University of West Bohemia
Department of Computer Science and Engineering
Univerzitní 8
30614 Plzeň
Czech Republic

Computer generated display holography

Summary of the results and the future work

Petr Lobaz

Technical Report No. DCSE/TR-2016-03
March 2016
Distribution: public

Technical Report No. DCSE/TR-2016-03
March 2016

Computer generated display holography

Summary of the results and the future work

Petr Lobaz

Abstract

This text summarizes my work in the field of computer generated display holography. It tries to explain what motivated me to explore certain aspects of its fundamental principles and briefly explains the results. The reader interested in details finds them in my research articles; they are included as appendices. My longest text, the draft of the intended book "Computer generated display holography", is available as a separate document.

Copies of this report are available on
`http://www.kiv.zcu.cz/en/research/publications/`
or by surface mail on request sent to the following address:

University of West Bohemia
Department of Computer Science and Engineering
Univerzitní 8
30614 Plzeň
Czech Republic

Contents

Summary of the results and the future work

This text summarizes my work in the field of computer generated display holography. It tries to explain what motivated me to explore certain aspects of its fundamental principles and briefly explains the results. The reader interested in details finds them in my research articles; they are included as appendices. My longest text, the draft of the intended book "Computer generated display holography", is available as a separate document.

I entered the world of holography in 2005 thanks to prof. Ing. Václav Skala, CSc. and his involvement in the European project "3DTV – Integrated Three-Dimensional Television – Capture, Transmission and Display". The computer graphics group from the University of West Bohemia, I am a member of, worked mainly in two areas – compression of dynamic meshes and digital holography. I was involved mainly in digital holography, together with Ing. Ivo Hanák and Ing. Martin Janda.

As a computer scientist specialized in computer graphics, I quickly realized that computer generated holography is a big challenge – among high-performance programming and algorithm design, it requires a deep understanding of optics (especially Fourier optics), digital signal processing, and indeed optical holography. Other areas such as computer graphics, 3-D display technology, numerical mathematics, Fourier analysis, vision science or even chemistry and precision mechanics are more than welcome.

I also quickly realized that a lot of texts about (computer generated) holography are mostly written by opticians for the audience with strong background in optics. Indeed, some texts were written by non-opticians – and here things became complicated. Some authors talked about e.g. the Rayleigh-Sommerfeld diffraction integral and described it by a particular equation; other authors talked about the Rayleigh-Sommerfeld diffraction integral as well but used a different equation. Some authors based their reasoning on formulas expressed in the frequency domain, but the algorithms used the discrete Fourier transform without bothering if the discretization is correct. Some authors used a certain light propagation calculation while the others used a different one without any clear reason. Some authors used zero padding in FFT calculations, the others did not. And so on.

In such a situation, scepticism creeps into mind: do the authors really know what are they doing? Or is it I just do not understand some common knowledge? Unfortunately, I was not able to find any monograph on computer generated holography written for non-opticians that would start from the very beginning and followed a single track – towards computer generated holography.

I, therefore, decided I have to create my own reference tool that allows me to compare results of various procedures to the result well supported by rigorous physics. Especially for a beginner, it is extremely hard to guess the correct result and differentiate implementation errors from approximate calculation artifacts and to decide if the artifacts are serious or not. As the basic task in computer generated holography is light propagation calculation between parallel planes, I focused to this area. I created an algorithm based on discrete convolution in the spatial domain that allows to propagate light between planes of different sizes and different sampling distances, see [5] (included as the Appendix E on page 55). Its main advantage is that it allows the trade-off between the time and the memory complexity; I was then able to calculate light propagation that would normally require thousands of gigabytes of main memory.

As a hologram is just a diffractive structure, and the simplest diffraction structure is a diffraction grating, it became obvious that it would be helpful to understand them before diving in computer generated holography. Unfortunately, the only available spatial light modulator for displaying computer generated holograms allows just binary output, and binarization significantly affects the pattern behaviour. When I was analysing effects of binarization and ways to avoid them (together with L. Kovář from the Brno Univerity of technology, see [6] for the results; included as the Appendix F on page 65), I noticed a significant discrepancy between real light behaviour and the result of my reference tool – which was disturbing as the tool was designed to match reality.

A similar problem was reported by many other authors and they also reported its cause – aliasing of the propagation kernel. However, in other cases, when the propagation kernel was aliased as well, my algorithm gave precise results. It was obvious that explaining the discrepancy just by aliasing is not sufficient, and I sought for a more fundamental explanation, possibly physically based and intuitive. Thanks to the previous study of fundamental principles of wave phenomena I realized that the dreaded aliasing is actually not an error – the fact is that the calculation of light propagation based on discrete convolution in the spatial domain solves a bit different physical problem. In particular, if the problem of light propagation is discretised in a naive and the most straightforward way (and by far the most common one!), it should be interpreted as a propagation of wave emitted by discrete dipoles, or simply said point light sources.

