The Visual Computer
https://doi.org/10.1007/s00371-023-03060-0

ORIGINAL ARTICLE

Fast rendering of central and peripheral human visual aberrations across the entire visual field with interactive personalization

István Csoba[1,2] · Roland Kunkli[1]

Accepted: 9 August 2023

Abstract
With the recent progress made in areas such as head-mounted displays and vision-correcting devices, there is a growing interest in fast and personalized algorithms for simulating aberrated human vision. Existing vision-simulating approaches are generally hindered by the lack of personalization, computational cost of rendering, and limited types of supported aberrations. This paper presents a fast vision simulation method with interactive personalization capabilities for simulating arbitrary central and peripheral aberrations of the human eye. First, we describe a novel, neural network-based solution for efficiently estimating the physical structure of the simulated eye and calculating the necessary Zernike aberration coefficients for computing the point-spread functions with varying pupil sizes, focus distances, and incidence angles. Our new approach operates in the sub-second regime and produces highly accurate outputs, facilitating the interactive personalization of vision simulation. Next, we present an improved PSF interpolation method for an existing tiled PSF splatting algorithm for rendering. The proposed algorithm significantly improves the computational performance and memory efficiency of the previous approach, allowing the simulation of peripheral vision with arbitrary visual aberrations in low-latency applications. Following the description of our new techniques, we evaluate their performance characteristics and simulation accuracies on several different eye conditions and test scenarios and compare our results to several previous vision simulation algorithms.

Keywords Human vision simulation · Eye structure estimation · Aberration estimation · Neural networks · Convolution · Point-spread function interpolation

1 Introduction

The field of ophthalmology has seen an increasing interest in vision simulation methods, owing largely to recent advancements in general computing and rendering. A great number of applications exist [1–4], including the evaluation of visual acuity [5–8], vision testing [9], studying the mechanisms of the human eye [10–12], evaluating the impact of eye diseases on our ability to perform real-world tasks [13], and the visualization of eye diseases in synthetic [14–16], virtual reality [17–19], and augmented reality [19–21] environments.

Personalized vision simulations are enormously useful for the proper planning of invasive vision-correcting processes (such as intraocular lens implants [22] and laser surgery [23]) because of the inherent risk of post-surgery complications posed by these operations. Non-invasive vision-correcting solutions (such as progressive lenses [24–27], vision-correcting displays [28–32], head-mounted displays [33], and holographic lenses [34]) are also rapidly evolving and benefit greatly from improved vision simulation algorithms.

Head-worn displays are common building blocks of many vision-correcting devices and can cause discomfort over an extended period of use, owing largely to the disparity between the vision of the eye and the synthetic images shown to the user. Accounting for the characteristics of the wearer's visual system via the inclusion of depth-of-field (DOF) [35–38], chromatic aberration [39, 40], and the eye's own visual aberrations [41] can significantly reduce the discomfort experienced by the user. Special-purpose fonts have also been developed to reduce the fatigue from rapid context switches

✉ István Csoba
csoba.istvan@inf.unideb.hu

Roland Kunkli
kunkli.roland@inf.unideb.hu

1 Faculty of Informatics, University of Debrecen, Debrecen, Hungary

2 Doctoral School of Informatics, University of Debrecen, Debrecen, Hungary

Published online: 29 September 2023

[42], a common issue with augmented reality applications. Fast, accurate, and personalizable vision simulation methods are essential in developing and implementing these features.

Wavefront aberrations are highly impactful for all optical systems, and algorithms for synthetically generating images that properly account for these effects have a long history in computer graphics [43–47]. Such aberrations greatly affect the human eye as well, and thus, numerous solutions exist for simulating visual aberrations [48]. These methods are often hindered by the disregard of peripheral aberrations, computational cost of the simulation, and lack of easy personalization with user-specific information. Recently, Csoba and Kunkli [49] solved the performance and personalization issues at the expense of a long precomputation step. Furthermore, their algorithm ignored peripheral aberrations, which is a common limitation of convolution-based approaches. Peripheral vision is an important source of information for many vision-dependent tasks [50–53] and can behave in highly unexpected ways when eye diseases are present. As an example, we illustrate *relative peripheral hyperopia* in Fig. 1, whereby the blurriness of a myopic eye decreases as the angle of incidence increases, causing the sharp region to fall outside the central vision [54, 55].

In this paper, we present our new solution for the interactively personalizable simulation of ocular wavefront aberrations, with arbitrary compositions of low- and high-order aberrations, for central and peripheral vision. We improve upon [49] in several critical ways; our main contributions are:

- A neural network-based method for estimating the eye structure and its aberrations in various states, which achieves sub-second computational performance, facilitating the interactive personalization of the simulation.
- An improved PSF interpolation strategy for simulating central visual aberrations, which vastly reduces the memory footprint of the previous process and leads to a fully real-time performance profile.
- An extension of our new PSF interpolation approach, which facilitates the simulation of peripheral vision across the entire visual field with an approximately real-time performance.

2 Previous work and motivation

2.1 Algorithms with an offline performance profile

Among the earliest works utilizing vision simulation, Camp et al. used paraxial ray tracing with a simplified eye model to efficiently calculate the PSF of the eye and performed convolution with the PSF to simulate vision [56]. Greivenkamp et al. improved the accuracy of this approach by using a full aspherical eye model, exact ray tracing, and a model of the Stiles–Crawford effect via an apodizing filter [7].

To facilitate personalized vision simulations, Barsky utilized a Shack–Hartmann aberrometer to calculate the PSFs from measurement data [57]. Ray tracing was employed with the measured aberration surface to compute a small set of depth-dependent PSFs, and the vision-simulated imagery was acquired by partitioning the input image into depth slices and performing convolution on each slice with the corresponding depth-dependent PSFs. This approach causes visible artifacts along the edges of the depth slices, which Barsky solved using object detection [58].

To simulate peripheral aberrations of progressive addition lenses, Rodríguez Celaya et al. used a small 3D PSF grid (with different axes corresponding to the horizontal angle, vertical angle, and object-space distance) and trilinear interpolation for approximating the per-pixel PSF of the lens [26]. Alternatively, Gonzalez-Utrera utilized a similar 3D PSF grid with Barsky's depth-dependent image-splitting approach [27]. A critical limitation of the two methods is that the simulated peripheral area is highly restricted and the axial resolution of the PSF grid is small for accurate simulations.

Instead of convolution, Mostafawy et al. utilized the Gullstrand schematic eye model and a flat retina for simulating vision using distributed ray tracing [59]. Personalization was achieved by placing additional lenses in front of the eye model and utilizing several corneal zones with varying refractive profiles. To properly handle peripheral vision, Wu et al. utilized the aspherical Navarro eye model [60] and Dias et al. investigated the impact of retina shape on vision [61]. The contributions of these works were collected by Lian et al. and further extended with other aspects (such as spectral ray tracing [62] for simulating chromatic aberration and stochastic ray-bending to model diffraction effects) in their open-source vision simulation tool called *ISET3d* [63].

For an even more detailed and personalizable eye structure representation, Fink and Micol modeled the refracting surfaces of the eye using the Zernike circle polynomials and used ray tracing with 2D images to simulate vision [64]. A significant drawback of this method is the high cost of the iterative intersection calculations, which can be solved by utilizing polygon meshes to describe the refracting elements, facilitating the use of much faster, triangle-based intersection computation tools, as demonstrated by Wei et al. [65].

Another severe issue with ray tracing is the high levels of noise stemming from its stochastic nature. To overcome this issue, Vu et al. employed an interpolation-based approach, whereby forward ray tracing and a coarse ray grid were used with a 2D input image to compute a small set of retinal samples, with triangulation and interpolation utilized to fill in the gaps in the grid and obtain a noise-free simulation [66].

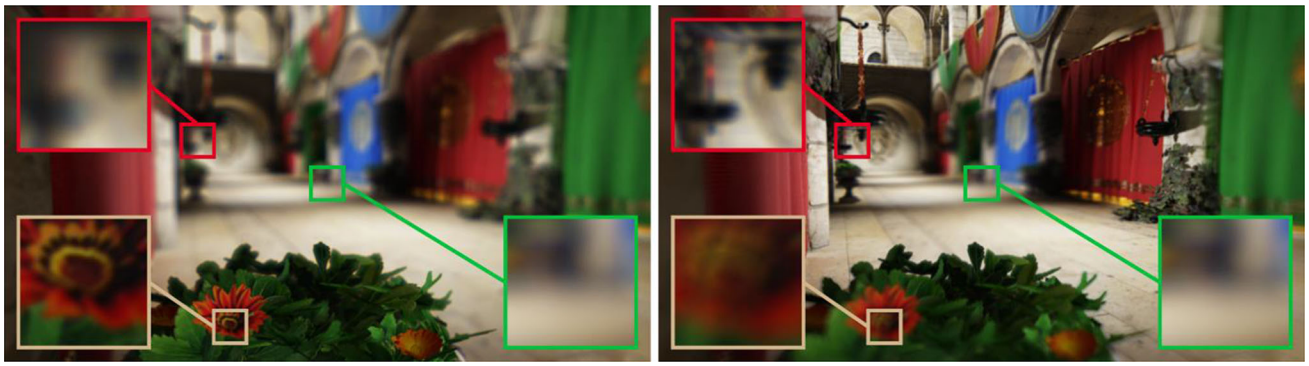


Fig. 1 Examples of visual aberration simulations generated using our proposed approach for a highly myopic eye for demonstrating the differences between ignoring (left) and simulating (right) peripheral visual aberrations. Eye structure and aberration estimation combined took 0.28 s (left) and 0.66 s (right) on an AMD Ryzen 7 1700 CPU. Rendering took 6.68 ms (left) and 12.38 ms (right) on an NVIDIA TITAN Xp GPU

2.2 Real-time vision simulation algorithms

The algorithms described thus far are only suitable for offline applications. As an inexpensive alternative to PSF-based convolution, Tang and Xiao utilized Gaussian kernels and a neural network-based per-pixel blurriness map generated with a schematic eye model [67]. This approach facilitates the real-time simulation of peripheral vision, albeit limited to low-order aberrations due to the characteristics of the kernel.

Alternatively, Lima et al. utilized a tree data structure for encoding the list of image-space sample locations on layered input images for several sample points from the pupil disk [68]. This solution facilitates the real-time simulation of low-order aberrations and efficiently handles partial occlusion, a common issue with convolution-based algorithms. Peripheral vision, however, was not considered by this study.

For the real-time simulation of arbitrary central visual aberrations, Csoba and Kunkli utilized a coarsely sampled PSF grid [49], similar to Barsky's concept. Tiled PSF splatting and a GPU-based per-pixel PSF interpolation method were employed to achieve interactive performance. To derive the unknown aberration coefficients for the entire PSF grid from a single aberration measurement, the authors described a custom parametric eye model and eye structure estimation process, which facilitates the simulation of chromatic aberration, varying pupil sizes, and configurable focus distances but hinders the performance of the system, because the eye estimation process can take hours to complete.

Recently, Xiao et al. used convolutional neural networks and deep learning for simulating the DOF of the human eye [35]. However, their main goal was to reduce the vergence-accommodation conflict when wearing head-mounted displays, and thus, their algorithm is limited to healthy vision.

2.3 Summary and motivation

In summary, despite the significant number of vision simulation algorithms, personalization is a mostly overlooked aspect of the system. Although *ocular wavefront tomography* [69] can be utilized for overcoming the issue, it requires a lengthy optimization process and a large number of aberration measurements. Alternatively, to estimate the eye structure from a single on-axis measurement, Csoba and Kunkli [49] used an optimization-based approach with a custom parametric eye model; however, their method is still very time-consuming and often takes several hours to produce an output. To overcome these limitations, we leverage the massive generalization capabilities of neural networks, which facilitate the interactive estimation of the eye structure and aberration coefficients from a single measurement for producing the PSFs of convolution-based vision simulation algorithms.

Furthermore, there is a lack of high-performance algorithms for simulating peripheral vision, with the existing approaches either being unsuitable for interactive environments or unable to fully reproduce the true aberration structure of the eye. We solve these problems by utilizing the tiled PSF splatting approach of Csoba and Kunkli [49] during the rendering stage and proposing a significantly faster and more memory-efficient PSF interpolation method, facilitating the efficient simulation of arbitrary central and peripheral visual aberrations in highly interactive and real-time environments.

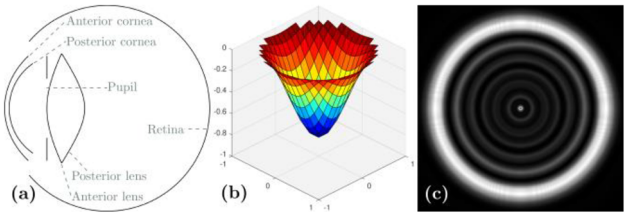


Fig. 2 Optical concepts of the human eye discussed in this section. **a** Cross sectional view of the eye model proposed in [49]. **b** Plot of a wavefront aberration surface calculated using this eye model. **c** The PSF computed using the aberration surface shown in (**b**)

3 Mathematical background of eye optics

In this section, we provide a brief overview of the necessary concepts of human eye optics to describe our new results. The main concepts discussed are visualized in Fig. 2.

3.1 Structure of the parametric eye model

Despite the existence of numerous schematic human eye models in the scientific literature, most models focus on accurately representing a healthy eye. To handle arbitrary eye conditions and estimate the simulated eye's physical structure, Csoba and Kunkli constructed a simplified parametric eye model with the aim of balancing the number of model parameters and physical accuracy of the model. Figure 2a displays a cross sectional view of their eye model.

