Volume 125, Article No. 125024 (2020) https://doi.org/10.6028/jres.125.024
Journal of Research of National Institute of Standards and Technology

A Toolbox for Isophase-Curvature Guided Computation of Metrology Holograms

Ulf Griesmann[1], **Johannes A. Soons**[1], **and Gufran S. Khan**[2]

[1]National Institute of Standards and Technology,
Gaithersburg, MD 20899, USA

[2]Indian Institute of Technology Delhi,
New Delhi 110016, India

ulf.griesmann@nist.gov

Software DOI: https://doi.org/10.18434/M32177

Key words: aspheric surfaces; computer-generated hologram; contour line; diffractive optics; freeform surfaces; Hilbert phase; isophase contour; lithographic layout; optical interferometry; phase function; surface metrology.

Accepted: July 31, 2020

Published: August 13, 2020

https://doi.org/10.6028/jres.125.024

1. Summary

We describe the algorithmic foundations of an open-source numerical toolbox, written in the Octave language [1], for the creation of computer-generated binary and multi-level holograms used in interferometric form error measurements of complex aspheric and free-form precision surfaces and wavefronts. In a typical measurement setup for this type of surface, a hologram is used to generate a test wavefront that has the design shape of the surface, which is then compared to a fabricated part using an imaging laser interferometer. The optical function of the hologram in the measurement is generally modeled with optical ray-tracing software and it can be encapsulated by a scalar optical phase function $\phi : \mathbb{R}^2 \to \mathbb{R}$. The toolbox converts phase functions into equivalent binary holograms that generate the desired test wavefronts for an interferometric form error measurement. The algorithms in this toolbox take advantage of the relationship between the local properties of phase functions and the local geometry (curvature) of isophase lines. It forms the core of an efficient algorithm for the computation of optical holograms. Holograms are created in a format that can be processed by most laser- or e-beam lithography systems. While the toolbox is chiefly aimed at the creation of hologram layouts needed for measurements of precision surfaces and wavefronts, we show that the isophase-following algorithm is easily extended to phase functions with singularities and discontinuities. Such phase functions result in holograms with zone bifurcation and they can be used to generate helical wavefronts. Light beams with helical wavefronts have applications beyond surface and wavefront metrology. The toolbox also includes a family of functions for the efficient estimation and evaluation of Zernike polynomials, which are widely used in optical applications.

How to cite this article:
Griesmann U, Soons JA, and Khan GS (2020) A Toolbox for Isophase-Curvature Guided Computation of Metrology Holograms. *J Res Natl Inst Stan* 125:125024. https://doi.org/10.6028/jres.125.024.

2. Software Specifications

NIST Operating Unit	Physical Measurement Laboratory, Sensor Science Division, Surfaces and Interfaces Group
Category	Computer-generated holograms and diffractive optics
Targeted Users	Researchers in optics and metrologists in the optics industry, who need to design and fabricate optical holograms for metrology applications and other diffractive optics.
Operating Systems	Cross-platform
Programming Language	GNU Octave 4.4 and above with the Optimization and, optionally, the Parallel and Symbolic packages installed [1, 2]. The toolbox requires additional open-source toolboxes for Boolean set algebra with planar polygons, and for the creation of lithographic layout files in GDSII format [3].
Inputs/Outputs	The user must supply a problem-specific Octave script that defines one or more optical phase functions and associated parameters needed to characterize optical performances and physical shapes of a set of diffractive optical elements. The output is a lithographic layout of a diffractive optical element, or hologram, capable of generating the desired optical phase distributions, or wavefront, when illuminated with coherent light. The layout is generated in a format suitable for submission to a fabrication facility. Layout files can be inspected with viewers or editors for electronics design layouts, such as the open-source software KLayout [4].
Documentation	The toolbox is accompanied by a user manual and several commented example scripts, that illustrate a range of application scenarios.
Accessibility	Provided by the operating system.
Disclaimer	https://www.nist.gov/director/licensing

3. Background

Traditional optical manufacturing has perfected the fabrication of imaging systems consisting of optical elements with spherical surfaces together with the development of optics design methods that sought to wrest excellent imaging performance from optical lenses and systems constructed from spherical optical elements with imperfect imaging properties [5, 6]. It has been understood since the construction of early telescopes that aspheric lens and mirror surfaces enable imaging system designs with fewer surfaces and yet far better performance, but the challenge of fabricating aspheric surfaces has tended to restrict their use to high-value applications, for example in astronomical imaging systems (telescopes), or a in certain class of earth-observing satellites. The introduction of computer-controlled fabrication systems into the manufacturing of precision optics, the availability of ever more processing power to computer-aided optical design, and the emergence of digital interferometry has catalyzed rapid advances in the design and fabrication of optical elements and imaging systems over the past three decades. Reduced fabrication costs have enabled the wide adoption of aspheric, and even free-form [7], optical elements to take advantage of

better performance, reduced stray light, and compact lens sizes, of aspheric designs. Aspheric lens designs are increasingly found even in consumer optics. A striking example is the remarkable imaging performance of cameras in mobile electronic devices, which contain sophisticated aspheric imaging objectives in a volume of only about 100 mm^3 [8].

Computer-generated holograms (CGHs) are an indispensable tool for interferometric form error measurements of ultra-precise non-spherical surfaces that require low measurement uncertainty [9–13]. CGHs can be used for surface form metrology in several different interferometer configurations. The measurement configuration sketched in Fig. 1 is common because it can be implemented easily with unmodified commercial Fizeau interferometers. In a Fizeau interferometer, coherent light from a nearly monochromatic laser is emitted by a point source, P in Fig. 1 (in practice often a slightly extended source to reduce the spatial coherence, which attenuates the deleterious effects of coherent stray light), and collimated by a lens or mirror (C). The beam is expanded by a lens pair (E), and a Fizeau objective, or transmission sphere, creates a converging beam, or sometimes a diverging beam, with a spherical wavefront (F). The hologram is placed into this test beam to generate the desired non-spherical test wavefront, typically in the first diffraction order. At the correct distance from the hologram the wavefront matches the surface of the part under test (S). All rays impinging on the test surface are normal to the surface, when the test part is free of form errors. Light reflected by the spherical reference surface of the Fizeau objective (R), and the light reflected by the test surface (S), is directed by a beam splitter towards a camera (K). An imaging system with variable magnification (Z) images the test surface onto the camera sensor. The interference fringes created by these two beams are recorded by the camera, and phase decoding techniques are used to measure their phase difference [14], which can be converted into a distance because the light wavelength is known. The measurement result is the difference between the design shape of the surface, which is encoded in the hologram, and the actual, fabricated shape of the test surface (S). The type of hologram used in this measurement setup is often called a “null-compensator” because no interferometer fringes are visible—the interferometer is “nulled”—when the surface under test has the design shape, and the test surface is correctly aligned to the hologram.

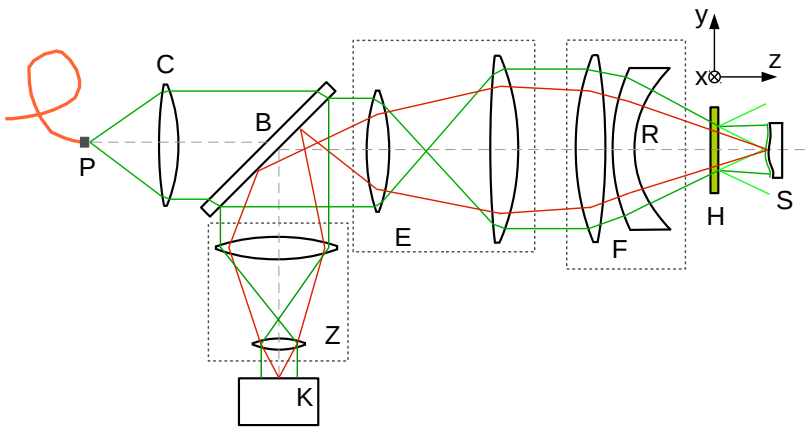

Fig. 1. Schematic of a Fizeau interferometer configured to measure the shape error of non-spherical surfaces. Components of the interferometer setup are point source (P), collimator (C), beam divider (B), beam expander (E), Fizeau objective or transmission sphere (F), zoom objective (Z), camera (K), hologram (H), spherical reference surface (R), and non-spherical surface under test (S). Illumination rays are shown in green, imaging rays in red. (Distances and angles are not drawn to scale.)

The lithographic technology needed for CGH fabrication is now widely available at research or commercial micro-fabrication foundries. However, many popular optical design software packages do not support the design of metrology holograms in lithographic layout formats (GDSII[1] [3] or OASIS [15]) that can be submitted to an open foundry. In this paper, we describe the algorithms we have developed over the past decade for the calculation of surface metrology holograms from the optical prescription of the hologram. The optical function of the hologram is calculated with optical modeling software. With the wide availability of multi-core computers, the calculation of hologram layouts is no longer the computational challenge it was merely a decade ago [16]. While our toolbox was originally developed with the goal to create improved metrology holograms at NIST, the publication of our algorithms and toolbox will enable computer-savvy optics designers and fabricators to create their own metrology holograms for precision surface form measurements. We also hope the toolbox will encourage the wider inclusion of similar algorithms and capabilities in commercial optical design software. This would make surface metrology with computer-generated holograms more widely available, and the metrology cost would be lowered.

