Texture-Based hologram generation using triangles

Matthias Koenig[a], Oliver Deussen[b], Thomas Strothotte[a]

[a]University of Magdeburg, FIN-ISG, 39016 Magdeburg, Germany
[b]Dresden University of Technology, ISMT, 01062 Dresden, Germany

ABSTRACT

The synthesis of holograms by computer requires the calculation of the complex amplitude emitted by an object in the hologram plane. Several approaches exist that decompose the object into primitives like points and lines. We assume that the object to be imaged can be approximately decomposed into congruent triangles. In a preprocessing step, we calculate wavefields for the triangle whose transformed copies build the object surface. The computed wavefields represent triangles rotated by different angles and positioned in different depths. The resulting wavefields are stored as conventional color images with alpha channel (textures) in a lookup table indexed by rotation angle and distance from the hologram plane. Each pixel in the texture codes a complex number. Every triangle of the input object has a corresponding entry in the lookup table. The rotation angles and the distance of the triangle determine the selection of the appropriate texture. The textures are rendered using special graphics hardware, and interference is simulated. The lookup table helps to react immediately to transformations of the input object. Texture swapping and repositioning according to the object movings lead to full-parallax hologram generation for small objects in real-time.

Keywords: Computer-generated holograms, computer graphics

1. INTRODUCTION

In research of 3d-display technology, an important goal is to provide techniques that support most of the visual cues that are necessary to convey spatial impression. These depth cues include, for example, binocular disparity, ocular accommodation, and motion parallax,[1] all of them being provided by holographic displays.[2] Holography enables to record and reconstruct a full copy of the original wavefield emitted by an illuminated object.

The synthetical generation of holograms by computer requires the calculation of the complex amplitude resulting from object wave and reference wave in the hologram plane. The complex amplitude is then appropriately converted for materialization to produce a computer-generated hologram[3] or for direct modulation of a light beam.[4,5] The reduction of the computational effort for real-time applications is given first priority. Therefore, sophisticated concepts that face this task have been presented in recent years. Some authors decompose the object into simple primitives like points[6] and lines[7] in order to reduce the calculation of the complete complex amplitude to the computation of the complex amplitudes of the primitives. Other authors applied special kinds of graphics hardware to support the synthesis of stereograms[8] and full-parallax holograms.[9]

In this paper, we apply graphics hardware to hologram rendering using complex-valued textures. Such textures represent emitted wavefields of simple objects like points and lines[9] or plane primitives.[10] The use of textures enables hardware-assisted rendering to speed up calculation time. In our case the primitives are triangles. We assume that an object to be imaged can be approximately decomposed in congruent triangles that build the object's surface, but triangulation methods fulfilling this decomposition are not discussed here as they are investigated in geometric mathematics.[11,12] This assumption enables us to reduce the amount of pre-calculated wavefields.

We divide the computation of the complex amplitude in several steps in order to use hardware developed specially for fast computer graphics. In a preprocessing step, we calculate wavefields for the triangle whose transformed copies build the approximated object surface. The computed wavefields represent the triangle rotated by different angles and positioned in different distances to the hologram plane. For this computation, the Fresnel transform for rotated planes is applied.[13] The resulting wavefields are coded as conventional color images with alpha channel and stored in a lookup table indexed by rotation angle and distance from the hologram plane.

M. König's e-mail address: koenig@isg.cs.uni-magdeburg.de

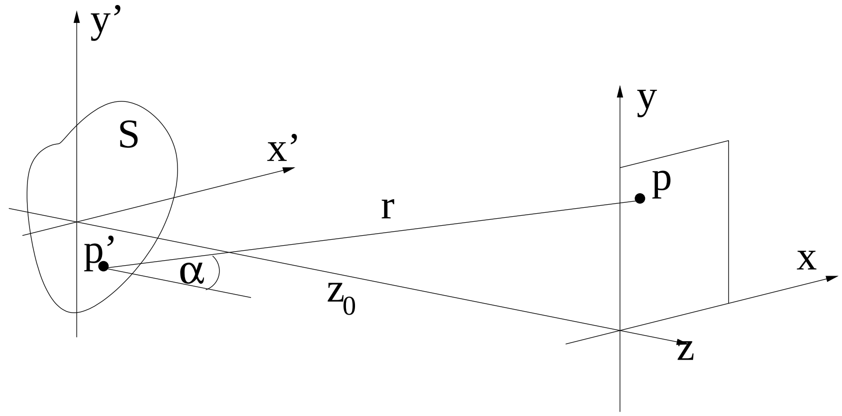


Figure 1: Presumed geometric relationship between object S and hologram H for solving the Kirchhoff diffraction integral.