Their parametric model is based on Navarro's unaccommodated aspherical eye model [70] and uses a spherical surface as the retina, a plane with a variable circular hole as the pupil, and four refracting surfaces to model the cornea and crystalline lens: the front (anterior) and back (posterior) surfaces for both components. These four surfaces were modeled as quadrics of revolution with the following formula:

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

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

Additionally, the authors split the radii of curvature of the two corneal surfaces into separate terms for simulating corneal astigmatism and added an offset function to the front surface for modeling its fine-scale irregularities. Thus, their proposed anterior cornea surface can be defined as follows:

$$z_c^A(x,\, y) = z_c(x,\, y) + \Delta(x,\, y), \tag{2}$$

$$z_c(x,\, y) = \frac{\frac{x^2}{R_x} + \frac{y^2}{R_y}}{1 + \sqrt{1 - (1+k) \cdot \left(\frac{x^2}{R_x^2} + \frac{\frac{R_y}{R_x} \cdot y^2}{R_y^2}\right)}}, \tag{3}$$

$$\Delta(x,\, y) = \sum_{n,m} \alpha_n^m \cdot U_n^m(x,\, y), \tag{4}$$

where z_c is the astigmatic surface, Δ is the offset function, R_x and R_y are the astigmatic radii of curvature, U_n^m is the Zernike polynomial of radial order n and azimuthal order m (as defined by (2.19) in [71], with $n \geq |m| \geq 0$, $n - |m|$ even), and α_n^m is the expansion coefficient for U_n^m.

Finally, the authors also added a rotation term (around the optical axis) to the cornea and a set of tilt and decentration parameters (along the x- and y-axis) to the lens.

3.2 Wavefront aberrations and the PSF of the eye

The PSF of the human eye is the projection of an infinitesimal point source on the retina and heavily depends on many parameters, including light wavelength, distance of the point source from the eye, horizontal and vertical incidence angles, pupil size, and focus distance. The terms *on-axis* and *off-axis* are generally used to refer to the origin of the point source with respect to the optical axis. Among the main uses of the PSF is the simulation of the eye's optical performance, whereby the PSF is utilized as a kernel to perform a convolution on the input image.

One way of calculating the PSF is using diffraction theory and wavefront aberrations of the eye. Wavefront aberrations describe the differences in optical path length between a reference wavefront (locus of points with the same phase) and the emerging wavefront as incoming waves pass through the optical system. The wavefront aberrations of the human eye are often described using Zernike polynomials [72]:

$$W(\rho,\, \phi) = \sum_{n,m} \alpha_n^m \cdot U_n^m(\rho,\, \phi), \tag{5}$$

where (ρ, ϕ) are polar coordinates in the exit pupil plane, U_n^m is the circle polynomial of Zernike of radial order n and azimuthal order m (with the same definition as in (4)), and α_n^m is the aberration coefficient corresponding to U_n^m. An example of an aberration surface is shown in Fig. 2b.

To obtain the aberration coefficients of the human eye, a *wavefront aberrometer* is often employed for measuring the aberrations for a set of parameters (pupil size, focus distance, and incidence angles), with the results often being made available in eye aberration studies. Conversion formulas also exist for simulating common eye conditions (such as myopia, hyperopia, and astigmatism) using corrective spectacle lens prescriptions [73]. Alternatively, ray tracing can

be employed if the physical structure of the eye is available, which can be obtained using eye structure estimation from aberration measurements, as demonstrated in [49], where a generalized pattern search (GPS) algorithm was utilized to estimate the physical eye structure from on-axis aberrations.

Given an input set of wavefront aberration coefficients, the extended Nijboer-Zernike (ENZ) diffraction theory [74] can be utilized for efficiently calculating the corresponding PSF at arbitrary defocus. The ENZ theory provides several different formulations of the PSF; for the human eye and arbitrary defocus, the large-defocus scalar formula is suitable:

$$U(r,\,\phi,\,f) = 2\sum_{n,m} \beta_n^m i^{|m|} V_n^{|m|}(r,\,f)\exp\{\mathrm{im}\phi\}, \tag{6}$$

where U is the PSF, $(r,\,\phi)$ are polar coordinates in the image plane (normalized by the diffraction unit λ/NA, where NA is the image-side numerical aperture of the system), f is the defocus parameter ($f = 0$ at best focus and $f = \frac{-2\pi u_0}{\lambda} z$, where z is the amount of focus shift in free-space, $u_0 = 1 - \sqrt{1 - s_0^2}$, $s_0 = \mathrm{NA}/n$, and n is the refraction index in the image space of the optical system). Additionally, the V_n^m functions are basic integrals of the Zernike radial polynomials and the β_n^m terms are complex expansion coefficients that can be derived from the real-valued α_n^m phase-aberration coefficients. The authors of [49] used the Bessel-Bessel series expression for V_n^m (as given by (26) in [72]) and the fitting strategy described by (30) in [75] to calculate the β_n^m values corresponding to a set of α_n^m aberration coefficients. An example of a PSF generated using (6) is shown in Fig. 2c.

The ENZ PSF formulation has several benefits. Firstly, it facilitates the computation of the depth-dependent PSF without the need for additional aberration coefficients. Secondly, the V_n^m terms can be cached for significantly shortening the PSF computation process. Finally, the V_n^m terms can also be rearranged in a GPU-friendly manner for computing large sets of PSF in a highly efficient, GPU-based manner [76].

4 Our proposed method

An overview of the main steps of our proposed vision simulation approach is presented in Fig. 3, which comprises three main phases: training, precomputation, and rendering. During training, we construct a set of neural networks to estimate the unknown eye structure and aberration coefficients. To simulate vision, we adopt the tile-based rendering approach of Csoba and Kunkli [49] and utilize the trained networks during all subsequent precomputation and rendering steps.

During precomputation, we generate the PSF grid for the online rendering stage using a GPU-based PSF computation approach [76]. We employ the networks trained during the training phase for efficiently estimating the structure of the eye in the relaxed and focused states and computing the aberration coefficients for an input set of PSF parameters (comprising object distance d, angles of incidence h and v, light wavelength λ, aperture diameter A, and focus distance f).

In the rendering stage, we utilize tiled convolution [49], whereby screen-aligned tiles are constructed out of the pixels of the input color and depth images and traversed for each output pixel to generate the vision-simulated image, utilizing a 3D GPU texture to approximate the PSF with hardware acceleration. We improve this approach with a much more memory-efficient and thus significantly faster sampling scheme for on-axis aberrations and an extension facilitating the fast and efficient rendering of off-axis aberrations.

5 Estimation of eye parameters and wavefront aberrations using neural networks

5.1 Overview of our neural networks

Conceptually, estimating the eye structure and its unknown aberrations requires the following main steps:

1. Find a set of parameters for the relaxed eye with aberrations closely matching the input aberrations.
2. Modify the crystalline lens parameters to focus the eye at each user-defined focus distance.
3. Compute the aberration coefficients using the resulting eye models for each PSF parameter combination.

We constructed a separate network to solve these tasks; the exact network inputs and outputs are summarized in Table 1. First, we built an *eye structure estimator* and a *focused eye estimator* network to estimate the relaxed and focused eye parameters from an input set of aberration measurements and focus distances. Next, we created an *aberration estimator* for computing the Zernike aberration coefficients using the focused eye parameters and input PSF parameters. Finally, we also constructed a *discriminator* network for computing the on-axis aberrations for an input set of eye parameters, which we used during the training of the eye structure estimator, as described in Sect. 5.4.

Despite the theoretical possibility of merging certain networks, our approach facilitates the use of multiple smaller networks, and thus, significantly reduces their individual memory footprints, increases the training accuracy and performance, and keeps the inference costs low. Furthermore, the estimated eye model parameters can be directly observed, providing additional knowledge about the simulated eye.

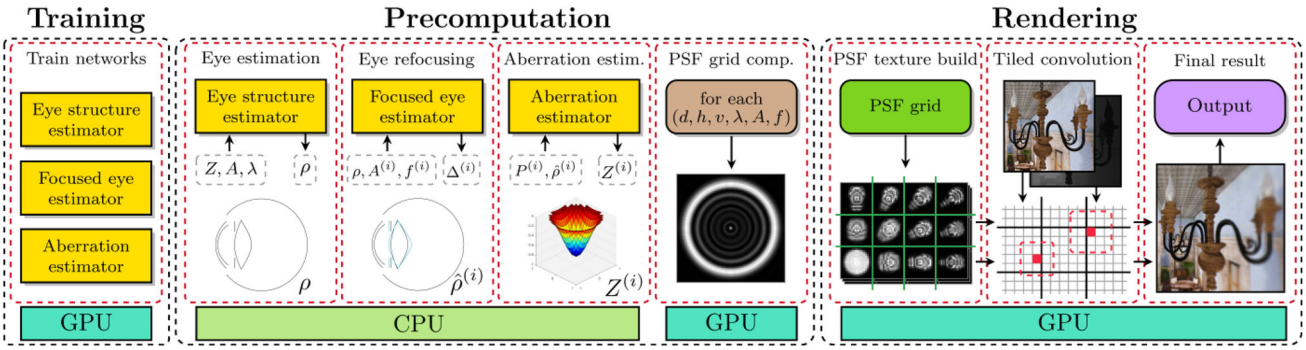


Fig. 3 Overview of our proposed solution for simulating aberrated human vision. A set of neural networks are first trained to estimate the relaxed (ρ) and focused ($\hat{\rho}^{(i)}$) eye structure parameters using the input aberration measurement settings (Z, A, λ), aperture sizes ($A^{(i)}$), and focus distances ($f^{(i)}$), and another network to compute the aberration coefficients ($Z^{(i)}$) of the focused eye for a set of PSF parameters ($P^{(i)}$). During precomputation, these networks are used to construct a coarse PSF grid, stored on the GPU. At runtime, vision is simulated using the tiled PSF-splatting approach of [49] and an improved PSF interpolation method proposed in this paper, whereby a 3D PSF texture is constructed from the precomputed PSF grid, and the output vision simulation is generated using tiled convolution with an input color map, depth map, and the per-sample approximation of the PSF

Table 1 Inputs and outputs of our neural networks

Network	Inputs	Outputs
Eye structure estimator	Z_{2-28}, A, λ	ρ_{1-45}
Discriminator	ρ_{1-45}, A, λ	Z_{2-28}
Focused eye estimator	ρ_{1-45}, A, f	Δ_{L_D}, Δ_{A_T}
Aberration estimator	ρ_{1-45}, A, λ, h, v	Z_{2-28}

The variable names are described in Sect. 5.2

5.2 Training data generation

The use of the custom parametric eye model described in Sect. 3.1 necessitated the construction of custom datasets for network training. We generated a different dataset for each estimator because the network inputs and outputs (and the means to obtain them) vary considerably per network.

To further reduce the complexity of the eye model and make its parametrization better suited for neural network learning, we modified the modeling of the cornea. Instead of defining the radius of curvature separately for the anterior and posterior surfaces, we defined the posterior radius of curvature using an anterior-to-posterior ratio parameter and used a dedicated astigmatism term (ratio of R_y to R_x). The resulting list of parameters is listed in Table 2.

5.2.1 Aberration datasets

Because the eye structure estimator, aberration estimator, and discriminator networks are all based on aberrations and eye structure parameters, we used the same approach to generate the training datasets for all three networks. First, we constructed a modified set of eye parameter domains (summarized in Table 2) using the eye population data described in [49]. We then generated a set of samples for each dataset by randomly sampling (with a uniform distribution) the eye parameters ρ_{1-45} (comprising the *Cornea*, *Aqueous*, *Lens*, and *Eye* parameters in Table 2) and other network inputs (the PSF parameters A, λ, h, and v). Next, we calculated the corresponding aberration coefficients Z_{2-28} (the first six degrees of Zernike polynomials, with the Z_1 piston term ignored) using ray tracing. Finally, we formed feature and target pools from the columns of the eye parameters and corresponding aberration coefficients, based on the roles of the columns in the network the dataset is being generated for.

As noted by the authors of [49], the cost of aberration computation contributed greatly to the length of their eye structure estimation step, because the GPS optimizer computes aberrations for a high number of samples. To reduce the cost of eye reconstruction, the authors skipped the steps where the ray bundles are aligned to the pupil center and the outgoing ray grid is fit to the physical pupil disk. These simplifications introduce inaccuracies into the resulting aberration coefficients, because the non-centered samples alter the distribution of asymmetric coefficients and the incomplete wavefront information causes deviations from the true aberrations. Although the GPS algorithm can recover from noisy sample locations, an inaccurate dataset can substantially hinder the training of neural networks. Furthermore, the authors only considered on-axis input locations, which further exacerbates the impact of the simplifications outlined above, because our goal of estimating peripheral aberrations also requires the computation of off-axis aberrations.

To overcome the aforementioned limitations, we extended the aberration computation approach described in [49] with

Table 2 Parameters and ranges used to compute our training datasets

Parameter		Unit	Value	Parameter		Unit	Value
Cornea	T	mm	[0.5, 0.6]	Aqueous	T	mm	[1.5, 4.5]
	R^A	mm	[7, 8.6]	Lens	V	mm^3	[150, 165]
	A^P		[0.9, 1.1]		k^A		[−10, −1]
	r_R^P		[0.8, 0.85]		k^P		[−6, −1]
	k^A		[−1.5, −0.01]		Δ_x	mm	[−0.2, 0.2]
	k^p		[−1.5, −0.01]		Δ_y	mm	[−0.2, 0.2]
	ϕ^A	deg	[−45, 45]		α_x	deg	[−3, 3]
	ϕ^P	deg	[−45, 45]		α_y	deg	[−3, 3]
	Z_{1-2}^A	mm	[−0.1, 0.1]	Eye	T	mm	[22.5, 25.5]
	Z_{3-4}^A	mm	[−0.06, 0.06]				
	Z_{5-6}^A	mm	[−0.03, 0.03]				

Parameter		Unit	Eye	Focus	Aberration
Lens	D	mm	[9.25, 9.75]	[9.25, 9.75]	[8.4, 9.75]
Pupil	D	mm	[3, 6]	[2, 7]	[2, 7]
Focus	f	D	–	[0.125, 8.125]	–
Rays	λ	nm	[450, 900]	–	[450, 700]
	h	deg	–	–	[−45, 45]
	v	deg	–	–	[−26, 26]

The main eye parameter types are: thickness (T), radius of curvature (R), astigmatism (A), ratio of anterior-to-posterior radius of curvature (r_R), diameter (D), conic constant (k), rotation around the optical axis (Φ), Zernike offset surface coefficient of radial order n (Z_n), volume (V), translation (Δ), and tilt (α). The superscripts indicate whether a parameter belongs to the anterior or posterior surface

several extra steps, leading to a much more physically correct approach that significantly improves the accuracy of the wavefront aberration computation process. The main steps of our modified algorithm are shown in Fig. 4 and an in-depth description is provided in Appendix 1.

