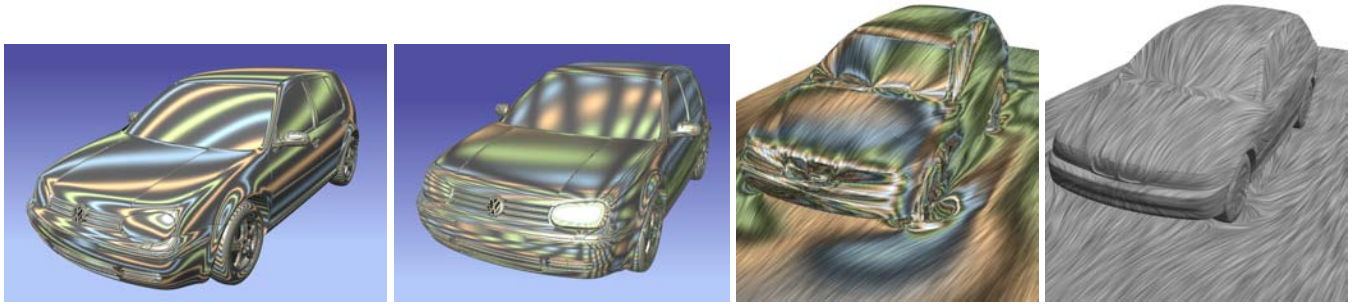
Cycle Shading for the Assessment and Visualization of Shape in One and Two Codimensions

Daniel Weiskopf
Simon Fraser University
GrUVi Lab, Burnaby, Canada
weiskopf@cs.sfu.ca

Helwig Hauser
VRVis Research Center
Vienna, Austria
Hauser@VRVis.at



(a) (b) (c) (d)

Figure 1: Surface rendering with cycle shading (a) and hatched cycle shading (b); flow visualization with hatched cycle shading for streamlines (c) and surface-based Phong illumination (d). Data sets courtesy of Volkswagen AG (two left images) and the BMW Group (two right images).

ABSTRACT

In this paper we propose cycle shading and hatched cycle shading as new local shading techniques for shape assessment and visualization. Natural surface highlights are extended to not only appear in isolated parts of a surface, but to reappear throughout the surface in a regular and easy-to-control pattern. Thereby even small surface variations become visible, wherever they are located on the surface. We further extend (hatched) cycle shading to curves in 3D, i.e., to shapes of higher codimension. We demonstrate how (hatched) cycle shading improves 3D vector field visualization by showing higher-order discontinuities of streamlines, pathlines, or streaklines. Our visualization approach is generic, simple, efficient, and can readily be used where Phong illumination is applicable because information on curvature or mesh connectivity is not required. The effectiveness of cycle shading for the assessment of surface quality is demonstrated by a user study. Finally, this paper addresses issues of anti-aliasing, parameter control, applications, and efficient GPU implementations.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation; I.3.7 [Computer Graphics]: 3D Graphics and Realism

Keywords: Surface assessment, shape visualization, vector field visualization, non-photorealistic rendering, curve rendering, GPU methods.

1 INTRODUCTION

Surfaces rule the world of perception—human beings are very effective in perceiving a 3D environment by visually "reading" the appearance of surfaces [9]. Both spatial and temporal variations in the shading and texturing of a surface strongly support 3D shape perception. In computer graphics, the importance of surface shading is well understood—without reasonable shading, surfaces appear to be flat and unstructured. Surface highlights, in particular, play an important role in shape perception [7]. In automotive engineering, for example, reflection lines (specular highlights from a set of light tubes) are used to assess the quality of car bodies [6]. In addition to shading, appropriate surface texturing (i.e., substructuring) can be used to support shape perception [8, 23, 26]; in particular, textures based on surface curvature can be helpful [10, 14, 17].

In this paper, we follow up this line of argumentation and propose a new shading technique that might be useful for shape visualization. The motivation for our technique ties in with the human ability to mentally reconstruct 3D shape from specular highlights and surface texture, and in particular from their temporal variations. We demonstrate by means of a user study (Section 8) that our shading method supports the user in detecting surface imperfections, i.e., our method is effective in visualizing and assessing the quality aspect of shape. The perceptual effectiveness in promoting the full understanding of 3D shape is not investigated in this paper but remains an open question for future work.

For our shading approach, we substructure view-dependent surface shading to distribute highlight-like information throughout the surface—in contrast to Phong highlighting [21], which only affects relatively small parts of the surface. We denote this approach as *cycle shading* because it results in concentric shading patterns

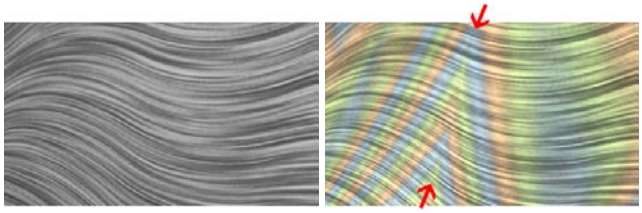


Figure 2: Flow visualization with dense streamlines: surface shading (left), illuminated streamlines with cycle shading (right). The red arrows denote usually invisible G^2 curve discontinuities.

around regular specular highlights and because it introduces cyclic secondary highlight structures when the angle between view direction and direction of perfect reflection is increasing. Cycle shading is interesting for existing and future applications as it can be readily used wherever Phong illumination is applicable; only surface normals are required, but no information on curvature or mesh connectivity. Due to its high sensitivity to surface smoothness, cycle shading is particularly useful for the quality assessment of surfaces. Figure 1a shows a typical application of shape interrogation for CAD models. In addition, cycle shading might be employed for non-photorealistic surface illustrations (see Section 7 and Figures 9a and 9b), although further investigations are needed to understand the effectiveness of cycle shading in the context of 3D shape perception.

With cycle shading we can successfully increase the texture frequency along *one* dimension across the surface: radially from the surface points of perfect reflection in the sense of a polar coordinate system. With *hatched cycle shading* (see Figure 1b), we additionally increase the texture frequency across the concentric shading patterns and thereby cover the complete 2D surface with shape information in *both* dimensions. Therefore, even subtle surface features such as tiny bumps or curvature changes become visible, regardless of their orientation or position on the surface.

Due to its simplicity, (hatched) cycle shading can be generalized for objects of higher codimensions, such as curves in 3D space. In Figures 1c and 2, for example, the tangential flow across a surface is visualized with a dense set of streamlines. With cycle shading, first-order derivative information of the curves is directly visualized instead of zero-order positional information (see, e.g., Figure 2, right): we can see a G^2 curve discontinuity that is invisible with traditional shading. Therefore, we propose (hatched) cycle shading as a method for shape visualization and assessment in higher codimensions.