4. Computer-Generated Holograms

The simplicity of an interferometer's (near-) point source illumination system, shown in Fig. 1, ensures that the hologram is illuminated with a monochromatic spherical wave of wavelength λ (with some spherical aberration resulting from the hologram substrate). The calculation of a hologram layout from first principles remains an enormous computational challenge and it is impractical for the large-area holograms required for metrology applications. The algorithms for hologram computation we describe here are instead based on the *Huygens-Fresnel principle* [17–21]. Despite its seeming simplicity, the Huygens-Fresnel principle enables the calculation of hologram layouts that generate test wavefronts with sufficient accuracy for form error measurements of advanced optical surfaces. The purpose of the hologram in Fig. 1 is to transform an incident spherical wave into an outgoing wave that matches the test surface after propagation through the space between hologram and test part. At the hologram, the incoming phase front must be advanced or retarded by a phase $\phi(\mathbf{x})$ ($\mathbf{x}$ is a point in the hologram plane), such that the outgoing phase front will match the shape of the test part at the correct distance. The domain of the hologram can be partitioned into half-period Fresnel zones with constant-phase boundaries that are separated in phase by π (equivalent to $\lambda/2$):

$$\phi(\mathbf{x}) = k\pi, \quad k \in \mathbb{Z}\,. \tag{1}$$

The shape of the zone boundaries depends on the shape of the test part. In the simplest case, the outgoing wave is spherical and the half-period zone boundaries will be circles. When, as in the case of the Fresnel zone plate, every other Fresnel zone is made opaque, the Huygens wavelets from the remaining transparent zones will interfere constructively to generate the desired test wavefront. Alternatively, a phase difference of π (equivalent to $\lambda/2$) can be introduced in every other zone which results in a hologram that generates the test wavefront without loss of light at the opaque zones. The type of hologram that results from the Fresnel construction is called a "binary hologram" due to the alternating zone structure. It can be viewed as a generalized Fresnel zone plate [18]. The qualitative summary given here can be put on a sound mathematical foundation within the framework of scalar diffraction theory, which also clarifies the limitations of the hologram construction outlined here [22–25]. In particular, it can be shown that the width of the half-zones must be several times larger than the wavelength of light, a condition that limits the range of allowable diffraction angles and can occasionally be difficult to meet. Computer-generated holograms are

[1] Certain commercial equipment, instruments, software, or materials are identified in this paper to foster understanding. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

generally fabricated either in the form of a patterned, opaque chromium layer on a glass substrate, which modulates the amplitude of an incoming beam such that transmitted light interferes constructively (amplitude hologram). Alternatively the hologram pattern is etched into the glass substrate to cause a phase delay of π, corresponding to $\lambda/2$, between neighboring half-zones, which enhances constructive interference (phase hologram).

The design of a metrology hologram for interferometric surface shape error measurements requires three steps. First, a phase function $\phi(\mathbf{x})$ must be found that encodes the test wavefront for a given test part. This step is typically performed using optical ray-tracing software. All optical modeling software has the option to describe the function of a CGH using an ideal, plane optical element that simply adds a phase $\phi(\mathbf{x})$ term to the optical system. In most cases, this phase function can be modeled with a suitable bivariate polynomial, for example a Zernike polynomial (see Appendix Sec. 9) or a Q-polynomial [26]. The phase function resulting from optical modeling can also compensate aberrations, for example those caused by the hologram substrate. The second step calculates the Fresnel half-zone boundaries for the phase function with the level of accuracy determined by the required form error uncertainty. Finally, the zone boundaries must be translated into a description of the zone geometry that can be processed by the fabrication system. Metrology holograms are generally fabricated with either photo-lithography or e-beam lithography. In a lithographic layout for most modern lithography systems, the Fresnel zone boundaries of a hologram must be approximated by closed polygons and stored in a layout format such as GDSII [3] or OASIS [15]. In the following sections we describe algorithms suitable for the efficient calculation of approximation polygons for phase boundary lines in Fresnel holograms *and* for the assembly of the boundary lines into closed polygons that are needed for the full description of the layout geometry.

5. Algorithms

At first glance, it may seem that the calculation of half-zone boundaries should not pose a problem. An efficient algorithm for the calculation of iso-surfaces was described decades ago [27], and its two-dimensional sibling, the marching squares algorithm for the calculation of iso-contours for functions $f : (x, y) \mapsto z$, is a staple of computer science teaching [28]. Implementations of the marching squares algorithm are available for many programming languages, including Octave [*contourc*][2]. A simple calculation, however, shows that the marching squares algorithm is not well suited for the calculation of the Fresnel half-zone boundaries defined by Eq. (1) in optics. The marching squares algorithm operates on a sampled grid of function values. For an optical hologram, the position of the polygon vertices describing Fresnel zones must have a position uncertainty well below the resolution of the lithography system. We typically use a maximum deviation of 5 nm of the polygon approximation from the isophase curve, but 10 nm would be acceptable. The spacing of isophase curves in a hologram is generally larger than the wavelength of light, usually several μm. For an average-sized circular hologram with 70 mm diameter the number of required phase function evaluations is approximately 3.8×10^{13} when the sample spacing is 10 nm. The goal of the algorithms described in the following sections is to reduce the number of required function evaluations as much as possible, because the naive application of the marching squares algorithm would result in an exorbitant number of wasted function evaluations far from the isophase curves. An additional complication is that the marching squares algorithm, at least in its common implementations, assumes the underlying function to be continuous, which is not always the case for optical phase functions.

Application specific algorithms for resource-sparing calculations of isophase curves are still required. Recently a contouring algorithm, the Pilot Approximation Trajectory (PAT) algorithm, a derivative of the

[2]References to the names of toolbox functions and other Octave functions are included in square brackets throughout the paper. The Octave help command can be used to display the location of a function in the directory tree, and the help text of the function.

marching squares algorithm, was developed for the contouring of functions with very time consuming evaluation [29]. Our algorithms similarly address the need to minimize the number of function evaluations, but unlike the PAT algorithm, they make use of the local geometry of isophase contours, which can be calculated from the local properties of the optical phase function. The algorithms are well suited to the calculation of optical holograms because phase functions are generally smooth functions that vary slowly, and derivatives can be calculated reliably. We further show in Sec. 5.5 how our geometry based algorithms can be extended to work with phase functions that have singularities and discontinuities, which occur for an important class of diffractive optics.

We begin with a brief summary of the relevant geometric relationships that are the foundation of the algorithms presented in the following sections. A curve in the plane is a function $\gamma : s \mapsto \mathbf{x}(s)$, where s is the arc length commonly used to parametrize curves [30], and $\mathbf{x}(s) = (x(s), y(s))$ is a point on the curve in the plane. An isophase curve has a constant phase along the curve:

$$\phi(\mathbf{x}(s)) = \text{const} \, . \tag{2}$$

Calculating the derivative of Eq. (2) with respect to the arc length parameter yields:

$$\frac{d}{ds}\phi(\mathbf{x}(s)) = \frac{\partial\phi}{\partial x}\frac{dx}{ds} + \frac{\partial\phi}{\partial y}\frac{dy}{ds} = \nabla\phi \cdot \frac{d\mathbf{x}(s)}{ds} = 0 \, . \tag{3}$$

The symbol $\cdot$ denotes the scalar (inner) product. The factor $d\mathbf{x}(s)/ds$ in Eq. (3) is the unit tangent vector at the isophase curve when the curve is parametrized with the arc length [30]. From Eq. (3) follows the simple but very useful conclusion that the phase gradient field, $\nabla\phi(\mathbf{x}(s))$, *is normal to isophase curves*. Equation (3) implies that a linear approximation to an isophase curve at any point on the curve can be obtained from the unit gradient vector field of the phase field $\phi(\mathbf{x})$:

$$\mathbf{n} = \frac{\nabla\phi}{\|\nabla\phi\|} \, , \tag{4}$$

because an isophase tangent vector $\mathbf{t}$ can always be found for which $\mathbf{t} \cdot \mathbf{n} = 0$. In the CGH toolbox the sign of $\mathbf{n}$ is chosen such that the exterior product $\mathbf{t} \wedge \mathbf{n} > 0$. The pair of vectors $\mathbf{t}(s)$ and $\mathbf{n}(s)$ then defines a local, right-handed coordinate system at each point of an isophase curve. A second-order approximation of an isophase curve can be obtained by considering the local curvature of the isophase. As in the case of the tangent and normal vectors, the isophase curvature $\kappa(s)$ can be expressed solely through the partial derivatives of the phase function,

$$\kappa(s) = -\frac{\phi_{xx}\phi_y^2 - 2\phi_{xy}\phi_x\phi_y + \phi_{yy}\phi_x^2}{\left(\phi_x^2 + \phi_y^2\right)^{3/2}} \, , \tag{5}$$

where the abbreviated notation ϕ_x denotes the partial derivative $\partial\phi(\mathbf{x}(s))/\partial x$, etc. [*vphderiv*]. Equation (5) is central to the algorithms described in the following sections and we provide a derivation in Appendix Sec. 8. The inverse of the curvature can be interpreted as the radius $R = 1/\kappa$ of a circle at a point $\mathbf{x}(s)$ of the isophase, the osculating circle, that shares tangent and normal vectors with the isophase curve, as shown in Fig. 2.