To solve the original propagation problem, i.e. the propagation of a continuous optical field defined in the plane $z = 0$, it is necessary to introduce

interpolation between samples to the discrete algorithm. This leads to a slightly different formulation of the propagation formula; see [3] for the complete discussion and an efficient algorithm for the calculation (included as the Appendix C on page 29).

The result also allowed me to solve another fundamental problem that was bothering me – while light propagation calculation can be equivalently formulated in both the spatial and the frequency domains, the numerical algorithms behave differently. Again, a common explanation just stated that the numerical algorithms are different due to aliasing, and many authors proposed "a solution" – to throw away the aliased part of the signal. First of all, no evidence was given that such a solution has a reasonable physical interpretation and if the results of a "corrected" algorithm still match reality. Second, I felt that it should be possible to discretise the problem carefully and to obtain a numerical algorithm that works in both the spatial and the frequency domain.

I found the solution and published it in [1] (included as the Appendix A on page 13). Moreover, I also showed that a careful discretisation in the frequency domain *requires* to introduce additional terms to the calculation, and that these terms would be normally considered as aliasing.

These insights would not be possible without an intuitive understanding of light behaviour, which leads me to some aspects of my Ph.D. study. Every Ph.D. student at the University of West Bohemia has to pass three exams, and I was very lucky that all three teachers allowed me to combine their subject with holography. In particular, Ing. Šárka Němcová, Ph.D. from the Czech Technical University allowed me to learn optical holography in the diffractive optics laboratory and taught me "an engineering approach" to thinking about light and holograms. Moreover, I prepared the unique technical exhibition of holograms that has been presented to public since then many times.

In the second subject, lead by prof. Dr. Ing. Ivana Kolingerová, I was allowed to learn aspects of floating point arithmetic; I summarized it in the technical report [7] (not included as an appendix here as it is too long). This gave me a chance to spot and solve a numerical problem common to many algorithms of computer generated holography. In short, a common knowledge states that certain calculations should be done in double precision arithmetic, while single precision is sometimes sufficient – regrettably without stating precisely which calculations and when it is sufficient. I was able to analyse the problem, to give the precise prediction of problems and to describe why certain calculations suffer from single precision while others do not. The results were published in [4] (together with P. Vaněček; included as the Appendix D on page 43).

Finally, in the third subject, lead by doc. Ing. Marek Brandner, Ph.D., I was allowed to learn about numerical methods of light simulation, which gave me much deeper insight to computer generated holography. The results are summarized in an unpublished document that tries to explain numerical

methods of light simulation from zero to advanced techniques such as linear canonical transforms (the text is not included as an appendix here as it is too long).

Here I again realized that a similar document on computer generated holography is simply missing – there are indeed monographs about optical holography, about technical digital holography (such as digital holographic microscopy or digital holographic interferometry), but there is no introductory text on computer generated display holography.

Every Ph.D. student is obliged to describe "state of the art". I must confess I have read many STAR reports and I consider most of them useless for a common reader; they are usually very brief and hardly useful for a non-expert. It is expected that a reviewer of my Ph.D. STAR report will not be an expert in computer generated display holography, as there are quite a few of them. It follows it would be desirable to write an "extended STAR report" that would be suitable even for a non-expert, and could be eventually considered as a textbook. This is why I wrote the text "Computer generated display holography". It does not reference every published idea, but tries to explain the important ideas step by step; I consider it a first draft of a book I would like to publish.

The form of the "book" reflects my experience with teaching holography, both computer generated and optical. I explained holography to many people (hundreds at least) in talks targeted to a various audience – from popularizing public talks to invited university lectures. I also explained holography to dozens of students in intensive, "face to face" courses – the audience covers both undergraduate students and university/industry researchers, both computer scientists and opticians. I learned that every type of a student lacks some fundamental knowledge – opticians have a poor understanding of digital signal processing, computer scientists do not understand wave optics, undergraduate students usually do not understand Fourier analysis, and so on. This is the main reason why the presented "book" seems to explain obvious facts more than necessary – the truth is that *obvious* for an optician is not *obvious* for a computer scientist and vice versa.

I am aware that the presented "book" is far from perfect; especially Chapters 5 (Light propagation calculation), 6 (Computer generated holography algorithms for 3-D scenes) and 7 (Computer generated hologram display) should be expanded considerably. I would like to finish them, but not necessarily as a part of my Ph.D. study.

