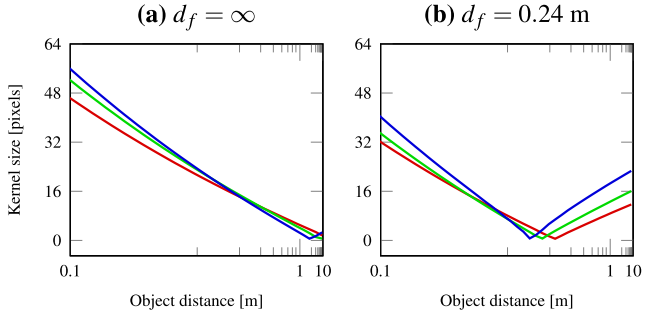


Figure 7: *Through-focus projected PSF sizes of a healthy eye in two focus states. The x-axis follows a diopter-based scaling, demonstrating that the blur size scales linearly with dioptres. Also, observe that each channel focuses at a different distance, which highlights the significance of treating all channels separately.*

Following the computation of the PSF grid, the last step of kernel computation is the transformation from physical to screen-space coordinates. Our PSF sampling yields sub-pixel image-space resolutions; our goal here is to downscale the kernels such that a one-to-one mapping can be established between the generated PSFs and convolution kernels. We thus need to compute the projected PSF sizes (in pixels), which we define in the following way:

$$D_p = \left\lceil \frac{f}{2\pi} \right\rceil \cdot N \cdot \sigma, \qquad \sigma = \frac{\mu_s \cdot H}{\mu_\theta \cdot fov_y}, \tag{6}$$

where f is the defocus parameter, fov_y is the vertical field of view of the camera used for rendering the input image (in degrees), H is the input image height, μ_θ is the size of a one-degree area in the foveal part of the retina (approximately 288 μm [DF74]), and σ is the minification factor. Figure 7 visualizes the projected PSF sizes for a healthy eye model computed using our formula. With $fov_y = 60^\circ$ and $H = 720$, the minification factor is $\sigma \approx 0.1667$, which results in 6×6 samples in the full-resolution PSFs per image-space pixel. We also show a few example outputs of our kernel computation approach in Figure 8.

Since a single PSF may span several different blur radii (depending on the sizes of its neighbouring PSFs), we need to accommodate for every blur size that might occur in the runtime step for the interpolation process to work properly. We thus use Equation (6) to

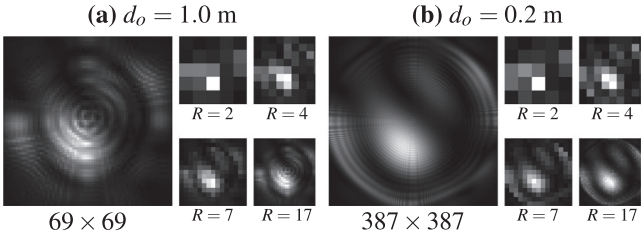


Figure 8: *Kernel computation step demonstrated on two different out-of-focus PSF images. The full-resolution images display the results of the PSF computation process, with the empty regions removed. Next to them are the downscaled kernels that we use for splatting, at multiple different blur radii. As can be seen, while the full-resolution PSFs capture the high-frequency details of the PSFs very well, the downscaled versions vary much less with object distance, which is what makes our interpolation strategy viable.*

compute and store the projected sizes of each kernel, but we downscale each PSF to each radius in the $[0, R_{max}]$ interval, where R_{max} denotes the largest blur radius found among the projected PSFs.

Finally, we would like to point out that the sampling scheme described above results in moderate amounts of empty space in our PSF images. We overcome this issue by introducing a cropping step between the computation and projection phases. For each PSF, we identify the centre area where the sum of weights is sufficiently close to the total sum. We then drop an equal number of rows and columns from all four sides of the PSFs, which ensures that the resulting PSFs are still correctly centred.

3.3. Runtime: Kernel splatting

The runtime step of our algorithm is responsible for producing the output renderings of the aberrated vision. Our approach uses tiled kernel splatting to efficiently convolve the input image with the spatially-varying kernels. Figure 9 provides an overview of our extended rendering algorithm. This section outlines the details of our main contributions involving vision simulation; the reader should consult the original work of Franke et al. [FHSS18] for a more elaborate explanation of tiled splatting.

Intermediate PSF buffer. As described earlier, our PSF convolution approach relies on interpolation to fill in the gaps in the coarse kernel grid. We take the PSFs generated in the pre-computation step (corresponding to the sparse grid points) and perform the actual interpolation on-the-fly, in the runtime step of the simulation.

One very important property of our generated kernels is that while the pre-computation process aims to cover the entire range of possible input parameters, we typically only need a fraction of these kernels to produce a single output image. More specifically, we can treat the focus distance and pupil diameter parameters as constants throughout the lifespan of a single frame. This observation essentially means that we can interpolate our PSFs across these parameters before splatting, storing the results in a smaller buffer dedicated to this purpose. During splatting, we can then employ on-demand

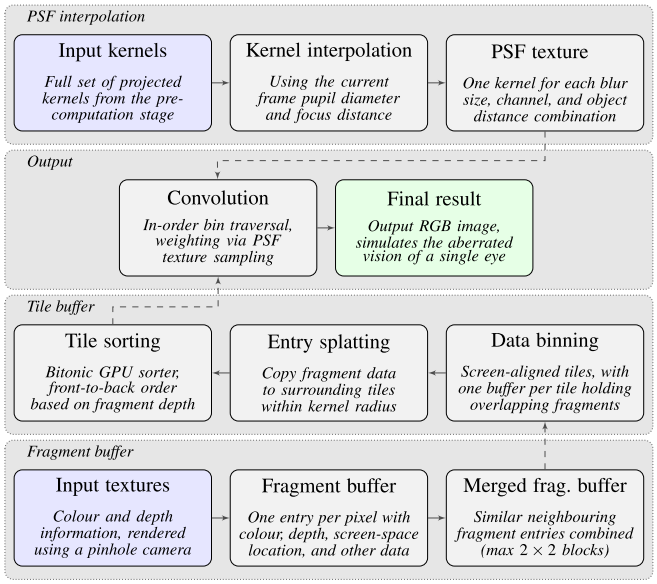


Figure 9: *Overview of the runtime stage of our algorithm.*

interpolation using the per-pixel object distances, for all three of the RGB channels.

To this end, we treat the focus distance as a configurable input, but determine the current pupil size automatically, using the Moon and Spencer formula [MS44, WY12]. With these parameters, we first interpolate the blur radii for each unique combination of light wavelength and object distance and store the results in a per-frame PSF parameter buffer.