The main contributions of this paper are (1) a surface illumination model for (hatched) cycle shading along with (2) a generalization to higher codimensions, (3) an anti-aliasing solution comparable to MIPmapping, (4) an efficient GPU implementation, (5) a demonstration of typical application scenarios, and (6) a user study that validates the effectiveness of cycle shading for the assessment of surface quality.

2 Related Work

Cycle shading is a non-photorealistic extension of the Phong lighting model [21] with the goal of shape visualization and assessment. It is related to cool/warm shading [12], with the difference that specular illumination instead of diffuse illumination is extended. This side-by-side position with respect to the joint Phong reference model allows us to consider cool/warm shading as a role model for cycle shading. In general, much research has been conducted in the field of non-photorealistic shape visualization [13, 25]. For example, silhouette and contour lines [16, 3, and references therein] or feature lines such as ridges are used as effective visual cues. Principal curvature directions and principal curvatures can guide the placement of strokes [10, 14, 17]. Curvature information can also be visualized directly by color coding [4] or indirectly via the Gauss map [19]. The computation of curvature information for discrete surfaces is a topic of ongoing research [11].

Another related field of research is geometric surface assessment. One approach is surface interrogation as used in geometric design to examine surface quality [15]. For example, reflection lines [18], which are the specular reflections of elongated light sources, have a long tradition in surface assessment. Real-time reflection mapping [28] can be used to compute respective images. Another related method is based on isophotes, i.e., surface curves with a constant angle between light direction and normal vector [22]. Isophotes and reflection lines are complementary, but not disjoint [27], and they form a discrete counterpart to cycle shading.

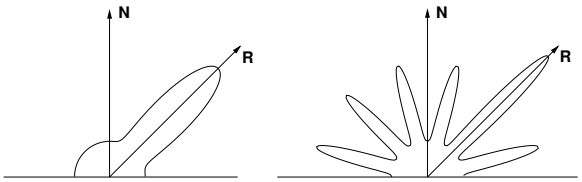


Figure 3: Comparison of BRDFs for incident light at 45 degrees: Phong model (left) and cycle shading (right).

Finally, our generalization of cycle shading to arbitrary dimension and codimension follows the extension of the Phong illumination model by Banks [1] and the computational methods for illuminated streamlines [32].

3 Cycle Shading

The original Phong model [21] can be split into ambient, diffuse, and specular components to compute the reflected radiance

$$L_{\text{Phong}} = (k_a + k_d(\mathbf{L}\cdot\mathbf{N}) + k_s(\mathbf{R}\cdot\mathbf{V})^{n_s})L_{\text{light}} \quad ,$$

with the ambient coefficient k_a, the diffuse coefficient k_d, the specular coefficient k_s, shininess n_s, and the radiance L_{light} incident from the light source. In this paper, we use a convention in which the light direction vector $\mathbf{L}$, the normal vector $\mathbf{N}$, the reflection vector $\mathbf{R}$, and the viewing vector $\mathbf{V}$ point away from the position of incidence on the surface. All these vectors are assumed to be normalized to unit length. One-sided lighting is usually applied, i.e., the diffuse and specular contributions are set to zero for surfaces that are not oriented towards the light source. The above equation is formulated for a single color channel; colored images are produced by applying the illumination model for the different color channels separately. In the remainder of this paper, we focus on the specular term $\sigma_c = (\mathbf{R}\cdot\mathbf{V})^{n_s}$, which can be written as

$$\sigma_c = f_c(\mathbf{R}\cdot\mathbf{V}) \quad , \tag{1}$$

with $f_c(d) = d^{n_s}$. We generalize this illumination model to *cycle shading* by changing f_c from a monotonic mapping to a function that exhibits recurring minima and maxima. Figure 3 illustrates the BRDFs (bidirectional reflection distribution functions) for Phong illumination and cycle shading.

A special family of functions f_c is required to achieve an approximately equidistant spacing between neighboring maxima of f_c on the image plane. To obtain equidistant visual structures, a mapping via arccosine is necessary,

$$\sigma_c = f_c(\mathbf{R}\cdot\mathbf{V}) = g_c(\arccos(\mathbf{R}\cdot\mathbf{V})) \quad , \tag{2}$$

provided g_c has equidistant maxima and minima. Essentially, the arccosine compensates for the fact that the dot product yields the cosine of the angle between two vectors. Figure 4 illustrates the effect of the arccosine mapping for a shaded sphere.

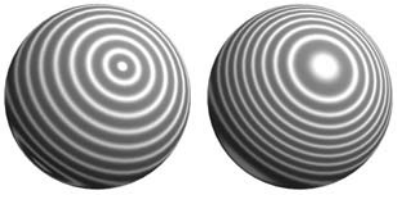


Figure 4: Cycle shading for a sphere, with arccosine mapping (left) and without arccosine mapping (right).

The mapping function g_c is of the type

$$g_c : [0,\pi] \to \mathbb{R}, \quad \alpha \longmapsto \sum_{i=0}^{i_{\max}} w_i p(\alpha - i\Delta\alpha) \quad . \tag{3}$$

The prototypical function p has compact support (like a triangle hat function) or quasi-compact support (like a Gaussian function) and describes the behavior around a single maximum. The size of the support determines the width of cycle patterns. Equidistant copies of p are added at intervals $\Delta\alpha$. The width $\Delta\alpha$ controls the spacing of cycle patterns. The number of maxima is determined by $i_{\max}$, and w_i are possibly varying weights. The domain of g_c covers the full range of possible input angles and so does the support of typical choices for g_c—cycle shading provides two-sided lighting, in contrast to one-sided illumination by the Phong model. Two-sided lighting is important to introduce visual structure at all visible parts of an object. The function g_c can represent gray-scale intensities and colors alike. Color can be introduced by specifying separate and varying weights w_i for different color channels.

Cycle shading is combined with traditional surface illumination to improve shape perception. It is useful to modulate the result of cycle shading with the Lambertian contribution according to $\sigma_c' = w(\max(0, \mathbf{L}\cdot\mathbf{N}))\sigma_c$, where w is a weight function with domain $[0,1]$ and range $[0,1]$. A good choice is $w(d) = \tilde{w}d + (1-\tilde{w})$, with $\tilde{w} = 3/4$. Finally, traditional illumination can be added to the modulated value σ_c'. Figure 8 compares different illumination alternatives, all of them based on a sum of Gaussians for g_c. Figure 8a illustrates colored cycle shading without additional surface illumination, whereas Figure 8b adds Phong illumination to colored cycle shading. The alternating colors green, blue, and orange are chosen approximately isoluminant. Figure 8c shows a combination of gray-scale Phong illumination and gray-scale cycle shading. Finally, Figure 8d is based on a combination of cool/warm surface shading [12] and luminance cycle shading with additional specular highlights. We recommend using a combination of traditional illumination and cycle shading because both methods seem to be compatible and mutually strengthen their shape cues. Figure 8a is only included to show the effect of pure cycle shading, but is not recommended for real applications. The spatial frequency of patterns from cycle shading depends on the rate at which normal vectors change with respect to varying viewing angle. Therefore, cycle pattern density is a direct measure for surface curvature.