Instead, I would like to continue introducing new ideas based on my results to computer generated holography/digital holography. I have published a method that significantly accelerates calculation speed of radially symmetric functions, e.g. light propagation kernels; see [2] (included as the Appendix B on page 23). Unfortunately, the publishers decided to limit length of the paper to four pages; the original description and analysis of the algorithm (14 pages plus

appendices) now waits for a small enhancement, then it will be resubmitted for publication elsewhere.

I have also derived a method for improved light approximation in the frequency domain that could be used in e.g. computer generated holographic stereogram calculation. The method has to be implemented and tested.

Conference papers (in order of importance)

[1] Lobaz, P. "Discrete calculation of the off-axis angular spectrum based light propagation". In: *Journal of Physics: Conference Series*. Vol. 415. 1. 2013, p. 012040.

[2] Lobaz, P. "Double lookup table method for fast light propagation calculations". In: *Proceedings of the 10th International Symposium on Display Holography*. 2015, in press.

Journal papers (in order of importance)

[3] Lobaz, P. "Memory-efficient reference calculation of light propagation using the convolution method". In: *Opt. Express* 21.3 (Feb. 2013), pp. 2795–2806. DOI: 10.1364/OE.21.002795.

[4] Lobaz, P. and Vaněček, P. "Safe range of free space light propagation calculation in single precision". In: *Opt. Express* 23.3 (Feb. 2015), pp. 3260–3269. DOI: 10.1364/OE.23.003260.

[5] Lobaz, P. "Reference calculation of light propagation between parallel planes of different sizes and sampling rates". In: *Opt. Express* 19.1 (Jan. 2011), pp. 32–39. DOI: 10.1364/OE.19.000032.

[6] Lobaz, P. and Kovář, L. "Binarizace počítačem generovaného hologramu pomocí ditheringu [Binarisation of a computer generated hologram using dithering]". In: *Jemná mechanika a optika* 56.10, 11-12 (2011), pp. 290, 303–305. ISSN: 0447-6441.

Conference tutorial

- *Computer generated holography for computer graphics.* [In English] A tutorial presented at the 3DTV Conference 2011, May 15, Antalya, Turkey. DOI: 10.1109/3DTV.2011.5877153.

Selected talks (in order of year)

- *Fourier transform and associated transforms in optics.* [In Czech] Institute of Mathematics of the Academy of Sciences of the Czech Republic, 2015.
- *Holography and computer science.* [In Czech] Czech Technical University, Prague, Czech Republic, 2015.

- *Computer generated holography: 3D vision and beyond.* [In Czech] Series of six lectures. University of West Bohemia, Pilsen, Czech Republic, 2013.
- *Computer generated holography: 3D vision and beyond.* [In Czech] Comenius University in Bratislava, Bratislava, Slovakia, 2013.
- *Computer generated holography.* [In English] Czech Technical University in Prague, Prague, Czech Republic, 2012.
- *Computer generated holography for computer graphics.* [In English] University of Maribor, Maribor, Slovenia, 2011.
- *Computer generated holography: 3D vision and beyond.* [In English] University of Minho, Braga, Portugal, 2011.
- *Computer generated holography.* [In Czech] The Palacký University, Olomouc, Czech Republic, 2010.

Selected popular talks and exhibitions (in order of year)

- *Hologramy [Holograms].* [In Czech] A workshop for grammar school teachers on using holography in physics education. At 13th International conference "Dílny Heuréky – Náchod 2014", Czech Republic.
- *Holograms.* An exhibition of art holograms (concepts by students of Intermedia art at the University of West Bohemia, holographer P. Lobaz) at Galerie Září, Prague, Czech Republic, 2014.
- *Holografie pro střední školy [Holography for grammar schools].* [In Czech]. A seminar and a hands-on course for grammar school teachers held in Hazuka hotel, Pilsen, Czech Republic, 2014.
- *Holography.* [In Czech] 3D Film Fest Prague, Prague, Czech Republic, 2013.
- *Holography.* An exhibition and hands-on courses at Dny vědy a techniky (Days of Science and Technology), Pilsen, Czech Republic, 2013.
- *Kouzlo? Ne. Holografie [Magic? No. Holography].* Half technical, half artistic exhibition of holography. Muzeum jižního Plzeňska v Blovicích, Blovice, Czech Republic, 2013.