Next, we proceed with the interpolation of the kernel weights. Here we need to account for the fact that our kernels could be sampled at an arbitrary radius; we thus need to produce a PSF buffer that can support this requirement. We build a 3D texture for this purpose, which has the following dimensions:

$$W = 2R_{max} + 1, \quad H = 2R_{max} + 1, \quad D = N_o \cdot (R_{max} + 1), \quad (7)$$

where W, H, and D are the width, height, and depth of the resulting texture, R_{max} is the largest possible blur radius, and N_o is the number of different object distances. We then lay out the kernels continuously across the texture layers, sorted by kernel radius first, then object distance, with the different wavelengths encoded in the RGB channels. This layout is also visualized in Figure 10.

One benefit of this PSF layout is that we can rely on hardware acceleration when looking up the kernel weights, as only a single texture sample is needed to interpolate between two neighbouring object distances. Since each kernel resides in a different texture layer and starts at the same pixel, we also avoid complex texture coordinate computations and all bleeding artefacts, which are common with texture atlases. The only pitfall is that the lower layers contain empty regions, and thus the layout is wasteful of texture space. In our experience, however, the increased texture size rarely impacts the rendering performance; we thus consider the single-texture

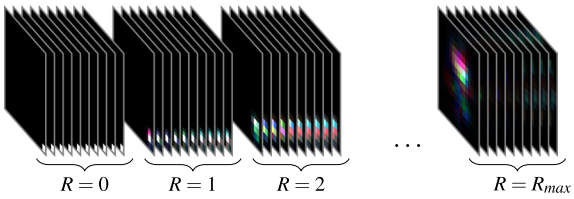


Figure 10: *Layout of the PSF texture. Each layer holds three out-of-focus kernels (one per channel), downscaled to a single blur radius. We place the kernels in the texture layers in a continuous fashion, sorting them by blur radius first, then by object distance. This layout allows us to sample two neighbouring PSFs with a single, hardware-accelerated texture lookup.*

approach a simple and efficient alternative to atlasing or using multiple textures.

PSF evaluation. The second part of the interpolation process is carried out on-demand, during the per-pixel traversal of the tile buffers (*Convolution* step in Figure 9). We perform a single interpolation operation for each channel of every sample fragment, to compute the weighting of the individual channels in the front-to-back blending process. The description below outlines the main steps of evaluating the kernel for a single channel, which we repeat for all three channels.

To approximate the PSF, we first have to find the indices of the two closest pre-computed kernels, which we do in the following way:

$$I_s = \frac{1}{D_\Delta}\left(\frac{1}{d_s} - D_{min}\right), \tag{8}$$

where d_s is the object space depth of the sample, D_{min} is the depth (in dioptres) of the PSF corresponding to the closest object distance, and D_Δ is the distance between two neighbouring PSFs (also in dioptres). This index uniquely identifies the two point-spread functions that we need to interpolate and gives an interpolation factor in the fractional part of the number.

Given the fractional PSF index, we first compute the corresponding blur radius R_s, using $\mathrm{frac}(I_s)$ as the interpolation factor. We then take two texture samples, one for each neighbouring integer blur size, with the PSF centred on the output fragment. We use the fractional PSF index I_s and the integer blur radii to compute the third texture coordinates of the two samples, which ensures that the samples are properly interpolated with respect to the object depth. Lastly, we finish the process by interpolating between the two samples using $\mathrm{frac}(R_s)$. Figure 11 demonstrates the process for one channel of a single PSF sample.

It is possible to halve the number of per-pixel texture samples by rounding the PSF radius upwards only, but that would also lead to visible jumps as the blur radius of one-pixel shifts from one discrete value to the next. The issue is fairly well hidden in still images by the multi-channel PSF convolution but becomes jarring as we start moving around in the environment. The main advantage of the sampling strategy described above is that it gradually fades between the different kernel sizes. Because of this, our solution is completely

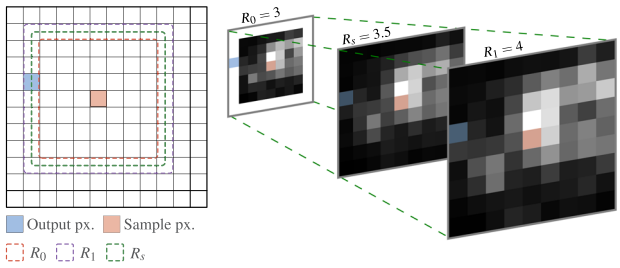


Figure 11: *Overview of the PSF interpolation process for a single sample pixel. We first round the interpolated fractional blur radius (R_s) to the two neighbouring integers (R_0 and R_1) and sample the PSF texture at both blur radii. Given the two PSF values, we obtain the final sample weight by interpolating between the two values using the fractional of the blur radius. This visualization also demonstrates how the lower level PSF samples fall outside the main kernel region near the PSF edges, which is the main reason why our sample placement strategy results in smooth transitions.*

devoid of this problem and produces smooth outputs across the entire focal region, and the results remain stable under motion as well. The downside of this approach is that it also leads to higher computation times, but in our experience, the increase in output quality vastly outweighs its impact on performance.

4. Results

Implementation. To evaluate the quality and the performance of our algorithm, we created a proof-of-concept implementation. We integrated it into a custom real-time renderer that was written in C++, using the OpenGL graphical library for rendering. As such, we also made the reference implementation using the same tools.

The pre-computation phase of our implementation runs purely on the CPU. Of particular interest are the eye-related parts of this pre-computation step (eye structure estimation and aberration measurements), for which we used the MATLAB software environment. We considered it a good choice for our goals, due to the vast array of well-documented algorithms, the large library of third party samples, and the ability to interface with it from C++.

As for the runtime stage, we implemented all the sub-steps using GLSL compute shaders. We run the entire process on the GPU since we already have all the pre-computed PSF data available at the start of this stage. The algorithm takes the colour and depth textures that were produced in the current frame, with anti-aliasing already applied to the colour buffer. We then perform the runtime step either before or after the tone mapper stage, depending on the type of input dynamic range (HDR or LDR) we would like to use.

Test setup. In an attempt to cover a wide range of different visual aberrations, we tested our algorithm with four unique eye conditions. First, we constructed a healthy eye model by training with an input set of phase coefficients comprising only zeros. Next, we created models for eyes with moderate amounts of myopia and astigmatism, for which we obtained the aberration coefficients from the

corresponding corrective lens prescriptions, using a conversion formula [Dai08]. Lastly, we also tested our algorithm with a vision suffering from keratoconus, using coefficients that resulted from a real eye aberration measurement.

For all four eye conditions, we simulated the eye in a relaxed state, setting the focus on the far plane. We tested several different test scenes, from which we present two in this paper: a simple setup with just three basic objects and a complex scene that represents a real-world environment. We also present our algorithm with a third scene, the famous Sponza atrium, in the supplemental videos. The pinhole camera used in all our simulations had a 60-degree vertical field of view, which resulted in a horizontal field of view of about 106 degrees with the aspect ratio of our input images.

