# Simplifying numerical ray tracing for characterization of optical systems

Yakir Luc Gagnon,[1,*] Daniel I. Speiser,[2] and Sönke Johnsen[1]

[1]Department of Biology, Duke University, Durham, North Carolina 27708, USA

[2]Department of Ecology, Evolution, and Marine Biology, University of California Santa Barbara, Santa Barbara, California 93106, USA

*Corresponding author: 12.yakir@gmail.com

Received 20 March 2014; revised 11 June 2014; accepted 13 June 2014; posted 19 June 2014 (Doc. ID 208575); published 18 July 2014

Ray tracing, a computational method for tracing the trajectories of rays of light through matter, is often used to characterize mechanical or biological visual systems with aberrations that are larger than the effect of diffraction inherent in the system. For example, ray tracing may be used to calculate geometric point spread functions (PSFs), which describe the image of a point source after it passes through an optical system. Calculating a geometric PSF is useful because it gives an estimate of the detail and quality of the image formed by a given optical system. However, when using ray tracing to calculate a PSF, the accuracy of the estimated PSF directly depends on the number of discrete rays used in the calculation; higher accuracies may require more computational power. Furthermore, adding optical components to a modeled system will increase its complexity and require critical modifications so that the model will describe the system correctly, sometimes necessitating a completely new model. Here, we address these challenges by developing a method that represents rays of light as a continuous function that depends on the light's initial direction. By utilizing Chebyshev approximations (via the *chebfun* toolbox in MATLAB) for the implementation of this method, we greatly simplified the calculations for the location and direction of the rays. This method provides high precision and fast calculation speeds that allow the characterization of any symmetrical optical system (with a centered point source) in an analytical-like manner. Next, we demonstrate our methods by showing how they can easily calculate PSFs for complicated optical systems that contain multiple refractive and/or reflective interfaces. 

*OCIS codes:* (080.0080) Geometric optics; (080.1510) Propagation methods; (080.1753) Computation methods; (080.4225) Nonspherical lens design; (080.4228) Nonspherical mirror surfaces.

http://dx.doi.org/10.1364/AO.53.004784

## 1. Introduction

Biologists and engineers often want to compare the performance of optical systems that differ in their optical components (e.g., pupils, lenses, mirrors, etc.) or evaluate how altering the components of a particular system will influence the quality of the images that it forms [e.g., 1–16]. These differences and/or effects are usually not trivial and may require a model that encompasses most of the factors in play. Geometric optics is one of the primary methods used to evaluate the performance of optical systems [17]. In geometric optics, light trajectories are traced as rays that extend from a light source in a straight line until they are refracted, reflected, or absorbed by the interfaces in an optical system. Following the trajectories of these rays allows us to track light from its

1559-128X/14/214784-07$15.00/0

source, through an optical system, to its termination point.

One of ray tracing's main uses is quantifying how well an optical system resolves spatial detail. This is achieved by tracing rays from a point source through an optical system until they reach the system's image plane, and then sampling those rays in order to form an image. This image describes the amount of blur inherent to the optical system and is referred to as a point spread function (PSF). Researchers often use PSFs to compare different optical systems [18]. The Fourier transform of an optical system's PSF gives the system's modulation transfer function (MTF), which can be used to quantify image contrast as a function of a signal's spatial frequency after it has passed through the system.

In all ray-tracing algorithms, the accuracy of the results directly depends on the number of discrete rays used in the calculation [19]. This limitation is obvious particularly when estimating the PSF of an optical system, because the statistical estimate (accuracy) of the PSF improves with a larger number of samples (rays) [20]. Since this dependence is specific to each ray-tracing algorithm and the optical setup it emulates, extensive testing is necessary to see how altering the number of rays affects the results. Indeed, Portilla and Barbero showed that convergence is slow and may result in a very large number of discrete rays. Some solutions have been suggested in an attempt to solve this problem [20–22].

An additional difficulty in ray tracing is the accurate representation of the refractive/reflective interfaces in an optical system. The curves of optical interfaces are usually represented with either a straight line, simple polynomial, circle, ellipse, or some segmentation or combination thereof [17]. These methods are useful for calculating the trajectories of rays in many optical systems [19], but they do not encompass all of the possible curvatures that may exist in a system. This limitation necessitates various approximations and shortcuts (e.g., fitting a circle to an irregularly shaped interface) that introduce unpredictable errors to ray-tracing algorithms.