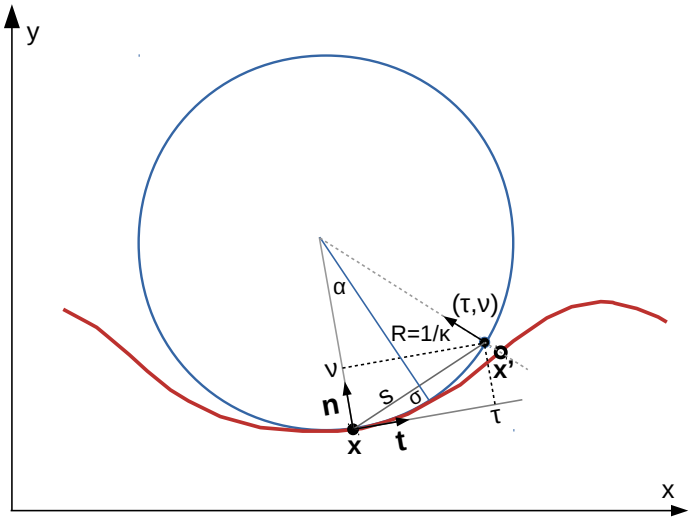

Fig. 2. Osculating circle (blue) at a point $\mathbf{x} = (x, y)$ on a planar isophase curve (red). Unit tangent and normal vectors, **t** and **n**, define local coordinate frames with coordinates (τ, ν) at each point of the isophase. At point **x** of the curve the isophase curvature κ is positive.

5.1 Domain Tilings

For the algorithms described here, the first step is to subdivide the hologram's domain into a set of rectangular tiles [*polytile*]. This approach has several advantages. Since the hologram in each tile area can be calculated independently of other tiles, the hologram's computation can take advantage of the multi-processor hardware that is now commonplace. The final layout of a hologram is described by a set of closed planar polygons. Subdividing the hologram domain is also an efficient way to ensure that the number of polygon vertices does not exceed the maximum number of vertices permitted in the layout files that are used by lithography tools (8192 in the case of GDSII); smaller tiles directly result in shorter polygons. Subdividing the hologram's domain can also be used to avoid exceptional points within a tile, which can result in closed or intersecting isophase curves. When an exceptional point of the phase function is detected in the interior of a tile, the tile is subdivided such that the exceptional point is located on the dividing line. Phase functions with singularities are handled similarly to ensure that phase singularities only occur on the edge of a tile. Every isophase curve then has exactly two tile edge intersections. The tiling also permits the intersections of the isophase curves with the tile edge to be ordered according to their distance from the tile origin along the edge, and isophase approximation polygons can be oriented such that their initial vertices are closer to the tile origin than their terminal vertices. Tile intersection ordering and polygon orientation is important for the isophase fill algorithms described in Sec. 5.4.

Figure 3 shows the example of a tiling for a circular hologram domain with a 20 mm diameter and a central circular opening with 4 mm diameter. The width of the square tiles shown is 2 mm. The lower left corner of a tile is considered the tile origin. Each tile has four edges that circumscribe the tile in counter-clockwise direction, where adjacent edges are orthogonal. Once a hologram is calculated for all tile areas, any part of the hologram extending beyond the domain boundaries is removed ("clipped") using Boolean set algebra functions [*polybool*].

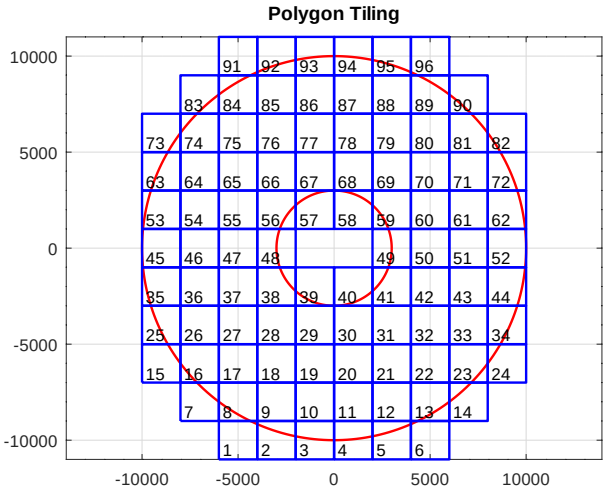

Fig. 3. An example of a tiling with square tiles (blue lines) covering the domain of an annular hologram [*polytile*]. Two polygons (red circles) define the outer and inner boundaries of the domain. Horizontal and vertical dimensions in this example are given in μm.

5.2 Pilot Approximation

Octave is an environment for numerical computing. It provides several building blocks to implement efficient algorithms for calculating optical isophase curves. We first describe an algorithm we call the pilot approximation algorithm [*phase2cgh_tile_pa*], that draws on the marching squares contouring algorithm [28] [*contourc*] and on an implementation of an algorithm for finding roots of continuous functions [*vfzero*] [31].

Our algorithm is similar to the Pilot Approximation Trajectory (PAT) contouring algorithm [29]. It uses the marching squares algorithm to first calculate a coarse approximation, the pilot approximation, of the isophase curves on a tile. The phase function is typically sampled on a grid with a resolution between 100×100 and 500×500 samples and the coarse contours are calculated for this grid of phase values using the built-in *contourc* function [*isophase_approx*]. The vertices of the approximate isophase polygons will generally not be located on the isophase curve with the desired vertex position tolerance (typically 0.1 nm). Improved vertex positions are found by searching for the isophase at each vertex of the approximate isophase polygon in a direction normal to the polygon [*vfmatch*]. The result is a set of isophase polygons P_k that have phase values Φ_k where distances between the vertices and the true isophase curve are less than a predefined tolerance. Unless the curvature of the isophase curves is very low, the number of polygon vertices in the pilot approximation is not sufficient to represent the isophase without exceeding the acceptable deviation of the approximation polygon from the isophase curve, and additional vertices must be added to the polygon. The number of additional vertices required between the vertices of the pilot approximation polygon can be estimated using the osculating circle approximation of the isophase described in Sec. 5. Using Eq. (5) we calculate the isophase curvature κ at the midpoint between two vertices of the pilot approximation polygon. From the radius of curvature, $R = 1/\kappa$, and the allowable deviation σ of the isophase from the polygon segment, an allowable segment length

$$S = 2\sqrt{\sigma\,(2R - \sigma)} \tag{6}$$

can be calculated by Eq. (6) (see Fig. 2). The number of additional vertices then follows from the ratio of the vertex distances in the pilot approximation polygon and the maximum allowable segment lengths.

Additional vertices are placed on the line between vertices of the approximation polygon at equal distances and the exact positions are calculated by searching for the location with the correct phase value in the direction normal to the polygon segment [*vfmatch*]. In a final step, the intersections of the isophase polygons with the tile edge are calculated [*isophase_clip*]. The intersections are the initial and terminal vertices of isophase polygons traversing a tile. The steps of the pilot approximation algorithm are summarized in Alg. 1. In the implementation of the algorithm [*isophase_refine*] the inner loops in Alg. 1 are replaced by vectorized expressions to speed up the computation. Once the isophase polygons over a tile are calculated, they are assembled into closed polygons using the algorithm described in Sec. 5.4.

Algorithm 1: Structure of the pilot approximation (pa) algorithm.

Input : A phase function ϕ, phase function parameters, phase levels $\{\Phi_k\}$, tile definition, sampling grid, CGH tolerances.

Output: A set of polygons $\{P_k\}$ approximating isophase curves on a rectangular tile. Each polygon is a sequence of vertex points, $P_k = \{\mathbf{x}_{k,1}, \mathbf{x}_{k,2}, \mathbf{x}_{k,3}, \dots\}$.

$\{\tilde{P}_k, \Phi_k \leftarrow$ contourc `#approximate isophase polygons on coarse grid`
for $k \leftarrow 1$ **to** $\#\{\tilde{P}_k\}$ **do**
  for $i \leftarrow 1$ **to** $\#\tilde{P}_k$ **do**
    Find $\mathbf{x}_{k,i}$ such that $\tilde{\mathbf{x}}_{k,i} \to \mathbf{x}_{k,i}$ and $\phi(\tilde{\mathbf{x}}_{k,i}) \to \Phi_k$
  end
  for $i \leftarrow 1$ **to** $\#\tilde{P}_k - 1$ **do**
    $\mathbf{D}_i \leftarrow \mathbf{x}_{k,i+1} - \mathbf{x}_{k,i}$ `#vertex distance`
    $\mathbf{m}_i \leftarrow \mathbf{x}_{k,i} + \frac{1}{2}\mathbf{D}_i$ `#vertex midpoint`
    $S_i \leftarrow 2\sqrt{\sigma\,(2/\kappa(\mathbf{m}_i) - \sigma)}$ `#approximate allowable segment length`
    $n_i \leftarrow \lceil \|\mathbf{D}_i\|/S_i \rceil - 1$ `#number of additional vertices`
    for $j \leftarrow 1$ **to** n_i **do**
      $\mathbf{x}'_{i,j} \leftarrow \mathbf{x}_{k,i} + j\,(\mathbf{D}_i/\|\mathbf{D}_i\|)$
      Find $\mathbf{x}_{i,j}$ such that $\mathbf{x}'_{i,j} \to \mathbf{x}_{i,j}$ and $\phi(\mathbf{x}'_{i,j}) \to \Phi_k$
    end
  end
  Replace $\mathbf{x}_{k,1}$ and $\mathbf{x}_{k,\mathrm{end}}$ with tile edge intersections
end

5.3 Isophase Following

The algorithm described in Sec. 5.2 works well for many phase functions that are encountered in precision surface metrology, but the reliance on the marching squares algorithm [*contourc*] results in limitations. One prominent example is the Hilbert phase function

$$\phi_H(\mathbf{x}, p) = \arg\left[\exp\left(ip \arctan\left(\frac{y - y_0}{x - x_0}\right)\right)\right], \tag{7}$$

which has a phase singularity at (x_0, y_0) and, for $p = 1$, a phase jump of 2π on the half-line $\{\mathbf{x} | x < x_0, y = y_0\}$, when it is implemented using the *atan2* function [*phase_hilbert*]. The factor $p \in \mathbb{N}$, also called "topological charge", determines the angular slope of the phase and the number of lines with discontinuous phase [32–34]. The pilot approximation algorithm described in Sec. 5.2 fails with phase

Algorithm 2: Structure of the isophase following (fi) algorithm.