Quality. To evaluate the visual quality of our approach, we compare our results with two different convolution-based approaches. First, we implemented Barsky's existing vision simulation algorithm [Bar11], which generates its outputs by splitting up the input images to separate depth slices. The thus computed partial images are convolved with a single PSF corresponding to the centre of the depth regions. We used the same ENZ PSF model for the through-focus PSF computation, with the goal of minimizing the differences stemming from the different PSF models and to focus solely on the rendering parts of the algorithms. Furthermore, we relied on our estimated eye models to derive the necessary Zernike coefficients, which facilitates the simulation of chromatic aberration with the previous algorithm as well. All outputs shown in this paper for the previous technique were rendered using the same 41 depth slices (and corresponding PSFs) that we used with our algorithm.

The second algorithm computes the ground truth images by evaluating the true PSFs at every input pixel. This reference algorithm works by assigning each pixel to a diopter-based bin (with a precision of 3 decimal places), which resulted in 4085 unique bins for the primitives scene, and 8 088 bins for San Miguel. The algorithm then produces the final output using front-to-back blending on all the pixels with overlapping PSFs. We consider this approach as the upper limit of what is possible with such convolution-based techniques; our goal is to match the quality of these renderings while keeping the rendering times as low as possible.

Our outputs for the test cases that we described earlier are shown in Figure 12 and Figure 13. Additionally, we also include four example videos (one for each eye condition) as supplemental material. These videos demonstrate our algorithm under motion, and – in the case of myopia, astigmatism, and keratoconus – compare the vision with that of the healthy eye. We exported these outputs by generating 60 unique frames for each second of the videos, which resulted in smooth and synchronous outputs for the comparison.

As can be seen from the results, our algorithm interpolates the kernels smoothly throughout the entire focal region, with no visible banding artefacts. The different characteristics of the individual eye conditions are clearly visible as well. Furthermore, our outputs also exemplify the importance of handling chromatic aberration and show that separately processing the individual channels is necessary to produce plausible vision simulations. This observation is in line with the results of Cholewiak et al. [CLS*17], and as concluded by the authors, a significant contributor to realism.

In comparison to the previous technique, the biggest difference in output quality results from the correct per-pixel alpha blending. Because the depth slices in Barsky's solution are blurred using a traditional convolution process, the handling of partial occlusion is limited to the depth slice boundaries. Disregarding per-pixel depth ordering can result in morphed object boundaries, which can be best observed in Figure 12. Our proposed solution handles these cases correctly.

We note that the handling of partial occlusion in our examples is limited to visible surfaces. This is not a limitation for our runtime stage, as the tiled kernel splatting approach already works with layered inputs, and our PSF interpolation process is only concerned with the object-space depth of the sample pixel. On the other hand, the hidden layers would require computing the true coverage ratio to properly work with our physically-based kernels. Since the resulting errors are limited to object boundaries, we decided to only process the visible front layer of the scene, which increases the overall performance of our simulation.

Next, we follow with the quantification of differences between the reference images and the outputs of both our algorithm and Barsky's solution. To this end, we first computed the peak signal-to-noise-ratio (PSNR) for each output-reference pairing, which can be used to get an objective measure of the overall per-pixel noise levels. Besides PSNR, we also calculated the structural similarity index (SSIM) and HDR-VDP [MKRH11] probability of error detection metrics, both of which can be used to get a better understanding of the higher level similarity of the compared images. The per-image mean SSIM and HDR-VDP probability of detection values are given in Table 2, with false-colour visualizations of the corresponding per-pixel maps shown in Figure 12 and Figure 13.

As demonstrated by the similarity metrics, there is no substantial difference between our renderings and the ground-truth images. Despite the very small number of pre-computed kernels, our PSF interpolation process faithfully reconstructs the true, dense PSFs. Our algorithm also correctly handles partial coverage and well preserves the overall shape and size of the PSFs, with only a slight overblur in certain areas. These differences can be best observed near object edges and can mostly be attributed to the fragment merging step. Nevertheless, considering the overall similarity of our results and the ground-truth images, we can safely conclude that the proposed algorithm is capable of producing images that are comparable to the reference renderings, while also achieving higher visual quality with respect to the previous algorithm.

Performance. The performance of our algorithm was measured on an AMD Ryzen 7 1700 3.00 GHz with an NVIDIA TITAN Xp graphics card, using our reference OpenGL implementation, at 720p resolution. Table 3 summarizes the computation times using the same test scenarios and parameter settings that we used for the outputs shown in Figure 12 and Figure 13. Besides the convolution step of Barsky's solution and the rendering stage of our method, we include the measurements for the pre-computation stage as well. To minimize the impact of frame-to-frame variance on the results, we used the average of 100 consecutive frames for measuring the