In the following study, we present a novel way of circumventing these limitations by using Chebyshev approximations to describe both the optical interfaces in an optical system and the rays of light that interact with the system as a *continuous* function of the rays' initial location and direction. While describing the location and direction of light and the interfaces with analytical functions becomes impossible for even relatively simple optical systems, Chebyshev approximations immensely simplify the required calculations (e.g., differentiation, integration, function inverse, root finding, etc.). Here, we implement Chebyshev approximations for analyzing the performances of symmetrical optical systems (i.e., systems with objects and a point source that are symmetrical about the optical axis) using the MATLAB *chebfun* toolbox [23]. All the code [24] directly runs on any MATLAB session that includes the *chebfun* toolbox.

## 2. Simple Case Study

As an introduction and a basic test for our methods, we will consider a simple camera-eye system that includes one isotropic point source, a lens (made of two refractive surfaces), and an image plane. Since the system is rotationally symmetrical around the optical axis, ray tracing can be easily simplified by calculating trajectories at only one of the planes that contains the optical axis (Fig. 1). We will later adjust our model for this simplification and extend its applicability to 3D.

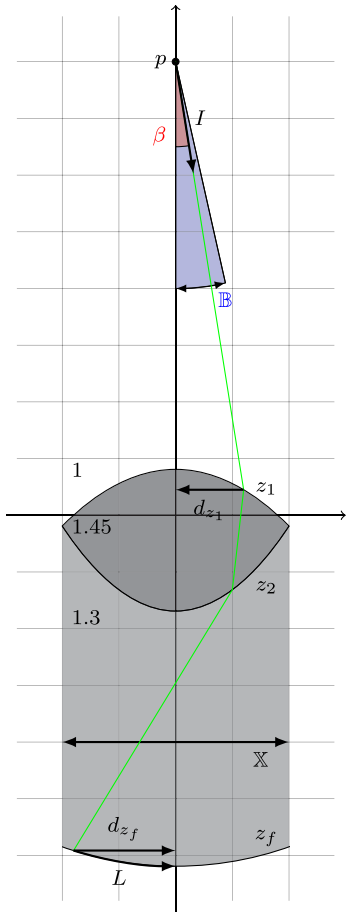


Fig. 1. Optical setup of the simple case study. The $y$ axis is the system's optical axis and the aperture is located on the $x$ axis. The grid lines represent 0.5 unit increments. The system is defined across the domain of $x$, $x:\mathbb{X} = [-1, 1]$, and includes $z_1$, $z_2$, and $z_f$ with refractive indices of 1, 1.45, and 1.3, respectively. The point source is located at $p = 0 + 4i$, and an example ray (green line) with exit angle $\beta$ (red highlighted angular area) has direction $I$. The domain of $\beta$ can vary within the blue highlighted angular area (i.e., $\beta:\mathbb{B} = [0, 12.7°]$). The example ray intersects the first interface, $z_1$, at a point that is $d_{z_1}$ from the optical axis. After following an unrealistic hypothetical trajectory (chosen only for explanatory reasons), the ray intersects the image plane at "direct" distance $d_{z_f}$ and arc length $L$ from the optical axis. See further specifications in the code at [24].

Diffraction may cause interference patterns and affect the PSF of an optical system when components of the system have dimensions comparable to the wavelength of the light passing through them (e.g., around 500 nm for visible light) [18]. Similar to other ray-tracing algorithms, we ignore the effects of diffraction in our study, limiting the functionality of our model to cases where diffraction's effects are negligible. Without diffraction, the units in this example are therefore arbitrary. In addition, it is important to note that we allow rays to travel from one interface to the next in a *predetermined* and specific order: light starts at the point source; it then intersects with the closest lens surface, then the second lens surface, and lastly with the image plane (and not in any other order). Scenarios where individual light rays can intersect with any of the included optical components in an arbitrary order are not within the scope of this study (i.e., no stray light effects are included).

Due to the ease of complex-number representation in MATLAB and in *chebfun*, we will describe the location and direction of rays of light as complex Chebyshev functions (*chebfun*) that depend on the exit angle, $\beta$, which is the angle between the ray exiting the light source and the optical axis of the optical system. This approach allows us to follow the trajectory of any ray of light from the light source, through the optical system, to the image plane as a function of $\beta$.