Input : A phase function ϕ, phase function parameters, phase levels $\{\Phi_k\}$, tile definition, CGH tolerances.

Output: A set of polygons $\{P_k\}$ approximating isophase curves on a rectangular tile. Each polygon is a sequence of vertex points, $P_k = \{\mathbf{x}_{k,1}, \mathbf{x}_{k,2}, \mathbf{x}_{k,3}, \dots\}$.

for edge $\leftarrow 1$ **to** 4 **do**
  Find all $\mathbf{x}_k^{\text{edge}}$ such that $\phi(\mathbf{x}_k^{\text{edge}}) \rightarrow \Phi_k$
end
$\{\mathbf{x}_k^{1-4}\} \leftarrow \text{sort}\left(\{\mathbf{x}_k^{1-4}\}\right)$
repeat
  $\mathbf{x}^{\text{curr}} \leftarrow \{\mathbf{x}_k^{1-4}\}$ `#next unconnected edge intersection`
  $i \leftarrow 1$
  repeat
    $\mathbf{x}^{\text{pred}} \leftarrow \mathbf{x}^{\text{curr}} + \tau\mathbf{t} + \nu\mathbf{n}$ `#next predicted vertex`
    $i \leftarrow i + 1$
    Find $\mathbf{x}_i$ such that $\mathbf{x}^{\text{pred}} \rightarrow \mathbf{x}_i$, and $\phi(\mathbf{x}_i) \rightarrow \Phi_k$ `#isophase polygon vertex i`
    $\mathbf{x}^{\text{curr}} \leftarrow \mathbf{x}_i$
  until $\mathbf{x}_i$ is outside the tile
  Replace $\mathbf{x}_i$ with the nearest edge intersection
  $P_k \leftarrow \{\mathbf{x}_i\}$
until All edge intersections are connected

functions containing Hilbert terms shown in Eq. (7). Later in this section we describe an alternative algorithm, the isophase following algorithm, that can be extended to work with discontinuous phase functions such as those containing Hilbert phase terms.

Instead of calculating a coarse approximation of the isophase curves, the isophase following algorithm proceeds from calculating the isophase curve–tile edge intersections on all four edges of a tile [*isophase_edges*]. The intersections can be ordered on the tile edge according to their distance from the tile origin (lower left tile corner) [*edge_sort*]. Starting at the first tile intersection the approximate location of the first isophase polygon vertex inside the tile can be estimated using the osculating circle approximation of the isophase curve as shown in Fig. 2 [*predict_xy*]. For a point on the isophase curve, the unit normal vectors $\mathbf{t}$ and $\mathbf{n}$, and the isophase curvature $\kappa = 1/R$ can be calculated using Eqs. (3) and (5). For an allowable deviation σ of the polygon segment from the isophase curve, the polygon segment length S can be calculated with Eq. (6), and the coordinates (τ, ν) of the next polygon vertex (approximating $\mathbf{x}'$ in Fig. 2) in the $\mathbf{t}, \mathbf{n}$ coordinate frame can be estimated as

$$\begin{aligned} \tau &= S\sqrt{1 - (S/2R)^2} \\ \nu &= \frac{S^2}{2R}\,. \end{aligned} \tag{8}$$

The Frenet equations [30] (also see Appendix Sec.9),

$$\frac{d\mathbf{t}}{ds} = \kappa\mathbf{n}\,, \quad \text{and} \quad \frac{d\mathbf{n}}{ds} = -\kappa\mathbf{t}\,, \tag{9}$$

which describe the evolution of the $\mathbf{t}, \mathbf{n}$ coordinate frame along the curve, can be used to estimate the

isophase normal vector $\mathbf{n}_p$ at the predicted vertex. When the osculating circle arc length from $\mathbf{x}$ to (τ, ν)

$$L = 2R \arcsin\left(\frac{S}{2R}\right) \tag{10}$$

is used to estimate the isophase curve length in the second of the Frenet equations in Eq. (9), the normal vector $\mathbf{n}'$ at $\mathbf{x}'$ can be approximated by

$$\mathbf{n}_p = \mathbf{n} - \kappa L \mathbf{t} \,. \tag{11}$$

The exact vertex position $\mathbf{x}'$ on the isophase curve (open circle in Fig. 2) is again obtained using a level finding algorithm [*vfmatch*] in the direction of $\mathbf{n}_p$. This process is repeated until the isophase polygon exits the tile area. The intersection of the last polygon segment with the tile edge is then replaced with the closest isophase–tile edge intersections initially calculated. The initial and terminal tile edge intersections are removed from the the list of unconnected edge intersections. The calculation of the next isophase polygon proceeds from the next available unconnected edge intersection until all intersections are connected by isophase polygons [*isophase_curves*]. A summary of the isophase following algorithm is shown in Alg. 2.

The inner loop of the isophase following algorithm, Alg. 2, is again implemented using the vector idiom of the Octave language to achieve adequate processing speed [*isophase_curves*]. This need to "vectorize" the computation creates an additional problem. The isophase following procedure must proceed from only half of the isophase–tile edge intersections, and the start intersections must be chosen such that no isophase curve is traced in duplicate. The algorithm for the selection of start intersections is shown in Alg. 3 [*isophase_vertex_init*]. It is predicated on the assumption that tiles were designed such that all isophase curves have two tile edge intersections and isophase curves never cross.

Algorithm 3: Selection of isophase polygon start vertices.

Input : Phase levels $\{\Phi_k\}$ of the ordered isophase–tile edge intersections.
Output: A set of indices $\{m\}$ of the selected polygon start intersections.

Partition $\{\Phi_k\}$ into equivalence classes $\check{\Phi}$ and $\hat{\Phi}$ containing lower and upper Fresnel zone boundaries
Add intersection $k = 1$ to start intersections
isi $\leftarrow$ **true**
for $k \leftarrow 2$ **to** $\#\{\Phi_k\}$ **do**
  if $(\Phi_k \in \check{\Phi} \wedge \Phi_{k-1} \in \check{\Phi}) \vee (\Phi_k \in \hat{\Phi} \wedge \Phi_{k-1} \in \hat{\Phi})$ **then**
    isi $\leftarrow \sim$isi
  end
  if isi **then**
    Add k to start intersections
  end
end

5.4 Isophase Filling

The geometry of a hologram layout must be described by a set of closed polygons for most common fabrication systems that use the GDSII or OASIS [15] layout description standards. Isophase curves must be connected into closed polygons that circumscribe areas with phase values corresponding to opaque (or phase-retarded) Fresnel zones, as described in Sec. 4. An algorithm for the filling (or shading) of adjacent isophase curves [*isophase_fill*] is illustrated in Fig. 4. The polygons approximating isophase curves can be

oriented such that their initial vertices are closer to the tile origin than their terminal vertices as shown in Fig. 4. The algorithm picks an unconnected polygon (e.g., the polygon that is marked with red arrows and starts on edge 1) and then looks up the neighboring isophase intersections of the end vertex of the polygon in negative, $\mathbf{x}^-$, and positive, $\mathbf{x}^+$, edge direction. The phase values on the line between one of the neighbors will be in the correct phase range of the Fresnel zone, and the polygon that starts or ends on this neighboring intersection must be connected to the first polygon (see red arrows). The end of the combined polygons again has two neighbors. When the neighbor in the negative direction is the start vertex of the original polygon, a closed polygon is complete. Otherwise, another polygon needs to be connected to the already connected polygons until the condition for a closed polygon is met (see blue arrows in Fig. 4). When a polygon traverses a tile corner, it is also possible that the negative neighbor of a polygon is its own start vertex and the polygon is then connected to itself when the phase on the edge connection is in the desired phase range. This situation is illustrated with the green arrows in Fig. 4. In the CGH toolbox, the loop in Alg. 4 is implemented through recursion.

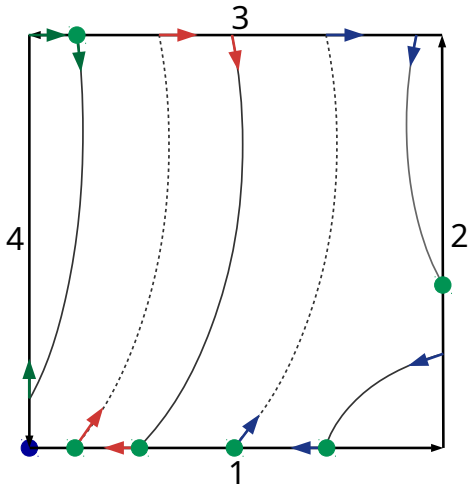

Fig. 4. Polygons approximating isophase curves crossing a tile area. Dashed isophase curves indicate the boundary with the lowest phase value in a Fresnel zone, solid lines the boundary with the highest phase value. The first and last vertices of each polygon are tile edge crossings. Green dots indicate the location of the first (start) vertex of a polygon. The blue dot denotes the tile origin. Arrows indicate paths for the assembly of isophase or phase boundary lines into closed polygons.

5.5 Discontinuous Phase Functions

The isophase following algorithm of Sec. 5.3 can be extended to phase functions with discontinuities and phase singularities. Examples are the phase functions with the Hilbert terms of Eq. (7), which will be discussed in detail in this section. Hilbert phase terms generate helical optical wavefronts characterized by an angular momentum distinct from the spin angular momentum related to polarization [34–36]. Helical beams are finding a growing number of applications in advanced imaging systems [37–39].