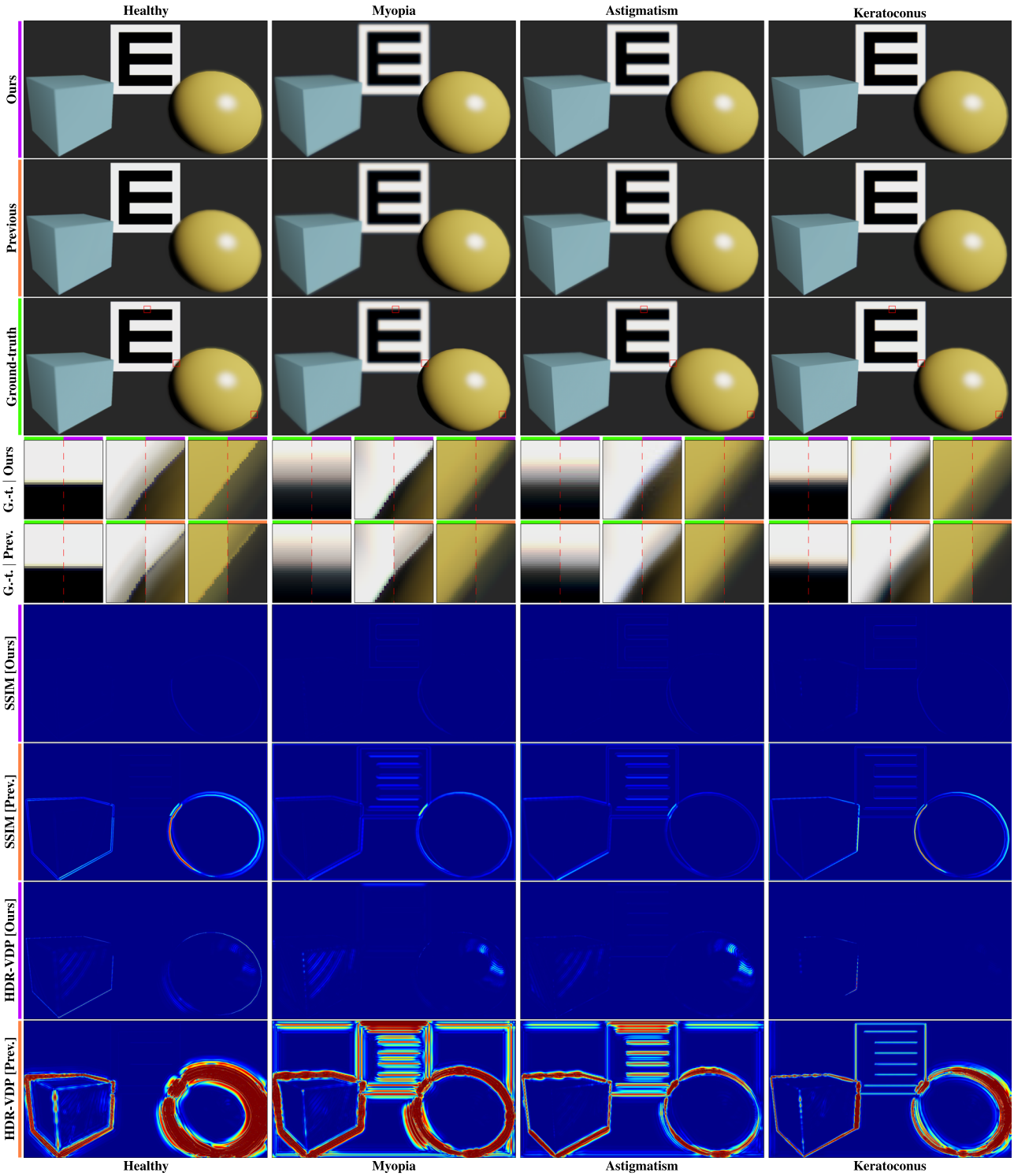


Figure 12: *Results of simulating four different eye conditions on a simple scene. The object distances are 0.2 m (sphere), 0.5 m (cube), and 7.5 m (letter), which leads to different amounts of blurring for each object. The first three rows show, respectively, the outputs of our algorithm, a previous technique [Bar11], and the ground-truth images. A side-by-side comparison of the ground-truth and the other two algorithms is included for a few smaller regions in the fourth and fifth rows. The last four rows show a visualization of the SSIM and HDR-VDP metrics for the two algorithms, with brighter pixels representing a lower structural similarity and a higher probability of error detection.*

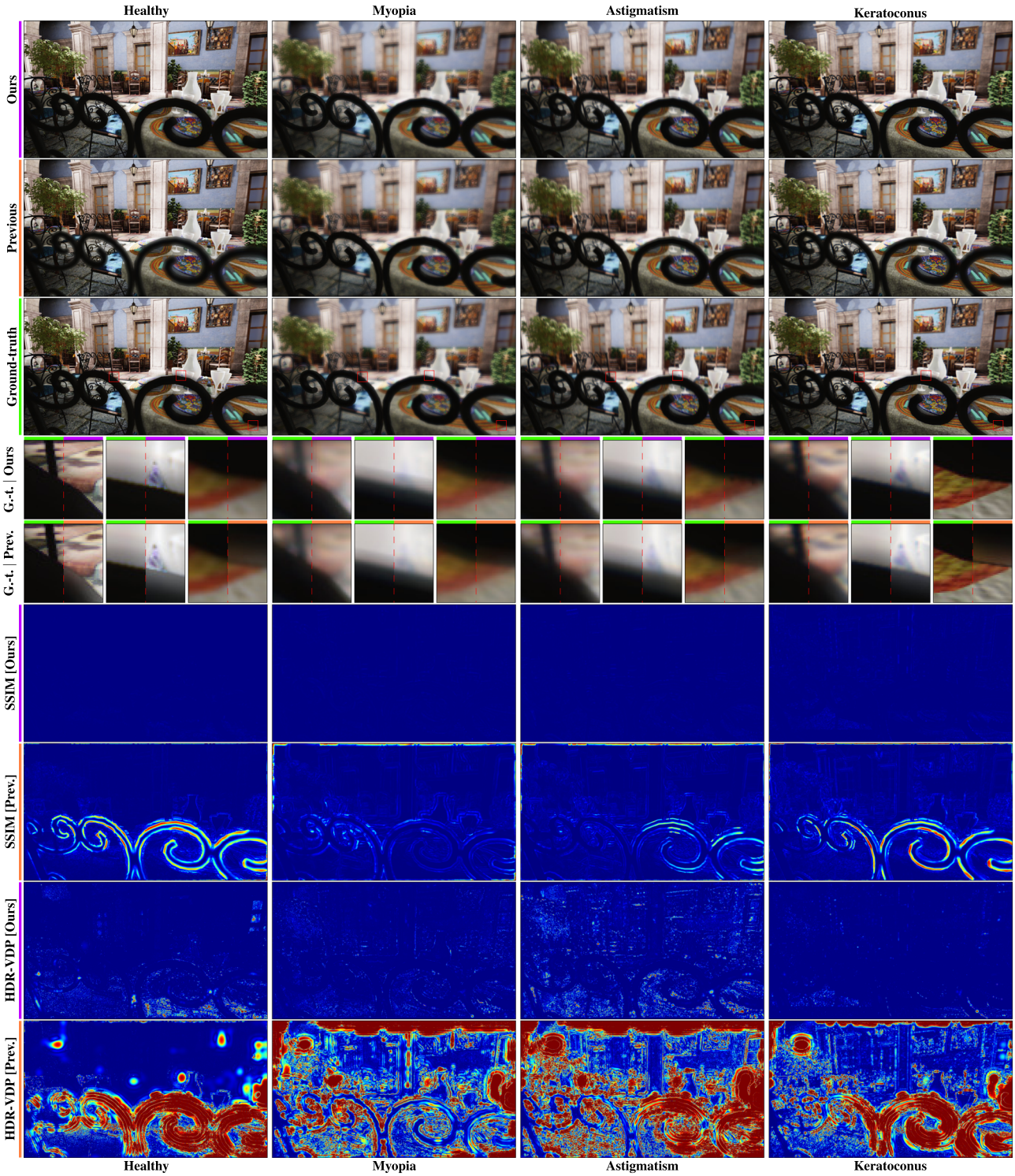


Figure 13: *Results of simulating four different eye conditions on the famous San Miguel hacienda test scene. The images follow the same layout that we used in Figure 12.*

Table 2: *Peak signal-to-noise ratio (PSNR), mean structural similarity index (MSSIM), and HDR-VDP-3 probability of error detection (HDR-VDP) metrics for the outputs shown in Figure 12 and Figure 13.*

	Primitives (Figure 12						San Miguel (Figure 13					
	PSNR		MSSIM		HDR-VDP		PSNR		MSSIM		HDR-VDP	
	Prev.	Ours	Prev.	Ours	Prev.	Ours	Prev.	Ours	Prev.	Ours	Prev.	Ours
Healthy	33.57 dB	50.82 dB	0.984	0.998	0.998	0.062	27.26 dB	45.61 dB	0.938	0.997	0.988	0.353
Myopia	33.45 dB	49.10 dB	0.983	0.998	0.970	0.172	29.99 dB	47.81 dB	0.966	0.995	0.994	0.100
Astigmatism	34.55 dB	48.84 dB	0.985	0.997	0.784	0.244	29.90 dB	47.22 dB	0.961	0.995	0.995	0.317
Keratoconus	34.40 dB	46.19 dB	0.982	0.998	0.963	0.069	27.99 dB	43.53 dB	0.941	0.993	0.993	0.113