Appendix A

Discrete calculation of the off-axis angular spectrum based light propagation

Journal of Physics: Conference Series **415** (2013) 012040 doi:10.1088/1742-6596/415/1/012040

Discrete calculation of the off-axis angular spectrum based light propagation

P Lobaz

Department of Computer Science and Engineering, University of West Bohemia, Univerzitni 8, 306 14 Plzen, Czech Republic

E-mail: lobaz@kiv.zcu.cz

Abstract. Light propagation in a free space is a common computational task in many computer generated holography algorithms. A solution based on the angular spectrum decomposition is used frequently. However, its correct off-axis numerical implementation is not straightforward. It is shown that for long distance propagation it is necessary to use digital low-pass filtering for transfer function calculation in order to restrict source area illumination to a finite area. It is also shown that for short distance propagation it is necessary to introduce frequency bands folding in transfer function calculation in order to simulate finite source area propagation. In both cases it is necessary to define properly interpolation filters that reconstruct continuous nature of the source area out of its sampled representation. It is also necessary to zero-pad properly source area sampling in order to avoid artifacts that stem from the periodic nature of the fast Fourier transform.

1. Introduction

To calculate coherent light propagation in a free space, scalar approximation is used frequently. A common task is such a calculation where complex amplitudes of light are given in the area *source* in a plane $z = 0$ and we look for complex amplitudes in the area *target* in a plane $z = z_0$, $z_0 > 0$. A common procedure leads to the Rayleigh-Sommerfeld integral of the first kind [1], or to its mathematically equivalent form, the angular spectrum decomposition [2]. Approximations of these formulas are used frequently, namely Fresnel and Fraunhofer approximations. However, these approximations are used in paraxial regime while in computer generated holography an off-axis solution is often needed (e. g. [3]); therefore we will not discuss them.

The problem has to be solved numerically in computer generated holography. This leads to discretization of signals. Discretization of the Rayleigh-Sommerfeld solution is not straightforward [4, 5] and discretization of the angular spectrum decomposition is tricky [6, 7]. Onural [6] describes the discretization process in general and shows that it leads to formation of signal copies in spatial domain that can be filtered out. However, he does not discuss implementation issues. Matsushima [7] deals with the implementation and solves a troublesome aliasing problem by local frequency estimation; however, he does not analyse the effect of hard frequency clipping.

This article focuses on reference calculation of light propagation between parallel planes using angular spectrum decomposition. We will point out what makes the discretization difficult

and how to overcome the problems so that the method provides the same results as the Rayleigh-Sommerfeld method. We will deal with both large propagation distances discussed by Matsushima [7] and small propagation distances that were not discussed in literature yet. We will explain the meaning of "large" and "small" distance later.

Structure of the article is as follows. At first we will precisely show how to discretize the propagation calculation based on convolution (i. e. based on Rayleigh-Sommerfeld integral). We will show that it is necessary to introduce spatial limitation of a convolution kernel for successful discretization. This leads to frequency limitation of the transfer function used in the angular spectrum decomposition method. We will show that the frequency limited transfer function can be calculated using digital signal processing methods. At last we will deal with the case where this frequency limitation is actually useless due to small propagation distance. We will show that, in this case, we have to deal with exact nature of sampling and reconstruction of signals involved. We will show that the choice of reconstruction leads to introduction of artificial alias.

2. How to read the article

The mathematical explanation presented may be unpleasant to follow. Readers are therefore encouraged to go through the presentation packed as a "multimedia" attachment to this article. It shows pictures containing various problems that appear when calculating the propagation numerically. I have decided to attach these pictures as a separate media for two reasons. The first one is: the pictures show mainly problems with aliasing. It is therefore needed to control the display of these images precisely which is not possible in a PDF reader or in printed media. The second one is: separate media gives the opportunity to show much more images than any printed media allows. I should note that the presentation does not contain any information not covered by the article.

3. Convolution discretization

Let us assume that we know complex amplitudes $u(x, y, 0)$ of monochromatic coherent light of wavelength λ in the area *source* ($x_{s\min} \leq x < x_{s\max}$, $y_{s\min} \leq y < y_{s\max}$, $z = 0$) and we want to calculate complex amplitudes $u(x, y, z_0)$ in the area *target* ($x_{t\min} \leq x < x_{t\max}$, $y_{t\min} \leq y < y_{t\max}$, $z = z_0 > 0$). The solution is given by the Rayleigh-Sommerfeld integral of the first kind:

$$u(x, y, z_0) = \frac{-1}{2\pi} \iint_{-\infty}^{\infty} u(\xi, \eta, 0) \frac{\partial}{\partial z} \frac{\exp(\mathrm{j}kr)}{r} \,\mathrm{d}\xi \,\mathrm{d}\eta \tag{1}$$