The full hologram is rendered in the next step. The interference of the wavefields is simulated by texture swapping and texture superposition using the accumulation buffer, a special graphics hardware unit. Fast hologram generation is achieved in comparison with standard computational methods using main processor and main memory.

The remainder of the paper is organized as follows: First, we briefly introduce synthetic holography and explain the methods for calculating the wavefields of rotated planar elements. Then we present our approach to simulate interference of rotated planar elements and show results. We close with an outline of future work.

2. SYNTHETIC HOLOGRAPHY

The determination of the complex amplitude in the hologram plane is done by computer in synthetic holography. Several methods exists for this in general.[3,14] All of these methods can be derived from the Kirchhoff diffraction integral:

$$U(p) = \frac{1}{i\lambda} \int_S u_0(p') \frac{\exp(ikr)}{r} \cos(\alpha) ds. \tag{1}$$

In Equation 1 $U(p)$ determines the electric field strength at the point p. As Figure 1 illustrates, $U(p)$ results from the field strength $u_0(p')$ contributed from each point p' of the surface S; λ represents the wavelength, r the distance between p and p' and α the angle between $p' - p$ and the incident illumination wave. The intensity of $U(p)$ results in a hologram.

Several methods making use of the fast Fourier transform can be applied to approximate Kirchhoff's formula for planar objects parallel or rotated to the hologram plane. Usually the angular spectrum of plane waves or the Fresnel transform is performed to determine the wavefield in the hologram plane for such objects.[14,13,15] The Fresnel transform is studied closely here.

If the conditions $|x|, |y|, |x'|, |y'| << z_0$ are fulfilled, the Fresnel approximation of Equation 1 can be applied to calculate the diffraction patterns of planar objects. For an object parallel to the hologram plane, this can be implemented by a convolution or a Fourier transform with an additional multiplication of a quadratic phase factor. Is the object rotated to the hologram plane, a Fourier transform with a coordinate transform before the multiplication of the phase factor is necessary.

Figure 2 demonstrates the geometry of a planar object rotated about the x-axis. A plane wave illuminates the object

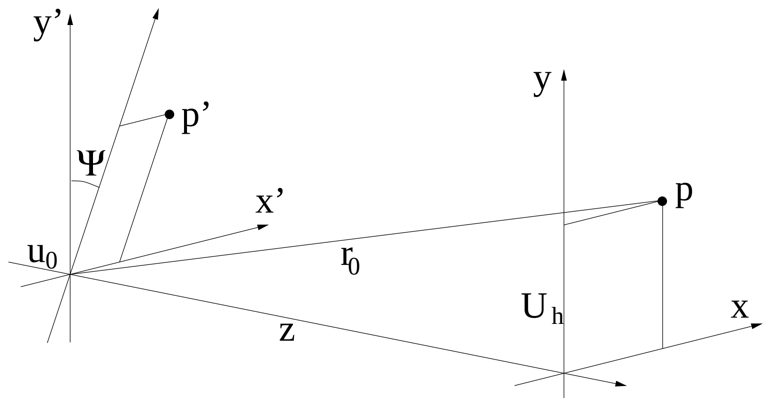


Figure 2: Determining the wavefield of a rotated planar object.