Table 3: *Running times of the pre-computation step, the previous vision simulation method [Bar11], and the rendering stage of our algorithm, for the same test scenarios and parameter settings that we used to generate the outputs shown in Figure 12 and Figure 13.*

				Primitives (Figure 12)				San Miguel (Figure 13)			
	Pre-computation			Proposed Algorithm			Previous	Proposed Algorithm			Previous
	Coeff.	PSFs	#PSFs	PSFs	Conv.	Total	Conv.	PSFs	Conv.	Total	Conv.
Healthy	141.07 s	622.06 s	2 460	0.47 ms	10.23 ms	**14.88** ms	*11.54* s	0.45 ms	9.91 ms	**14.40** ms	*12.13* s
Myopia	119.82 s	196.00 s	2 460	0.15 ms	13.56 ms	**16.67** ms	*9.43* s	0.16 ms	20.48 ms	**24.49** ms	*10.31* s
Astigmatism	115.52 s	246.17 s	2 460	0.20 ms	10.54 ms	**13.49** ms	*9.65* s	0.19 ms	13.89 ms	**17.32** ms	*9.31* s
Keratoconus	127.60 s	450.58 s	2 460	0.28 ms	8.51 ms	**11.86** ms	*9.65* s	0.24 ms	12.60 ms	**16.73** ms	*10.75* s

running times of our algorithm. Lastly, we only include the new and modified parts of the rendering stage, since only those are affected by our extended approach; we omit the rest for brevity.

As can be seen from the results, our algorithm achieves the desired interactive performance and clearly outperforms the previous approach. Since we do the vast majority of the interpolations in the tile traversal step, the cost of the PSF texture generation step is negligible. Most of the processing time is spent in the convolution step, which is a consequence of treating each channel separately. We identified the memory bandwidth (texture accesses) to be the most limiting factor; replacing the PSF texture lookup with a simple weight (based on the inverse kernel area) significantly reduced the cost of the convolution step.

Furthermore, as the performance measurements demonstrate, the relative processing times follow our expectations and change with the overall blurriness of the output. The one exception to this pattern can be observed in the simulation of the healthy eye with primitives test scene, where the processing time of the mostly sharp image is higher than that of most other eye conditions. The phenomenon is caused by the blur radius threshold of the merge process, which we use to preserve the sharpness of the image in the focused region. This particular output is thus a good demonstration of the importance of fragment merging on the computation times.

Memory. The memory consumption of the PSF buffer is mainly defined by the maximum blur radius R_{max}; we used $R_{max} = 48$ in our examples. Using such a large R_{max} not only facilitates the rendering of arbitrary conditions, but it also leans more towards the worst-case scenario that can be expected. With this configuration, the full PSF grid consumes 1 472 MB of GPU memory (74 MB for each unique combination of focus distance and pupil diameter), with another 72 MB used by the PSF texture.

Storing all the kernels on the GPU allows the eye parameters to be dynamically changed at runtime, which is a very large benefit for interactive applications. On the other hand, extending the approach with support for off-axis aberrations is not trivial. Our eye model can be used to obtain the necessary off-axis aberration coefficients, but the asymmetric nature of the eye increases the number of light angles that we need to handle. We believe that the combination of PSF texture packing and streaming the kernels from main memory (to handle parameter changes) would make it possible to extend our method with support for off-axis PSFs, but further experiments are needed to verify the practicality of such an approach.

Eye estimation. Closing off, we also evaluate the performance characteristics and reconstruction precision of the eye estimation procedure. We used the `patternsearch` optimizer in MATLAB and the loss function described in Section 3.2 to implement this process. We also used manual parameter normalization and a very conservative initial mesh size of 0.01. With this setup, the overall convergence speed depends mainly on the input eye condition: on an AMD Ryzen 7 1700 CPU, with the simple, low-order aberrations, the optimizer gets very close to the final output in less than an hour. On the other hand, the highly complex eye with

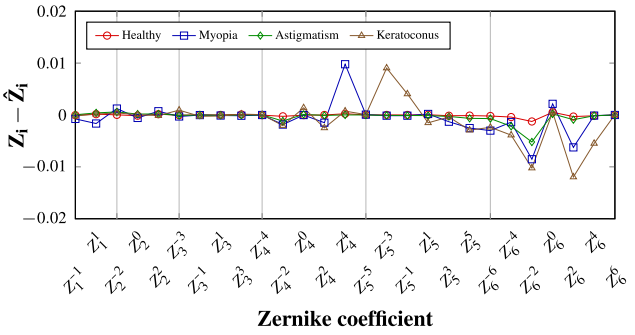


Figure 14: *Difference between the input (Z_i) and generated ($\hat{Z}_i$) Zernike coefficients, for the four estimated eye models corresponding to eye conditions shown in this paper. For reference, the RMS wavefront aberrations are: 0.009 µm (healthy), 5.409 µm (myopia), 4.488 µm (astigmatism), and 6.01 µm (keratoconus).*

keratoconus took longer to estimate, with a total running time of about 4 hours.

The differences between the input and generated aberration coefficients for the demonstrated eye conditions are visualized in Figure 14. As can be seen, the errors are mainly present in the very high Zernike orders, which is likely the outcome of the simplified structure of the crystalline lens that we used in our model. However, the results also clearly show that the error magnitudes are very low, and thus the accuracy of our estimation process is perfectly suitable for our vision simulation purposes. Furthermore, despite the significant variance of the input aberration coefficients for the tested eye conditions, our proposed parametric eye model handles each scenario efficiently, as demonstrated by the resulting error metrics. We can thus also see from this test that our proposed eye model is capable of representing the various kinds of visual aberrations.

5. Conclusions

In this paper, we presented an efficient new method for simulating the visual aberrations of the human eye. Our approach significantly outperforms previous techniques and runs at interactive, near real-time performance, which we achieve by combining the rendering power of tiled splatting with physically-based point-spread functions. We also require only a minimal amount of input data and support arbitrary visual aberrations, offering a high degree of usability. As demonstrated, our proposed solution produces vision simulations that closely match the reference images, while also approaching real-time rendering performance on commodity hardware.

As for future work, we would first like to decrease the running time of the pre-computation step. We believe that deep learning could be useful for the eye estimation process, as we can generate an arbitrary number of training data with the help of our parametric eye model. Moving the PSF computations to the GPU would significantly reduce the length of the pre-computation step as well. We also plan on extending our PSF sampling strategy to support off-axis aberrations. Since the centre of the visual field typically receives the most attention, our current approach of only using on-axis aberrations can generate convincing results. Our eye estimation method is capable of generating the required aberration coefficients, and as such, we see integrating off-axis aberrations into our existing pipeline as another promising area for future research. Lastly, we would also like to further improve the performance of the runtime step, with the end goal of supporting partial occlusion with layered inputs.