where $r = \sqrt{(x-\xi)^2 + (y-\eta)^2 + z_0^2}$, $k = 2\pi/\lambda$, $\mathrm{j}^2 = -1$ and $u(\xi, \eta, 0) = 0$ for $[\xi, \eta, 0] \notin$ *source*. The second term of the multiplication inside the integral depends on $(x - \xi)$ and $(y - \eta)$ only, which means that the integral can be rewritten as the convolution with the Rayleigh-Sommerfeld kernel $h(x, y, z)$:

$$u(x, y, z_0) = u(x, y, 0) \otimes h(x, y, z_0) = \iint_{-\infty}^{\infty} u(\xi, \eta, 0) h(x - \xi, y - \eta, z_0) \,\mathrm{d}\xi \,\mathrm{d}\eta \tag{2}$$

$$h(x, y, z) = \frac{-1}{2\pi} \frac{\partial}{\partial z} \frac{\exp(\mathrm{j}kr)}{r} = \frac{-z}{2\pi} \left(\mathrm{j}k - \frac{1}{r}\right) \frac{\exp(\mathrm{j}kr)}{r^2}$$

$$r = \sqrt{x^2 + y^2 + z^2}$$

The kernel $h(x, y, z)$ can be interpreted easily: the value $h(x_d, y_d, z_d)$ describes the change of the complex amplitude of light travelling from the point $[x_s, y_s, z_s]$ to the point $[x_s + x_d, y_s + y_d, z_s + z_d]$.

To calculate $u(x, y, z_0)$ in the area *target* using (1) we have to know the kernel $h(x, y, z_0)$ for $x_{t\min} - x_{s\max} \leq x < x_{t\max} - x_{s\min}$, similarly for y. The value $h(x, y, z_0)$ is not important for other x, y because in this case $u(x, y, 0) = 0$. We will use this fact in a while.

We discretize (1) easily by changing integrals to sums and differentials to differences. The sums will have finite extent of the indices thanks to (in fact) finite domain of the integration. Therefore their calculation will be easy, although the computational complexity will be high.

To reduce computational complexity, let us rewrite the equation (2):

$$u(x, y, z_0) = u(x, y, 0) \otimes h(x, y, z_0) = \mathcal{F}^{-1}\left\{\mathcal{F}\{u(x, y, 0)\} \cdot \mathcal{F}\{h(x, y, z_0)\}\right\} \tag{3}$$

where $\mathcal{F}$ and $\mathcal{F}^{-1}$ are 2-D Fourier and inverse Fourier transform respectively. We will try to use the discrete Fourier transform (DFT) implemented as the fast Fourier transform (FFT) after discretization.

The discrete Fourier transform can be defined if the original continuous functions are periodic before discretization (sampling); then we take into account one period of functions $u_p(x, y, z_0)$, $u_p(x, y, 0)$ and $h_p(x, y, z_0)$ derived from functions $u(x, y, z_0)$, $u(x, y, 0)$ and $h(x, y, z_0)$. We can define the DFT another way if the functions to be transformed are spatially limited; in this case we can assume just one period of the functions $u_p(x, y, z_0)$, $u_p(x, y, 0)$ a $h_p(x, y, z_0)$. Both ways lead to the same results. This means that the results of the DFT can be interpreted in both ways. We will choose the way that will be more suitable in a particular situation.

Let us briefly give a hint what is the meaning of the functions $u_p(x, y, 0)$, $u_p(x, y, z_0)$ and $h_p(x, y, z_0)$ before we define them precisely. Their period in x direction is equal to the sum of widths of the *source* and the *target* (similarly in y). One period of the function $u_p(x, y, 0)$ is composed of values of $u(x, y, 0)$ in the way that one corner of the *source* is translated to the origin. The meaning of the function $u_p(x, y, z_0)$ is similar. The $h_p(x, y, z_0)$ is restricted and shifted version of the function $h(x, y, z_0)$. The meaning of the value $h_p(0, 0, z_0)$ is the change of the complex amplitude of light travelling from the point $[x_{s\min}, y_{s\min}, 0]$ to the point $[x_{t\min}, y_{t\min}, z_0]$, see fig. 1.