4 Hatched Cycle Shading

We propose *hatched cycle shading* as a means of generating additional visual structures that are essentially perpendicular to the patterns of cycle shading. The idea is to project the up vector (vertical axis on the image plane) and the reflection vector into a plane that is perpendicular to the view direction, and then to compute the angle between the two projected vectors. Two orthonormal basis vectors are chosen to describe the image plane: the up vector $\mathbf{U}$ and the vector $\mathbf{X}$ that points to the right. We use a notation in which $\mathbf{A}_V$ denotes the normal part of a generic vector $\mathbf{A}$ with respect to a projection to the view vector $\mathbf{V}$, i.e., $\mathbf{A}_V = \mathbf{A} - (\mathbf{A}\cdot\mathbf{V})\mathbf{V}$. Then, $\mathbf{X}_V$ and $\mathbf{U}_V$ form a basis of a 2D plane that is perpendicular to the view direction. This basis is well-defined and its basis vectors have finite length because the view vector cannot be collinear with $\mathbf{U}$ or $\mathbf{X}$ for orthographic or perspective projections. (A 4π sterad panorama camera is not considered here.) Although $\mathbf{X}_V$ and $\mathbf{U}_V$ are not necessarily normalized to unit length or perpendicular to each other, a corresponding orthonormal basis $(\mathbf{X}_V', \mathbf{U}_V')$ can always be constructed:

$$\mathbf{U}_V' = \Lambda_1(\mathbf{U}_V), \quad \mathbf{X}_V' = \Lambda_1(\mathbf{X}_V - (\mathbf{X}_V\cdot\mathbf{U}_V')\mathbf{U}_V') \quad .$$

In general, we denote by Λ_1 an operator that normalizes an arbitrary input vector $\mathbf{A}$ to unit length, i.e., $\Lambda_1(\mathbf{A}) = \mathbf{A}/||\mathbf{A}||$. Then, the reflection vector is projected onto the 2D plane spanned by $(\mathbf{X}_V', \mathbf{U}_V')$:

$$\mathbf{R}_V = (\mathbf{R}\cdot\mathbf{U}_V')\mathbf{U}_V' + (\mathbf{R}\cdot\mathbf{X}_V')\mathbf{X}_V' \quad .$$

The dot product of the projected up and reflection vectors is used to define the analog of the cycle term,

$$\sigma_h = f_h\left(\Lambda_1(\mathbf{R}_V)\cdot\mathbf{U}_V'\right) \quad . \tag{4}$$

This equation can be evaluated efficiently by using cross-product computations to project vectors onto a plane perpendicular to the viewing direction. In this way, the hatched version of cycle shading can alternatively be computed by

$$\sigma_h = f_h\left(\Lambda_1(\mathbf{V}\times\mathbf{R})\cdot\Lambda_1(\mathbf{V}\times\mathbf{U})\right) \quad . \tag{5}$$

This equation leads to the same result as Eq. (4) because the rotations by $\pi/2$ that are introduced by the cross-product terms are canceled out in the dot-product computation.

For view vectors in a neighborhood around the reflection vector, cycle shading leads to circular isolines of σ_c that are concentric around the direction of perfect reflection. In contrast, isolines of σ_h form star-shaped rays that originate from a common center point at the direction of perfect reflection. Similarly to cycle shading, f_h should contain recurring maxima and minima. To achieve equidistant visual patterns, a mapping via arccosine is required again: $f_h(d) = g_h(\arccos(d))$.

We recommend using the result of the hatched shading version only in combination with cycle shading: the hatched structures do not originate from an illumination metaphor (such as specular reflection is the basis for cycle shading), but are introduced to add structure to the cycle patterns. Hatched structures are faded out around the direction and the opposite direction of perfect reflection. The fading out is based on a weighting factor that is constructed from the first ($i = 0$) and last ($i = i_{\max}$) prototypical function for cycle shading in Eq. (3).

Figure 10 illustrates hatched cycle shading. Here, colored cycle patterns are modulated with the result of the cross-product computation from Eq. (5). The two spheres on the left are illuminated by directional light that is incident from top-right. The images of the two spheres only differ in the width of cycle maxima. The top sphere contains gray gaps between both kinds of patterns (cycle and hatched). In contrast, the bottom sphere has wide, slightly overlapping cycle maxima, which result in an almost isoluminant cycle structure. In this way, cycle shading is mainly based on chromatic contrast and the hatched patterns are based on luminance contrast. For most applications of hatched cycle shading, we recommend a separation into luminance and chromatic contrast in order to perceptually separate the cycle and hatching patterns. The role of luminance and chromatic contrast could also be exchanged so that cycle shading would be related to luminance contrast and the hatching aspect would be related to chromatic contrast.

The right image of Figure 10 illustrates hatched cycle shading for the height field of a sinusoidal function of which the amplitude decreases with increasing distance from the center. The examples in Figure 10 show that the star-shaped hatched patterns are perpendicular to the circular patterns of cycle shading in a large neighborhood around perfect reflection. Since temporal variations of highlight structures support shape perception especially well, (hatched) cycle shading is most effective in interactive applications in which the user can control viewpoint and light direction (cf. the accompanying video [29]).

At this point, the range of valid input values for the hatched version of cycle shading needs to be discussed. As mentioned before, the operations on vectors $\mathbf{U}$ and $\mathbf{X}$ are well-defined for orthographic

or perspective projections. The vectors $\mathbf{R}_V$ or, alternatively, $\mathbf{V} \times \mathbf{R}$ have zero length only at the point of perfect reflection and the point opposite of perfect reflection. This degeneracy can be neglected because hatched cycle shading only uses the well-defined highlights from cycle shading at these points.

5 Shading in Higher Codimensions

Following Banks [1], we extend cycle and hatched cycle shading to shapes of arbitrary dimension and codimension. We assume that an object of dimension $k > 0$ can be described as a k-manifold M embedded in Euclidean space of dimension $n > k$. The difference $n-k$ is the codimension of the object. The previous discussion in Sections 3 and 4 covers the case of a 2-manifold (a 2D surface) with codimension 1 (i.e., $n = 3$ and $k = 2$). Curves, hair, or fur embedded in 3D space are examples of another important class of objects: 1-manifolds with codimension 2 (i.e., $n = 3$ and $k = 1$).

Banks relies on Fermat's principle with light paths of minimal length to derive a model for diffuse and specular illumination. We use his generalization of Phong lighting to illuminate objects of any codimension. Both cycle and hatched cycle shading are primarily based on the reflection vector $\mathbf{R}$, and so is the specular term in Banks' model. We adopt his computation of the reflection vector to derive a codimension-invariant generalization of cycle and hatched cycle shading.