Closing off, we hope that our proposed method will help increase the graphical fidelity of vision simulations. We would mainly hope to see more medical applications make use of such techniques, as we believe the area could greatly benefit from these types of algorithms. We also hope that sophisticated vision simulations will eventually find their way to entertainment software as well, as we believe it could raise public awareness of the significance of these conditions.

Acknowledgements

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the TITAN Xp GPU used for this research. This work was supported by the construction EFOP-3.6.3-VEKOP-16-2017-00002. The project was supported by the European Union, co-financed by the European Social Fund.

References

[AV15] Antonello J., Verhaegen M.: Modal-based phase retrieval for adaptive optics. *Journal of the Optical Society of America A 32*, 6 (06 2015), 1160–1170.

[Bar11] Barsky B. A.: Vision-Realistic Rendering: Simulation of the Scanned Foveal Image with Elimination of Artifacts due to Occlusion and Discretization. In *VISIGRAPP 2010: Computer Vision, Imaging and Computer Graphics. Theory and Applications* (Berlin, Heidelberg, 2011), Richard P., Braz J., (Eds.), Springer Berlin Heidelberg, pp. 3–27.

[BP17] Barbero S., Portilla J.: Simulating real-world scenes viewed through ophthalmic lenses. *Journal of the Optical Society of America A 34*, 8 (08 2017), 1301–1308.

[BTA10] Berrio E., Tabernero J., Artal P.: Optical aberrations and alignment of the eye with age. *Journal of Vision 10*, 14 (12 2010), 269–275.

[BvHJD08] Braat J. J., van Haver S., Janssen A. J., Dirksen P.: Assessment of optical systems by means of point-spread functions. *Progress in Optics 51* (2008), 349–468.

[BW54] Bhatia A. B., Wolf E.: On the circle polynomials of *Zernike* and related orthogonal sets. *Mathematical Proceedings of the Cambridge Philosophical Society 50*, 1 (1954), 40–48.

[CBV*04] Cheng H., Barnett J. K., Vilupuru A. S., Marsack J. D., Kasthurirangan S., Applegate R. A., Roorda A.: A population study on changes in wave aberrations with accomodation. *Journal of Vision 4*, 4 (04 2004), 3.

[CLB18] Cholewiak S. A., Love G. D., Banks M. S.: Creating correct blur and its effect on accommodation. *Journal of Vision 18*, 9 (09 2018), 1:1–1:29.

[CLS*17] Cholewiak S. A., Love G. D., Srinivasan P. P., Ng R., Banks M. S.: *ChromaBlur*: Rendering chromatic eye aberration improves accommodation and realism. *ACM Transactions on Graphics 36*, 6 (11 2017), 210:1–210:12.

[CMDlCSNM*16] Cavas-Martí nez F., De la Cruz Sánchez E., Nieto Martínez J., Fernández Cañavate F. J., Fernández-Pacheco D. G.: Corneal topography in keratoconus: State of the art. *Eye and Vision 3* (02 2016), 5:1–5:12.

[CPC84] Cook R. L., Porter T., Carpenter L.: Distributed ray tracing. *ACM SIGGRAPH Computer Graphics 18*, 3 (01 1984), 137–145.

[Dai08] Dai G.-m.: *Wavefront Optics for Vision Correction*. SPIE Press, 2008.

[DF74] Drasdo N., Fowler C. W.: Non-linear projection of the retinal image in a wide-angle schematic eye. *British Journal of Ophthalmology 58*, 8 (08 1974), 709–714.

[DSVdH06] Dubbelman M., Sicam V. A. D. P., Van der Heijde R. G.: The shape of the anterior and posterior surface of the aging human cornea. *Vision Research 46*, 6–7 (03 2006), 993–1001.

[DVdH01] Dubbelman M., Van der Heijde R. G. L.: The shape of the aging human lens: curvature, equivalent refractive index and the lens paradox. *Vision Research 41*, 14 (06 2001), 1867–1877.

[dVWT*11] de Vries N. E., Webers C. A., Touwslager W. R., Bauer N. J., de Brabander J., Berendschot T. T., Nuijts R. M.: Dissatisfaction after implantation of multifocal intraocular lenses. *Journal of Cataract & Refractive Surgery 37*, 5 (2011), 859–865.

[DWRW16] Dias C., Wick M., Rifai K., Wahl S.: Peripheral retinal image simulation based on retina shapes. In *Proceedings of the 37th Annual Conference of the European Association for Computer Graphics: Short Papers* (Goslar, DEU, 2016), Santos L. P., Bashford-Rogers T., (Eds.), EG '16, Eurographics Association, pp. 61–64.

[FHSS18] Franke L., Hofmann N., Stamminger M., Selgrad K.: Multi-layer depth of field rendering with tiled splatting. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 1*, 1 (07 2018), 6:1–6:17.

[FLB*09] Fatahalian K., Luong E., Boulos S., Akeley K., Mark W. R., Hanrahan P.: Data-parallel rasterization of micropolygons with defocus and motion blur. In *Proceedings of the Conference on High Performance Graphics 2009* (New York, NY, USA, 08 2009), Spencer S. N., (Ed.), HPG '09, Association for Computing Machinery, pp. 59–68.

[FM06] Fink W., Micol D.: simEye: computer-based simulation of visual perception under various eye defects using Zernike polynomials. *Journal of Biomedical Optics 11*, 5 (2006), 054011:1–054011:12.

[Gar17] Garcia K.: Circular Separable Convolution Depth of Field. In *ACM SIGGRAPH 2017 Talks* (New York, NY, USA, 2017), SIGGRAPH '17, ACM, pp. 16:1–16:2.

[GD07] Goncharov A. V., Dainty C.: Wide-field schematic eye models with gradient-index lens. *Journal of the Optical Society of America A 24*, 8 (08 2007), 2157–2174.

[GGR*99] Guirao A., González C., Redondo M., Geraghty E., Norrby S., Artal P.: Average optical performance of the human eye as a function of age in a normal population. *Investigative Ophthalmology & Visual Science 40*, 1 (01 1999), 203–213.

[Goo17] Goodman J. W.: *Introduction to Fourier Optics*, 4 ed. W.H. Freeman, Macmillan Learning, 2017.

[GSMM95] Greivenkamp J. E., Schwiegerling J., Miller J. M., Mellinger M. D.: Visual acuity modeling using optical raytracing of schematic eyes. *American Journal of Ophthalmology 120*, 2 (08 1995), 227–240.

[HA90] Haeberli P., Akeley K.: The accumulation buffer: Hardware support for high-quality rendering. *ACM SIGGRAPH Computer Graphics 24*, 4 (09 1990), 309–318.

[Hec86] Heckbert P. S.: Filtering by repeated integration. *ACM SIGGRAPH Computer Graphics 20*, 4 (08 1986), 315–321.

[Her80] Herrmann J.: Least-squares wave front errors of minimum norm. *Journal of the Optical Society of America 70*, 1 (01 1980), 28–35.