5.2.2 Focused eye parameters dataset

Our approach for generating the training dataset of the focused eye estimator was similar to the one described in Sect. 5.2.1. We generated a set of random input samples (each comprising the 45 eye parameters ρ_{1-45}, a pupil diameter A, and a diopter-based focus distance f) and calculated the corresponding changes of the eye parameters (lens diameter Δ_{L_D} and aqueous chamber thickness Δ_{A_T}) that focus the sampled eye models on the object distances of the samples.

To calculate Δ_{L_D} and Δ_{A_T}, we used the eye-refocusing method described in [49]. It attempts to mimic the real focusing process of the human eye and works by testing a set of modified eye models with shrunken crystalline lens diameters, adapting the radii of curvature of the lens surfaces accordingly for retaining a constant lens volume [78]. A small grid of rays is traced through each test eye to obtain a set of retinal sample locations for each model, selecting as the result the model with the least spread of the test ray grid.

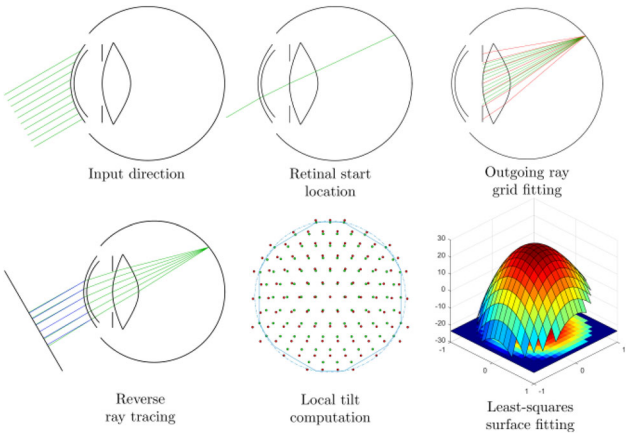


Fig. 4 Visualization of the main steps of our extended aberration computation process used to generate our aberration-based training datasets. First, a single ray is cast through the center of the pupil to find the retinal starting location for reverse ray tracing. An outgoing ray grid is then fit to the physical pupil and traced through the optical system to find the output coordinates. Finally, the ideal and emerging ray locations are used to calculate the local wavefront tilts, from which the resulting aberration coefficients are calculated via least-squares fitting [77]. The full algorithm is provided in Appendix 1

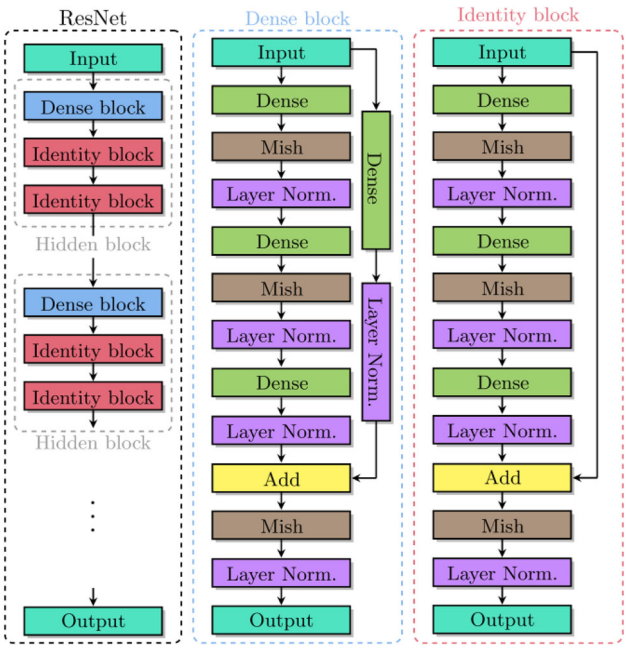


Fig. 5 Schematic of our modified ResNet architecture. Several hidden blocks are placed between the input and output layers, with each comprising fully connected (dense), Mish activation, and Layer Normalization layers. The residual blocks also contain shortcut paths for facilitating the efficient training of deep networks

To ensure that each sample eye in the training dataset is included in the highest number of different focus states possible, we sampled the focus distance separately from the other parameters. A set of samples were generated for both the focus distance and eye model parameters, using linear and uniform random sampling, respectively. To produce the final training dataset, we then obtained the feature vectors by combining the two sets of samples using a Cartesian product and utilized the focus estimation approach outlined above for computing the corresponding target (Δ_{L_D} and Δ_{A_T}) values.

5.3 Neural network structure

Due to the large number of eye parameters, we used a modified version of the residual neural network (ResNet) architecture. ResNet is a model purposefully designed for the efficient training of deep networks and has recently been adopted by Chen et al. for solving deep regression problems [79]. ResNet utilizes fully connected (dense) layers, an activation function to transform the layer outputs, and normalization layers to stabilize and speed up the training procedure. What differentiate ResNet from traditional feed-forward networks are the shortcut paths that facilitate the skipping of dense layers, providing ResNet with its outstanding deep network-training performance. An overview of our modified version of the architecture by Chen et al. is shown in Fig. 5.

The rectified linear unit (ReLU) [80] activation function is commonly used by deep neural networks and was utilized by the ResNet regressor described in [79]. However, the loss of gradient information from clamping negative inputs to zero causes the *Dying ReLU* phenomenon, which we avoided by using the Mish activation function instead [81]:

$$f(x) = x \tanh\left(\ln\left(1 + e^x\right)\right). \tag{7}$$

We only used Mish in the hidden blocks and employed a linear output in the last network layers. However, we used the tanh function in the output layer of the eye structure estimator, because of its ability to force the network outputs within the range of valid population-based eye parameters if the target columns are transformed into the range $[-1, 1]$.

Finally, instead of the Batch Normalization (BN) [82] approach utilized in [79], we used Layer Normalization (LN) [83] based on our experimental results. Furthermore, unlike the original ResNet regressor proposed in [79], we placed the normalization layers both after the activation function and before the shortcut connections, because this placement performed the best in our experiments.

5.4 Discriminator-based eye estimator training

With the highly nonlinear relationship between the eye model parameters and aberration coefficients and the possibility of inducing the same aberrations by multiple different model parametrizations, simply comparing the true and predicted eye model parameters is dissatisfactory for describing the goodness of an eye estimation. To overcome this issue, we utilized our discriminator network during the training of the eye estimator for computing the proper inputs of the loss function. Discriminators are often employed during the training of neural networks, and although training the discriminator in parallel with the network is a common practice, pretraining the discriminator has been shown to substantially improve and stabilize the training process [84, 85].

Therefore, we also pretrained our discriminator network and proceeded with the training of the eye structure estimator afterward. An overview of our process is shown in Fig. 6. During each training step of the eye estimator, we used the discriminator to compute the aberration coefficients ($\hat{Z}_{2-28}$) corresponding to the predicted eye parameters ($\hat{\rho}_{1-45}$) for the input aberration coefficients of the training sample (Z_{2-28}). We then computed the training loss by evaluating the network's loss function using Z_{2-28} and $\hat{Z}_{2-28}$ as inputs. This approach measures the goodness of the estimation using the functional performance of the eye model instead of the raw model parameters. Consequently, the predicted eye models can reproduce the input aberrations significantly better than a network trained via the naïve approach.

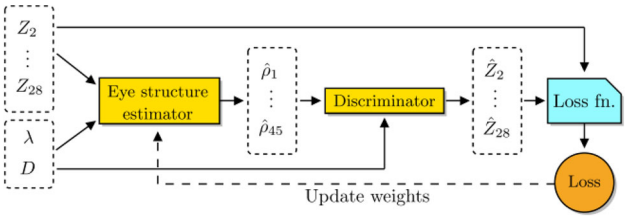


Fig. 6 Our discriminator-based approach for computing the training loss of the eye structure estimator. The discriminator is used for estimating the aberrations of the eye parameters predicted by the network. The training loss is then calculated by evaluating the loss function with the input (target) and predicted aberrations

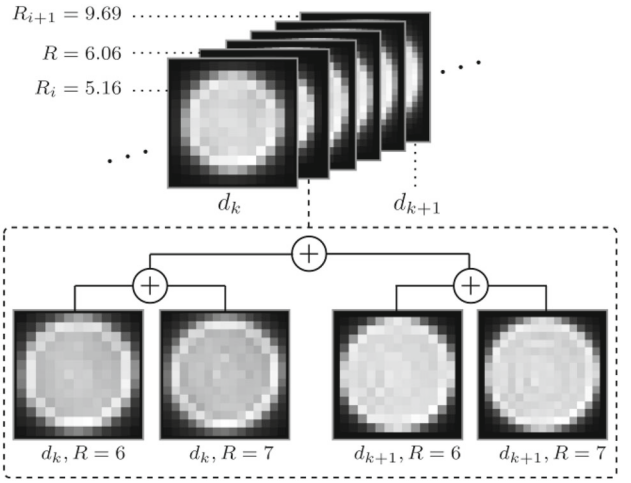


Fig. 7 Our on-axis PSF texture layout for a single PSF wavelength. Each object distance d_k in the precomputed PSF grid is assigned a depth layer. Next, transitional layers are inserted between neighboring slices, using the differences in blur sizes for determining the number of layers inserted. Each layer is computed by interpolating the corresponding PSFs of the grid along the blur size and object distance axes

6 Improved GPU-based kernel interpolation

6.1 On-axis aberrations

The texture-based PSF interpolation approach described in [49] works by storing each precomputed PSF in every possible blur radius and sampling the PSF texture twice per channel. This approach requires six texture samples to approximate a single chromatic PSF value, significantly reducing the overall throughput of the interpolation process. Furthermore, the memory requirements are also considerably high, increasing the cost sampling and making it non-trivial to extend the layout with off-axis object locations.

To overcome these issues, we used the object-space distance as the z-axis of the 3D PSF texture. This approach substantially reduces the memory footprint and facilitates the computation of a full chromatic PSF sample using only a single texture lookup. Our layout and the computation of a single depth layer are visualized in Fig. 7.

Each 2D layer of our PSF texture corresponds to a distinct object distance and is constructed by computing the interpolated blur radius and neighboring precomputed PSFs along the blur size and object distance axes for each channel. The PSFs are also aligned to the center of the texture, with empty regions filled with zeros and the texture dimensions computed from the maximum possible PSF sizes.

To sample the PSF texture for an output pixel p_o and sample pixel p_s, we calculate the 2D sample coordinates using the image-space coordinates of p_s relative to p_o and utilize it as an offset from the texture center. The object-space depth of p_s is then used to calculate the third sample coordinate based on the smallest and largest PSF object distances. Because the texture is padded with zeros, samples falling outside the PSF will naturally not contribute to the output, which is critical for properly handling the varying per-channel blur radii stemming from chromatic aberration.

6.2 Non-uniform depth sampling

The sampling of the object-space depth parameter is crucial when constructing the texture layers. A dense sampling causes excessive texture sizes, wasting GPU memory and negatively impacting performance. However, a large distance between two slices can lead to blur size differences exceeding one, producing visible artifacts. To avoid these problems, we employ non-uniform sampling, whereby we take a set of layers corresponding to the object-space depths of the PSF grid and insert a varying number of layers in-between. The number of transitional layers is computed as:

$$L_d^k = \lceil |R_{k+1} - R_k| \rceil - 1, \tag{8}$$

where d_k is the k th object distance in the precomputed PSF grid, L_d^k is the number of transitional layers inserted after the layer corresponding to d_k, and R_k is the largest blur radius (across all wavelengths) of the precomputed PSFs at d_k.

This sampling scheme guarantees that the total number of depth slices is minimal and ensures that the blur size difference between two neighboring slices never exceeds one, resulting in fast and artifact-free PSF interpolations. The layers in Fig. 7 were computed using this approach, with $L_d^k = 4$.

6.3 Off-axis aberrations

A significant benefit of our on-axis texture layout is that the reduced sample counts and memory footprint facilitate its extension with off-axis PSFs by using the on-axis texture as a block corresponding to a single horizontal and vertical angle.

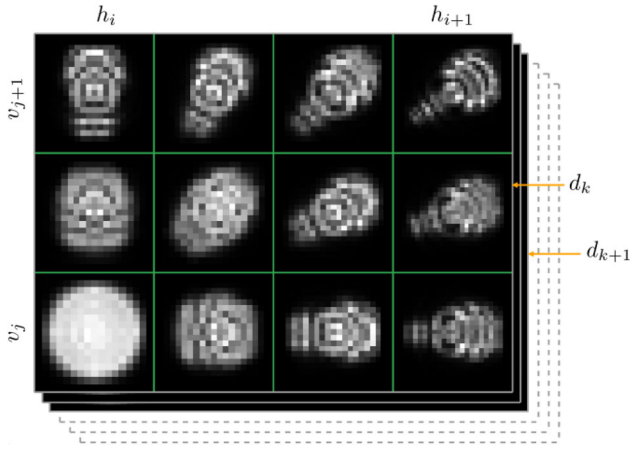


Fig. 8 Our off-axis PSF texture layout, an extension of our on-axis layout shown in Fig. 7, for a single wavelength. A layer is assigned to each unique PSF parameter combination (h_i, v_j, d_k). Transitional layers are then inserted along each axis and the neighboring precomputed PSFs are interpolated to produce the final texture values

Multiple blocks are simply laid out horizontally and vertically for multiple input directions. We start with one layer corresponding to each unique combination of precomputed incidence angles and object distance (denoted (h_i, v_j, d_k)) and place transitional layers in-between along the three axes, with the number of layers inserted determined as follows:

$$L_h^i = \max_{\substack{j=0,\ldots,N_v-1 \\ k=0,\ldots,N_d-1}} L\big(R_{i,j,k}, R_{i+1,j,k}\big), \tag{9}$$

$$L_v^j = \max_{\substack{i=0,\ldots,N_h-1 \\ k=0,\ldots,N_d-1}} L\big(R_{i,j,k}, R_{i,j+1,k}\big), \tag{10}$$

$$L_d^k = \max_{\substack{i=0,\ldots,N_h-1 \\ j=0,\ldots,N_v-1}} L\big(R_{i,j,k}, R_{i,j,k+1}\big), \tag{11}$$

$$L(R_1, R_2) = \lceil |R_1 - R_2| \rceil - 1, \tag{12}$$

where $R_{i,j,k}$ is the largest blur radius (across all wavelengths) of the precomputed PSFs at (h_i, v_j, d_k), N_h, N_v, and N_d are the number of horizontal, vertical, and depth samples in the PSF grid, and L_h^i, L_v^j, and L_d^k are, respectively, the numbers of transitional layers inserted along the horizontal, vertical, and depth axes after the layer at (h_i, v_j, d_k). This layout is visualized in Fig. 8 with $L_h^i = 2$, $L_v^j = 1$, and $L_d^k = 1$.