on-axis. Then the Fresnel transform is given by the following equations:

$$U_h(x,y) = \exp(ikr_0(x,y,z))U(\nu(x,y),\mu(x,y)), \tag{2}$$

$$U(\nu(x,y),\mu(x,y)) = \iint\limits_{-\infty}^{\infty} u(x',y')\exp(-i2\pi(\nu x' + \mu y'))dx'dy', \tag{3}$$

$$u(x',y') = u_0(x',y')\exp(\frac{i\pi}{\lambda z}(x'^2 + y'^2)), \tag{4}$$

with $r_0 = \sqrt{z^2 + x^2 + y^2}$. Moreover, the coordinate transform that performs the rotation of the angle ψ about the x-axis is:

$$\begin{aligned} \nu(x,y) &= \frac{x}{\lambda r_0}, \\ \mu(x,y) &= \frac{y\cos(\psi)}{\lambda r_0} + \frac{(z - r_0)\sin(\psi)}{\lambda r_0}. \end{aligned} \tag{5}$$

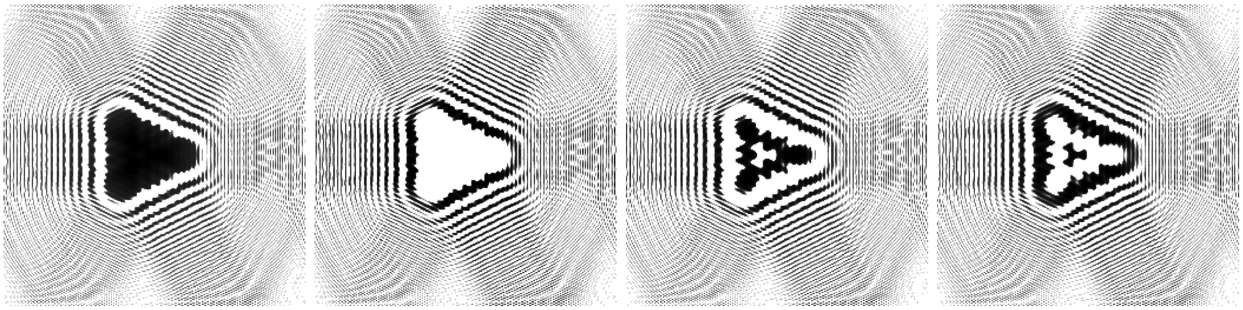
Further, the free positioning of a planar object in 3d space requires a combination of rotations about different axes. This leads to other coordinate transformations. For example, a coordinate transform for rotation about the x-axis followed by a rotation about the y-axis yields to the coordinate transformation:

$$\begin{aligned} \nu(x,y) &= \frac{x\cos(\phi)}{\lambda r_0} + \frac{(z - r_0)\sin(\phi)}{\lambda r_0}, \\ \mu(x,y) &= \frac{y\cos(\psi)}{\lambda r_0} + \frac{(z - r_0)\sin(\psi)\cos(\phi)}{\lambda r_0} + \frac{x\sin(\psi)\sin(\phi)}{\lambda r_0}, \end{aligned} \tag{6}$$

where ϕ is the angle about the y-axis with positive rotation z to x.

3. HARDWARE-BASED RENDERING OF HOLOGRAMS

A speed-up is achieved for rendering of holograms by graphics hardware. Cross sections through the wavefields emitted by each primitive building the input object are coded as complex textures for this. A complex texture is a conventional color image with three color channels and one alpha channel. Usually the alpha channel codes information about transparency of an image. We store in each channel parts of the complex numbers of a wavefield. Each pixel in a texture T codes in the red channel the positive real part, in the green channel the negative real part,



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

Figure 3: Color channels of a complex texture representing a cross section through a wavefield emitted by a triangle: (a) red, (b) green, (c) blue and (d) alpha channel.

in the blue channel the positive imaginary part and in the alpha channel the negative imaginary part:

$$T_{red}(x,y) = \begin{cases} \frac{|real(U(x,y;z))|-min_c}{max_c-min_c} & real(u(x,y;z)) > 0 \\ 0.0 & \text{else,} \end{cases}$$
$$T_{green}(x,y) = \begin{cases} \frac{|real(U(x,y;z))|-min_c}{max_c-min_c} & real(U(x,y;z)) < 0 \\ 0.0 & \text{else,} \end{cases}$$
$$T_{blue}(x,y) = \begin{cases} \frac{|imag(U(x,y;z))|-min_c}{max_c-min_c} & imag(U(x,y;z)) > 0 \\ 0.0 & \text{else,} \end{cases}$$
$$T_{alpha}(x,y) = \begin{cases} \frac{|imag(U(x,y;z))|-min_c}{max_c-min_c} & imag(U(x,y;z)) < 0 \\ 0.0 & \text{else,} \end{cases}$$