The n-dimensional Euclidean space is decomposed into the k-dimensional tangent space $TM_{\mathbf{p}}$ and the $(n-k)$-dimensional normal space $NM_{\mathbf{p}}$—both at the point $\mathbf{p}$ of the manifold M. The reflection vector can then be split into a tangential part $\mathbf{R}_T$ and a normal part $\mathbf{R}_N$ by projection. In general, a subscript T denotes the tangential part of a vector and a subscript N denotes the normal part. Due to Fermat's principle, the reflection vector can be computed from the light and view vectors according to

$$\mathbf{R} = \mathbf{R}_T + \mathbf{R}_N, \text{ with } \mathbf{R}_T = -\mathbf{L}_T, \ \mathbf{R}_N = ||\mathbf{L}_N|| \Lambda_1(\mathbf{V}_N) \quad . \quad (6)$$

With this reflection vector, the codimension-invariant cycle contribution $g_c(\arccos(\mathbf{R} \cdot \mathbf{V}))$ can be computed in the same way as in Eq. (2). Similarly, hatched cycle shading can be applied in higher codimensions. We still assume that the image plane is two-dimensional and, thus, can be described by two orthonormal basis vectors $\mathbf{U}$ and $\mathbf{X}$. Then, Eq. (4) is suitable for any dimension n and any codimension $(n-k)$ because it is only based on well-defined vector quantities and dot-product computations. The alternative formulation with cross-product computations according to Eq. (5) can also be applied in arbitrary codimension, but only for the special case of 3D Euclidean space $n = 3$. Only in 3D space, the cross product can be used to emulate the projection operation as used for Eq. (5). Moreover, cross-product computations (with two input vectors from one space and an output (pseudo) vector from the same space) are not defined in Euclidean space of arbitrary dimension.

Illumination of curves in 3D Euclidean space is the most important application of lighting in higher codimensions. Similarly to Zöckler et al. [32], the specular coefficient can be expressed solely in terms of the light vector $\mathbf{L}$, the view vector $\mathbf{V}$, and the tangent vector $\mathbf{T}$:

$$\mathbf{R} \cdot \mathbf{V} = -(\mathbf{L} \cdot \mathbf{T})(\mathbf{V} \cdot \mathbf{T}) + \sqrt{1-(\mathbf{L} \cdot \mathbf{T})^2}\sqrt{1-(\mathbf{V} \cdot \mathbf{T})^2} \quad . \quad (7)$$

Based on this equation, $g_c(\arccos(\mathbf{R} \cdot \mathbf{V}))$ can be computed for cycle shading. In analogy, we derive an expression for the hatched version of cycle shading that is only based on the view, up, and tangent vectors. The key element is to rewrite the term $(\mathbf{V} \times \mathbf{R})$ with the use of Eq. (6) and several algebraic transformations:

$$\begin{aligned} \mathbf{V} \times \mathbf{R} &= (\mathbf{V}_T + \mathbf{V}_N) \times (\mathbf{R}_T + \mathbf{R}_N) \\ &= \mathbf{V}_T \times \mathbf{R}_N + \mathbf{V}_N \times \mathbf{R}_T = C\,(\mathbf{V} \times \mathbf{T}) \quad , \end{aligned}$$

with the scalar-valued factor

$$C = -\frac{(\mathbf{V} \cdot \mathbf{T})||\mathbf{L}_N|| + (\mathbf{L} \cdot \mathbf{T})||\mathbf{V}_N||}{||\mathbf{V}_N||} \quad .$$

We are interested in the normalized vector, which removes the scaling factor C,

$$\Lambda_1(\mathbf{V} \times \mathbf{R}) = \alpha_{\text{sign}} \Lambda_1(\mathbf{V} \times \mathbf{T}) \quad .$$

The variable α_{sign} has value ± 1, depending on the sign of C. Assuming that the mapping function f_h is even (i.e., $f_h(-d) = f_h(d)$), the hatched version of cycle shading can be evaluated according to

$$\sigma_h = f_h\left(\Lambda_1(\mathbf{V} \times \mathbf{T}) \cdot \Lambda_1(\mathbf{V} \times \mathbf{U})\right) \quad . \quad (8)$$

Figure 11 illustrates shading in codimension 2. The illuminated curves are streamlines originating from a tangential vector field around a torus. The streamlines are generated with an image-space approach [30] for surface LIC (Line Integral Convolution), which results in images analogous to planar LIC [2]. Figure 11a shows illuminated streamlines according to Zöckler et al. [32]. Figure 11b demonstrates cycle shading based on Eq. (7). In analogy to surface rendering, the result of cycle shading is combined with the traditional illumination of streamlines. Figure 11c illustrates hatched cycle shading based on Eq. (8).

6 Anti-Aliasing

The density of cycle-shading patterns is determined by the curvature of an object as seen by the camera, i.e., by curvature with respect to image space. Therefore, the pattern density for a fixed geometry increases with increasing distance from the camera. The influence of varying shape curvature for different objects is taken into account by a user-controlled specification of cycle-shading frequency. To reduce depth-induced aliasing, we propose an approach similar to MIPmapping [31]. A hierarchy of different resolutions of g_c is built in a preprocessing step. Generic low-pass filtering would be inadequate because it would lead to blurred or even uniformly colored images without clear visual patterns. Instead, different resolutions are generated by applying Eq. (3) with different values of $\Delta\alpha$. As for MIPmapping, the resolutions of two neighboring hierarchy levels are related by a factor of two, and so are the respective values $\Delta\alpha$.

Intermediate, non-integer hierarchy levels could be generated on-the-fly by linear interpolation between two neighboring levels. Alternatively, we propose an explicit construction of intermediate non-integer levels. For intermediate levels, functions g_c of neighboring integer resolutions are interpolated linearly and, at the same time, their prototypical functions p are computed with changing widths. In this way, the prototypical functions p maintain a fixed width with respect to screen space even for non-integer levels. Figure 5 shows the plot of g_c with explicitly constructed non-integer

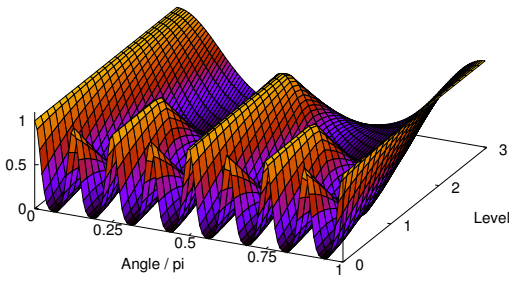


Figure 5: Two-dimensional mapping function g_c, depending on $\arccos(\mathbf{R} \cdot \mathbf{V})$ and the MIPmapping level.