Exit angles, $\beta$, can take values between zero and the arctan of the radius of the optical system's aperture, $r$, divided by the distance between the light source and the aperture of the optical system, $L$. We therefore define $\beta$'s domain, $\mathbb{B}$, to those values (Eq. 1). The light's initial position, $p$, and direction, $I$, are described as a function of $\beta$. Note that the direction of the light, $I$, is relative to the $x$ axis of the system and therefore requires the $-90°$ rotation of $\beta$ in Eq. (3):

$$\beta{:}\mathbb{B} = [0, \arctan(r/L)], \tag{1}$$

$$p(\beta) = 0 + Li, \tag{2}$$

$$I(\beta) = \exp(i(\beta - \pi/2)). \tag{3}$$

Next comes the categorization of the refractive interfaces of the lens and the image plane. We describe these surfaces as real-domain functions that depend on $x$ and can take any shape. In our example, the lens surfaces, $z_1$ and $z_2$ (distally and proximally positioned relative to the image plane), are a second-degree polynomial (i.e., a parabola) and the image plane, $z_f$, is the lower half of a circle (see Fig. 1). It is most convenient to group $z_1$, $z_2$, and $z_f$ in a matrix of functions, $z$. The domain of $z$, $\mathbb{X}$, must be larger than the optical system's aperture in order to include rays that refracted further away from the optical axis, e.g., between $-2r$ and $2r$. Since the distance of the light source from the aperture should be equal to $L$, we translate all the surfaces in $z$ along the $y$ axis so that the aperture lies on the $x$ axis of the system and equals $2r$ (see Fig. 1):

$$z = \begin{bmatrix} z_1 \\ z_2 \\ z_f \end{bmatrix}, \tag{4}$$

$$z{:}\mathbb{X} = [-2r, 2r], \tag{5}$$

$$z = z - z_1(r). \tag{6}$$

Tracing the rays through the interfaces of our simple optical system occurs sequentially by calculating the location of the rays and their direction after refraction as a function of $\beta$. The intersection between the light and a specific interface, $z_i$, (element $i$ of the matrix $z$) is calculated by solving the difference between the interface function and the ray expressed as a function of $x$. For each specific exit angle, $\beta = \beta_1$, there is one singular point of light, $p_{\beta_1}$, with a single direction, $I_{\beta_1}$. That directional light is described as a linear function, $f$, with its own slope, $a$, and intercept, $b$, and intersects the interface's surface function at a specific $x$ value. Both functions share the same domain, namely $\mathbb{X}$:

$$a = \Im(I_{\beta_1})/\Re(I_{\beta_1}), \tag{7}$$

$$b = \Im(p_{\beta_1}) - a\Re(p_{\beta_1}), \tag{8}$$

$$f = ax + b, \tag{9}$$

$$z_i - f = 0. \tag{10}$$

While analytically solving Eq. (10) for $x$ becomes impossible even for relatively simple refractive interfaces, solving it with the *chebfun* toolbox in MATLAB (specifically the *roots* function) is straightforward.

The ray's new direction after interacting with each refractive surface is calculated by first finding the angle between the normal of the surface, $N$, and the ray before the refraction, $\alpha_1$. The angle between the normal and the $x$ axis, $\angle N$, is equal to the arctan of the derivative of the surface function plus $\pi/2$. We then use Snell's law to calculate the refracted angle, $\alpha_2$, and the angle of the refracted ray, $\theta$, and the light's new direction, $I$ (see [24] for the implementation):

$$\angle N = \arctan\left(\frac{dz}{dx}\right) + \frac{\pi}{2}, \tag{11}$$

$$\alpha_1 = \angle N(\Re(p)) - \arctan \Im(I)/\Re(I), \tag{12}$$

$$\alpha_2 = \arcsin(n_1 \sin(\alpha_1)/n_2), \tag{13}$$

$$I = \exp(i(\angle N + \alpha_2 - \pi)). \tag{14}$$

In order to validate our methodology, the same optical system was constructed in the Zemax software package and ray traced to produce a "spot diagram" (a hexapolar pattern with a sampling density of 18 rings was used). The spot diagram indicates the location of individual rays at the entrance pupil and at the image plane (after traveling through the system). We constructed an equivalent spot diagram (one for the entrance pupil and one for the image plane) and found no visible differences between the spot diagrams from Zemax and ours [see Fig. 5(a) in Appendix A].