The isophase following algorithm relies on the derivatives of a phase function to calculate isophase normal vectors and isophase curvatures. While the derivatives exist everywhere except at the phase singularity in Eq. (7), conventional methods for numerical differentiation fail at the discontinuities. This problem can be overcome by considering the complex *continuous* function

$$Z(\mathbf{x}) = e^{i\phi(\mathbf{x})} = \cos\phi(\mathbf{x}) + i\sin\phi(\mathbf{x})\ . \tag{12}$$

Algorithm 4: Structure of the isophase filling algorithm.

Input : A set of polygons $\{P_k\}$ approximating isophase curves on a rectangular tile. Each polygon is a sequence of vertex points, $P_k = \{\mathbf{x}_{k,1}, \mathbf{x}_{k,2}, \mathbf{x}_{k,3}, \dots\}$.

Output: A set of closed polygons circumscribing Fresnel zones with phase values within the specified phase interval.

```
repeat
    Get next unconnected polygon P_u
    loop
        Look up tile edge neighbor x^- of the terminal vertex x_{u,end} of P_u
        if x^- = x_{u,1} then
            if Phase of any edge point between x_{u,end} and x_{u,1} is in phase interval then
                P_u ← Connect polygon containing vertex x^- to P_u
                Connect ends of P_u and add to closed output polygons
                Mark polygon P_u as connected
                break
            end
        end
        Look up tile edge neighbor x^+ of the terminal vertex x_{u,end} of P_u
        if Phase of any edge point between x_{u,end} and x^- is in phase interval then
            P_u ← Connect polygon containing vertex x^- to P_u
        else if Phase of any edge point between x_{u,end} and x^+ is in phase interval then
            P_u ← Connect polygon containing vertex x^+ to P_u
        else
            Error condition
        end
        Mark polygon connected to P_u as connected
    end
until All polygons are connected
```

Expressions for the derivatives of discontinuous phase functions suitable for numerical differentiation can be derived by calculating the partial derivatives of $Z(\mathbf{x})$ in Eq. (12).

$$\phi_x = -i\frac{Z_x}{Z}, \quad \text{and} \quad \phi_y = -i\frac{Z_y}{Z}\,, \tag{13}$$

because $Z(\mathbf{x})$ is continuous and its partial derivatives can be calculated at or near the discontinuities of $\phi(\mathbf{x})$ using standard methods for numerical differentiation (the abbreviated notation for partial derivatives introduced for Eq. (5) is also used in this section). The second numerical derivatives of $\phi(\mathbf{x})$ can similarly be calculated by using the derivatives of $Z(\mathbf{x})$ [*vwphderiv*],

$$\phi_{xx} = -i\left(\frac{Z_{xx}}{Z} + \phi_x^2\right), \quad \phi_{xy} = -i\left(\frac{Z_{xy}}{Z} + \phi_x\phi_y\right), \quad \text{and} \quad \phi_{yy} = -i\left(\frac{Z_{yy}}{Z} + \phi_y^2\right)\,, \tag{14}$$

where ϕ_x and ϕ_y are calculated using Eq. (13). Using Eqs. (13) and (14) the isophase following algorithm in Sec. 5.3 can be adapted for the calculation of holograms with discontinuous phase functions simply by replacing the method of numerical differentiation. Only one additional change is required. Fresnel zone boundaries are no longer lines of constant phase because the phase changes by 2π every time a zone

boundary crosses a discontinuity. The implementation of the boundary following algorithm [*boundary_curves*] must test for the presence of discontinuities and adjust the phase whenever a zone boundary is crossed. The algorithm for determining the boundary polygon start vertices [*boundary_vertex_init*] takes into account that a phase singularity with undefined phase may be present on one of the tile edges. Similarly the boundary filling algorithm [*boundary_fill*] is expanded to handle the $2p$ zone boundaries that meet at a singularity. First, boundaries are connected at the singularity, then boundary filling proceeds as usual with the algorithm described in Sec. 5.4.

6. Methods for Validation

The functionality of the toolbox was validated in several ways. For elementary cases, such as a rotationally symmetric Fresnel zone lens illuminated by a collimated beam, the hologram zone radii can be derived analytically [17, 19] and can be used to confirm the zone radii calculated by the toolbox from the phase function of the Fresnel lens. Holograms can be validated locally because the gradient of a phase function, $\nabla\phi(\mathbf{x})$, is related to the local grating pitch $p(\mathbf{x})$, the width of a Fresnel zone pair at $\mathbf{x}$,

$$\|\nabla\phi(\mathbf{x})\| \cong \frac{2\pi}{p(\mathbf{x})}\,, \tag{15}$$

because the phase change corresponding to the zone pair is 2π. The relationship in Eq. (15) was used for spot checks of holograms either after the layout is calculated or after fabrication of the hologram by comparing the zone widths to the width that is expected from the phase gradient. Zone widths of finished holograms can be measured with a microscope. Finally, the toolbox was validated through the use of holograms that were made with the toolbox in a variety of interferometric tests and measurements. Some of these measurements are described in Refs. [40–45].

7. An Example

The user manual for the toolbox [46] describes several applications of varying complexity. Here we illustrate the toolbox with the example of a Fresnel zone lens with an additional Hilbert phase term. A Fresnel zone lens that brings a collimated beam of light with wavelength λ to a focus at a distance f from the zone lens (in first diffraction order) has the spherical phase function

$$\phi_F(\mathbf{x}) = \frac{4\pi}{\lambda}\left(\sqrt{f^2 + \|\mathbf{x}-\mathbf{x}_0\|^2} - f\right). \tag{16}$$

$\mathbf{x}_0$ is the center coordinate of the phase function. In this example we place the Fresnel phase function at the coordinate origin, $\mathbf{x}_0 = \mathbf{0}$, and assume a focal distance f of 100 mm and a wavelength of 0.632 82 µm. The phase distribution of $\phi_F + \phi_H$ with an added Hilbert term ϕ_H in Eq. (7) with topological charge $p = 2$ at an offset (0 mm, 0.8 mm) is shown in Fig. 5(a). The binary hologram that generates this phase front when illuminated with collimated light is shown in Fig. 5(b). The boundaries between dark and light zones are lines with phase values $3\pi/4 + k\pi$, $k \in \mathbb{Z}$. Fig. 5 illustrates the characteristic forking of Fresnel zones into $p+1$ zones at the singularity of a Hilbert phase term.

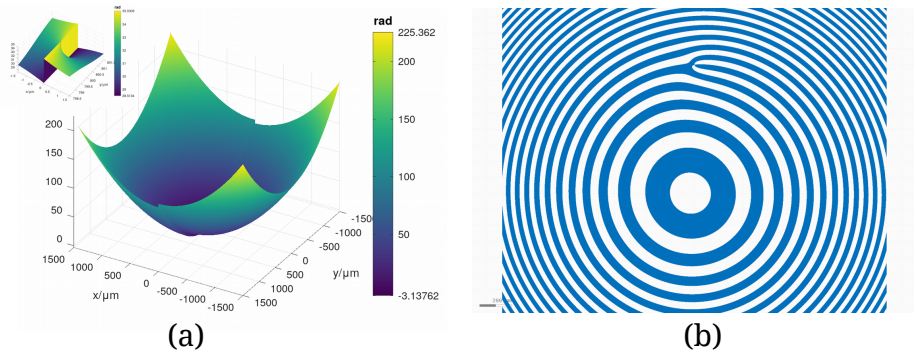

Fig. 5. (a) Central 3 mm × 3 mm domain of a phase function consisting of a Fresnel term centered at (0 mm, 0 mm) and a Hilbert term with topological charge $p = 2$ centered at (0 mm, 0.8 mm). The inset shows a 3 µm × 3 µm patch centered at the singularity of the Hilbert term. (b) The resulting binary hologram calculated with the phase boundary following algorithm described in Sec. 5.5. The hologram shows the zone forking characteristic of phase functions with Hilbert phase terms.

8. Appendix: Isophase Curvature

Equation (5) relates the isophase curvature to the partial derivatives of a scalar phase field. In this section we give an elementary derivation of this connection, because it is central to the algorithms described in this paper. Consider a circle that tangentially touches an isophase curve at a location $\mathbf{x}(s)$ on the curve and intersects the curve at $\mathbf{x}'(s')$ (also see Fig. 2). Two lines normal to the isophase curve can be defined at $\mathbf{x}$ and $\mathbf{x}'$,

$$\begin{aligned} \lambda &\mapsto \mathbf{x} + \lambda \mathbf{n} \\ \mu &\mapsto \mathbf{x}' + \mu \mathbf{n}' \,, \end{aligned} \tag{17}$$

where $\lambda, \mu \in \mathbb{R}$. $\mathbf{n}$ is the unit normal vector at $\mathbf{x}$, and $\mathbf{n}'$ the unit normal vector at $\mathbf{x}'$. The two lines intersect where

$$\mathbf{x} + \lambda \mathbf{n} = \mathbf{x}' + \mu \mathbf{n}' \,. \tag{18}$$

A solution for the parameter μ is found when Eq. (18) is multiplied by $\mathbf{t}$, the unit tangent vector at $\mathbf{x}$,

$$\mu = -\frac{\mathbf{t} \cdot (\mathbf{x}' - \mathbf{x})}{\mathbf{t} \cdot \mathbf{n}'} \,, \tag{19}$$

because $\mathbf{t} \cdot \mathbf{n} = 0$. When $\mathbf{x}'$ is moved along the isophase curve towards $\mathbf{x}$, the circle becomes the osculating circle of the isophase curve, shown in Fig. 2, at position $\mathbf{x}$ in the limit $\mathbf{x}' \to \mathbf{x}$. In the limit, the lines in Eq. (17) intersect at the center of the circle. Since $\mathbf{n}$ is a unit vector, μ must be the radius, R, of the circle. $\mathbf{x}'$ and $\mathbf{n}'$ in Eq. (19) can be linearized in the vicinity of $\mathbf{x}$. For a small arc distance δs from $\mathbf{x}$, $\mathbf{x}'$ and $\mathbf{n}'$ can be written as linear expressions in δs,

$$\begin{aligned} \mathbf{x}' &= \mathbf{x} + \mathbf{t}\,\delta s + O(\delta s^2) \\ \mathbf{n}' &= \mathbf{n} + \frac{d\mathbf{n}}{ds}\delta s + O(\delta s^2) \,. \end{aligned} \tag{20}$$