[HPD*09] Hermans E. A., Pouwels P. J. W., Dubbelman M., Kuijer J. P. A., van der Heijde R. G. L., Heethaar R. M.: Constant Volume of the Human Lens and Decrease in Surface Area of the Capsular Bag during Accommodation: An MRI and Scheimpflug Study. *Investigative Ophthalmology & Visual Science 50*, 1 (01 2009), 281–289.

[HQL*10] Hou Q., Qin H., Li W., Guo B., Zhou K.: Micropolygon ray tracing with defocus and motion blur. *ACM Transactions on Graphics 29*, 4 (07 2010), 64:1–64:10.

[HS17] Hillesland K., Skelton S.: Cinematic depth of field: How to make big filters cheap. Presented in Game Developers Conference, San Francisco, 2017.

[HSH*02] He J. C., Sun P., Held R., Thorn F., Sun X., Gwiazda J. E.: Wavefront aberrations in eyes of emmetropic and moderately myopic school children and young adults. *Vision Research 42*, 8 (04 2002), 1063–1070.

[HSS97] Heidrich W., Slusallek P., Seidel H.-P.: An image-based model for realistic lens systems in interactive computer graphics. In *Proceedings of the Conference on Graphics Interface '97* (CAN, 05 1997), Davis W. A., Mantei M.,

Klassen R. V., (Eds.), Canadian Information Processing Society, pp. 68–75.

[Jan02] Janssen A. J.: Extended *Nijboer–Zernike* approach for the computation of optical point-spread functions. *Journal of the Optical Society of America A 19*, 5 (05 2002), 849–857.

[JBD04] Janssen A. J., Braat J. J., Dirksen P.: On the computation of the *Nijboer-Zernike* aberration integrals at arbitrary defocus. *Journal of Modern Optics 51*, 5 (2004), 687–703.

[JvHBD07] Janssen A. J., van Haver S., Braat J. J., Dirksen P.: Strehl ratio and optimum focus of high-numerical-aperture beams. *Journal of the European Optical Society - Rapid publications 2* (2007), 07008:1–07008:9.

[JvHJ*08] Janssen O. T. A., van Haver S., Janssen A. J. E. M., Braat J. J. M., Urbach H. P., Pereira S. F.: Extended Nijboer-Zernike (ENZ) based mask imaging: efficient coupling of electromagnetic field solvers and the ENZ imaging algorithm. In *Optical Microlithography XXI* (03 2008), Levinson H. J., Dusa M. V., (Eds.), vol. *6924*, SPIE, pp. 363–371.

[KHB09] Kosloff T. J., Hensley J., Barsky B. A.: Fast filter spreading and its applications. *EECS Department, University of California, Berkeley, Technical Report UCB/EECS-2009-54* (2009).

[KMH95] Kolb C., Mitchell D., Hanrahan P.: A realistic camera model for computer graphics. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 09 1995), Mair S. G., Cook R., (Eds.), SIGGRAPH '95, Association for Computing Machinery, pp. 317–324.

[KMNN10] Kovács I., Miháltz K., Németh J., Nagy Z. Z.: Anterior chamber characteristics of keratoconus assessed by rotating *Scheimpflug* imaging. *Journal of Cataract & Refractive Surgery 36* (07 2010), 1101–1106.

[KPV*18] Khan A., Pope J. M., Verkicharla P. K., Suheimat M., Atchison D. A.: Change in human lens dimensions, lens refractive index distribution and ciliary body ring diameter with accommodation. *Biomedical Optics Express 9*, 3 (02 2018), 1272–1282.

[KTB09] Kosloff T. J., Tao M. W., Barsky B. A.: Depth of field postprocessing for layered scenes using constant-time rectangle spreading. In *Proceedings of Graphics Interface 2009* (Toronto, Ont., Canada, Canada, 2009), Gooch A., Tory M., (Eds.), GI '09, Canadian Information Processing Society, pp. 39–46.

[KTMN07] Kakimoto M., Tatsukawa T., Mukai Y., Nishita T.: Interactive simulation of the human eye depth of field and its correction by spectacle lenses. *Computer Graphics Forum 26*, *3* (2007), 627–636.

[KTN10] Kakimoto M., Tatsukawa T., Nishita T.: An eyeglass simulator using conoid tracing. *Computer Graphics Forum 29*, 8 (12 2010), 2427–2437.

[KvB03] Křivánek J., Žára J., Bouatouch K.: Fast depth of field rendering with surface splatting. In *Proceedings Computer Graphics International 2003* (2003), Werner B., (Ed.), IEEE Computer Society Press, pp. 196–201.

[Lar79] Larsen J. S.: Axial length of the emmetropic eye and its relation to the head size. *Acta Ophthalmologica 57*, 1 (02 1979), 76–83.

[LES09] Lee S., Eisemann E., Seidel H.-P.: Depth-of-field rendering with multiview synthesis. *ACM Transactions on Graphics 28*, 5 (12 2009), 134:1–134:6.

[LES10] Lee S., Eisemann E., Seidel H.-P.: Real-time lens blur effects and focus control. *ACM Transactions on Graphics 29*, 4 (07 2010), 65:1–65:7.

[LH13] Lei K., Hughes J. F.: Approximate depth of field effects using few samples per pixel. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 03 2013), Spencer S. N., (Ed.), I3D '13, ACM, pp. 119–128.

[LMB*19] Lian T., MacKenzie K. J., Brainard D. H., Cottaris N. P., Wandell B. A.: Ray tracing 3D spectral scenes through human optics models. *Journal of Vision 19*, 12 (10 2019), 23:1–23:17.

[LSR18] Leimkühler T., Seidel H.-P., Ritschel T.: Laplacian kernel splatting for efficient depth-of-field and motion blur synthesis or reconstruction. *ACM Transactions on Graphics 37*, 4 (08 2018), 55:1–55:11.

[LSS01] Loos J., Slusallek P., Seidel H.-P.: Using Wavefront Tracing for the Visualization and Optimization of Progressive Lenses. *Computer Graphics Forum 17*, 3 (12 2001), 255–265.

[LT19] Liu T., Thibos L. N.: Customized models of ocular aberrations across the visual field during accommodation. *Journal of Vision 19*, 9 (08 2019), 13:1–13:24.

[LU07] Lundströ m L., Unsbo P.: Transformation of *Zernike* coefficients: scaled, translated, and rotated wavefronts with circular and elliptical pupils. *Journal of the Optical Society of America A 24*, 3 (03 2007), 569–577.

[McG14] McGraw T.: Fast bokeh effects using low-rank linear filters. *The Visual Computer 31*, 5 (05 2014), 601–611.

[MH14] Moersch J., Hamilton H. J.: Variable-sized, circular bokeh depth of field effects. In *Proceedings of Graphics Interface 2014* (Toronto, Ontario, Canada, 2014), Kry P. G., Bunt A., (Eds.), GI '14, Canadian Information Processing Society, pp. 103–107.