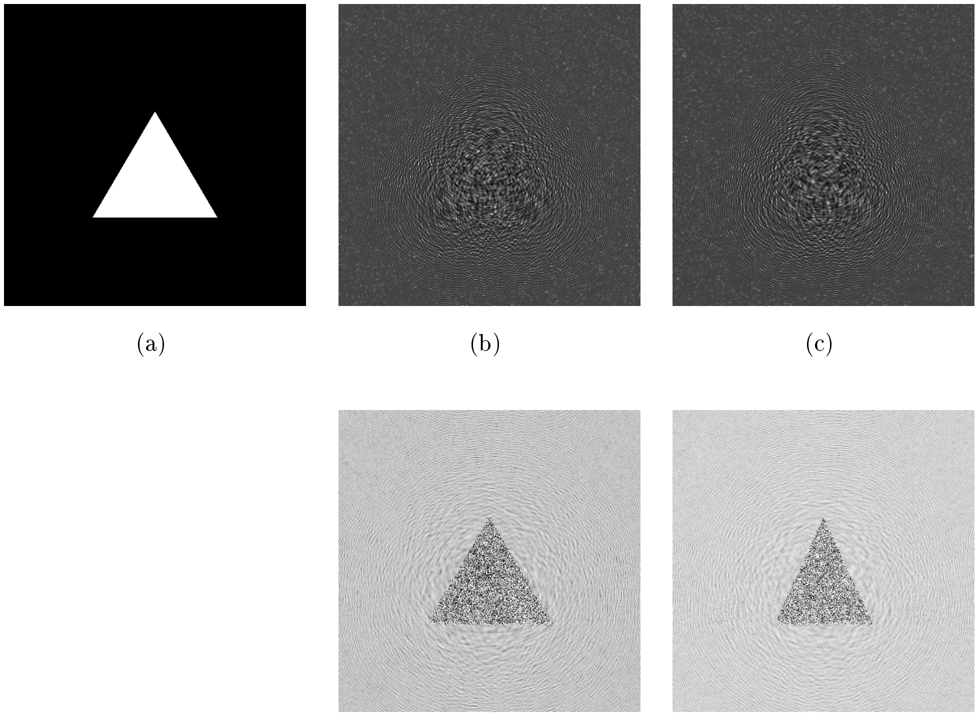
where $real(x)$ denotes the real part of a complex number x and $imag(x)$ the imaginary part. The numbers are standardized for a texture of size $N_x \times N_x$ with:

$$max_c = max(|real(U(x,y;z))|, |imag(U(x,y;z))|)\ \forall(x,y) \in (N_x, N_y),$$
$$min_c = min(|real(U(x,y;z))|, |imag(U(x,y;z))|)\ \forall(x,y) \in (N_x, N_y).$$

We pre-compute a set of complex textures evaluating Equations 2-6 for a planar object. The calculation is fast for planar objects rotated by the angles ψ and ϕ. A planar object is described as a simple two-dimensional field of complex numbers. At first, a phase factor (Equation 4) is applied. This phase factor can be neglected if a random phase distribution is assigned to the object. Then, a fast Fourier transform (Equation 3) is performed. These two steps are identical for all rotation angles and distances to the hologram plane. Therefore, they need to be applied only once for an object. The information about the rotation is processed by the coordinate transform (Equation 6) and the information about the distance to the hologram plane by the outer multiplication of a quadratic phase factor (Equation 2). Unfortunately, the last steps are insufficient in many discrete cases due to the changes of the sample rate. The same sample rate can be achieved using a second Fourier transform and a further multiplication of a phase factor.