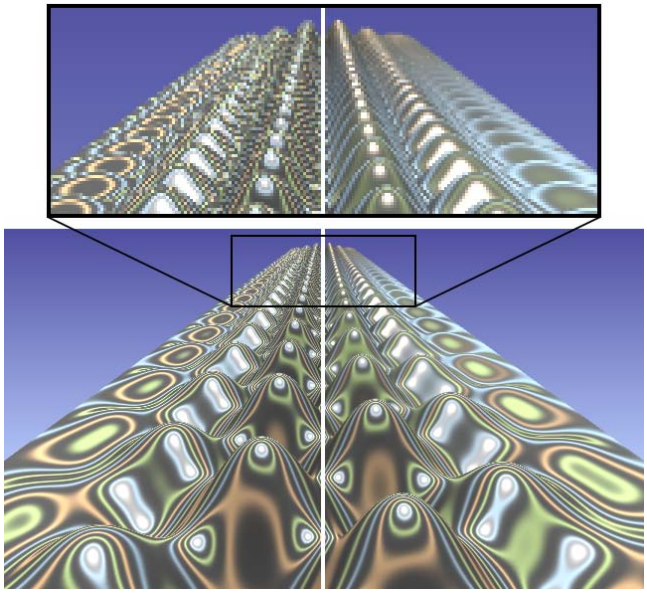


Figure 6: Anti-aliasing for cycle shading on a sinusoidal height field: without filtering (left part), with filtering (right part), and close-up views for both approaches (top image).

resolutions. Figure 6 demonstrates anti-aliasing by comparing a non-filtered version with a filtered version.

Since hatched cycle shading is subject to the same aliasing problems as cycle shading, an analogous MIPmapping method is applied, with a pre-filtering of the mapping function g_h.

7 Applications

We believe that (hatched) cycle shading can be employed primarily for shape assessment and shape visualization. This section discusses which parameter choices can be useful in these application areas in order to obtain good results from our shading approach. We think that the spacing of maxima for (hatched) cycle shading is the only crucial parameter that needs to be controlled by the user—besides control of camera and light source, which should be part of any interactive viewer. The spacing of maxima is important because it determines the density of shading patterns on the image plane and allows the user to take into account the variations in curvature between different objects.

We first discuss shape visualization as one proposed application area. Figures 9a and 9b demonstrate the combination of cycle shading with non-photorealistic rendering techniques. Figure 9a shows an image with chromatic cycle shading and additional silhouette lines, while Figure 9b uses luminance cycle shading in combination with cool/warm shading and silhouette lines. Figure 9c shows an example of a volume illustration; other aspects of non-photorealistic volume rendering are discussed by Ebert and Rheingans [5]. This volume rendering is directly generated from a scalar data set of a medical CT scan with 256^3 voxels by means of GPU-based ray casting [24]. For shape visualization, we recommend using functions g_c and g_h with a large spacing between maxima and with a wide prototype function to achieve a smooth looking image with only little luminance contrast from cycle shading.

Shape assessment is another application area. Figure 9d is based on hatched cycle shading with a mapping function that introduces gray gaps between maxima in both shading directions. Figure 9e shows cycle shading with an alternative mapping: an asymmetric prototype function p is chosen to emphasize discontinuities by kinks in sharp shading edges. The two close-up views focus on those surface imperfections. Figures 1a and 1b illustrate another typical example for shape assessment: the evaluation of CAD models. In general, shape interrogation should make use of mapping functions g_c and g_h that introduce clear luminance and color contrasts to emphasize discontinuities.

Figure 12 shows an application of hatched cycle shading for codimension 2. Here, streamlines are used to visualize the tangential vector field that originates from a simulation of air flow around a car (based on a surface LIC technique [30]). This figure demonstrates that hatched cycle shading can be successfully applied to typical computational fluid dynamics data from industry. Figure 12a gives an overall picture of the car with hatched cycle shading. Figure 12b compares traditional surface shading (top image) and hatched cycle shading (bottom image) for a detailed view on the area beneath the front left wheel of the car. Hatched cycle shading is used to enhance the streamlines (represented by luminance contrast of high spatial frequency perpendicular to streamline directions) by chromatic contrast (for cycle shading) and luminance contrast (for the hatched part). The spatial frequency of the luminance contrast for hatched cycle shading is lower than the frequency of the streamline texture to make both distinguishable from each other. Hatched cycle shading is useful for illustrating global flow behavior (as in Figure 12a), but can also be applied for flow inspection. Higher-order discontinuities of the flow, which are caused by changing resolution of the underlying computational grid, become clearly visible, e.g., in the lower part of Figure 12b (bottom image). In contrast, traditional surface shading fails to show these discontinuities (see Figure 12b, top image).

Corresponding animations are contained in the accompanying video [29].

8 User Study

We have conducted a user study to validate the usefulness of cycle shading for the visual assessment of surface quality. The goal of the study is to compare the users' efficiency in locating surface imperfections for two different surface shading approaches: traditional Phong shading and cycle shading. The user study tests the usefulness of those shading techniques for the detection of G^2 surface discontinuities, i.e., it tests the usefulness for shape assessment.

A sphere is used as test surface. We choose this simple geometric shape because our study targets the users' efficiency in perceiving surface irregularities, not in understanding complex shapes. Two types of imperfections that exhibit discontinuous normal vectors are applied to an otherwise smooth sphere—a line-like, thin imperfection covering an angle of 30 degrees of the sphere in the long direction, and a triangular imperfection that covers less than 10 degrees in both directions. The location and orientation of these imperfections is randomly initialized, and the user is asked to identify the type of imperfection by interactively inspecting the surface. Only grayscale visualizations are used, i.e., in none of the tests a chromatic contrast is employed. The inspection process is restricted to rotating the sphere via the mouse. The position of the sphere is located at a fixed distance in a way that it fills large parts of the screen. Illumination direction is fixed, incident from top–right. The two types of imperfections and the two types of shading models are chosen randomly, leading to a random collection of four different configurations. Figure 7 depicts the four combinations of stimuli: the left image shows an overall view of the test scene, the other three images show close-up views of the imperfections. A stratified random sampling is applied so that each user has to identify a constant number of the four configurations in a random order. This random order of presentations of stimuli was used to avoid any learning effects. In this study, each user was asked to identify 80 imperfections, 20 for each configuration. The user can start a single time measurement by pressing the space bar on the keyboard; then,

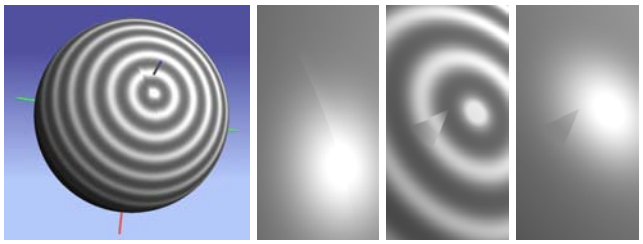


Figure 7: Stimuli for user study: line-shaped (two left images) and triangle-shaped (two right images) imperfections.

the object is shown and can be rotated. After users have identified an imperfection, they indicate its type by pressing one of two other, fixed keys. The time between initial presentation of the object and its identification is recorded. The time span for identifying one imperfection is restricted to 20 seconds in order to avoid an overly long identification process. We added a training phase before the actual measurements: training includes an introduction to the user study and an actual practicing phase. The training phase takes approximately five minutes, the measurements some ten minutes.