Once the location of the light is traced to the surface of the image plane, $z_3$, the arc lengths between these intersection points and the optical axis are calculated. These arc lengths encompass all the possible distances (between the optical axis and the intersection point of the ray and the image plane) the rays can take, described as a function of $\beta$. We define arc length as a function of $x$, $l(x)$. We center $l(x)$ so that $x$ values smaller than zero will result in a negative distance from the optical axis and vice versa:

$$l(x) = \int_{\mathbb{X}} \left| \frac{\mathrm{d}(x + iz_3)}{\mathrm{d}x} \right| \mathrm{d}x, \tag{15}$$

$$l(x) = l(x) - l(0). \tag{16}$$

Using function composition, $l$ takes the real part of the rays' positions on the image plane $\Re(p)$, and results in all possible arc lengths as a function of $\beta$ (which $p$ is). While some of these lengths might be negative (i.e., intersect with the image plane on the negative end of the system's $x$ axis), the rotational symmetry of our hypothetical optical system ensures that those negative lengths exist on the positive end of the $x$ axis as well. The absolute value of this composition is therefore necessary:

$$L = |l \circ \Re(p)|. \tag{17}$$

Sampling $L$ across $\beta$'s domain, $\mathbb{B}$, results in a discrete distribution of arc length deviations from the optical axis (much like sampling from an inverse cumulative distribution function). This distribution of how much the light spread is the optical system's PSF (Fig. 2).

### A. Adjustments for 3D

In the previous section, we outlined methods for calculating a PSF for a simple two-dimensional optical system. However, real optical systems are not 2D. In rotationally symmetrical imaging systems, adjusting the calculations that we use for 2D systems so that they work in 3D is straightforward and involves a few alterations to our model that we describe in this section (i.e., there is no need for an actual 3D calculation).

The simplification of calculating trajectories of light rays in only one plane misrepresents the relative intensities of light as a function of distance from the optical axis. These misrepresentations occur at two places in the system (but affect it throughout): the aperture and the image plane. While an optical system's aperture is a disk in 3D, it is represented as a

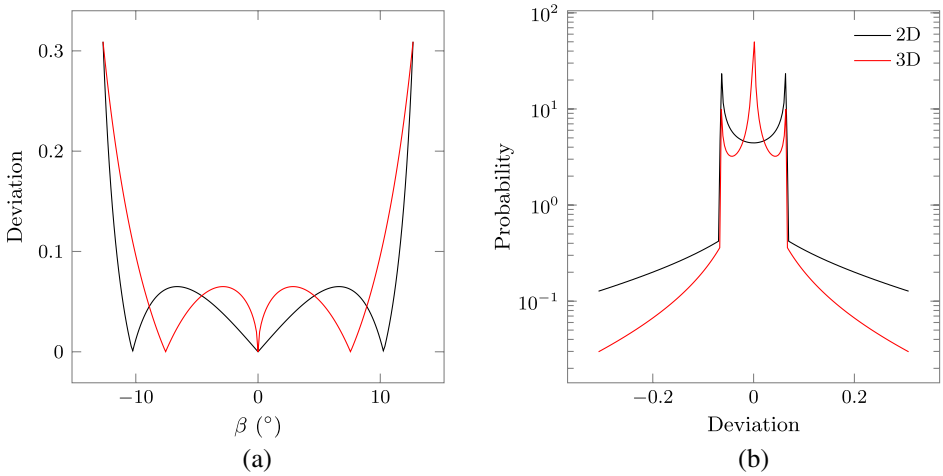


Fig. 2. Deviation of rays from the optical axis in the simple case study. (a) Deviations of rays from the optical axis as a function of $\beta$ (the initial angle between the optical axis and an exiting ray of light) in 2D (black) and 3D (red). The $x$ axis is $\beta$ across its domain. The $y$ axis is the arc length (along the image plane) between the optical axis and the point where each ray intersected with the image plane [calculated as $L$ in Eq. (17)]. (b) PSFs of the optical system in 2D (black) and 3D (red). The $x$ axis is the deviation sampled from (a). The (log-scale) $y$ axis describes the corresponding probability each deviation has in the system.