Replacing $\mathbf{x}'$ and $\mathbf{n}'$ in Eq. (19) with their linearized forms in Eqs. (20), and taking the limit $\delta s \to 0$, results in an equation for the curvature $\kappa = 1/R$ of the osculating circle (and the isophase curve) at $\mathbf{x}$:

$$\kappa = -\frac{d\mathbf{n}}{ds} \cdot \mathbf{t} \,. \tag{21}$$

Equation (21) also follows from the second of the Frenet equations in Eqs. (9) when it is multiplied with **t**. The derivative of the normal vector along a curve in Eq. (21),

$$\frac{d\mathbf{n}}{ds} = \frac{\partial \mathbf{n}}{\partial x}\frac{dx}{ds} + \frac{\partial \mathbf{n}}{\partial y}\frac{dy}{ds} \,, \tag{22}$$

can be related to a phase function ϕ through the definition of the unit normal vector in Eq. (4),

$$\mathbf{n} = \frac{\nabla\phi}{\|\nabla\phi\|} = \mathcal{N}\nabla\phi \,, \quad \text{where} \quad \mathcal{N} = \left(\phi_x^2 + \phi_y^2\right)^{-\frac{1}{2}} \,. \tag{23}$$

For both components n_k ($k = 1, 2$) of **n**, Eq. (22) has the form

$$\frac{dn_k}{ds} = \nabla n_k \cdot \frac{d\mathbf{x}}{ds} = \nabla n_k \cdot \mathbf{t} \tag{24}$$

(numerical indices are used to distinguish the normal vector components from derivatives). Applying ∇ to each of the components n_k produces

$$\begin{aligned} \nabla n_1 &= \nabla(\mathcal{N}\phi_x) = \mathcal{N}\nabla\phi_x + \phi_x\nabla\mathcal{N} \\ \nabla n_2 &= \nabla(\mathcal{N}\phi_y) = \mathcal{N}\nabla\phi_y + \phi_y\nabla\mathcal{N} \,, \end{aligned} \tag{25}$$

which, when multiplied with **t** according to Eq. (24), and written in vectorial form again, gives

$$\begin{bmatrix} \nabla n_1 \cdot \mathbf{t} \\ \nabla n_2 \cdot \mathbf{t} \end{bmatrix} = \mathcal{N} \begin{bmatrix} \nabla\phi_x \cdot \mathbf{t} \\ \nabla\phi_y \cdot \mathbf{t} \end{bmatrix} + (\nabla\mathcal{N} \cdot \mathbf{t}) \begin{bmatrix} \phi_x \\ \phi_y \end{bmatrix} \,. \tag{26}$$

The curvature κ is finally obtained, when this expression for $d\mathbf{n}/ds$ is applied in Eq. (21):

$$\kappa = -\mathcal{N} \begin{bmatrix} \nabla\phi_x \cdot \mathbf{t} \\ \nabla\phi_y \cdot \mathbf{t} \end{bmatrix} \cdot \mathbf{t} \,. \tag{27}$$

The second term in Eq. (26) does not contribute to the curvature because it is a vector proportional to $\nabla\phi$ and thus orthogonal to **t** (Eq. 3). From the orthogonality of **t** and **n** it follows further that the unit tangent vector **t** can be written as

$$\mathbf{t} = \mathcal{N} \begin{bmatrix} \phi_y \\ -\phi_x \end{bmatrix} \,, \tag{28}$$

where $\mathcal{N}$ is the same normalization factor defined in Eq. (23) for the normal vector **n**. With Eq. (28) the sought after expression for the isophase curvature κ in Eq. (5) is obtained

$$\begin{aligned} \kappa &= -\mathcal{N}^2 \begin{bmatrix} \phi_{xx}\phi_y - \phi_{xy}\phi_x \\ \phi_{xy}\phi_y - \phi_{yy}\phi_x \end{bmatrix} \cdot \mathbf{t} \\ &= -\mathcal{N}^3 \left(\phi_{xx}\phi_y^2 - 2\phi_{xy}\phi_x\phi_y + \phi_{yy}\phi_x^2\right) \\ &= -\frac{\phi_{xx}\phi_y^2 - 2\phi_{xy}\phi_x\phi_y + \phi_{yy}\phi_x^2}{\left(\phi_x^2 + \phi_y^2\right)^{3/2}} \,. \end{aligned} \tag{29}$$

Interested readers can find a less pedestrian derivation by Goldman [47], that places Eq. (29) in the context of differential geometry.

9. Appendix: Zernike Polynomials

The CGH toolbox includes a suite of functions for the evaluation, estimation, conversion, and transformation of Zernike polynomials. Zernike polynomials are a complete set of orthogonal polynomials on a circular disk. They provide an efficient representation of arbitrary wavefront aberrations over a circular pupil. Various definitions exist for Zernike polynomials that differ in the coordinate system used, parameter numbering scheme, and normalization. Our implementation of Zernike polynomials is based on the definition described in the ANSI Z80.28-2017 standard [48, 49], which has the following elements:

- The Zernike polynomials are defined in terms of polar coordinates (ρ, θ) in the XY plane of a right-handed coordinate system. The normalized radial coordinate ρ ranges from 0 to 1, although the implementation in the toolbox permits values $\rho > 1$. The azimuth angle θ is defined relative to the positive X-axis and increases in counter-clockwise direction. Each Zernike term is the product of three components: a normalization factor, a polynomial defined on the radius ρ, and a sinusoid defined on the azimuth angle θ.
- Each Zernike term, except the offset (piston) term, is normalized so that its variance equals 1 on the unit disk. The Zernike terms are orthonormal.
- Each Zernike term is identified by two indices n and m. The radial order n is the degree (highest power) of the radial polynomial. The angular order m is the azimuthal frequency of the angular term. The latter term equals $\sin(|m|\theta)$ for $m < 0$ and $\cos(m\theta)$ for $m >= 0$. For each radial order n, the valid values for m equal $-n, -n+2, \cdots, n-2, n$. A Zernike term can also be identified by a single index $j = (n(n+2)+m)/2$.

Following this convention, a Zernike polynomial with coefficients a_n^m is defined as:

$$Z(\rho,\theta) = \sum_{n=0}^{k} \sum_{m=-n}^{n} a_n^m Z_n^m(\rho,\theta) \,, \tag{30}$$

with Zernike terms

$$Z_n^m(\rho,\theta) = N_n^m R_n^m(\rho) \begin{cases} \sin(|m|\theta) & \text{if } m < 0 \\ \cos(m\theta) & \text{if } m \geqslant 0 \end{cases} . \tag{31}$$

The radial polynomial term $R_n^m(\rho)$ is given by:

$$R_n^m(\rho) = \begin{cases} \sum_{k=0}^{(n-|m|)/2} \frac{(-1)^k (n-k)!}{k!\left(\frac{n+|m|}{2}-k\right)!\left(\frac{n-|m|}{2}-k\right)!} \rho^{n-2k} & \text{if } n-|m| \text{ is even} \\ 0 & \text{if } n-|m| \text{ is odd} \end{cases} , \tag{32}$$

and the normalization factor

$$N_n^m = \sqrt{\frac{2(n+1)}{1+\delta_m}} \,, \tag{33}$$

where δ_m is 1 for $m = 0$, and 0 otherwise. The first fifteen Zernike polynomial terms are summarized in Tables 1 and 2.