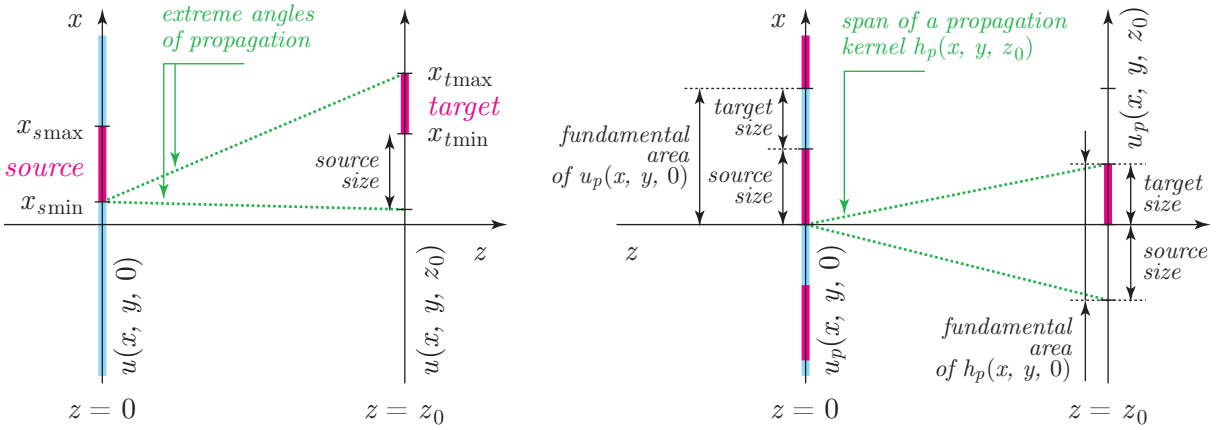


Figure 1. Geometry of the problem in xz slice. *Left image:* original setup. Red line in the source area denotes the complex amplitudes to be propagated, blue line in the same plane defines zero value. It should be clear that we need to know the values of a propagation kernel contained in a green wedge only. *Right image:* setup prepared for discretization. *Source* and *target* are shifted to the z axis, blue and red lines in the *source* plane show both zero padding and periodicity. The green wedge shows one period (fundamental area) of the function $h_p(x, y, z_0)$.

Let us define the functions precisely. The function $h_p(x, y, z_0)$ is defined in "the fundamental area" $x_{s\min} - x_{s\max} \leq x < x_{t\max} - x_{t\min}$ (similarly for y) as:

$$h_p(x, y, z_0) = h(x + x_{t\min} - x_{s\min}, y + y_{t\min} - y_{s\min}, z_0) \tag{4}$$

Definition outside the fundamental area depends on situation: it can be either zero or the function can be made periodic. However, as we will see, the values outside the fundamental area are not important, which means that the periodic nature of the function is not harmful.

Let us define the function $u_p(x, y, 0)$ in the fundamental area $0 \leq x < (x_{t\max} - x_{t\min}) + (x_{s\max} - x_{s\min})$ (similarly for y; notice that the size of the area is the same as for $h_p(x, y, z_0)$) as:

$$u_p(x, y, 0) = \begin{cases} u(x + x_{s\min}, y + y_{s\min}, 0) & \text{for } 0 \leq x < x_{s\max} - x_{s\min}, \\ & \quad 0 \leq y < y_{s\max} - y_{s\min} \\ 0 & \text{elsewhere in the fundamental area} \end{cases}$$

and again let us make it periodic.

Let us calculate $u_p(x, y, z_0) = u_p(x, y, 0) \otimes h_p(x, y, z_0)$. If we consider the function $u_p(x, y, 0)$ to be periodic and the function $h_p(x, y, z_0)$ to be zero outside the fundamental area, we can easily prove that $u_p(x, y, z_0)$ will contain correct result of the propagation from the *source* to the *target* in the area $0 \leq x < x_{t\max} - x_{t\min}$ (similarly for y); the values of $h_p(x, y, z_0)$ are meaningless for other x, y as they are damaged by periodicity of the function $u_p(x, y, 0)$. The proof easily follows from the geometry of the problem.

If we consider the function $u_p(x, y, 0)$ to be spatially limited and the function $h_p(x, y, z_0)$ to be periodic, we get the same result. If we consider both functions to be periodic, the meaning of the result remains the same; however, we are in fact in the world of discrete Fourier transform.

The form of the discrete calculation follows from the aforementioned ideas. The area *source* is discretized by $M_x \times M_y$ samples, the area *target* by $N_x \times N_y$ samples. For simplicity let us assume such parameters so that the sampling period Δ is the same in both directions x and y and in both *source* and *target* areas. Its value is then e. g. $\Delta = (x_{s\max} - x_{s\min})/M_x$.

The area *source* is discretized by samples $u_0[m, n] = u_p(m\Delta, n\Delta, 0)$, the convolution kernel is discretized by samples $h[m, n] = h_p(m\Delta, n\Delta, z_0)$ for $m \in \{0, 1, \ldots, M_x + N_x - 2\}$ (similarly n, see [4]). We consider functions $u_p(x, y, 0)$ and $h_p(x, y, z_0)$ to be periodic. Then we can calculate

$$u_{z_0}[\,] = \mathbf{IDFT}\left\{ \mathbf{DFT}\{u_0[\,]\} \cdot \mathbf{DFT}\{h[\,]\} \right\} \tag{5}$$

to get the array $u_{z_0}[\,]$ that contains complex amplitudes of the area *target* in the first $N_x \times N_y$ elements. In the equation (5), **DFT** and **IDFT** stands for forward and backward 2-D discrete Fourier transform, respectively, and $\cdot$ stands for elementwise product (Hadamard product).