Aberrations of off-axis object locations are often much higher, considerably increasing the sizes of their PSFs. Combined with the increased number of texture layers, the PSF texture dimensions grow exponentially, making the total memory requirements unsuitable in many scenarios. To alleviate this issue, we apply a layer reduction term to L in (12):

$$L'(R_1, R_2) = (s \cdot L(R_1, R_2))^{-p} \cdot L(R_1, R_2), \tag{13}$$

where s and p are user-controllable parameters. Alternatively, a fixed number of transitional layers can also be used ($L'(R_1, R_2) = C$, with C a configurable parameter), but our dynamic formulation in (13) is much more robust to coarse PSF parameter sampling and large blur size differences.

To obtain an interpolated off-axis chromatic PSF sample, we determine the closest two horizontal and vertical blocks, take a sample from all four blocks using the on-axis sampling strategy described in Sect. 6.1, and produce the output using a bilinear interpolation of the four samples, based on the fractional block indices corresponding to the sample.

We note that continuously storing blocks of PSF pixels is a viable alternative, whereby each block corresponds to a single PSF pixel and the number of blocks equals the largest PSF size. Although this layout facilitates the off-axis PSF computation using a single hardware-accelerated texture sample, it displayed highly suboptimal memory access patterns in our experiments, significantly degrading the rendering performance. Furthermore, the fragment-merging step of tiled PSF splatting causes fractional image-space sample locations, which necessitates the manual sampling and interpolation of the PSF across the visual field, further reducing the overall rendering throughput. Consequently, we consider this layout unsuitable for practical applications.

6.4 Variable pupil sizes and focus distances

To facilitate the on-the-fly change of the pupil size and focus distance parameters, Csoba and Kunkli extended their PSF grid with the corresponding dimensions and sampling [49] and constructed the PSF texture in each frame using the current pupil and focus settings. Because the inclusion of the incidence angles extends the PSF parameter space by two additional dimensions, computing a single pixel of our per-frame off-axis PSF texture requires 192 samples (64 per channel) from the precomputed PSF grid, which enormously increases the cost of the per-frame PSF construction process.

To overcome this problem, during precomputation, we construct a GPU texture for each combination of pupil size and focus distance sample using the strategy outlined in Sect. 6.3. To construct the per-frame texture, we sample the four relevant neighboring textures (selected using the current pupil and focus settings of the simulation) and use bilinear interpolation to compute the PSF pixel. Finally, to keep the memory costs tractable, we exclude transitional blocks from the caches and per-frame texture when utilizing dynamic eye settings. Consequently, the precomputed and per-frame PSF textures share the same dimensions, facilitating the sampling of the caches using a single texture lookup, significantly reducing the length of per-frame PSF texture preparation.

7 Results

7.1 Estimators

We first evaluate our new neural network-based estimators. We used the Python programming language for data generation, the TensorFlow [86] programming library to construct and train our neural networks, the C++ binding of TensorFlow (with a CPU backend) to utilize the trained networks in our rendering framework, and the *patternsearch* optimizer of MATLAB [87] to implement the previous GPS-based eye estimation method of Csoba and Kunkli [49] for comparing our results against. Finally, we used an AMD Ryzen 7 1700 3.00 GHz CPU for data generation and an NVIDIA TITAN Xp graphics card for network training.

7.1.1 Training data generation

We generated 2,000,000 samples for the aberration-based datasets, which took approximately five days per on-axis dataset (discriminator and eye structure estimator) and seven days for the off-axis dataset. Regarding the focused eye estimator, we generated 40 focus distance and 25,000 eye parameter samples, resulting in a dataset of 1,000,000 samples and taking approximately six days to generate. The statistical properties of the datasets are provided in Appendix 2.

Our experiments with smaller datasets displayed a considerable decrease in network accuracies, and thus, increasing the size of the training datasets would likely be beneficial to the precision of our estimators. Our largest dataset consumes about 1.1 GB of memory; therefore, space is not a limiting factor at all. However, reducing the length of the data generation procedure would be necessary for increasing the dataset sizes while keeping the computational costs tractable.

7.1.2 Network training

Before training, we split the training datasets into training (85%) and validation sets (15%) using uniform random sampling. The feature and target columns were then standardized as $\hat{x} = (x - \overline{x})/\sigma_x$, where x and $\hat{x}$ are the original and standardized columns, and $\overline{x}$ and σ_x are the mean and standard deviation of x across all samples. However, to ensure that predicted eye parameters are valid and because of the tanh output function of the eye estimator, we normalized the target columns of the eye estimator into the range $[-1, 1]$ (as described in Sect. 5.3) as $\hat{x} = -1 + 2(x - x_{\min})/(x_{\max} - x_{\min})$, where $x_{\min}$ and $x_{\max}$ denote, respectively, the minimum and maximum values of x across all samples.

To train the neural networks, we employed the Rectified Adam optimizer [88], a modified version of the commonly used Adam algorithm [89]. We also utilized the *lookahead* mechanism [90] for improving the training speed and weight decay [91] for regularization. Each network was trained for 30 epochs using mini-batches of size 1024 and exponentially decaying learning rates and weight decay rates. For the training loss, we utilized the mean absolute error (MAE) function:

$$\mathrm{MAE}(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|, \tag{14}$$

where y_i and $\hat{y}_i$ are, respectively, the i th columns of the true and predicted output vectors. Finally, training took two and a half hours (discriminator), six hours (eye estimator), three hours (aberration estimator), and one hour (focus estimator).

7.1.3 Ablation study

We tested the following network configurations for evaluating the accuracy of our estimators and training process:

- *FFN*: A feedforward network with hidden blocks comprising a dense, BN, and ReLU layer.
- *ResNet* [79]: The ResNet regressor proposed by Chen et al., with ReLU activations, BN normalization, and the placement of the BN layers following the model in [79].
- *ResNet (ours)*: Our modified ResNet architecture proposed in Sect. 5.3, with Mish activation, LN normalization, and our modified LN layer placement.

The networks were constructed such that the numbers of network parameters are similar in each network corresponding to the same estimator, which ensures that our results demonstrate the differences stemming from the network structures.

The results, summarized in Table 3, clearly demonstrate that our modified ResNet model performed the best for all estimators. Despite the ResNet regressor by Chen et al. being more accurate than the classic feedforward approach, our modified ResNet architecture showed significant further improvements. Additionally, unlike some of the tested network configurations, all the estimators built using our proposed architecture were sufficiently accurate for our eye structure and aberration estimation purposes, demonstrating the benefits of our proposed network configuration.

In general, off-axis aberration estimation was the most challenging problem, owing largely to aberrations being significantly larger for off-axis object locations, as demonstrated by the statistical properties of the datasets in Appendix 2. The MAE differences between the discriminator and aberration estimator are consistent with this observation.

Finally, we also trained the eye structure estimator without the discriminator for evaluating our proposed discriminator-based training approach. We used the discriminator on the

Table 3 Properties related to the structure and training of our neural networks

Network	Architecture	h	n	p	Val. MAE
Eye structure estimator	FFN	19	3.5 K	221 M	0.123248
	ResNet [79]	2	3.5 K	221 M	0.021555
	ResNet (ours)	2	3.5 K	221 M	0.009016
	ResNet (ours) (no discriminator)	2	3.5 K	221 M	0.075660
Discriminator	FFN	19	2 K	72 M	0.070620
	ResNet [79]	2	2 K	72 M	0.013066
	ResNet (ours)	2	2 K	72 M	0.003137
Focused eye estimator	FFN	19	2 K	72 M	0.003099
	ResNet [79]	2	2 K	72 M	0.001538
	ResNet (ours)	2	2 K	72 M	0.000429
Aberration estimator	FFN	19	2 K	72 M	0.087222
	ResNet [79]	2	2 K	72 M	0.037234
	ResNet (ours)	2	2 K	72 M	0.015308

h and *n* denote the number of hidden blocks and neurons in each fully connected layer, and *p* is the total number of network parameters. We also provide the final mean absolute errors (MAE) produced by the trained networks on the corresponding validation datasets

trained network to obtain the MAE of the aberrations on the validation dataset. The result, included in Table 3, demonstrates that the MAE of the network trained using our proposed approach (0.009) is an order of magnitude smaller than that of the straightforward approach (0.076), verifying that our proposed loss computation method is vital for estimating the physical eye structure with sufficient accuracy.

7.1.4 Computational performance

We measured the computation times of the eye structure, focus, and aberration estimation steps using the previous and our new approach. We used several different conditions with distinct aberration profiles; emmetropia, myopia, and astigmatism for eyes with only low-order aberrations, and keratoconus, cataract, and post-LASIK surgery to evaluate arbitrary aberration compositions. For PSF parameter sampling, we used $N_\lambda = 3$ wavelengths, $N_A = 6$ pupil sizes, and $N_f = 7$ focus distances for focus estimation and on-axis aberrations, and $N_h = 19$ horizontal angles, $N_v = 11$ vertical angles, a relaxed eye, and a 5 mm pupil for off-axis aberrations.

The results, shown in Table 4, demonstrate the suitability of our approach for interactive applications, with the combined length of the computations consistently staying in the sub-second regime. The most time-consuming step was aberration estimation, resulting from the large number of network evaluations. Compared to the previous approach, our new solution was at least three orders of magnitude faster in all computations, with the largest difference displayed in eye structure estimation. The cost of the previous process is the large number of aberration computations, which is fully circumvented by our new approach, whereby the eye structure is estimated using a single neural network evaluation.

7.1.5 Estimation accuracy

Despite our main goal being to reduce computational costs, we evaluated the six eye conditions described in Sect. 7.1.4 for verifying that our approach retains an accuracy similar to the previous method and testing our method on realistic inputs. Our results, listed in Table 5, contain the mean absolute errors produced by the previous and our proposed approach and the mean absolute target values as reference.

Our proposed approach achieved very high accuracy in these realistic scenarios, with the eye estimator showing similar accuracy as the previous solution, despite the substantially shorter computation times. Furthermore, our neural network-based approach even outperformed the previous method in one scenario (cataract), demonstrating the robustness of our proposed solution. In the other tasks, the previous method formed the baseline and thus produced no errors. For our approach, the most challenging problem was estimating focus, owing largely to the nonlinearities of the eye parameter changes that comprise the network targets. However, our estimators achieved high accuracies, making our proposed approach perfectly suitable for real-world applications, especially considering the significant reduction in computational costs compared to the previous method.

Table 4 Computation times (in seconds) of the eye and aberration estimation steps using the previous and our proposed approach for various eye conditions, with the number of invocations shown in parentheses

Cond	Eye (1)		Focus (42)		On-axis (126)		Off-axis (627)	
	Prev	Prop	Prev	Prop	Prev	Prop	Prev	Prop
Emmet	3521	0.079	77.53	0.046	25.51	0.170	213.45	0.638
Myop	6270	0.088	46.17	0.056	24.10	0.134	250.39	0.512
Astig	7306	0.064	56.19	0.063	21.06	0.129	208.38	0.548
Kerat	10,642	0.061	35.62	0.051	30.61	0.159	207.81	0.643
Cata	4582	0.073	59.33	0.065	28.44	0.146	201.71	0.583
LASIK	4696	0.085	32.97	0.059	31.72	0.175	194.82	0.617

Table 5 Mean absolute target values and estimation errors produced by the previous and our new eye and aberration estimation procedures for the different eye conditions evaluated in this paper

Task		Emmet	Myop	Astig	Kerat	Cata	LASIK
Eye estim	Target	0.0000	0.2005	0.3440	0.2894	0.1655	0.0583
• Prev	Error	0.0008	0.0010	0.0011	0.0076	0.0361	0.0029
• Ours	Error	0.0044	0.0047	0.0038	0.0083	0.0048	0.0024
Focus estim	Target	0.3171	0.0411	0.0122	0.2000	0.2130	0.4124
• Prev	Error	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
• Ours	Error	0.0111	0.0058	0.0020	0.0046	0.0174	0.0033
On-axis ab	Target	0.1484	0.2137	0.3376	0.3167	0.4254	0.2506
• Prev	Error	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
• Ours	Error	0.0079	0.0093	0.0093	0.0107	0.0143	0.0086
Off-axis ab	Target	0.7013	0.7009	0.7031	0.7872	0.4156	0.9545
• Prev	Error	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
• Ours	Error	0.0152	0.0152	0.0149	0.0182	0.0208	0.0197

7.2 Rendering

In the remainder of the section, we evaluate our improved on-axis and new off-axis rendering approach. To this end, we utilized the six eye conditions described in Sect. 7.1.4 and a 50° vertical field-of-view (FOV) with a 1280 × 720 resolution. The input images were rendered using a rasterization-based pipeline and pinhole camera model, with vision simulation performed as a post-processing filter. Alternatively, other image-forming methods (such as ray tracing or RGB-D cameras) could be used to obtain the input images, provided that per-pixel color and depth information is available.

Regarding the input test scenes, we utilized a simple setup (*Primitives*) and a more realistic test scene (*San Miguel*) to evaluate our proposed solution, which correspond to different application scenarios and display different overall characteristics. Simulations for other input scenes and eye configurations are also available as supplementary material.

7.2.1 On-axis aberrations

We implemented the tiled PSF-splatting approach described in [49] using the C++ programming language and OpenGL graphics library [92]. We utilized a GPU-based implementation [76] of the ENZ approach [74] to compute the PSFs and our new neural network-based solution to obtain the aberration coefficients; we used $N_d = 33$ object distances, $N_A = 6$ aperture diameters, $N_f = 7$ focus distances, and $N_\lambda = 3$ wavelengths to construct the grid, resulting in 4,158 unique PSFs and taking approximately fifteen seconds to compute.