In our case, one texture of the pre-calculated set represents the wavefield of the object in a certain depth and specific rotation angles. Figure 3 shows the different channels of a complex texture as grey scale images. A cross section through the wavefield of a triangle with constant phase distribution is demonstrated by the four separated parts of the complex number field.

The set of textures is stored in a lookup table. The access to a texture is performed by its rotation angles and distance between object and hologram plane; $T_{\psi,\phi,z}$ delivers the necessary texture. The number of textures to be



(a) (b) (c)

(d) (e)

Figure 4: Holograms generated from entries of a lookup table: (a) input object (b) hologram of the corresponding texture with no rotation, (c) hologram of the texture rotated 40° about the y-axis, (d) and (e) simulated reconstruction of (b) and (c).

pre-computed is determined by the interval distances for the rotation angles and distances. Figure 4 shows an input object, its holograms generated by superimposing an entry from a pre-calculated lookup table with a texture for the reference wave and the simulated reconstructions of the holograms. Typical speckle patterns occur because random phase was used.

To generate a hologram, the rotation angles ψ, ϕ and the distance z of the input object are determined and the appropriate complex texture $T_{\psi,\phi,z}$ is selected from the lookup table. The textures are now rendered to the accumulation buffer, a unit of special hardware graphics workstations[16] which performs the simulation of interference. The locations of the complex textures in the accumulation buffer for the rendering process result from a parallel projection of the input object. The positioning of the textures in the accumulation buffer is advantageous because translations $\mathcal{T}$ parallel to the hologram plane and rotations $\mathcal{R}$ about the z-axis are directly realized by this way. The hologram generation of an object consisting of N equal primitives is performed in the accumulation buffer using complex textures:

$$U[0\ldots N_x, \ldots N_y; 0] = \sum_{n=1}^{N} \mathcal{T}(\mathcal{R}_z(T_{\psi_x,\psi_y,z}[0\ldots N_x, 0\ldots N_y])),$$

here the square brackets denote that the addition of two-dimensional fields (the complex textures) of size $N_x \times N_y$ is accomplished in one step. A separate texture representing a reference wave is superimposed in an additional step. The intensity is read out of the accumulation buffer and constitutes the hologram. The hologram can be rendered in tiles to achieve high resolutions. All rendering steps are performed with *OpenGL*.[17]

Further, the use of the lookup table allows to react immediately to transformations of the input object. Texture swapping and repositioning according to object transformations leads to real-time hologram generation for small