Table 1. Zernike indices j for the first fifteen Zernike terms according to the ANSI, Optical Shop Testing 2nd edition (OST), Noll, and Fringe or University of Arizona (UoA) term ordering conventions [*zern_index_trans*]. The symbol $(-)$ indicates that the respective Zernike term is negated. The normalized terms Z_n^m in column 7 and their names are those found in the ANSI standard. Note that the OST and Fringe Zernike terms are not normalized.

n	m	ANSI [48]	OST [50]	Noll [51]	Fringe [52]	Z_n^m, see Eq. (31)	Common name [48]
0	0	0	1	1	0	1	piston
1	−1	1	3	3	2	$2\rho\sin(\theta)$	vertical tilt
1	1	2	2	2	1	$2\rho\cos(\theta)$	horizontal tilt
2	−2	3	4	5	5	$\sqrt{6}\rho^2\sin(2\theta)$	oblique astigmatism
2	0	4	5	4	3	$\sqrt{3}(2\rho^2-1)$	defocus (power)
2	2	5	$(-)6$	6	4	$\sqrt{6}\rho^2\cos(2\theta)$	astigmatism
3	−3	6	$(-)10$	9	10	$\sqrt{8}\rho^3\sin(3\theta)$	oblique trefoil
3	−1	7	9	7	7	$\sqrt{8}(3\rho^3-2\rho)\sin(\theta)$	vertical coma
3	1	8	8	8	6	$\sqrt{8}(3\rho^3-2\rho)\cos(\theta)$	horizontal coma
3	3	9	$(-)7$	10	9	$\sqrt{8}\rho^3\cos(3\theta)$	horizontal trefoil
4	−4	10	$(-)11$	15	17	$\sqrt{10}\rho^4\sin(4\theta)$	
4	−2	11	12	13	12	$\sqrt{10}(4\rho^4-3\rho^2)\sin(2\theta)$	secondary oblique ast.
4	0	12	13	11	8	$\sqrt{5}(6\rho^4-6\rho^2+1)$	primary spherical
4	2	13	$(-)14$	12	11	$\sqrt{10}(4\rho^4-3\rho^2)\cos(2\theta)$	secondary astigmatism
4	4	14	15	14	16	$\sqrt{10}\rho^4\cos(4\theta)$	

Table 2. The first sixteen Zernike terms Z_n^m and their index values j according to ANSI, Optical Shop Testing 2nd edition (OST), Noll, and Fringe (UoA) term ordering conventions. Note that the OST and Fringe (UoA) Zernike terms are not normalized.

j	ANSI [48]	OST [50]	Noll [51]	Fringe [52]
0	Z_0^0			Z_0^0
1	Z_1^{-1}	Z_0^0	Z_0^0	Z_1^1
2	Z_1^1	Z_1^1	Z_1^1	Z_1^{-1}
3	Z_2^{-2}	Z_1^{-1}	Z_1^{-1}	Z_2^0
4	Z_2^0	Z_2^{-2}	Z_2^0	Z_2^2
5	Z_2^2	Z_2^0	Z_2^{-2}	Z_2^{-2}
6	Z_3^{-3}	$-Z_2^2$	Z_2^2	Z_3^1
7	Z_3^{-1}	$-Z_3^3$	Z_3^{-1}	Z_3^{-1}
8	Z_3^1	Z_3^1	Z_3^1	Z_4^0
9	Z_3^3	Z_3^{-1}	Z_3^{-3}	Z_3^3
10	Z_4^{-4}	$-Z_3^{-3}$	Z_3^3	Z_3^{-3}
11	Z_4^{-2}	$-Z_4^{-4}$	Z_4^0	Z_4^2
12	Z_4^0	Z_4^{-2}	Z_4^2	Z_4^{-2}
13	Z_4^2	Z_4^0	Z_4^{-2}	Z_5^1
14	Z_4^4	$-Z_4^2$	Z_4^4	Z_5^{-1}
15	Z_5^{-5}	Z_4^4	Z_4^{-4}	Z_6^0

The Zernike polynomial related functions of the CGH toolbox can be grouped into the following categories:

- **Evaluation** of a Zernike polynomial [*zern_eval*]. In our implementation, we use the modified Kintner method [53, 54] for the fast and stable evaluation of the radial polynomial terms $R_n^m(\rho)$ using a recursion relation. Zernike terms can be calculated and evaluated in their symbolic form in both polar and cartesian coordinates [*zern_symbolic*]. This functionality requires installation of Octave's Symbolic package [2].
- **Estimation** of a Zernike coefficient vector from wavefront aberration data [*zern_estim*]. The estimation minimizes the least squares error and can be performed with or without the assumption of orthogonality.
- **Conversion** of a Zernike coefficient vector from one definition to another [*zern_index_trans*]. The implemented definitions are
  - (n,m) index pairs defined in the ANSI Z80.28-2017 standard.
  - Zernike term index j defined in the ANSI Z80.28-2017 standard. Starting with an index value of 0, a polynomial with a lower value of n is ordered first. For equal values of n, a polynomial with a lower value of m is ordered first.
  - Zernike indices and Zernike polynomials defined in the second edition of Optical Shop Testing [50]. This publication uses a similar index numbering approach to the ANSI Z80.28-2017 standard. However, the Zernike terms are not normalized, i.e., their values on the unit circle are in the interval $[-1,1]$. The lowest index value for j is 1. Furthermore, the azimuth angle θ' is defined relative to the positive Y-axis and increases in the clockwise direction. The sinusoidal term for an angular order l is $\sin(l\theta')$ for $l > 0$ and $\cos(l\theta')$ for $l <= 0$. Finally, for the same radial order n, cos terms have a lower Zernike index number than sin terms.
  - indices and Zernike polynomials defined by Noll [51] and also described in the third edition of Optical Shop Testing [55]. Here polynomials are defined using the same normalization and azimuth angle as the ANSI Z80.28-2017 standard. However, for a given Z_n^m term, the equivalent index j is defined in a different manner. Starting with $j = 1$, a polynomial with a lower value of n is ordered first. For equal values of n, a polynomial with a lower value of the azimuthal frequency $|m|$ is ordered first. An even j corresponds to a $\cos(m\theta)$ angular term whereas an odd j corresponds to a $\sin(|m|\theta)$ angular term.
  - "Fringe" indices and Zernike polynomials as defined by Loomis [52, 56]. Here the azimuth angle definition is the same as that of the ANSI Z80.28-2017 standard. The Zernike terms are not normalized. This term ordering scheme is inspired by the needs of optics fabricators and groups terms according to their $n+|m|$ values. Thus, the first group $n+|m| = 2$ contains the paraxial wavefront properties. The second group $n+|m| = 4$ contains the third order aberrations. The third group $n+|m| = 6$ contains the fifth order aberrations, etc. Starting with $j = 0$, a polynomial with a lower value of $n+|m|$ is ordered first. For polynomials with the same $n+|m|$ value, a polynomial with a higher value of the azimuthal frequency $|m|$ is ordered first. Finally, $\cos(m\theta)$ terms are ordered before $\sin(|m|\theta)$. The toolbox also contains a conversion for Zernike coefficient vectors that follow this definition, but where the angle is defined relative to the positive Y-axis in clockwise direction.
- **Transformation** of a Zernike coefficient vector to describe a mirror operation, scaling, rotation, or translation [57] [*zern_transform*]. These functions can also be used to obtain the Zernike coefficient vector of a circular sub-aperture.

Acknowledgments

We are indebted to our former NIST colleague Dr. Quandou Wang, now with Facebook, Inc., for the fabrication of numerous metrology holograms that were created using earlier versions of the toolbox described in this paper. We gratefully acknowledge the many years of support by the Center for Nanoscale Science and Technology (CNST) at NIST through the tools, processes, and expertise provided by the center.

10. References

[1] Eaton JW, *et al* (1996-2019) GNU Octave: A high-level interactive language for numerical computations. Available at https://octave.org
[2] Octave Forge: Extra packages for GNU Octave. Available at https://octave.sourceforge.io
[3] Griesmann U (2008-2019) An Octave and Matlab toolbox for layout files in GDSII format. Available at https://github.com/ulfgri/gdsii-toolbox
[4] Köfferlein M (2006-2019) KLayout - High performance layout viewer and editor. Available at https://www.klayout.de
[5] Twyman F (1988) *Prism and lens making* (Adam Hilger/IOP Publishing) 2nd Ed.
[6] Kingslake R (1978) *Lens Design Fundamentals* (Academic Press) 1st Ed.
[7] Fang FZ, Zhang XD, Weckenmann A, Zhang GX, Evans CJ (2013) Manufacturing and measurement of freeform optics. *CIRP Annals–Manufacturing Technology* 62:823–846. https://doi.org/10.1016/j.cirp.2013.05.003
[8] Steinich T, Blahnik V (2012) Optical design of camera optics for mobile phones. *Advanced Optical Technologies* 1:51–58. https://doi.org/10.1515/aot-2012-0002
[9] Pruß C, Garbusi E, Osten W (2008) Testing aspheres. *Optics and Photonics News* 19:25–29. https://doi.org/10.1364/OPN.19.4.000024
[10] Creath K, Wyant JC (1995) Use of computer-generated holograms in optical testing. *Handbook of Optics*, ed Bass M (McGraw-Hill Professional, New York, NY) Vol. II, 2nd Ed., pp 31.1–31.11.
[11] Burge JH (1995) Applications of computer-generated holograms for interferometric measurement of large aspheric optics. *International Conference on Optical Fabrication and Testing*, ed Kasai T International Society for Optics and Photonics (SPIE), Vol. 2576, pp 258–269. https://doi.org/10.1117/12.215609
[12] Fercher A (1976) Computer-generated holograms for testing optical elements: Error analysis and error compensation. *Optica Acta: International Journal of Optics* 23(5):347–365. https://doi.org/10.1080/713819270
[13] MacGovern AJ, Wyant JC (1971) Computer generated holograms for testing optical elements. *Applied Optics* 10(3):619–624. https://doi.org/10.1364/AO.10.000619
[14] Burke J (2012) Phase decoding and reconstruction. *Optical Methods for Solid Mechanics: A Full-Field Approach*, eds Rastogi P, Hack E (Wiley-VCH Verlag, Weinheim) 1st Ed., pp 83–139.
[15] SEMI (2016) P39–Specification for OASIS - Open Artwork System Interchange Standard. Available at www.semi.org
[16] Schnabel B, Günther K, Eichhorn H, Gramß J, Beyer D, Kley EB, Rockstroh W Fast data preparation for high-quality and large-area computer generated hologram fabrication. *Microelectronic Engineering* 73-74:80–84. https://doi.org/10.1016/j.mee.2004.02.020
[17] Jenkins FA, White HE (1957) *Fundamentals of Optics, Ch. 18* (McGraw-Hill, New York, NY) 3rd Ed. pp 353–381.
[18] Ferrier WG (1969) The zone plate and its role in holography. *Contemporary Physics* 10:413–436. https://doi.org/10.1080/00107516908204795
[19] Born M, Wolf E (1999) *Principles of Optics, Ch. 8* (Cambridge University Press) 7th Ed. pp 412–516.
[20] Kress BC, Meyrueis P (2000) *Digital diffractive optics* (John Wiley & Sons, Chichester) 1st Ed. pp 275–306.
[21] Kazanskiy NL, Kotlyar VV, Soifer VA (2002) Wave front correction. *Methods for computer design of diffractive optical elements*, ed Soifer VA (John Wiley & Sons), pp 607–649.
[22] Harvey JE (1979) Fourier treatment of near-field scalar diffraction theory. *American Journal of Physics* 47:974–980. https://doi.org/10.1119/1.11600
[23] O'Shea D, Suleski TJ, Kathman AD, Prather DW (2003) *Diffractive Optics: Design, Fabrication, and Test* (SPIE Press) 1st Ed. Vol. TT62.
[24] Goodman JW (2005) *Introduction to Fourier Optics, Chs. 3-4* (Roberts & Company) 3rd Ed.
[25] Kress BC, Meyrueis P (2009) *Applied Digital Optics: from Micro-optics to Nanophotonics* (John Wiley & Sons) 1st Ed.
[26] Forbes GW (2012) Characterizing the shape of freeform optics. *Optics Express* 20:2483–2499. https://doi.org/10.1364/OE.20.002483
[27] Lorensen WE, Cline HE (1987) Marching cubes: A high resolution 3d surface construction algorithm. *ACM SIGGRAPH Computer Graphics* 21:163–169. https://doi.org/10.1145/37401.37422
[28] Wenger R (2013) *Isosurfaces: Geometry, Topology, and Algorithms* (CRC Press) 1st Ed.