To compare our results against, we implemented the original PSF texture layout proposed in [49] as well as our new, significantly improved approach. We also implemented Barsky's layered algorithm [58], using $N_d = 41$ object distances, $N_\lambda = 3$ wavelengths, and the ENZ PSF model. Finally, to produce the ground-truth reference simulations, we utilized a CPU-based method that uses the dense PSF grid, whereby each input pixel was assigned to a depth-based bin and convolved with the corresponding PSF. The binning process resulted in 595 (*Primitives*) and 419 (*San Miguel*) depth bins, taking approximately eleven minutes to compute each image. The simulations are shown in Figs. 9 and 10.

We first compare the per-frame PSF texture sizes and texture computation times of the layout proposed in [49] and our improved approach. The results are summarized in Table 6. As demonstrated by the results, our depth-based texture layout significantly reduces the number of texture layers, with

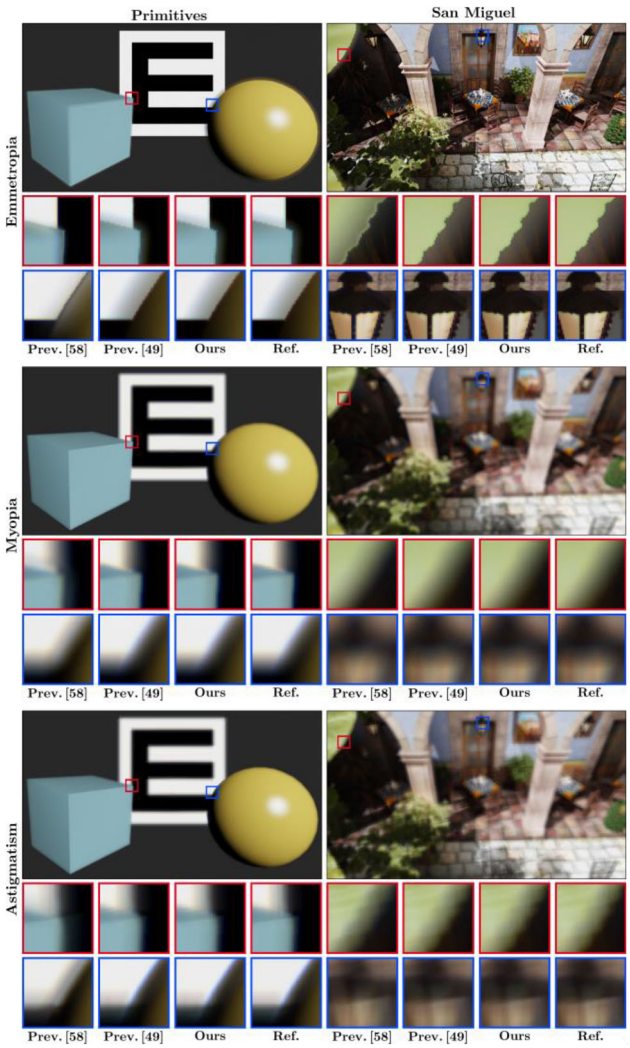


Fig. 9 On-axis simulations for the three eye conditions comprising only low-order aberrations

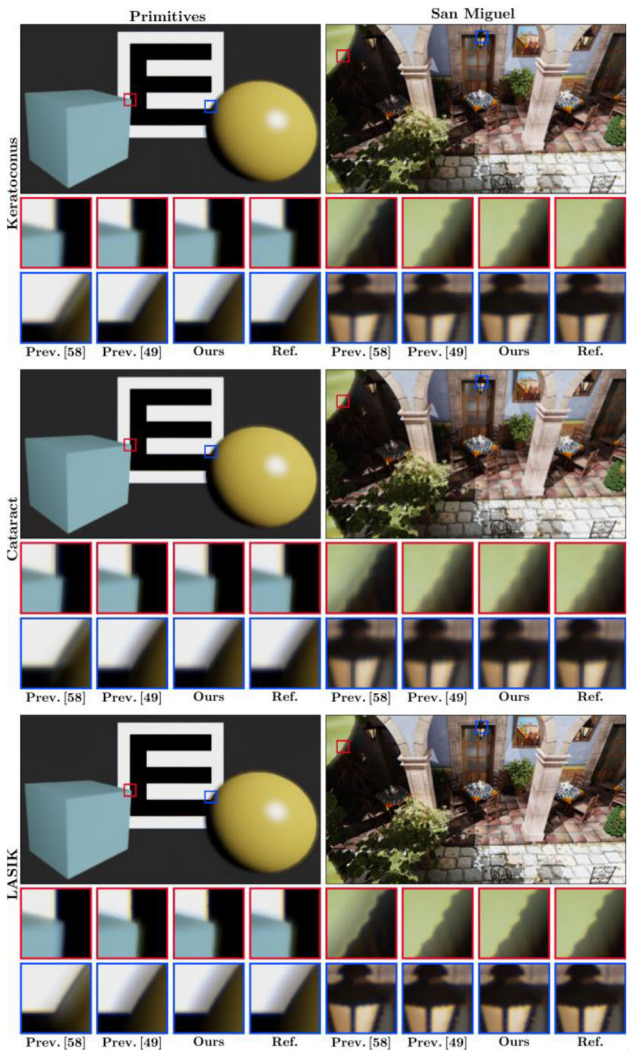


Fig. 10 On-axis simulations for the three eye conditions comprising mixed aberrations

our dynamic layer insertion strategy ensuring that the number of layers is minimal. Consequently, the memory footprint of the texture is also greatly reduced, significantly shortening the per-frame computation costs.

Next, we measured the total rendering times for Barsky's method [58] as well as the tiled approach with the previous [49] and our new PSF interpolation strategies. The results, summarized in Table 7, clearly demonstrate that our proposed method significantly outperforms both previous approaches. Since Barsky's algorithm relies on CPU-based convolution and object detection, its computation time is not suitable for interactive applications. However, our proposed solution achieves a large speedup compared to the previous tile-based approach as well, with the rendering performance more than doubled in all cases, owing mainly to the reduced memory footprint and texture sample counts. Critically, our proposed method also facilitates the simulation of arbitrary eye conditions in fully real-time environments, a task impossible with the previous approaches.

We also evaluated the accuracy of all approaches by computing the peak signal-to-noise ratio (PSNR) for all outputs using the ground-truth simulations as references. The results, summarized in Table 8, verify that our proposed approach achieves very low error levels and accurately approximates the true PSF. Additionally, our proposed method significantly outperformed Barsky's method in all scenarios, despite the lower number of depth-dependent PSFs used in the coarse grid of our approach. This behavior can be attributed mostly to the lack of proper per-pixel PSF approximation, and the handling of partial occlusion being limited to the edges of depth slices in Barsky's algorithm. Finally, our proposed texture layout caused no meaningful increase in the errors compared to the tile-based approach of Csoba and Kunkli

Table 6 Properties of the previous and our proposed PSF texture layouts

Condition	Layers		Memory		Interpolation	
	Prev	Prop	Prev	Prop	Prev	Prop
Emmetropia	1419	65	39.11 MB	1.79 MB	0.63 ms	0.26 ms
Myopia	957	65	11.86 MB	0.81 MB	0.26 ms	0.21 ms
Astigmatism	891	65	9.55 MB	0.70 MB	0.26 ms	0.17 ms
Keratoconus	1320	65	31.43 MB	1.55 MB	0.45 ms	0.24 ms
Cataract	957	65	11.86 MB	0.81 MB	0.38 ms	0.23 ms
LASIK	1221	65	24.82 MB	1.32 MB	0.49 ms	0.19 ms

Table 7 Total rendering times of the two previous approaches and our proposed on-axis aberration simulation method

Cond	Primitives			San Miguel		
	Barsky [58]	Csoba and Kunkli [49]	Ours	Barsky [50]	Csoba and Kunkli [49]	Ours
Emmet	21.93 s	24.68 ms	10.57 ms	13.41 s	16.13 ms	7.43 ms
Myop	12.33 s	21.73 ms	9.53 ms	13.51 s	23.06 ms	8.66 ms
Astig	11.98 s	26.51 ms	7.51 ms	13.29 s	30.62 ms	10.70 ms
Kerat	17.10 s	27.08 ms	12.93 ms	14.62 s	24.66 ms	9.87 ms
Cata	15.53 s	24.50 ms	9.38 ms	15.22 s	17.09 ms	6.81 ms
LASIK	16.46 s	21.29 ms	8.67 ms	11.86 s	20.60 ms	8.73 ms

Table 8 PSNRs for the simulations generated using the two previous and our proposed on-axis approaches

Cond	Primitives			San Miguel		
	Barsky [58]	Csoba and Kunkli [49]	Ours	Barsky [58]	Csoba and Kunkli [49]	Ours
Emmet	31.92 dB	46.52 dB	46.43 dB	33.70 dB	46.29 dB	46.14 dB
Myop	33.39 dB	46.80 dB	46.77 dB	33.52 dB	50.75 dB	50.60 dB
Astig	33.61 dB	47.07 dB	47.09 dB	33.89 dB	49.35 dB	49.33 dB
Kerat	33.64 dB	45.74 dB	45.77 dB	32.61 dB	48.91 dB	48.83 dB
Cata	31.53 dB	45.55 dB	45.53 dB	30.81 dB	45.51 dB	45.45 dB
LASIK	31.07 dB	45.76 dB	45.73 dB	30.89 dB	48.10 dB	47.95 dB

either, despite the substantial reduction in memory requirements and computational costs. Our new method essentially works by discarding all unused data from the PSF texture, leading to no significant loss of information.

Finally, we also highlighted a few areas with the largest deviations from the references in Figs. 9 and 10 for all outputs. As can be seen, our proposed approach approximates the ground-truth reference simulations with high accuracy, producing almost identical outputs to the ground-truth references. Although the previous method by Csoba and Kunkli is capable of the same accuracy, our new method achieves these results with significantly less memory and computation costs. The limitations of Barsky's approach are also apparent, as the inaccurate convolution kernels and the lack of per-pixel treatment of partial occlusion led to visible deviations from the references.

7.2.2 Off-axis aberrations

We extended our implementation of the tiled PSF-splatting approach described in Sect. 7.2.1. with our proposed off-axis PSF layout. Due to the significantly larger amount of PSF data, we considered two distinct use cases:

- *quality* mode: The PSF grid was sampled using $N_h = 31$, $N_v = 21$, $N_d = 9$, $N_A = 1$, $N_f = 1$, and $N_\lambda = 3$ samples, resulting in 17,577 total PSFs and taking ca. two minutes to compute. Transitional layers were utilized in the per-frame PSF texture using (13), with $s = 1$ and $p = 1/2$ chosen empirically, providing a good balance between quality and texture memory.
- *dynamic* mode: The PSF grid was sampled with $N_h = 23$, $N_v = 13$, $N_d = 9$, $N_A = 5$, $N_f = 5$, and $N_\lambda = 3$

samples, resulting in 201,825 total PSFs and taking ca. twenty minutes to compute. No transitional layers were used in the per-frame PSF texture.

To compare our results against, we implemented the algorithm of Rodríguez Celaya et al. [26]. We constructed a PSF grid using the ENZ PSF model with $N_h = 5$, $N_v = 3$, $N_d = 2$, and $N_\lambda = 3$ samples (covering the near and far plane of the input 50° vertical FOV) and performed convolution with per-pixel PSF interpolation to obtain the output. We also implemented the layered algorithm by Gonzalez-Utrera [27] with $N_h = 11$, $N_v = 11$, $N_d = 3$, and $N_\lambda = 3$ PSF grid samples. The input images were split into three depth-based layers, which were convolved using a per-pixel interpolation of the off-axis PSFs and composited to obtain the outputs. Finally, we utilized a CPU-based method using the dense PSF grid for producing the ground-truth reference simulations. Each input pixel was assigned to a unique bin (based on depth and incidence angles) and convolved with the corresponding PSF. The binning process resulted in 194,243 (*Primitives*) and 127,548 (*San Miguel*) PSF bins, with the total computation taking several hours per image. The simulations are shown in Figs. 11 and 12.

First, we evaluate the properties of the per-frame PSF texture, which are summarized in Table 9. The only memory cost of the *quality* setup is the per-frame PSF texture, which is smaller in the *dynamic* setup, owing to the smaller PSF grid and omission of transitional layers. The *dynamic* setup also requires several cached PSF textures, the main bottleneck of the setup. However, it also represents the worst-case scenario of memory requirements, demonstrating the suitability of our approach for commodity graphics hardware. Moreover, the PSF grid sampling can also be reduced for a smaller memory footprint of the cache. Finally, because the aperture and focus parameters are static, the *quality* setup also facilitates the skipping of the per-frame texture-building step, which is an extra cost in the *dynamic* setup.

Next, we measured the total rendering times of the previous algorithms and the two setups of our proposed approach. The results, summarized in Table 10, demonstrate that both existing approaches are highly inadequate for interactive environments, with the computations taking at least one hour per image. The computation times also vary highly with the input eye condition. Our new approach, however, displays performance characteristics suitable even for real-time applications, with a very small difference between the two setups. Furthermore, the rendering times are also much more consistent across all eye conditions, with the main differentiating factor proving to be the input scene. Overall, *Primitives* appeared to be more challenging, owing largely to the higher number of memory transactions needed to access the per-frame PSF texture, as demonstrated by the higher number of depth bins generated by the ground-truth algorithm.