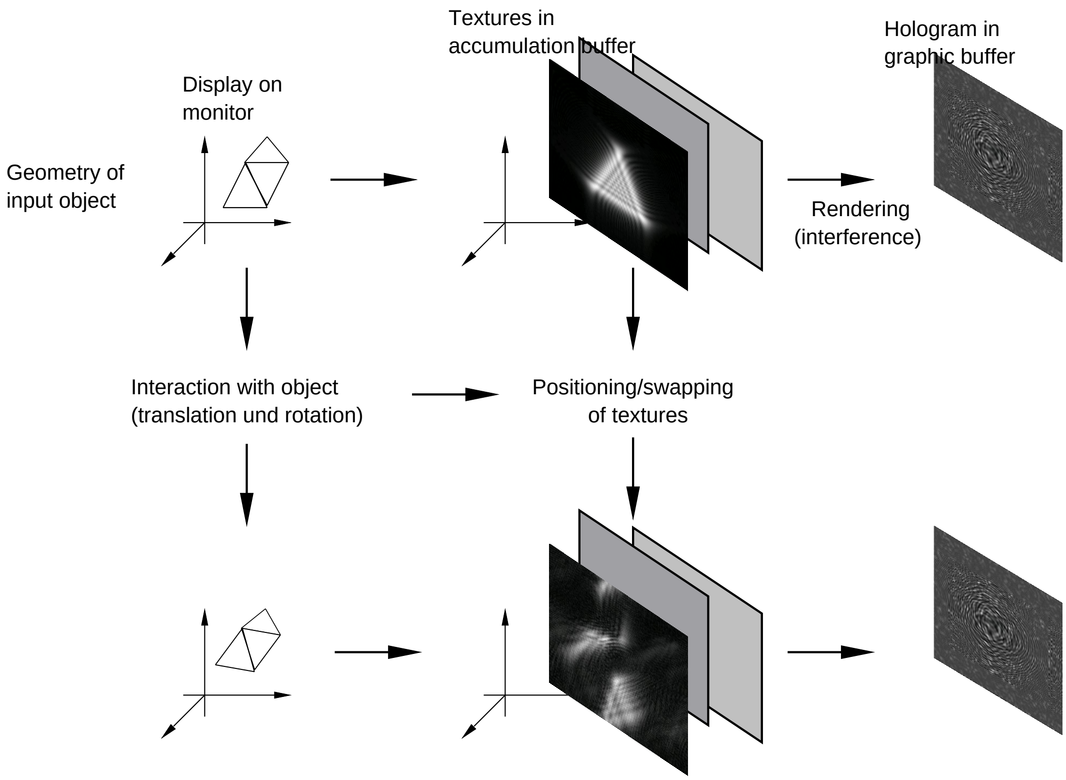


Figure 5: Positioning the complex textures related to the geometry of primitives of the input object.

objects. The process is illustrated in Figure 5. A geometric object composed of planar segments corresponds with complex textures rendered to a hologram.

4. RESULTS

A lookup table stores complex textures of one primitive for a certain distance and two rotation angles. We calculated 3600 complex textures for one primitive with size of 512×512 points. This corresponds to a precision of three degrees for the rotation angles and a fixed distance between primitive and hologram plane. This led to a calculation time of ca. five hours on an SGI Onyx2 with two R10000 (195 MHz) processors and InfinityReality graphics. Because of the symmetry of the rotation, textures can be mirrored and reused. This allows to halve the amount of textures needed. Superposition of the complex textures using the accumulation buffer was up to a factor of 25 faster than using main processor and main memory doing the same task.

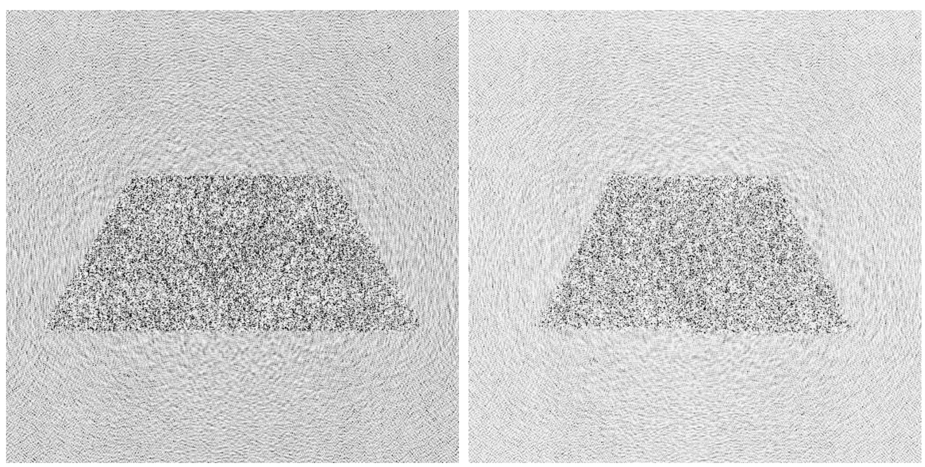
Figure 6 depicts three congruent triangles building an input object in different orientations to the hologram plane and the reconstructions of the related holograms. Because of the congruence of the triangles it is sufficient to calculate one lookup table for one triangle. The hologram is generated in real-time for arbitrary rotations of the input object. Although in Figure 6 no hidden surface problem is apparent, this problem is inherent with the presented method.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we described a method combining the traditional Fresnel transform and methods profiting from computer graphics hardware for fast hologram generation. The Fresnel transform for inclined planar objects is used to calculate textures. These textures are superimposed to simulate interference using graphics hardware. Full-parallax holograms for small objects are generated in real-time.

Our results show that the overhead caused by the pre-computation of the textures is amortized by the real-time hologram generation. Further, the textures can be saved to a file once calculated.