[MKL97] Mostafawy S., Kermani O., Lubatschowski H.: Virtual eye: Retinal image visualization of the human eye. *IEEE Computer Graphics and Applications 17*, 1 (01 1997), 8–12.

[MKRH11] Mantiuk R., Kim K. J., Rempel A. G., Heidrich W.: HDR-VDP-2: A calibrated visual metric for visibility and quality predictions in all luminance conditions. *ACM Transactions on Graphics 30*, 4 (07 2011), 40:1–40:13.

[MMB01] McLellan J. S., Marcos S., Burns S. A.: Age-related changes in monochromatic wave aberrations of the human eye. *Investigative Ophthalmology & Visual Science 42*, 6 (05 2001), 1390–1395.

[MRD12] McIntosh L., Riecke B. E., DiPaola S.: Efficiently simulating the bokeh of polygonal apertures in a post-process depth of field shader. *Computer Graphics Forum 31*, 6 (03 2012), 1810–1822.

[MS44] Moon P., Spencer D. E.: On the *Stiles-Crawford* effect. *Journal of the Optical Society of America A 34*, 6 (06 1944), 319–329.

[NKN*14] Namba H., Kawasaki R., Narumi M., Sugano A., Homma K., Nishi K., Murakami T., Kato T., Kayama T., Yamashita H.: Ocular higher-order wavefront aberrations in the *Japanese* adult population: the *Yamagata* study (*Funagata*). *Investigative Ophthalmology & Visual Science 56*, 1 (12 2014), 90–97.

[NSB85] Navarro R., Santamaría J., Bescós J.: Accommodation-dependent model of the human eye with aspherics. *Journal of the Optical Society of America A 2*, 8 (08 1985), 1273–1280.

[NSG12] Nießner M., Sturm R., Greiner G.: Real-time Simulation and Visualization of Human Vision Through Eyeglasses on the GPU. In *Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry* (New York, NY, USA, 2012), Spencer S. N., (Ed.), VRCAI '12, ACM, pp. 195–202.

[PAA*10] Piñero D. P., Alió J. L., Alesón A., Vergara M., Miranda M.: Corneal volume, pachymetry, and correlation of anterior and posterior corneal shape in subclinical and different stages of clinical keratoconus. *Journal of Cataract & Refractive Surgery 36*, 5 (05 2010), 814–825.

[PC81] Potmesil M., Chakravarty I.: A lens and aperture camera model for synthetic image generation. *ACM SIGGRAPH Computer Graphics 15*, 3 (08 1981), 297–305.

[PGCW01] Porter J., Guirao A., Cox I. G., Williams D. R.: Monochromatic aberrations of the human eye in a large population. *Journal of the Optical Society of America A 18*, 8 (08 2001), 1793–1803.

[PGO08] Pruss C., Garbusi E., Osten W.: Testing aspheres. *Optics and Photonics News 19*, 4 (04 2008), 24–29.

[RAK*12] Rozema J. J., Atchison D. A., Kasthurirangan S., Pope J. M., Tassignon M.-J.: Methods to Estimate the Size and Shape of the Unaccommodated Crystalline Lens In Vivo. *Investigative Ophthalmology & Visual Science 53*, 6 (05 2012), 2533–2540.

[RAT11] Rozema J. J., Atchison D. A., Tassignon M.-J.: Statistical Eye Model for Normal Eyes. *Investigative Ophthalmology & Visual Science 52*, 7 (06 2011), 4525–4533.

[RRNT16] Rozema J. J., Rodriguez P., Navarro R., Tassignon M.-J.: SyntEyes: A Higher-Order Statistical Eye Model for Healthy Eyes. *Investigative Ophthalmology & Visual Science 57*, 2 (02 2016), 683–691.

[SBHL99] Simard P., Bottou L., Haffner P., LeCun Y.: Boxlets: A fast convolution algorithm for signal processing and neural networks. In *Advances in Neural Information Processing Systems 11* (1999), Kearns M. J., Solla S. A., Cohn D. A., (Eds.), MIT Press, pp. 571–577.

[Shi94] Shinya M.: Post-filtering for depth of field simulation with ray distribution buffer. In *Proceedings of Graphics Interface '94* (Toronto, Ontario, Canada, 1994), Davis W. A., Joe B., (Eds.), GI 1994, Canadian Information Processing Society, pp. 59–66.

[Sou13] Sousa T.: *CryEngine* 3 graphics gems. Presented in Advances in Real-Time Rendering in Games, SIGGRAPH Course, Los Angeles, 08 2013.

[TFTA16] Tse S. M., Farley N. D., Tomasko K. R., Amin S. R.: Intraoperative LASIK complications. *International Ophthalmology Clinics 56*, 2 (2016), 47–57.

[THBC02] Thibos L. N., Hong X., Bradley A., Cheng X.: Statistical variation of aberration structure and image quality in a normal population of healthy eyes. *Journal of the Optical Society of America A 19*, 12 (12 2002), 2329–2348.

[TX15] Tang N., Xiao S.: Real-time Human Vision Rendering Using Blur Distribution Function. In *Proceedings of the 14th ACM SIGGRAPH International Conference on Virtual Reality Continuum and its Applications in Industry* (New York, NY, USA, 10 2015), Spencer S. N., (Ed.), VRCAI '15, ACM, pp. 39–42.

[VDC*15] Vinas M., Dorronsoro C., Cortes D., Pascual D., Marcos S.: Longitudinal chromatic aberration of the human eye in the visible and near infrared from wavefront sensing, double-pass and psychophysics. *Biomedical Optics Express 6*, 3 (03 2015), 948–962.

[VH10] Van Haver S.: *The Extended Nijboer-Zernike Diffraction Theory and its Applications*. PhD thesis, Delft University of Technology, 2010.

[Wat15] Watson A. B.: Computing human optical point spread functions. *Journal of Vision 15*, 2 (02 2015), 26:1–26:25.

[WBB11] White J., Barré-Brisebois C.: More Performance! Five Rendering Ideas from Battlefield 3 and Need for Speed: The Run. Presented in Advances in Real-Time Rendering in Games, SIGGRAPH Course, Vancouver, 08 2011.

[WPP14] Wei Q., Patkar S., Pai D. K.: Fast ray-tracing of human eye optics on Graphics Processing Units. *Computer Methods and Programs in Biomedicine 114*, 3 (05 2014), 302–314.

[WY12] Watson A. B., Yellott J. I.: A unified formula for light-adapted pupil size. *Journal of Vision 12*, 10 (09 2012), 12:1–12:16.

[WZJ*03] Wang Y., Zhao K., Jin Y., Niu Y., Zuo T.: Changes of higher order aberration with various pupil sizes in the myopic eye. *Journal of Refractive Surgery 19*, 2 (03 2003), S270–S274.

Supporting Information

Additional supporting information may be found online in the Supporting Information section at the end of the article.

Data videoS1

Data videoS2

Data videoS3

Data videoS4