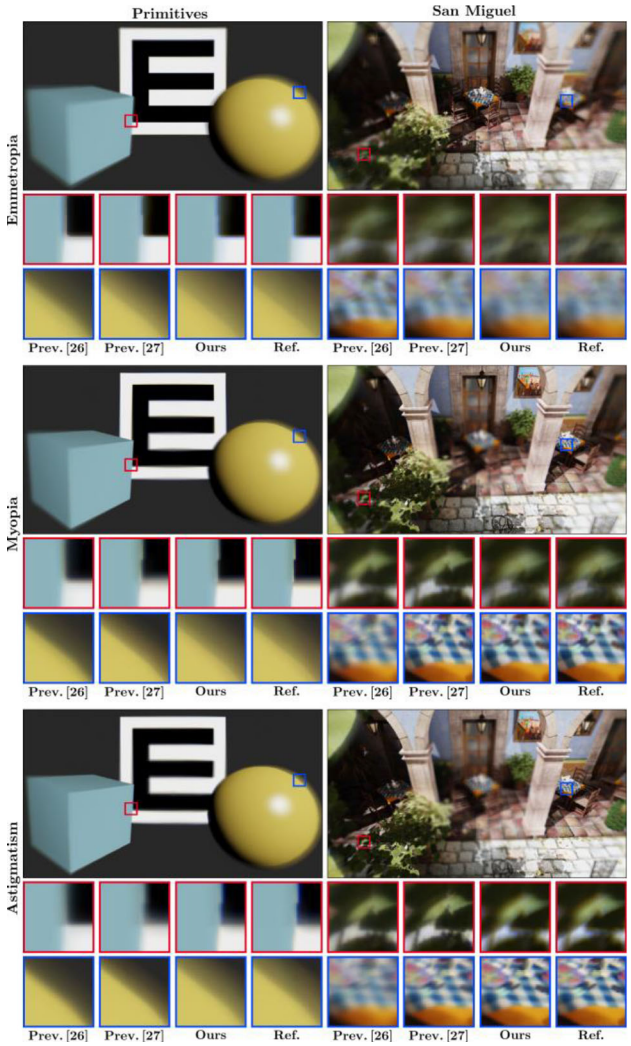


Fig. 11 Off-axis simulations for the three eye conditions comprising only low-order aberrations

Critically, however, our proposed method also facilitates the interactive change of eye settings, which was impossible with the previous approaches.

We also evaluated the accuracy of all methods using the ground-truth reference images, with the resulting PSNRs summarized in Table 10. The results for the outputs generated using our approach indicate perfect suitability for real-world applications, with the *quality* setup consistently outperforming the *dynamic* setup due to the significantly more information in the per-frame PSF texture. However, compared to the previous methods, our proposed approach achieved significantly higher accuracy, owing mostly to the ability of our new method to utilize substantially higher amounts of PSF information, properly handle per-pixel partial occlusion, and much more accurately approximate the true, per-pixel PSF of the simulated eye.

A few areas with the largest mismatches between the reference images and our simulations are also highlighted in

Fig. 12 Off-axis simulations for the three eye conditions comprising mixed aberrations

Figs. 11 and 12, using the *quality* setup of our approach. These regions clearly demonstrate that the previous methods are unable to faithfully simulate the true blurriness levels and PSF shapes of the simulated eye, with the object boundaries also highly morphed in many regions. However, both configurations of our new approach closely approximate the ground-truth simulations and avoid all these artifacts. The varying sharpness over the visual field is also faithfully replicated by our approach, which is critical to properly evaluate the performance of the simulated eye and study the effects of conditions like *relative peripheral hyperopia* (which was outlined in Sect. 1). The previous techniques are unable to properly handle such conditions and give a false representation of the true optical performance of the simulated eyes.

Finally, rendered videos using our approach are also available as supplementary material, which demonstrate the stability and lack of interpolation artifacts under motion, and provide further examples for the outputs of our method.

8 Conclusion

In this paper, we presented an efficient solution for simulating the central and peripheral visual aberrations of the human eye. We first proposed a novel, neural network-based method for estimating the physical structure and unknown aberration coefficients of the simulated eye. Our new method is substantially faster than the previous approach and operates in the sub-second regime, facilitating the fully interactive exploration of visual aberrations. We also described an improved PSF interpolation strategy for a tiled PSF-splatting approach for efficiently computing the per-pixel PSFs and extended it for peripheral vision as well. We demonstrated that our proposed solution significantly outperforms previous convolution-based approaches both in terms of rendering speed and memory requirements, without sacrificing rendering quality. Critically, our new solution also facilitates the simulation of off-axis aberrations with high accuracy and an approximately real-time performance profile, which substantially increases the plausibility of the resulting vision simulations and was previously impossible with such low computational costs.

Regarding future research directions, other neural network architectures and dataset generation strategies could lead to further improvements in accuracy and computation

Table 9 Per-frame PSF texture properties for the two off-axis configurations and various eye conditions evaluated in this paper

Condition	Quality		Dynamic		
	Layers	Memory	Cache	Texture	Interpolation
Emmetropia	51 × 27 × 17	1.54 GB	5.71 GB	234.06 MB	1.94 ms
Myopia	50 × 26 × 17	1.13 GB	4.70 GB	192.67 MB	1.90 ms
Astigmatism	50 × 26 × 17	1.13 GB	4.43 GB	181.58 MB	1.93 ms
Keratoconus	55 × 27 × 17	2.14 GB	7.68 GB	314.38 MB	2.43 ms
Cataract	37 × 30 × 17	0.46 GB	3.09 GB	85.01 MB	1.36 ms
LASIK	56 × 32 × 17	2.32 GB	6.50 GB	266.08 MB	2.12 ms

Table 10 Total rendering times and PSNRs of the two previous approaches and the two setups of our proposed off-axis aberration simulation method

Cond	Primitives						San Miguel					
	Rodríguez Celaya et al. [25]		Gonzalez-Utrera [26]		Ours (Quality)	Ours (Dynamic)	Rodríguez Celaya et al. [25]		Gonzalez-Utrera [26]		Ours (Quality)	Ours (Dynamic)
Emmet	1.7 h	31.4 dB	4.7 h	33.1 dB	20.9 ms	45.6 dB	22.8 ms	45.0 dB	1.7 h	30.3 dB	4.6 h	31.3 dB	14.3 ms	46.4 dB	14.5 ms	45.1 dB
Myop	1.7 h	31.1 dB	3.5 h	32.6 dB	14.7 ms	45.1 dB	16.2 ms	43.9 dB	1.7 h	30.3 dB	3.6 h	30.8 dB	11.7 ms	43.8 dB	11.9 ms	42.0 dB
Astig	1.8 h	29.7 dB	3.7 h	31.8 dB	13.5 ms	45.4 dB	14.9 ms	44.5 dB	1.9 h	29.7 dB	3.6 h	31.5 dB	11.8 ms	43.9 dB	11.7 ms	42.0 dB
Kerat	1.8 h	32.1 dB	5.9 h	36.2 dB	19.6 ms	44.3 dB	20.1 ms	43.6 dB	1.8 h	29.8 dB	5.7 h	32.2 dB	14.3 ms	44.2 dB	14.0 ms	42.2 dB
Cata	1.0 h	33.8 dB	1.8 h	35.9 dB	13.4 ms	44.8 dB	12.8 ms	44.4 dB	1.0 h	32.0 dB	1.8 h	34.5 dB	10.7 ms	44.8 dB	11.3 ms	44.2 dB
LASIK	1.6 h	31.2 dB	4.8 h	36.3 dB	24.4 ms	44.3 dB	24.9 ms	43.2 dB	1.6 h	29.2 dB	5.7 h	32.6 dB	20.6 ms	44.9 dB	19.1 ms	42.2 dB

times, and the recent advancements in hardware-accelerated ray tracing could facilitate the efficient generation of much larger datasets. Extending the eye estimator with multiple input aberration measurements (such as off-axis aberrations) could be beneficial as well. Experimenting with different eye elements (such as gradient-index lenses and intraocular lens implants) is straightforward and could further increase the applicability of our model, which our rendering method naturally supports. Regarding rendering, reducing the GPU memory usage of the precomputed off-axis PSF cache would make it easier to support dynamic pupil and focus changes. Furthermore, our approach could be adapted for other optical systems to provide even more application areas. Finally, experimenting with neural network-based solutions for the full simulation of aberrated vision could be a promising future research direction, with the DeepFocus method of Xiao et al. [35] being a great starting point.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s00371-023-03060-0.

Acknowledgements We gratefully acknowledge the support of NVIDIA Corporation with the donation of the TITAN Xp GPU used for this research.

Funding Open access funding provided by University of Debrecen. The authors have no relevant financial or non-financial interests to disclose.

Data availability The datasets generated during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Appendix 1: Our extended approach for computing the aberration coefficients using reverse ray tracing

The aberration estimation process plays a critical role in the training dataset generation step, and we made some important

changes to the process proposed in [49]. Therefore, in Algorithm 1, we provide an in-depth description of our extended algorithm for computing the aberration coefficients of our aberration-based datasets, which contains all the modifications that we outlined in Sect. 5.2.1.

Algorithm 1: Our modified method for computing the wavefront aberration coefficients using reverse ray tracing and least-squares fitting

Input: light wavelength λ, object distance d, incidence angles h, v, ray grid sizes n_{p} and n_Z, pupil fit threshold τ_c, grid contraction factors γ_c, γ_Z

Output: Zernike aberration coefficients Z_{2-28}

1. $\boldsymbol{g}_p^e \leftarrow$ Construct a parallel grid of n_{p} number of rays (of wavelength λ) that covers the entire anterior cornea, each starting outside the eye, using d, h, v to determine their starting locations and directions
2. $\boldsymbol{g}_p^p \leftarrow$ Trace $\boldsymbol{g}_p^e$ through the eye until it intersects the pupil plane
3. $\boldsymbol{r}_p^c \leftarrow$ Determine the ray closest to the pupil center in $\boldsymbol{g}_p^p$
4. If the distance of $\boldsymbol{r}_p^c$ from the pupil center $> \tau_c$: $\boldsymbol{g}_p^e \leftarrow$ construct a new grid of n_{p} rays outside the eye, centered on the starting location of $\boldsymbol{r}_p^c$ in $\boldsymbol{g}_p^e$, with the distance between neighboring rays reduced by a factor of γ_c with respect to the previous grid; go back to Step 2
5. $\boldsymbol{g}_p^r \leftarrow$ Trace $\boldsymbol{g}_p^p$ through the crystalline lens to the retina
6. $\boldsymbol{r}_r^c \leftarrow$ Determine the ray on the retina corresponding to $\boldsymbol{r}_p^c$
7. $\boldsymbol{g}_Z^r \leftarrow$ Construct a grid of n_Z number of rays (of wavelength λ) using spherical coordinates, each starting at $\boldsymbol{r}_r^c$ and going toward the posterior lens surface, with the grid covering the entire posterior lens and the ray in the center going in the opposite direction as $\boldsymbol{r}_r^c$
8. $\boldsymbol{g}_Z^p \leftarrow$ Trace $\boldsymbol{g}_Z^r$ backwards until it intersects the pupil plane
9. If there is a ray in $\boldsymbol{g}_Z^p$ that falls outside the pupil hole: $\boldsymbol{g}_Z^{\mathrm{p}} \leftarrow$ construct a new grid of n_Z rays similar to Step 7, but reduce the distance in spherical coordinates between neighboring rays by a factor of γ_Z; go back to Step 8
10. $\boldsymbol{r}_Z^p \leftarrow$ Determine the ray closest to the pupil center in $\boldsymbol{g}_Z^p$
11. $\boldsymbol{g}_Z^c \leftarrow$ Trace $\boldsymbol{g}_Z^p$ through the cornea to the anterior cornea surface
12. $p_e \leftarrow$ Construct a plane outside the eye that is perpendicular to the plane of $\boldsymbol{g}_p^e$
13. $\boldsymbol{g}_Z^e \leftarrow$ Trace $\boldsymbol{g}_Z^c$ to p_e using their current directions to obtain the 2D spots of the emerging wavefront
14. $\boldsymbol{g}_Z^r \leftarrow$ Project the rays in $\boldsymbol{g}_Z^c$ to p_e parallel to the normal of p_e to obtain the 2D spots of the reference wavefront
15. $\boldsymbol{r}_Z^e \leftarrow$ Determine the 2D location corresponding to $\boldsymbol{r}_Z^p$ in $\boldsymbol{g}_Z^e$
16. $\boldsymbol{g}_Z^e, \boldsymbol{g}_Z^r \leftarrow$ Offset $\boldsymbol{g}_Z^e, \boldsymbol{g}_Z^r$ by $-\boldsymbol{r}_Z^e$ such that the grid is centered on the projection of the pupil
17. $\boldsymbol{r}_Z^c \leftarrow$ Determine the radius of the circle circumscribed around $\boldsymbol{g}_Z^e$
18. $\boldsymbol{g}_Z^e, \boldsymbol{g}_Z^r \leftarrow$ Normalize the 2D locations of $\boldsymbol{g}_Z^e, \boldsymbol{g}_Z^r$ by $\boldsymbol{r}_Z^c$
19. $\boldsymbol{s}_Z^x, \boldsymbol{s}_Z^y \leftarrow$ Determine the local wavefront tilts along the x- and y-axis from $\boldsymbol{g}_Z^e, \boldsymbol{g}_Z^r$
20. $Z_{2-28} \leftarrow$ Use the least-squares fitting approach described in [77] to compute the resulting Zernike coefficients from $\boldsymbol{g}_Z^e, \boldsymbol{s}_Z^x, \boldsymbol{s}_Z^y$ and the partial derivatives of the Zernike polynomials

Appendix 2: Statistical properties of our synthetic datasets

Table 11 provides a summary of the statistical properties of the synthetic datasets generated for the training of our neural networks. We discussed the input domains and sampling strategies for obtaining the input vectors in Sect. 5.2; therefore, we only consider the corresponding computed values in this section. We also excluded the eye estimator dataset,

Table 11 Statistical properties of groups of target vectors computed over the full training datasets of the discriminator, aberration estimator, and focus estimator

	Discriminator			Aberration estimator			Focus estimator	
	$\boldsymbol{Z}_{2-6}$ ($n = 1, 2$)	$\boldsymbol{Z}_{7-15}$ ($n = 3, 4$)	$\boldsymbol{Z}_{16-28}$ ($n = 5, 6$)	$\boldsymbol{Z}_{2-6}$ ($n = 1, 2$)	$\boldsymbol{Z}_{7-15}$ ($n = 3, 4$)	$\boldsymbol{Z}_{16-28}$ ($n = 5, 6$)	$\boldsymbol{\Delta}_{L_D}$	$\boldsymbol{\Delta}_{A_T}$
Min	− 41.236	− 4.601	− 0.6095	− 114.723	− 42.732	− 8.2681	− 0.878	− 0.572
Max	26.862	4.744	0.6919	103.282	39.295	8.0194	0.000	0.000
Mean	− 0.572	− 0.018	0.0003	− 0.336	0.010	0.0023	− 0.622	− 0.198
Abs. mean	2.904	0.379	0.0200	4.231	0.701	0.0556	0.622	0.198
Std. dev.	4.138	0.568	0.0368	6.689	1.389	0.1192	0.331	0.125
Q_1	− 2.373	− 0.245	− 0.0085	− 2.696	− 0.253	− 0.0150	− 0.859	− 0.280
Q_2	− 0.223	− 0.007	0.0001	− 0.106	0.001	0.0003	− 0.839	− 0.200
Q_3	1.626	0.212	0.0089	2.242	0.268	0.0187	− 0.380	− 0.125

n is the radial order of the Zernike coefficients

because the training dataset of the discriminator was generated with the exact same input domain (only the random number generator seed differed) and thus displayed very similar statistical properties. Finally, we also grouped the aberration coefficients into three groups based on the radial order n ($n = 1,\ 2$, $n = 3,\ 4$, and $n = 5,\ 6$), because the magnitudes of aberration coefficients behaved similarly in these groups.