The user study was conducted with 20 participants, mostly graduate students. All of them have a background in computer science or a related field, and are familiar with using computers and 3D graphics applications. A total number of 1600 time measurements were recorded. 92.5% of the imperfections were identified correctly, 0.75% were identified incorrectly, and in 6.75% of the cases the timeout of 20 seconds was reached. These numbers show that the timeout limit, the test objects, and the task are adequately designed. Within the four sub-cases, the prorated numbers of timeout results were: Phong shading (12.75%), cycle shading (0.75%), line-like stimulus (7.38%), and triangular stimulus (6.13%). Incorrect answers were more evenly distributed: Phong shading (0.75%), cycle shading (0.75%), line-like stimulus (0.88%), and triangular stimulus (0.63%).

Only correct answers are considered in the following discussion. The users' performance for four different categories are: Phong shading (average time 7.38 seconds, standard deviation (SD) 4.79 seconds), cycle shading (4.64 s, SD: 3.31 s), line-like stimulus (6.00 s, SD: 4.27 s), and triangular stimulus (5.85 s, SD: 4.32 s). An ANOVA reveals a significantly shorter identification time for the experiments with cycle shading as compared to Phong shading ($F = 167.5$, $p < 0.0001$). On the other hand, there is no significant difference between identification times for the two stimuli ($F = 0.71$, $p > 0.4$). This study shows that users perform a surface interrogation task that targets the detection of G^2 surface discontinuities much faster with cycle shading than with Phong shading. A detailed comparison with other variants for surface shading, such as reflection lines or curvature-based methods, is out of the scope for this paper.

9 Implementation

Our shading model is implemented by extending traditional lighting with the corresponding computations and mappings for (hatched) cycle shading. The functions f_c and f_h can either be accessed from pre-computed lookup tables (1D tables or, for anti-aliasing, 2D tables) or evaluated on-the-fly by direct numerical operations. Due to non-linearities, per-pixel lighting is necessary for adequate image quality—as in the case of Phong illumination. Cycle shading can be directly included in CPU-based and GPU-based rendering systems by slightly modifying existing Phong computations.

Our GPU implementation of surface and streamline visualization is based on C++, Microsoft's DirectX 9.0, as well as High-Level Shading Language (HLSL) and FX files for GPU-based per-vertex and per-pixel computations. The implementation requires a Shader Model 2 compliant GPU and has been successfully tested on Windows PCs with various types of graphics boards (ATI Radeon 9700, Radeon X800, NVIDIA GeForce FX 5950, and GeForce 6800 Ultra). The FX file that implements (hatched) cycle shading on surfaces can be found on our project web page [29]. The implementation of volume rendering extends an existing code for GPU ray casting [24] and is based on C++ and OpenGL with ARB fragment and vertex programs.

The performance of any of these implementations of (hatched) cycle shading is comparable to a respective implementation of Phong illumination. The rendering pipeline is almost identical for our approach and traditional lighting, except for slight modifications in fragment processing. Performance measurements for a viewport-filling quadrilateral (1400×1100 viewport) on an NVIDIA GeForce 6800 Ultra GPU result in 294.2 fps (frames per second) for Phong illumination as well as cycle shading and 131.9 fps for hatched cycle shading. Performance differences are much smaller for less extreme cases. For example, the surface model of the automobile from Figure 12 (3 431 388 vertices and 1 143 796 triangles) renders Phong illumination and cycle shading at 68.8 fps, and hatched cycle shading at 53.3 fps. Similarly, there is hardly any performance difference for volume rendering or streamline shading.

10 Discussion and Conclusion

We have proposed cycle shading and hatched cycle shading as new rendering methods for shape assessment and visualization. Our approach is compact, generic, simple to implement, and computationally inexpensive. It can be used wherever Phong lighting is applicable, e.g., for triangle meshes without neighborhood information or for volume rendering. We have shown that (hatched) cycle shading can be used for objects of any dimension and codimension. Curvature is encoded by the density of shading patterns, while shape smoothness is visualized through the smoothness of shading patterns. Our approach is sensitive to variations in orientation and, thus, facilitates shape assessment. We see the main application of (hatched) cycle shading in user-based rapid inspection of information within engineering and scientific disciplines, e.g., in order to validate the smoothness of CAD models or the results of numerical flow simulations.

We briefly compare the approach of this paper with existing methods for shape visualization. Cycle shading is related to surface interrogation methods such as reflection lines or isophotes because all these approaches visualize second-order surface discontinuities by first-order discontinuities of visual patterns. The previous approaches focus on surface interrogation with a few line-like structures, whereas cycle shading targets a continuous shading. Advantages of (hatched) cycle shading are its high efficiency, a practical filtering mechanism (as compared to reflection mapping, for which anti-aliasing is an issue), a full coverage of the object by shading patterns, information encoding in two, essentially perpendicular, surface directions via hatched cycle shading, and the extensibility to higher codimensions.

Another important class of previous shape illustration methods utilizes curvature information, e.g., to place strokes along principal directions or to apply color coding. These methods and (hatched) cycle shading are related in the sense that both approaches represent curvature, either directly by second-order information from curvature or indirectly through the smoothness of first-order information from normals. Curvature-based rendering often is view-independent, while cycle shading is, as a matter of principle, view-dependent. An advantage of cycle shading is that the problem of computing curvature measures on polyhedral meshes is not present. On the other hand, the direct control of stroke directions can be used for a better control of visual patterns. Finally, silhouette, fea-

ture, and contour lines are effective in illustrating shape by means of a sparse visual representation. Cycle shading is complementary to these methods because it achieves continuous shading throughout the surface, whereas line-oriented methods generate curves that cover only a small portion of the surface. We believe that the combination of both approaches could be useful: contour and feature lines, for example, visually separate regions along zero-order and first-order surface discontinuities with respect to image space and, thus, separate zero-order discontinuities of cycle shading patterns.