[29] Huttunen JM, Stark PB (2012) Cheap contouring of costly functions: the Pilot Approximation Trajectory algorithm. *Computational Science and Discovery* 5:015006. https://doi.org/10.1088/1749-4699/5/1/015006
[30] do Carmo MP (1976) *Differential Geometry of Curves and Surfaces* (Prentice Hall) 1st Ed.
[31] Alefeld GE, Potra FA, Shi Y (1995) Algorithm 748: Enclosing zeros of continuous functions. *ACM Transactions on Mathematical Software* 21:327–344. https://doi.org/10.1145/210089.210111
[32] Khonina SN, Kotlyar VV, Shinkaryev MV, Soifer VA, Uspleniev GV (1992) The phase rotator filter. *Journal of Modern Optics* 39:1147–1154. https://doi.org/10.1080/09500349214551151
[33] Heckenberg NR, McDuff R, Smith CP, White AG (1992) Generation of optical phase singularities by computer-generated holograms. *Optics Letters* 17(3):221–223. https://doi.org/10.1364/OL.17.000221
[34] Berry MV (2004) Optical vortices evolving from helicoidal integer and fractional phase steps. *Journal of Optics A: Pure and Applied Optics* 6:259–268. https://doi.org/10.1088/1464-4258/6/2/018
[35] Larkin KG (2001) Natural demodulation of two-dimensional fringe patterns II. stationary phase analysis of the spiral phase quadrature transform. *Journal of the Optical Society of America A* 18:1871–1881. https://doi.org/10.1364/JOSAA.18.001871
[36] Padgett M, Courtial J, Allen L (2004) Light's orbital angular momentum. *Physics Today* 57:35–40. https://doi.org/10.1063/1.1768672
[37] Sakdinawat A, Liu Y (2007) Soft-x-ray microscopy using spiral zone plates. *Optics Letters* 32:2635–2637. https://doi.org/10.1364/OL.32.002635
[38] Maurer C, Barnet S, Ritsch-Marte M (2009) Refining common path interferometry with a spiral phase Fourier filter. *Journal of Optics A: Pure and Applied Optics* 11:1–7. https://doi.org/10.1088/1464-4258/11/9/094023
[39] Larkin KG, Bone DJ, Oldfield MA (2001) Natural demodulation of two-dimensional fringe patterns I. general background of the spiral phase quadrature transform. *Journal of the Optical Society of America* 18:1862–1870. https://doi.org/10.1364/JOSAA.18.001862
[40] Gao G, Lehan JP, Zhang WW, Griesmann U, Soons JA (2009) Computer-generated hologram cavity interferometry test for large x-ray mirror mandrels: design. *Optical Engineering* 48:063602–1–10. https://doi.org/10.1117/1.3153303
[41] Wang Q, Griesmann U, Burnett JH (2011) Binary amplitude holograms made from dyed photoresist. *Optics Letters* 36(10):1899–1901. https://doi.org/10.1364/OL.36.001899
[42] Wang Q, Soons JA, Griesmann U (2013) Holographic radius test plates. *Optical Manufacturing and Testing X*, eds Fähnle OW, Williamson R, Kim DW International Society for Optics and Photonics (SPIE), Vol. 8838, pp 131–137. https://doi.org/10.1117/12.2024890
[43] Griesmann U, Soons JA, Wang Q, Assoufid L (2013) Figure metrology for x-ray focusing mirrors with fresnel holograms and photon sieves. *Renewable Energy and the Environment: Postdeadline Papers* (Optical Society of America), p FW2B.5. https://doi.org/10.1364/FREEFORM.2013.FW2B.5
[44] Gong Q, Content DA, Dominguez M, Emmett T, Griesmann U, Hagopian J, Kruk J, Marx C, Pasquale B, Wallace T, Whipple A (2016) Wide-Field InfraRed Survey Telescope (WFIRST) slitless spectrometer: design, prototype, and results. *Space Telescopes and Instrumentation 2016: Optical, Infrared, and Millimeter Wave*, eds MacEwen HA, Fazio GG, Lystrup M, Batalha N, Siegler N, Tong EC International Society for Optics and Photonics (SPIE), Vol. 9904, pp 403–420. https://doi.org/10.1117/12.2231665
[45] Dominguez MZ, Marx CT, Gong Q, Hagopian JG, Griesmann U, Burge JH, Kim DW (2018) Infrared computer-generated holograms: design and application for the WFIRST grism using wavelength-tuning interferometry. *Optical Engineering* 57(7):1–10. https://doi.org/10.1117/1.OE.57.7.074105
[46] Griesmann U (2019) A toolbox for isophase-curvature guided computation of metrology holograms: User Manual. Available at https://doi.org/10.18434/M32177
[47] Goldman RN (2005) Curvature formulas for implicit curves and surfaces. *Computer Aided Geometric Design* 22:632–658. https://doi.org/10.1016/j.cagd.2005.06.005
[48] The Accredited Committee Z80 for Ophthalmic Standards (2017) *ANSI Z80.28-2017–American National Standard for Ophthalmics: Methods for Reporting Optical Aberrations of Eyes* (American National Standards Institute, New York, NY).
[49] Thibos LN, Applegate RA, Schwiegerling JT, Webb R, Members OVST (2002) Standards for reporting the optical aberrations of eyes. *Journal of Refractive Surgery* 18:S652–S660. https://doi.org/10.3928/1081-597X-20020901-30
[50] Malacara D (1992) *Optical Shop Testing* (John Wiley & Sons) 2nd Ed. p 466.
[51] Noll RJ (1976) Zernike polynomials and atmospheric turbulence. *Journal of the Optical Society of America* 66(3):207–211. https://doi.org/10.1364/JOSA.66.000207
[52] Loomis J (1978) A computer program for analysis of interferometric data. *Optical Interferograms–Reduction and Interpretation*, eds Guenther A, Liebenberg D (ASTM International, West Conshohocken, PA), pp 86–1978. https://doi.org/10.1520/STP33971S
[53] Chong CW, Raveendran P, Mukundan R (2003) A comparative analysis of algorithms for fast computation of zernike moments. *Pattern Recognition* 36(3):731–742. https://doi.org/10.1016/S0031-3203(02)00091-2
[54] Singh C, Walia E (2010) Fast and numerically stable methods for the computation of zernike moments. *Pattern Recognition* 43(7):2497–2506. https://doi.org/10.1016/j.patcog.2010.02.005
[55] Malacara D (2007) *Optical Shop Testing* (John Wiley & Sons) 3rd Ed. p 517.
[56] Wyant JC, Creath K (1992) Basic wavefront aberration theory for optical metrology. *Applied Optics and Optical Engineering*, eds

Shannon RR, Wyant JC (Academic Press) Vol. XI, pp 2–53.
[57] Lundström L, Unsbo P (2007) Transformation of zernike coefficients: scaled, translated, and rotated wavefronts with circular and elliptical pupils. *Journal of the Optical Society of America A* 24(3):569–577. https://doi.org/10.1364/JOSAA.24.000569

About the authors: *Ulf Griesmann is a physicist with the Surfaces and Interfaces Group within the Sensor Science Division at NIST. His primary research focus is the development of metrology methods for optical precision surfaces and wavefronts using diffractive optics. Johannes A. Soons is a mechanical engineer in the Surfaces and Interfaces Group with an interest in developing computational methods to solve problems in precision engineering and metrology. Gufran S. Khan is a professor at the Instrument Design & Development Centre of the Indian Institute of Technology Delhi, India. His primary research areas are fabrication and metrology methods for optical precision surfaces. Our joint work on optical metrology holograms began during an extended research visit by GSK as a NIST associate.The National Institute of Standards and Technology is an agency of the U.S. Department of Commerce.*