See Table 11.

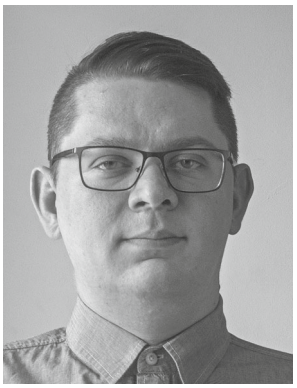
References

1. Li, T., Li, C., Zhang, X., Liang, W., Chen, Y., Ye, Y., Lin, H.: Augmented reality in ophthalmology: applications and challenges. Front. Med. **8**, 733241:1–733241:12 (2021). https://doi.org/10.3389/fmed.2021.733241
2. Aydindoğan, G., Kavakli, K., Şahin, A., Artal, P., Ürey, H.: Applications of augmented reality in ophthalmology [invited]. Biomed. Opt. Express **12**(1), 511–538 (2021). https://doi.org/10.1364/BOE.405026
3. Ong, C.W., Tan, M.C.J., Lam, M., Koh, V.T.C.: Applications of extended reality in ophthalmology: systematic review. J. Med. Internet Res. **23**(8), e24152:1-e24152:18 (2021). https://doi.org/10.2196/24152
4. Iskander, M., Ogunsola, T., Ramachandran, R., McGowan, R., Al-Aswad, L.A.: Virtual reality and augmented reality in ophthalmology: a contemporary prospective. Asia-Pac. J. Ophthalmol. **10**(3), 244–252 (2021). https://doi.org/10.1097/APO.0000000000000409
5. Watson, A.B., Ahumada, A.J., Jr.: Predicting visual acuity from wavefront aberrations. J. Vis. **8**(4), 17:1-171:9 (2008). https://doi.org/10.1167/8.4.17
6. Fülep, C., Kovács, I., Kránitz, K., Erdei, G.: Simulation of visual acuity by personalizable neuro-physiological model of the human eye. Sci. Rep. **9**(1), 7805:1–7805:15 (2019). https://doi.org/10.1038/s41598-019-44160-z
7. Greivenkamp, J.E., Schwiegerling, J., Miller, J.M., Mellinger, M.D.: Visual acuity modeling using optical raytracing of schematic eyes. Am. J. Ophtalmol. **120**(2), 227–240 (1995). https://doi.org/10.1016/S0002-9394(14)72611-X
8. Kordek, D., Young, L.K., Kremláček, J.: Comparison between optical and digital blur using near visual acuity. Sci. Rep. **11**(1), 3437:1-3437:8 (2021). https://doi.org/10.1038/s41598-021-82965-z
9. Zaman, N.: EyeSightVR: An immersive and automated tool for comprehensive assessment of visual function. Master's thesis, University of Nevada, Reno (2021)
10. Cheng, X., Bradley, A., Thibos, L.N.: Predicting subjective judgment of best focus with objective image quality metrics. J. Vis. **4**(4), 310–321 (2004). https://doi.org/10.1167/4.4.7
11. Tabernero, J., Benito, A., Alcón, E., Artal, P.: Mechanism of compensation of aberrations in the human eye. J. Opt. Soc. Am. A **24**(10), 3274–3283 (2007). https://doi.org/10.1364/JOSAA.24.003274
12. Tabernero, J., Berrio, E., Artal, P.: Modeling the mechanism of compensation of aberrations in the human eye for accommodation and aging. J. Opt. Soc. Am. A **28**(9), 1889–1895 (2011). https://doi.org/10.1364/JOSAA.28.001889
13. Krösl, K., Bauer, D., Schwärzler, M., Fuchs, H., Suter, G., Wimmer, M.: A VR-based user study on the effects of vision impairments on recognition distances of escape-route signs in buildings. Vis. Comput. **34**(6–8), 911–923 (2018). https://doi.org/10.1007/s00371-018-1517-7
14. Krueger, M.L., Oliveira, M.M., Kronbauer, A.L.: Personalized visual simulation and objective validation of low-order aberrations of the human eye. In: Guerrero, J.E., (ed) 2016 29th SIBGRAPI Conference on Graphics, Patterns and Images. SIBGRAPI'16, pp. 64–71. IEEE Computer Society, Los Alamitos, CA, USA (2016). https://doi.org/10.1109/SIBGRAPI.2016.018
15. Kanazawa, K., Nakano, Y., Moriya, T., Takahashi, T.: Visual appearance simulation method for exhibited objects considering viewers' eyesight and lateral inhibition. J. Inst. Image Electron. Eng. Jpn. **40**(1), 151–158 (2011). https://doi.org/10.11371/iieej.40.151
16. Xiong, Y.-Z., Lei, Q., Calabrèse, A., Legge, G.E.: Simulating visibility and reading performance in low vision. Front. Neurosci. **15**, 671121:1-671121:13 (2021). https://doi.org/10.3389/fnins.2021.671121
17. Jin, B., Ai, Z., Rasmussen, M.: Simulation of eye disease in virtual reality. In: Zhang, Y.T. (ed) Proceedings of the 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference. EMBC'05, pp. 5128–5131. IEEE Computer Society, Los Alamitos, CA, USA (2005). https://doi.org/10.1109/IEMBS.2005.1615631
18. Krösl, K., Elvezio, C., Hürbe, M., Karst, S., Wimmer, M., Feiner, S.: ICthroughVR: illuminating cataracts through virtual reality. In: Teather, R., Itoh, Y., Gabbard, J., Argelaguet, F., Olivier, A.-H., Keefe, D.F. (eds) 2019 IEEE Conference on Virtual Reality and 3D User Interfaces. VR 2019, pp. 655–663. IEEE Computer Society, Los Alamitos, CA, USA (2019). https://doi.org/10.1109/VR.2019.8798239
19. Krösl, K., Elvezio, C., Hürbe, M., Karst, S., Feiner, S., Wimmer, M.: XREye: simulating visual impairments in eye-tracked XR. In: O'Conner, L. (ed) 2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops. VRW 2020, pp. 830–831. IEEE Computer Society, Los Alamitos, CA, USA (2020). https://doi.org/10.1109/VRW50115.2020.00266
20. Krösl, K., Elvezio, C., Luidolt, L.R., Hürbe, M., Karst, S., Feiner, S., Wimmer, M.: CatARact: simulating cataracts in augmented reality. In: O'Conner L (ed) Proceedings of the 2020 IEEE International Symposium on Mixed and Augmented Reality. ISMAR 2020, pp. 682–693. IEEE Computer Society, Los Alamitos, CA, USA (2020). https://doi.org/10.1109/ISMAR50242.2020.00098
21. Ateş, H.C., Fiannaca, A., Folmer, E.: Immersive simulation of visual impairments using a wearable see-through display. In: Verplank, B., Ju, W., Antle, A.N, Mazalek, A., Mueller, F. (eds) Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction. TEI'15, pp. 225–228. Association for Computing Machinery, New York, NY, USA (2015). https://doi.org/10.1145/2677199.2680551
22. Tabernero, J., Piers, P., Benito, A., Redondo, M., Artal, P.: Predicting the optical performance of eyes implanted with IOLs to correct spherical aberration. Investig. Ophthalmol. Vis. Sci. **47**(10), 4651–4658 (2006). https://doi.org/10.1167/iovs.06-0444
23. Wang, W.: Intelligent planning for refractive surgeries: a modelling and visualisation-based approach. Ph.D. thesis, University of Liverpool (2020). https://doi.org/10.17638/03090577
24. Loos, J., Slusallek, P., Seidel, H.-P.: Using wavefront tracing for the visualization and optimization of progressive lenses. Comput. Graph. Forum **17**(3), 255–265 (2001). https://doi.org/10.1111/1467-8659.00272
25. Nießner, M., Sturm, R., Greiner, G.: Real-time simulation and visualization of human vision through eyeglasses on the GPU. In: Spencer, S.N. (ed) Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry. VRCAI '12, pp. 195–202. Association for Computing Machinery, New York, NY, USA (2012). https://doi.org/10.1145/2407516.2407565
26. Rodríguez Celaya, J.A., Brunet Crosa, P., Ezquerra, N., Palomar, J.E.: A virtual reality approach to progressive lenses simulation. In:

Isern, J.R., Perandrés, D.M. (eds.) Actas del XV Congreso Español de Informática Gráfica, pp. 43–52. Thomson-Paraninfo, Madrid (2005)
27. Gonzalez-Utrera, D.: Metrology and simulation with progressive addition lenses. Ph.D. thesis, The University of Arizona (2018)
28. Keleş, O., Anarim, E.: Adjustment of digital screens to compensate the eye refractive errors via deconvolution. In: Goularas, D., Maaref, H. (eds) 2019 Ninth International Conference on Image Processing Theory, Tools and Applications. IPTA 2019, pp. 1–6. IEEE Computer Society, Los Alamitos, CA, USA (2019). https://doi.org/10.1109/IPTA.2019.8936098
29. Zhao, J., Liu, L., Zhang, J., Wang, T.: Contrast enhancement of images on retina by adjusting deconvolved images. In: Zhu, C., Jiang, X., Dong, J. (eds) 2021 6th International Conference on Image, Vision and Computing. ICIVC 2021, pp. 202–208. IEEE Computer Society, Los Alamitos, CA, USA (2021). https://doi.org/10.1109/ICIVC52351.2021.9526982
30. Barsky, B.A., Huang, F.-C., Lanman, D., Wetzstein, G., Raskar, R.: Vision correcting displays based on inverse blurring and aberration compensation. In: Agapito, L., Bronstein, M.M., Rother, C. (eds.) Computer Vision - ECCV 2014 Workshops. ECCV 2014, Lecture Notes in Computer Science, vol 8927, pp. 524–538. Springer, Berlin, Heidelberg (2015)
31. Huang, F.-C., Wetzstein, G., Barsky, B.A., Raskar, R.: Eyeglasses-free display: towards correcting visual aberrations with computational light field displays. ACM Trans. Graph. **33**(4), 59:1–59:12 (2014). https://doi.org/10.1145/2601097.2601122
32. Zhu, Z., Toyoura, M., Go, K., Fujishiro, I., Kashiwagi, K., Mao, X.: Processing images for red–green dichromats compensation via naturalness and information-preservation considered recoloring. Vis. Comput. **35**(6–8), 1053–1066 (2019). https://doi.org/10.1007/s00371-019-01689-4
33. Itoh, Y., Klinker, G.: Vision enhancement: defocus correction via optical see-through head-mounted displays. In: Nanayakkara, S., Do, E.Y.-L., Rekimoto, J., Huber, J., Chen, B.-Y. (eds) Proceedings of the 6th Augmented Human International Conference. AH'15, pp. 1–8. Association for Computing Machinery, New York, NY, USA (2015). https://doi.org/10.1145/2735711.2735787
34. Yamamoto, K., Suzuki, I., Namikawa, K., Sato, K., Ochiai, Y.: Interactive eye aberration correction for holographic near-eye display. In: Häkkilä, J., Lopes, P., Kosch, T., Nishida, J., Strohmeier, P., Abdelrahman, Y. (eds) Proceedings AHs'21. Augmented Humans Conference 2021, pp. 204–214. Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3458709.3458955
35. Xiao, L., Kaplanyan, A., Fix, A., Chapman, M., Lanman, D.: DeepFocus: learned image synthesis for computational displays. ACM Trans. Graph. **37**(6), 200:1-200:13 (2018). https://doi.org/10.1145/3272127.3275032
36. Duchowski, A.T., House, D.H., Gestring, J., Wang, R.I., Krejtz, K., Krejtz, I., Mantiuk, R., Bazyluk, B.: Reducing visual discomfort of 3D stereoscopic displays with gaze-contingent depth-of-field. In: Spencer, S.N. (ed) Proceedings of the ACM Symposium on Applied Perception. SAP'14, pp. 39–46. Association for Computing Machinery, New York, NY, USA (2014). https://doi.org/10.1145/2628257.2628259
37. Mantiuk, R., Bazyluk, B., Tomaszewska, A.: Gaze-dependent depth-of-field effect rendering in virtual environments. In: Ma, M., Oliveira, M.F., Pereira, J.M. (eds) Proceedings of the Second International Conference on Serious Games Development and Applications. SGDA 2011, Lecture Notes in Computer Science, vol 6944, pp. 1–12. Springer, Berlin, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23834-5_1
38. Rokita, P.: Generating depth-of-field effects in virtual reality applications. IEEE Comput. Graph. Appl. **16**(2), 18–21 (1996). https://doi.org/10.1109/38.486676
39. Cholewiak, S.A., Love, G.D., Srinivasan, P.P., Ng, R., Banks, M.S.: ChromaBlur: rendering chromatic eye aberration improves accommodation and realism. ACM Trans. Graph. **36**(6), 210:1-210:12 (2017). https://doi.org/10.1145/3130800.3130815
40. Cholewiak, S.A., Love, G.D., Banks, M.S.: Creating correct blur and its effect on accommodation. J. Vis. **18**(9), 1–29 (2018). https://doi.org/10.1167/18.9.1
41. Xu, F., Li, D.: Software based visual aberration correction for HMDs. In: Kiyokawa, K., Steinicke, F., Thomas, B., Welch, G. (eds) 2018 IEEE Conference on Virtual Reality and 3D User Interfaces. VR 2018, pp. 246–250. IEEE Computer Society, Los Alamitos, CA, USA (2018). https://doi.org/10.1109/VR.2018.8447557
42. Arefin, M.S.: [DC] SharpView AR: enhanced visual acuity for out-of-focus virtual content. In: O'Conner, L. (ed) Proceedings of the 2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops. VRW 2021, pp. 731–732. IEEE Computer Society, Los Alamitos, CA, USA (2021). https://doi.org/10.1109/VRW52623.2021.00248
43. Barsky, B.A., Kosloff, T.J.: Algorithms for rendering depth of field effects in computer graphics. In: New Aspects of Computers. Proceedings of the 12th WSEAS International Conference on COMPUTERS. WSEAS Press, Athens, Greece (2008)
44. Cook, R.L., Porter, T., Carpenter, L.: Distributed ray tracing. ACM SIGGR. Comput. Graph. **18**(3), 137–145 (1984). https://doi.org/10.1145/964965.808590
45. Wu, J., Zheng, C., Hu, X., Xu, F.: Rendering realistic spectral bokeh due to lens stops and aberrations. Vis. Comput. **29**(1), 41–52 (2013). https://doi.org/10.1007/s00371-012-0673-4
46. McGraw, T.: Fast Bokeh effects using low-rank linear filters. Vis. Comput. **31**(5), 601–611 (2015). https://doi.org/10.1007/s00371-014-0986-6
47. Schuster, K., Trettner, P., Kobbelt, L.: High-performance image filters via sparse approximations. Proc. ACM Comput. Graph. Interact. Tech. **3**(2), 14:1-141:9 (2020). https://doi.org/10.1145/3406182
48. Csoba, I., Kunkli, R.: Rendering algorithms for aberrated human vision simulation. Vis. Comput. Ind. Biomed. Art **6**, 5:1-5:25 (2023). https://doi.org/10.1186/s42492-023-00132-9
49. Csoba, I., Kunkli, R.: Efficient rendering of ocular wavefront aberrations using tiled point-spread function splatting. Comput. Graph. Forum **40**(6), 182–199 (2021). https://doi.org/10.1111/cgf.14267
50. Rosenholtz, R.: Capabilities and limitations of peripheral vision. Annu. Rev. Vis. Sci. **2**, 437–457 (2016). https://doi.org/10.1146/annurev-vision-082114-035733
51. Gu, Y., Legge, G.E.: Accommodation to stimuli in peripheral vision. J. Opt. Soc. Am. A **4**(8), 1681–1687 (1987). https://doi.org/10.1364/JOSAA.4.001681
52. Odden, J.L., Mihailovic, A., Boland, M.V., Friedman, D.S., West, S.K., Ramulu, P.Y.: Assessing functional disability in glaucoma: the relative importance of central versus far peripheral visual fields. Investig. Ophthalmol. Vis. Sci. **61**(13), 23:1-23:8 (2020). https://doi.org/10.1167/iovs.61.13.23
53. Bickerdt, J., Wendland, H., Geisler, D., Sonnenberg, J., Kasneci, E.: Beyond the tracked line of sight - evaluation of the peripheral usable field of view in a simulator setting. J. Eye. Mov. Res. **12**(3), 9:1-9:13 (2021)
54. Seidemann, A., Schaeffel, F., Guirao, A., Lopez-Gil, N., Artal, P.: Peripheral refractive errors in myopic, emmetropic, and hyperopic young subjects. J. Opt. Soc. Am. A **19**(12), 2363–2373 (2002). https://doi.org/10.1364/JOSAA.19.002363
55. Shen, J.: Ocular aberrations and image quality, contact lens and MYOPIA progression. In: Davey, P.G. (ed) Ophthalmology - Current Clinical and Research Updates, pp. 177–205. IntechOpen, London, UK (2014). https://doi.org/10.5772/58456
56. Camp, J.J., Maguire, L.J., Robb, R.A.: An efficient ray tracing algorithm for modeling visual performance from corneal topography.