We can summarize the results as follows. For the discrete calculation of light propagation, we have to use a convolution kernel spatially limited to a certain area. The *source* has to be zero-padded to span the same area. The result of their convolution contains correct values in the area of the size proportional to the size of the *target*.

4. Angular spectrum discretization for large propagation distances

The equation (1) can be written in the equivalent form called the angular spectrum decomposition [2]:

$$u(x, y, z_0) = \mathcal{F}^{-1}\{U(f_x, f_y, 0) \cdot H(f_x, f_y, z_0)\}$$

where

$$U(f_x, f_y, 0) = \mathcal{F}\{u(x, y, 0)\}$$
$$H(f_x, f_y, z_0) = \exp\left(-\mathrm{j}\, 2\pi z_0 \sqrt{\lambda^{-2} - f_x^2 - f_y^2}\right)$$

and f_x, f_y are Fourier domain variables.

It follows from (3) that

$$u(x, y, z_0) = u(x, y, 0) \otimes h(x, y, z_0) = \mathcal{F}^{-1}\left\{U(f_x, f_y, 0) \cdot \mathcal{F}\{h(x, y, z_0)\}\right\},$$

that is $H(f_x, f_y, z_0) = \mathcal{F}\{h(x, y, z_0)\}$. It seems that the propagation calculation should be advantageous using the angular spectrum decomposition compared to convolution with the Rayleigh-Sommerfeld kernel because we have one Fourier transform less – the transform of the kernel is known in the analytic form.

In the following paragraphs, we will talk about various limitations of functions in both spatial and Fourier (frequency) domain. A function $f(x, y)$ defined in spatial domain is spatially limited if it is zero outside a bounded area in the plane (x, y); it is frequency limited if $\mathcal{F}\{f(x, y)\}$ is zero outside a bounded area in the plane (f_x, f_y). Similarly, a function $F(f_x, f_y)$ defined in frequency domain is spatially limited if it is zero outside a bounded area in the plane (f_x, f_y); it is frequency limited if $\mathcal{F}^{-1}\{F(f_x, f_y)\}$ is zero outside a bounded area in the plane (x, y).

We know from the previous section that we have to introduce certain functions to discretize the calculation. We have to define the periodic function $u_p(x, y, 0)$ based on the function $u(x, y, 0)$ and to calculate its (discrete) Fourier transform; we will follow this procedure exactly. We also need to spatially restrict the function $h(x, y, z_0)$, and to make its periodic form alternatively. However, we do not know the Fourier transform of the function $h_p(x, y, z_0)$ in the analytic form.

Fortunately we can use digital signal processing tools. If the signal is spatially limited in one domain (in our case spatial domain), it is frequency limited in the other domain (in our case frequency domain). We can calculate frequency limited signal using low-pass filter $l(f_x, f_y)$. Let us assume the function $h_p(x, y, z_0)$ to be spatially limited (i. e. not periodic). For propagation calculation, we have to use the transfer function $H_p(f_x, f_y, z_0) = H(f_x, f_y, z_0) \otimes l(f_x, f_y)$.

It follows from properties of the Fourier transform that the function $H_p(f_x, f_y, z_0)$ has to be spatially unlimited. This does not matter even in numerical calculation. Since the function $u_p(x, y, 0)$ is periodic, then $U_p(f_x, f_y, 0)$ is spatially limited; and we need to calculate the product $U_p(f_x, f_y, 0)H_p(f_x, f_y, 0)$. It also follows that the value of the function $H_p(f_x, f_y, z_0)$ can be arbitrary outside the important area, and therefore we can use its periodic form to introduce discrete calculation correctly.

We cannot limit the frequency content of the function $H(f_x, f_y, z_0)$ sharply using digital low-pass filtering; the frequency limitation is approximate. In the spatial domain it means that the transition between zero and non-zero part is gradual instead of sharp. This does not matter either. All we need to do is to enlarge the fundamental area of the functions $u_p()$ and $h_p()$, i. e. zero-padding of the array $u_p[]$ will be larger than defined in section 3, so that the gradual transition will not affect the *target* area.