line in 2D. Points along that line that lie further away from the optical axis represent larger circles in the 3D disk, and points that are closer to the optical axis represent smaller circles. This affects the distribution of light as a function of distance from the optical axis, i.e., more rays enter the system at its periphery than closer to its center. Likewise, we should also correct for the 2D distribution of light intensities at the image plane [i.e., the system's PSF; see black line in Fig. 2(b)]. As in the case of an optical system's aperture, greater distances from the optical axis represent larger circles in the system's image plane. A given intensity in the image plane with a small deviation from the optical axis is spread across less area than an intensity that lies further away from the optical axis. To correct for these discrepancies, we need to adjust the distribution of the incoming rays as a function of distance from the optical axis (i.e., $\beta$), and divide the PSF with a weighting function.

As mentioned, the distribution of incoming rays depends on the rays' distance to the optical axis. This distance depends on where the rays intersected the first refractive interface of the system, $z_1$. This can be calculated from the real part of the (complex) position of the light at the first refractive interface, $\Re(p_2)$. The distance between rays and the optical axis as a function of $\beta$ is therefore the inverse of $\Re(p_2)$ (denoted as $d_{z_1}$ in Fig. 1). Finding the inverse of a function analytically can be impossible even for simple cases, but with the *chebfun* toolbox it is straightforward (via the *inv* function).

A function that will generate the correct distribution of incoming rays is the inverse cumulative probability function (ICDF) of uniformly spaced points on a disk. The probability density function (PDF) of light hitting a disk quantified by its distance from the optical axis is $2d/r^2$, where $r$ is the aperture radius and $d$ is the distance to the optical axis. The cumulative distribution function of that PDF is $(d/r)^2$ and equals 1 for $d = r$. The ICDF is therefore $r\sqrt{\beta/\max(\beta)}$ for $\beta$, and is denoted as $F$.

Using function composition again allows us to adjust the arc lengths from Eq. (17) so that they will conform to the correct distribution of incoming light in a 3D system. Here, $L$ takes the distances between incoming rays and the optical axis, $d$, which in turn takes the ICDF for uniformly distributed points on a disk:

$$L_2 = L \circ d \circ \mathrm{F}. \tag{18}$$

The PSF's weighting function should express the density of uniformly spaced points on a disk quantified by their distance from the optical axis, namely their PDF, which was $2d/r^2$. Since the PSF will later be divided by its sum (to normalize its sum to one), that weighting function can be simplified to $d$, which is the distance from the rays to the optical axis (denoted as $d_{z_f}$ in Fig. 1). Since the PSF is defined across the arc length from the optical axis along the image plane surface, we will have to convert arc distances to "direct" distances. The function $l$ in Eq. (3) is arc length as a function of $x$. Since $x$ is the direct distance to the optical axis (which is the $y$ axis), the inverse of $l$ results in the weights as a function of arc length. The inverse of $l$ is easily computed with *chebfun*'s *inv* function. After incorporating these two adjustments, the resulting 3D PSF is substantially different from the 2D PSF in this simple case study (Fig. 2).

In order to test the validity of our algorithm, we compared our results to previously published geometric PSFs. Liu and Lin calculated the geometric PSF of a double-convex lens system using both standard ray counting as well as their own, more accurate, method (see Figs. 9 & 10 on page 134 in [22]). Using the same parameters, we modeled an identical double-convex lens system and calculated its PSF using our algorithm. It was straightforward to adapt our algorithm to both of the methods presented in Liu and Lin's study due to the high flexibility of our code scheme. The PSFs we produced