In: [1990] Proceedings of the First Conference on Visualization in Biomedical Computing. VBC'90, pp. 278–285. IEEE Computer Society, Los Alamitos, CA, USA (1990). https://doi.org/10.1109/VBC.1990.109333
57. Barsky, B.A.: Vision-realistic rendering: simulation of the scanned foveal image from wavefront data of human subjects. In: Spencer S.N. (ed) Proceedings of the 1st Symposium on Applied Perception in Graphics and Visualization. APGV'04, pp. 73–81. Association for Computing Machinery, New York, NY, USA (2004). https://doi.org/10.1145/1012551.1012564
58. Barsky, B.A.: Vision-realistic rendering: simulation of the scanned foveal image with elimination of artifacts due to occlusion and discretization. In: Richard, P., Braz, J. (eds) Computer Vision, Imaging and Computer Graphics. Theory and Applications. VISIGRAPP 2010, Communications in Computer and Information Science, vol 229, pp. 3–27. Springer, Berlin, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25382-9_1
59. Mostafawy, S., Kermani, O., Lubatschowski, H.: Virtual eye: retinal image visualization of the human eye. IEEE Comput. Graph. Appl. **17**(1), 8–12 (1997). https://doi.org/10.1109/38.576849
60. Wu, J., Zheng, C., Hu, X., Xu, F.: Realistic simulation of peripheral vision using an aspherical eye model. In: Avis, N.J., Lefebvre, S. (eds) Eurographics 2011 - Short Papers. EG 2011, pp. 37–40. The Eurographics Association, Goslar, DEU (2011). https://doi.org/10.2312/EG2011/short/037-040
61. Dias, C., Wick, M., Rifai, K., Wahl, S.: Peripheral retinal image simulation based on retina shapes. In: Santos, L.P., Bashford-Rogers, T. (eds) Proceedings of the 37th Annual Conference of the European Association for Computer Graphics: Short Papers. EG'16, pp. 61–64. The Eurographics Association, Goslar, DEU (2016). https://doi.org/10.2312/egsh.20161015
62. Sun, Y., Fracchia, F.D., Drew, M.S., Calvert, T.W.: A spectrally based framework for realistic image synthesis. Vis. Comput. **17**(7), 429–444 (2001). https://doi.org/10.1007/s003710100116
63. Lian, T., MacKenzie, K.J., Brainard, D.H., Cottaris, N.P., Wandell, B.A.: Ray tracing 3D spectral scenes through human optics models. J. Vis. **19**(12), 23:1-23:17 (2019). https://doi.org/10.1167/19.12.23
64. Fink, W., Micol, D.: simEye: computer-based simulation of visual perception under various eye defects using Zernike polynomials. J. Biomed. Opt. **11**(5), 054011:1-054011:12 (2006). https://doi.org/10.1117/1.2357734
65. Wei, Q., Patkar, S., Pai, D.K.: Fast ray-tracing of human eye optics on graphics processing units. Comput. Methods Progr. Biomed. **114**(3), 302–314 (2014). https://doi.org/10.1016/j.cmpb.2014.02.003
66. Vu, C.T., Stock, S., Fan, L.T., Stork, W.: Highly parallelized rendering of the retinal image through a computer-simulated human eye for the design of virtual reality head-mounted displays. In. Schelkens, P., Kozacki, T. (eds) Optics, Photonics and Digital Technologies for Imaging Applications VI, Proceedings of SPIE, vol. 11353, pp. 1135316:1–1135316:20. International Society for Optics and Photonics, SPIE (2020). https://doi.org/10.1117/12.2555872
67. Tang, N., Xiao, S.: Real-time human vision rendering using blur distribution function. In: Spencer, S.N. (ed) Proceedings of the 14th ACM SIGGRAPH International Conference on Virtual Reality Continuum and its Applications in Industry. VRCAI'15, pp. 39–42. ACM, New York, NY, USA (2015). https://doi.org/10.1145/2817675.2817686
68. Lima, A.R.C., Medeiros, A.M., Marques, V.G., Oliveira, M.M.: Real-time simulation of accommodation and low-order aberrations of the human eye using light-gathering trees. Vis. Comput. **37**(9–11), 2581–2593 (2021). https://doi.org/10.1007/s00371-021-02194-3
69. Wei, X., Thibos, L.: Modeling the eye's optical system by ocular wavefront tomography. Opt. Express **16**(25), 20490–20502 (2008). https://doi.org/10.1364/oe.16.020490
70. Navarro, R., Santamaría, J., Bescós, J.: Accommodation-dependent model of the human eye with aspherics. J. Opt. Soc. Am. A **2**(8), 1273–1280 (1985). https://doi.org/10.1364/JOSAA.2.001273
71. Bhatia, A.B., Wolf, E.: On the circle polynomials of Zernike and related orthogonal sets. Math. Proc. Camb. Philos. Soc. **50**(1), 40–48 (1954). https://doi.org/10.1017/S0305004100029066
72. Janssen, A.J.E.M., Braat, J.J.M., Dirksen, P.: On the computation of the Nijboer-Zernike aberration integrals at arbitrary defocus. J. Mod. Opt. **51**(5), 687–703 (2004). https://doi.org/10.1080/09500340408235546
73. Dai, G.-m.: Wavefront optics for vision correction. SPIE, Bellingham, WA, USA (2008)
74. van Haver, S.: The extended Nijboer-Zernike diffraction theory and its applications. Ph.D. thesis, Delft University of Technology (2010)
75. Antonello, J., Verhaegen, M.: Modal-based phase retrieval for adaptive optics. J. Opt. Soc. Am. A **32**(6), 1160–1170 (2015). https://doi.org/10.1364/JOSAA.32.001160
76. Csoba, I., Kunkli, R.: Fast, GPU-based computation of large point-spread function sets for the human eye using the extended Nijboer-Zernike approach. In: Fazekas, I. (ed) 2022 IEEE 2nd Conference on Information Technology and Data Science. CITDS 2022, pp. 69–73. IEEE Computer Society, Los Alamitos, CA, USA (2022). https://doi.org/10.1109/CITDS54976.2022.9914232
77. Herrmann, J.: Least-squares wave front errors of minimum norm. J. Opt. Soc. Am. A **70**(1), 28–35 (1980). https://doi.org/10.1364/JOSA.70.000028
78. 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. Investig. Ophthalmol. Vis. Sci. **50**(1), 281–289 (2009). https://doi.org/10.1167/iovs.08-2124
79. Chen, D., Hu, F., Nian, G., Yang, T.: Deep residual learning for nonlinear regression. Entropy **22**(2), 193:1–193:14 (2020). https://doi.org/10.3390/e22020193
80. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Fürnkranz, J., Joachims, T. (eds) Proceedings of the 27th International Conference on Machine Learning. ICML 2010, pp. 807–814. Omnipress, Madison, WI, USA (2010)
81. Misra, D.: Mish: a self regularized non-monotonic activation function. Paper presented at the 31st British Machine Vision Conference, BMVC 2020. British Machine Vision Association, Virtual event, 7–10 September 2020. https://bmvc2020-conference.com/conference/papers/paper_0928.html (2020). Accessed 3 March 2023
82. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Bach, F., Blei, D. (eds) Proceedings of the 32nd International Conference on Machine Learning. ICML'15, pp. 448–456. JMLR (2015)
83. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. Paper presented at the NIPS 2016 Deep Learning Symposium. 8 December 2016, Barcelona, Spain. https://arxiv.org/pdf/1607.06450v1.pdf (2016). Accessed 3 March 2023
84. Sauer, A., Chitta, K., Müller, J., Geiger, A.: Projected GANs converge faster. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P.S., Wortman Vaughan, J. (eds.) Advances in Neural Information Processing Systems 34, pp. 17480–17492. Curran Associates Inc, New York, NY, USA (2021)
85. Lyu, Q., Chen, M., Chen, X.: Learning color space adaptation from synthetic to real images of cirrus clouds. Vis. Comput. **37**(8), 2341–2353 (2021). https://doi.org/10.1007/s00371-020-01990-7

86. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., et al.: TensorFlow: a system for large-scale machine learning. In: Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation. OSDI '16, pp. 265–283. USENIX Association, Berkeley, CA, USA (2016)
87. MATLAB version 9.9.0.1495850 (R2020b). The Mathworks, Inc., Natick, MA, USA (2020)
88. Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., Han, J.: On the variance of the adaptive learning rate and beyond. Paper presented at the 8th International Conference on Learning Representations, ICLR 2020. Virtual event, 26–30 April 2020. https://iclr.cc/virtual_2020/poster_rkgz2aEKDr.html (2020) Accessed 3 March 2023
89. Kingma, D.P., Ba, J.L.: Adam: a method for stochastic optimization. Paper presented at the 3rd International Conference on Learning Representations, ICLR 2015. San Diego, CA, USA, 7–9 May 2015. https://arxiv.org/pdf/1412.6980v5.pdf (2015) Accessed 3 March 2023
90. Zhang, M.R., Lucas, J., Hinton, G.E., Ba, J.: Lookahead optimizer: k steps forward, 1 step back. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.A., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 9597–9608. Curran Associates Inc, New York, NY, USA (2019)
91. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. Paper presented at the 7th International Conference on Learning Representations, ICLR 2019. New Orleans, LA, USA, 6–9 May 2019. https://openreview.net/forum?id=Bkg6RiCqY7 (2019) Accessed 3 Mar 2023
92. Kessenich, J., Sellers, G., Shreiner, D.: OpenGL programming guide: the official guide to learning OpenGL, Version 4.5 with SPIR-V. 9th ed., Addison-Wesley Professional, Glenview, IL, USA (2016)



István Csoba received his M.Sc. degree in Computer Science from the University of Debrecen, Hungary in 2017. He is currently a Ph.D. student at the Doctoral School of Informatics, University of Debrecen, Hungary. His research interests mainly include computer graphics and real-time rendering, with a particular focus on massively parallel algorithms and the simulation of the characteristics of optical systems.

Roland Kunkli received his M.S. and Ph.D. degrees from the University of Debrecen, Hungary; he graduated as a teacher of mathematics and descriptive geometry (2008), and as a teacher of informatics (2012), then he received his Ph.D. degree in computer science in 2015. He currently works as an assistant professor at the Department of Data Science and Visualization, Faculty of Informatics, University of Debrecen, Hungary. His research interests are in computer aided geometric design, computer graphics, and data visualization.