We have conducted a user study that indicates the usefulness of cycle shading for surface quality assessment. Since the user study has been restricted to a comparison between Phong shading and cycle shading, further experiments that compare a larger variety of methods could be addressed as part of future work. A much more complex question is whether our shading approach is effective in promoting the understanding of 3D shape. Through our user study, we have only investigated the task of surface assessment but not the effectiveness for shape understanding. However, previous knowledge about visual perception leads us to the hypothesis that our approach might support shape perception. As view-dependent and light-source dependent specular illumination is an effective cue for shape perception [20], cycle shading—as an extension of, and in combination with, specular lighting—might also support shape perception. In particular, the combination with interactive visualization is promising because perceptual cues can be enhanced by dynamic scenes and camera motion. In future work, the usefulness of (hatched) cycle shading for shape visualization (with the goal of shape understanding) should be evaluated by user studies. Similarly, a combination of our approach with perceptual cues like other texture properties or motion needs to be examined in more detail.

A beneficial feature of (hatched) cycle shading is its applicability to objects of high dimension and codimension. We are not aware of any reasonably comparable, alternative shape visualization method for such a scenario and, thus, our shading approach could play an important role in displaying and investigating curve-oriented information, for example, for 3D vector field visualization. Again, the perceptual effectiveness of our shading approach for curve rendering needs yet to be evaluated by user tests.

ACKNOWLEDGMENTS

Parts of this work have been done at the VRVis Research Center in Vienna, Austria, which is funded by the K*plus* research program. We thank the anonymous reviewers for helpful comments and Christopher G. Healey for his feedback during the revision process. The basis code for GPU ray casting (for Figure 9c) was kindly provided by Simon Stegmaier. The model for Figures 1a and 1b is used by courtesy of Volkswagen AG, the data set in Figures 1c, 1d, and 12 is used by courtesy of the BMW Group. Special thanks to Michael Guthe, Martin Schulz, and Bettina Salzer for their help.

REFERENCES

[1] D. C. Banks. Illumination in diverse codimensions. In *Proc. ACM SIGGRAPH 1994*, pages 327–334, 1994.

[2] B. Cabral and L. C. Leedom. Imaging vector fields using line integral convolution. In *Proc. ACM SIGGRAPH 1993*, pages 263–270, 1993.

[3] D. DeCarlo, A. Finkelstein, R. Rusinkiewicz, and A. Santella. Suggestive contours for conveying shape. *ACM Transactions on Graphics (Proc. ACM SIGGRAPH 2003)*, 22(3):848–855, 2003.

[4] J. C. Dill. An application of color graphics to the display of surface curvature. *Computer Graphics (Proc. ACM SIGGRAPH '81)*, 15(3):153–161, 1981.

[5] D. Ebert and P. Rheingans. Volume illustration: Non-photorealistic rendering of volume models. In *Proc. IEEE Visualization 2000*, pages 195–202, 2000.

[6] G. Farin and D. Hansford. Reflection lines: Shape in automotive design. *ties Magazine*, pages 18–21, Mar. 2002.

[7] R. W. Fleming, A. Torralba, and E. H. Adelson. Specular reflections and the perception of shape. *Journal of Vision*, 4(9):798–820, 2004.

[8] J. J. Gibson. *The Perception of the Visual World*. Houghton Mifflin, Boston, 1950.

[9] J. J. Gibson. *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1986.

[10] A. Girshick, S. Haker, T. Lemoine, and V. Interrante. Line direction matters: An argument for the use of principal directions in 3D line drawings. In *Proc. NPAR*, pages 43–52, 2000.

[11] J. Goldfeather and V. Interrante. A novel cubic-order algorithm for approximating principal direction vectors. *ACM Transactions on Graphics*, 23(1):45–63, 2004.

[12] A. Gooch, B. Gooch, P. Shirley, and C. Cohen. A non-photorealistic lighting model for automatic technical illustration. In *Proc. ACM SIGGRAPH 1998*, pages 447–452, 1998.

[13] B. Gooch and A. A. Gooch. *Non-Photorealistic Rendering*. A. K. Peters, Natick, MA, 2001.

[14] G. Gorla, V. Interrante, and G. Sapiro. Texture synthesis for 3D shape representation. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):512–524, 2003.

[15] S. Hahmann. Surface interrogation: Visualization techniques for surface analysis. In C. Bajaj, editor, *Data Visualization Techniques*, pages 49–74. John Wiley, 1999.

[16] A. Hertzmann and D. Zorin. Illustrating smooth surfaces. In *Proc. ACM SIGGRAPH 2000*, pages 517–526, 2000.

[17] V. L. Interrante. Illustrating surface shape in volume data via principal direction-driven 3D line integral convolution. In *Proc. ACM SIGGRAPH 1997*, pages 109–116, 1997.

[18] R. Klass. Correction of local surface irregularities using reflection lines. *Computer-Aided Design*, 12(2):73–77, 1980.

[19] B. Lowekamp, P. Rheingans, and T. S. Yoo. Exploring surface characteristics with interactive Gaussian images (A case study). In *Proc. IEEE Visualization 2002*, pages 553–556, 2002.

[20] J. F. Norman, J. T. Todd, and G. A. Orban. Perception of three-dimensional shape from specular highlights, deformations of shading, and other types of visual information. *Psychological Science*, 15(8):565–570, 2003.

[21] B.-T. Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317, 1975.

[22] T. Poeschl. Detecting surface irregularities using isophotes. *Computer Aided Geometric Design*, 1(2):163–168, 1984.

[23] D. Schweitzer. Artificial texturing: An aid to surface visualization. *Computer Graphics (Proc. ACM SIGGRAPH '83)*, 17(3):23–29, 1983.

[24] S. Stegmaier, M. Strengert, T. Klein, and T. Ertl. A simple and flexible volume rendering framework for graphics-hardware-based raycasting. In *Proc. Workshop on Volume Graphics*, pages 187–195, 2005.

[25] T. Strothotte and S. Schlechtweg. *Non-Photorealistic Computer Graphics – Modeling, Rendering, and Animation*. Morgan Kaufmann, San Francisco, 2002.

[26] G. Sweet and C. Ware. View direction, surface orientation and texture orientation for perception of surface shape. In *Proc. Graphics Interface*, pages 97–106, 2004.

[27] H. Theisel. Are isophotes and reflection lines the same? *Computer Aided Geometric Design*, 18:711–722, 2001.