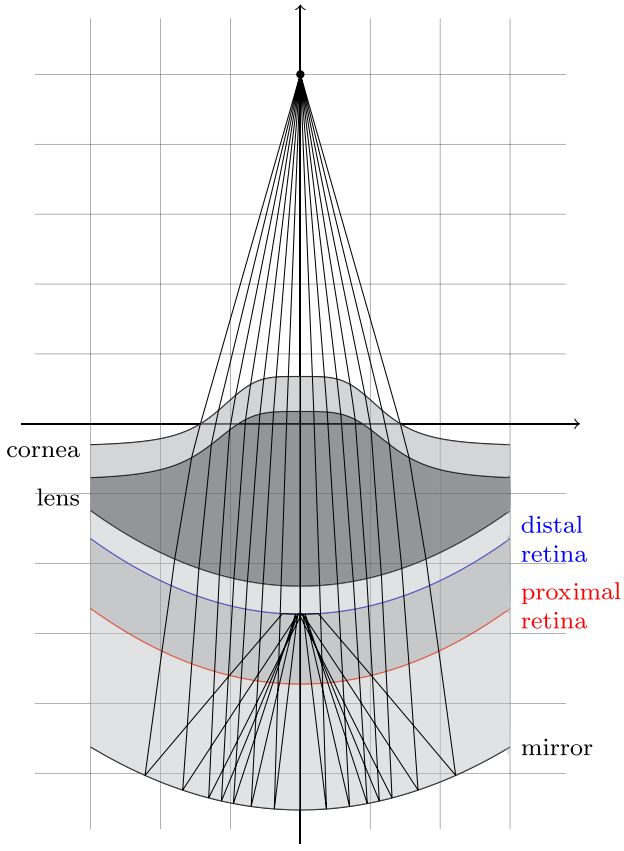


Fig. 3. Traced rays through the scallop eye. The $y$ axis is the system's optical axis and the aperture is located on the $x$ axis. The grid lines represent 100 μm increments. The system includes a cornea, lens, distal retina (in blue), proximal retina (in red), and mirror. The refractive indices of the surrounding medium, cornea, and lens are 1.34, 1.37, and 1.52, respectively, while 1.36, 1.38, and 1.36 are the refractive indices outside the distal retina, proximal retina, and the mirror, respectively. The point source is located 500 μm from the aperture (see more size specifications in [24]).

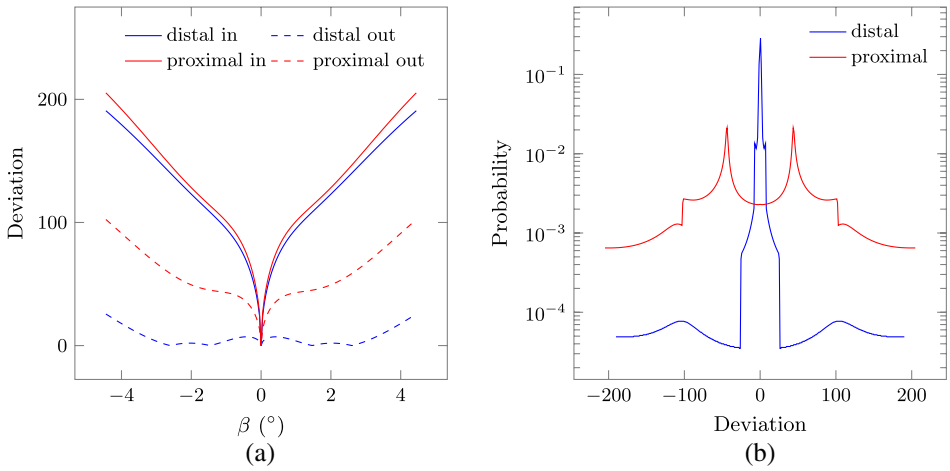


Fig. 4. Deviation of rays from the optical axis for a scallop's eye. (a) Deviations of rays from the optical axis as a function of $\beta$ in the distal retina (blue) and the proximal retina (red) for both incoming (solid) and exiting (dashed) light. The $x$ axis is $\beta$ across its domain. The $y$ axis is the arc length (along the image plane) between each ray–plane intersection point and the optical axis [calculated as $L$ in Eq. (17)]. (b) PSFs of the scallop eye at the distal retina (blue) and proximal retina (red). The $x$ axis is the deviation sampled from (a). The (log-scale) $y$ axis describes the corresponding probability each deviation has in the system.

with our algorithm were identical to Liu and Lin's PSFs [a comparison between Liu and Lin's suggested method and ours is in Fig. 5(b) in Appendix A]. These results corroborate the validity of our methods and algorithm.