(a)



(b)

Figure 6: Three congruent triangles building the input object (a) and the reconstruction of the respective holograms generated by means of the presented method (b).

In future studies the triangulation of the input object in congruent triangles need to be integrated. Applying such triangulation can simplify the process of hologram generation enormously because pre-calculated textures can be reused many times.

The storing of parts of scenes composed of some elements is an important property of complex textures. This will allow to reuse calculated scenes.

Parts of the SYNTHETIC HOLOGRAPHY TOOLKIT use the texture-based approach. This toolkit can be downloaded for free.[18] It is implemented in C++ and runs on different unix systems. The texture-based modules require systems supporting hardware-based accumulation buffers.

REFERENCES

1. J. Murray, "Some perspectives on visual cues for depth perception," *Computer Graphics* **28**(2), pp. 155–157, 1994.
2. M. McKenna and D. Zeltzer, "Three dimensional display systems for virtual environments," *Presence: Teleoperators and Virtual Environments* **1**(4), pp. 421–458, 1992.
3. O. Bryngdahl and F. Wyrowski, "Digital holography — computer-generated holograms," *Progress in Optics* **28**, pp. 1–86, 1990.
4. N. Hashimoto and S. Morokawa, "Motion-picture holography using liquid-crystal television spatial light modulators," in *Society for Information Display '95*, pp. 847–850, 1995.
5. M. Lucente, "Interactive three-dimensional holographic displays: seeing the future in depth," *Computer Graphics* **31**(2), pp. 63–67, 1997.
6. J. P. Waters, "Holographic image synthesis utilizing theoretical methods," *Applied Physics Letters* **9**(11), pp. 405–407, 1966.
7. C. Frere, D. Leseberg, and O. Bryngdahl, "Computer-generated holograms of three-dimensional objects composed of line segments," *J. Opt. Soc. Am.* **3**(5), pp. 726–730, 1986.
8. M. W. Halle and A. B. Kropp, "Fast computer graphics rendering for full parallax spatial displays," in *Practical Holography XI*, S. A. Benton, ed., pp. 105–112, Proc. SPIE 3011, 1997.
9. A. Ritter, J. Böttger, O. Deussen, M. König, and T. Strothotte, "Hardware-based rendering of full-parallax synthetic holograms," *Applied Optics* **38**(8), pp. 1364–1369, 1999.
10. M. König, O. Deussen, and T. Strothotte, "Texture-based composition of holograms using triangular elements," in *Proc. of Computer Graphics and Imaging (CGIM'99)*, pp. 162–167, 1999.
11. D. Shikhare, S. Gopalsamy, T. S. Reddy, A. Patgawkar, M. Mahapatra, S. P. Mudur, K. P. Singh, I. Narayanswamy, and L. Ravishankar, "Zeus: surface modeling, surface grid generation, tetrahedral volume discretization," *Computer and Graphics* **23**, pp. 59–72, 1999.
12. A. Fuchs, "Automatic grid generation with almost regular delaunay tetrahedra," *Int. Journal for Numerical Methods in Engineering* **40**, pp. 4595–4610, 1997.
13. D. Leseberg and C. Frere, "Computer-generated holograms of 3-d objects composed of tilted planar segments," *Applied Optics* **27**(14), pp. 3020–3024, 1988.
14. J. W. Goodman, *Introduction to Fourier Optics*, McGraw-Hill, New York, 2. ed., 1968.
15. T. Tommasi and B. Bianco, "Computer-generated holograms of tilted planes by a spatial frequency approach," *J. Opt. Soc. Am. A* **10**(2), pp. 299–305, 1993.
16. K. Akeley, "RealityEngine graphics," in *SIGGRAPH '93*, pp. 109–116, August 1993.
17. J. Neider, T. Davis, and M. Woo, *OpenGL Programming Guide: The Official Guide to Learning OpenGL*, Addison-Wesley, 1993.
18. M. König, J. Hantke, and V. Padur, "Synthetic Holography Toolkit." http://wwwisg.cs.uni-magdeburg.de/~makoenig/SHoT, 2000.