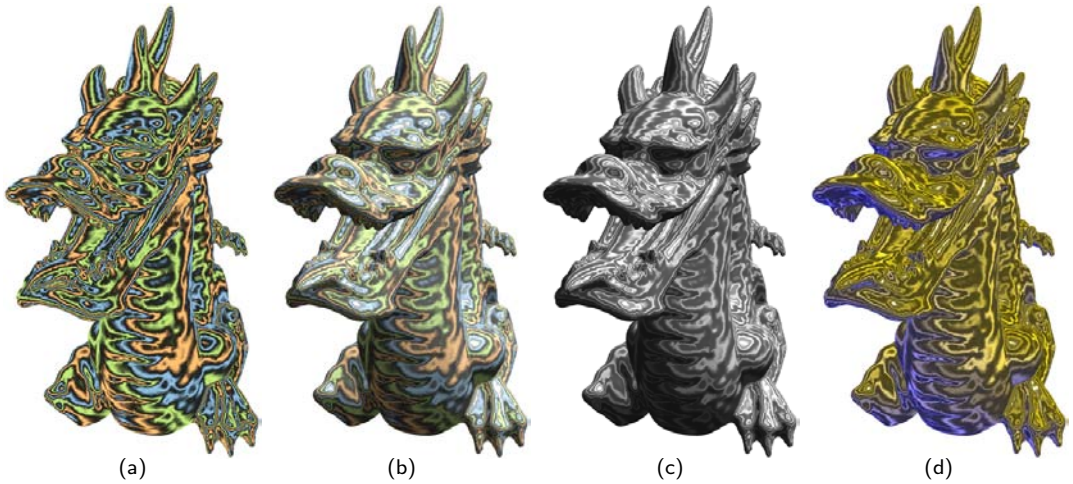
[28] D. Voorhies and J. Foran. Reflection vector shading hardware. In *Proc. ACM SIGGRAPH 1994*, pages 163–166, 1994.

[29] D. Weiskopf. Cycle shading web page. `http://www.cs.sfu.ca/~weiskopf/research/cycleshading`, 2006.

[30] D. Weiskopf and T. Ertl. A hybrid physical/device-space approach for spatio-temporally coherent interactive texture advection on curved surfaces. In *Proc. Graphics Interface*, pages 263–270, 2004.

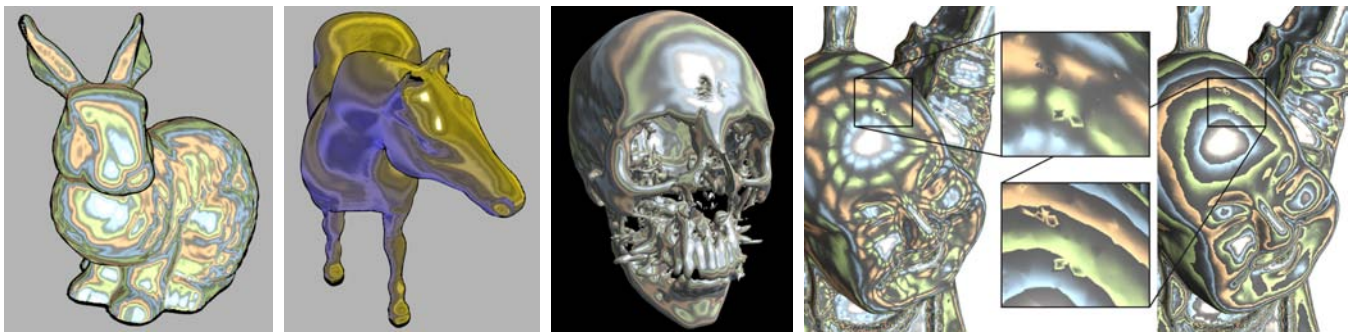
[31] L. Williams. Pyramidal parametrics. *Computer Graphics (Proc. ACM SIGGRAPH '83)*, 17(3):1–11, 1983.

[32] M. Zöckler, D. Stalling, and H.-C. Hege. Interactive visualization of 3D-vector fields using illuminated stream lines. In *Proc. IEEE Visualization 1996*, pages 107–113, 1996.



(a) (b) (c) (d)

Figure 8: Different illumination options: no surface illumination (not recommended, included just for comparison) in (a), colored cycle shading with gray-scale illumination (b), gray-scale cycle shading (c), and cool/warm surface shading with luminance cycle shading (d).



(a) (b) (c) (d) (e)

Figure 9: Chromatic cycle shading with silhouettes (a), luminance cycle shading with cool/warm shading (b), volume illustration of a CT scan via cycle shading (c), surface assessment in (d) and (e), including two close-up views.

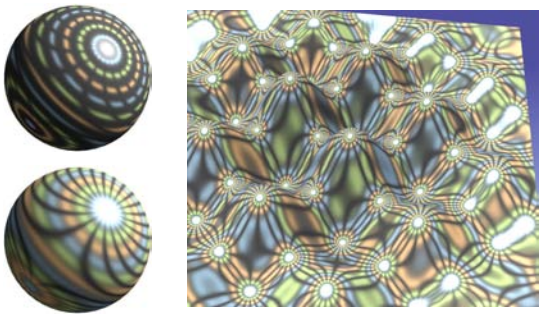
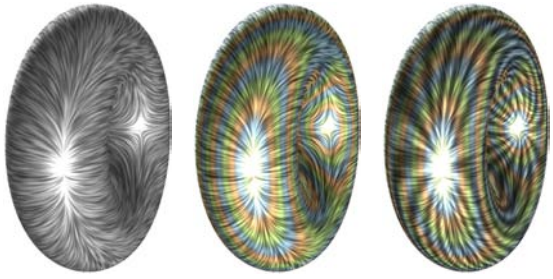
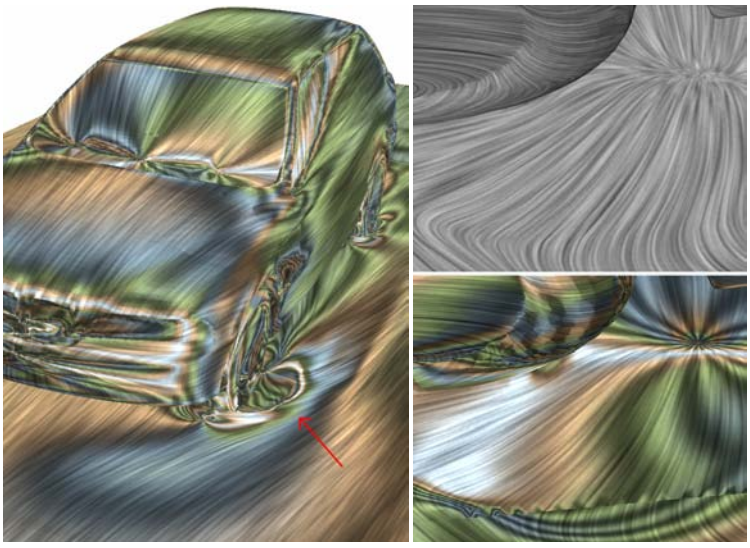


Figure 10: Hatched cycle shading.



(a) (b) (c)

Figure 11: Illumination in codimension 2: illuminated streamlines (a), cycle shading (b), and hatched cycle shading (c).



(a) (b)

Figure 12: Flow visualization: overall view with hatched cycle shading (a), comparison between surface illumination (b, top) and hatched cycle shading (b, bottom) for a detailed view on the area beneath the front left wheel of the car. The viewing area for the close-up is indicated by a red arrow. Data set courtesy of the BMW Group.