### B. Adding Complexity

The previous sections narrated the ray-tracing principles used in this study through a simple optical setup. Now, we will apply our methods to model a more complicated optical system. Building on the principles introduced earlier, we will demonstrate that we can evaluate the performance of almost any rotationally symmetrical optical system. The usefulness of our methods is demonstrated by modeling the optics of the concave mirror eyes of scallops (Bivalvia). Scallops are a group of swimming bivalves with surprisingly complex eyes. The eyes of scallops have an irregularly curved cornea with a thick biconvex lens with aspherical surfaces [25]. Behind the lens are two spherical retinas and behind those is a large spherical mirror. Light gets refracted by the cornea and lens, partially absorbed by the distal retina and then the proximal retina, and reflected back by the mirror (and converged) to the two retinas (this time in reverse order) [25]. All the refractive media may have different refractive properties (Fig. 3). Despite its five different refractive interfaces, reflective component, irregularly shaped curves, and multiple image planes, the scallop eye's PSFs are just as easily computed as in the simple case study using the same functions and an additional reflection function. PSFs for a scallop's eye are presented in Fig. 4.

### C. Advantages

Describing the location and direction of light in terms of a *chebfun* has many advantages that are hard or impossible to program in a more conventional discrete paradigm. The transition between a discrete finite number of rays to a continuous function describing the rays' location and direction is made trivial with our algorithm. High-precision and fast computation is a difficult combination to achieve in any algorithm, especially those used for ray tracing. Nevertheless, the methods that we present here result in machine-error precision ($\epsilon < 3 \cdot 10^{-16}$) and relatively high speeds (the scallop eye code in the previous section was completed in less than 5 s on an Intel Core i7-3520M CPU at 2.90 GHz). The *chebfun* framework produces very flexible code that allows quick adaptations to a large variety of applications, such as drawing comparisons between different imaging systems or evaluating how changing the components of a particular system alters its performance. The flexibility of our methods was evident when comparing the codes for the simple case study and the more complicated scallop eye: while the simple case study was written in 35 lines of code, the scallop eye took only 22 additional lines. Further, our methods allow for the accurate description of any irregularly shaped interface in an imaging system, eliminating the restrictions on shape that hinder more conventional algorithms.

## 3. Conclusions

We have presented a versatile method of tracing rays through increasingly complicated optical systems by describing rays as continuous functions that depend on the ray's exit angle from the light source. The framework of the *chebfun* MATLAB toolbox proved to be well adapted and intuitive to the adjustments needed to describe and ray trace a large variation of optical systems quickly and accurately.

## Appendix A

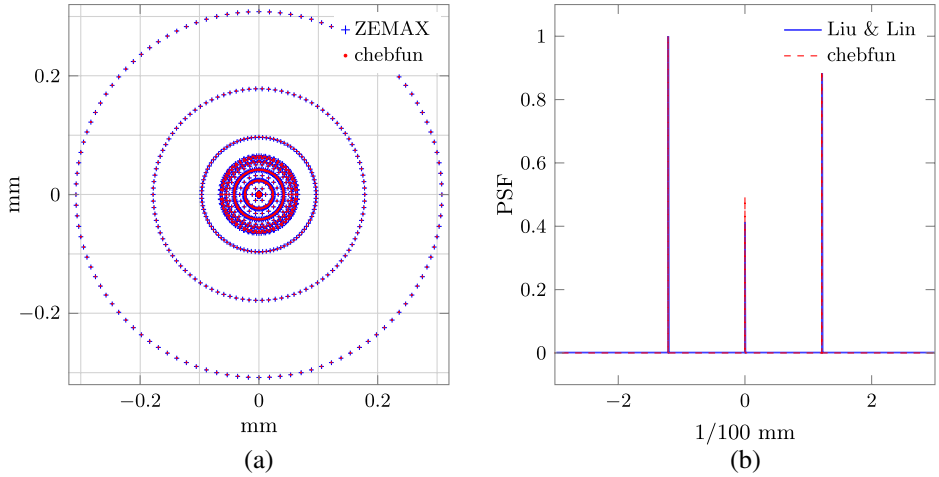


Fig. 5. Comparisons of the algorithm from this study to other methods. (a) Comparison of a spot diagram calculated by ZEMAX (blue cross) and our algorithm (red dot). (b) Comparison of a PSF calculated by Liu and Lin [22] (blue line) and our algorithm (red-dashed line).

The authors would like to thank Justin Haag for valuable help with the Zemax comparisons. This research was funded by MURI (grant number N00014-09-1-1053) from the Office of Naval Research.

## References

1. Y. L. Gagnon, R. H. H. Kröger, and B. Söderberg, "Adjusting a light dispersion model to fit measurements from vertebrate ocular media as well as ray-tracing in fish lenses," Vis. Res. **50**, 850–853 (2010).
2. L. F. Garner, G. Smith, S. Yao, and R. C. Augusteyn, "Gradient refractive index of the crystalline lens of the Black Oreo Dory (*Allocyttus niger*): comparison of magnetic resonance imaging (MRI) and laser ray-trace methods," Vis. Res. **41**, 973–979 (2001).
3. A. Sharma, D. V. Kumar, and A. K. Ghatak, "Tracing rays through graded-index media: a new method," Appl. Opt. **21**, 984–987 (1982).
4. W. S. Jagger, "The optics of the spherical fish lens," Vis. Res. **32**, 1271–1284 (1992).
5. Y. L. Gagnon, T. T. Sutton, and S. Johnsen, "Visual acuity in pelagic fishes and mollusks," Vis. Res. **92**, 1–9 (2013).
6. D. Y. C. Chan, "Determination and modeling of the 3-D gradient refractive-indexes in crystalline lenses," Appl. Opt. **27**, 926–931 (1988).
7. P. L. Chu, "Nondestructive measurement of index profile of an optical-fibre preform," Electron. Lett. **13**, 736–738 (1977).
8. Y. L. Gagnon and R. H. H. Kröger, "Gradient index models of monofocal and multifocal spherical fish lenses," Investig. Ophthalmol. Vis. Sci. **47**, 1211 (2006).
9. Y. L. Gagnon, B. Söderberg, and R. H. H. Kröger, "Optical advantages and function of multifocal spherical fish lenses," J. Opt. Soc. Am. A **29**, 1786–1793 (2012).
10. Y. L. Gagnon, B. Söderberg, and R. H. H. Kröger, "Effects of the peripheral layers on the optical properties of spherical fish lenses," J. Opt. Soc. Am. A **25**, 2468–2475 (2008).
11. R. H. H. Kröger, M. C. W. Campbell, R. D. Fernald, and H. J. Wagner, "Multifocal lenses compensate for chromatic defocus in vertebrate eyes," J. Comp. Physiol. A **184**, 361–369 (1999).
12. R. H. H. Kröger, M. C. W. Campbell, R. Munger, and R. D. Fernald, "Refractive index distribution and spherical aberration in the crystalline lens of the African cichlid fish Haplochromis burtoni," Vis. Res. **34**, 1815–1822 (1994).
13. O. E. Lind, A. Kelber, and R. H. H. Kröger, "Multifocal optical systems and pupil dynamics in birds," J. Exp. Biol. **211**, 2752–2758 (2008).
14. L. Matthiessen, "Ueber die beziehungen, welche zwischen dem brechungsindex des kerncentrums der krystalllinse und den dimensionen des auges bestehen," Pflüger's Archiv **27**, 510–523 (1882).
15. J. M. Schartau, B. Sjögreen, Y. L. Gagnon, and R. H. H. Kröger, "Optical plasticity in the crystalline lenses of the cichlid fish Aequidens pulcher," Curr. Biol. **19**, 122–126 (2009).
16. J. G. Sivak and R. O. Kreuzer, "Spherical aberration of the crystalline lens," Vis. Res. **23**, 59–70 (1983).
17. P. Mouroulis and J. Macdonald, *Geometrical Optics and Optical Design* (Oxford University, 1997).
18. E. Hecht, *Optics* (Addison-Wesley, 2002).
19. J. Arasa and J. Alda, *Real Ray Tracing* (Marcel Dekker, 2004).
20. J. Portilla and S. Barbero, "Accuracy of geometric point spread function estimation using the ray-counting method," Proc. SPIE **8550**, 855003 (2012).
21. O. N. Stavroudis and D. P. Feder, "Automatic computation of spot diagrams," J. Opt. Soc. Am. **44**, 163–164 (1954).
22. C.-S. Liu and P. D. Lin, "Computational method for deriving the geometric point spread function of an optical system," Appl. Opt. **49**, 126–136 (2010).
23. L. N. Trefethen, "Chebfun Version 4.2," The Chebfun Development Team (2011), http://www.chebfun.org/.
24. Y. L. Gagnon, "chebRay," (2014), https://github.com/yakir12/chebRay.
25. M. F. Land, "Activity in the optic nerve of *Pecten maximus* in response to changes in light intensity, and to pattern and movement in the optical environment," J. Exp. Biol. **45**, 83–99 (1966).