We will face a problem in a practical implementation. The function $H(f_x, f_y, z_0)$ is frequency unlimited – the local frequency [1, 7] in the point $[f_x, f_y]$ grows without bound as this point approaches a circle of a radius λ^{-1} centered in the origin. Moreover, if the propagation distance z_0 is large, then the local frequency will be large as well everywhere except in the origin. Therefore it is impossible to sample the function $H(f_x, f_y, z_0)$ correctly and then the low-pass filtering will not work properly.

However, we can use the same procedure as described by Matsushima [7] who realizes the aliasing problem. He evaluates local frequency when sampling the function $H(f_x, f_y, z_0)$, and if the local frequency is bigger than a half of the sampling frequency, he sets the function $H(f_x, f_y, z_0)$ to be zero. He solves the aliasing problem this way, on the other hand he introduces a spatial limitation of the function $H(f_x, f_y, z_0)$. It follows that the propagation kernel $\mathcal{F}^{-1}\{H_{\text{Matsushima}}(f_x, f_y, z_0)\}$ is then spatially unlimited which is not correct. It should be however emphasised that even though it gives remarkably good results.

The solution is easy. We can sample the function $H(f_x, f_y, z_0)$ using higher sampling frequency than desired and use Matsushima's procedure to avoid alias. Then we can filter it using $l(f_x, f_y)$ and downsample the result to get a correct sampling frequency. As the cutoff frequency of the filter $l(f_x, f_y)$ is derived from the sizes of the arrays used in the calculation, it is guaranteed that no aliasing appears when downsampling.

It follows from practical experiments that it is sufficient to sample the function $H(f_x, f_y, z_0)$ using sampling frequency twice as high as desired, and to use sinc low-pass filter with Hamming window as $l(f_x, f_y)$. The size of the filter should be chosen carefully – a long filter filters high frequencies well, but takes long to evaluate. Evaluation with a short filter is faster but slow attenuation of high frequencies has to be compensated with bigger zero-padding of the array $u_0[\,]$.

5. Discretization for short propagation distances

To calculate the propagation numerically (either using convolution approach or angular spectrum approach), we have to calculate (discrete) Fourier transform of the function $u_p(x, y, 0)$. Its result is the function $U_p(f_x, f_y, 0)$ limited to the area $A = (-1/(2\Delta), 1/(2\Delta)) \times (-1/(2\Delta), 1/(2\Delta))$, or its periodic form. We also need to calculate the function $H_p(f_x, f_y, z_0)$ in the same area, and then to calculate $\mathcal{F}^{-1}\{U_p(f_x, f_y, 0)H_p(f_x, f_y, z_0)\}$.

We can naturally ask a question: what happens if the propagation distance z_0 is so small that the low-pass filtering of the function $H(f_x, f_y, z_0)$ will not have any significant effect inside the area A? It is not easy to find the answer. Let us start with one more look into the convolution based approach described by the equation (2).

The convolution kernel $h(x, y, z_0)$ is a spatially unlimited function that is in fact frequency limited if we ignore evanescent waves. We can easily show that its local frequency grows as the point $[x, y]$ moves away from the origin. The smaler is z_0 the faster is the growth. It can therefore easily happen that the function $h_p(x, y, z_0)$ cannot be properly sampled using sampling period Δ for small propagation distances.

Physical meaning of wrong sampling is easy. Discretization is based on change of integrals to sums in the equation (1). It means that we change a continuous light field in the *source* to a number of point light sources. This replacement cannot be observed from a big distance or in on-axis case but makes a big difference close enough or in off-axis case.

We have to assume (especially in small propagation distances) that one sample of the function $u(x, y, 0)$ represents behaviour of the light field in a small neighbouring area. Let us denote the result of the sampling of the function $u(x, y, 0)$ by the function $u_s(x, y, 0) = u(x, y, 0)\,\mathrm{comb}(x/\Delta)\,\mathrm{comb}(y/\Delta)$, where $\mathrm{comb}(x)$ is the sampling function with period of samples 1 (see [1]). Then we can describe the reconstruction of the continuos form using convolution with a reconstruction kernel $r(x, y)$:

$$u(x, y, 0) \approx u_r(x, y, 0) = u_s(x, y, 0) \otimes r(x, y)$$

where $u_r(x, y, 0)$ is a continuos function reconstructed from the discrete samples. The function $u_r(x, y, 0)$ is more or less similar to the function $u(x, y, 0)$ depending on the shape of a reconstruction kernel $r(x, y)$ and size of the sampling period Δ.

We can therefore express the propagation as

$$u(x, y, z_0) \approx u_s(x, y, 0) \otimes \Big(r(x, y) \otimes h(x, y, z_0)\Big)$$

It is possible to put big parenthesses in the equation thanks to associativity of the convolution. It follows that we should not use the function $h(x, y, z_0)$ for discrete propagation calculation, but we shoud use its filtered version $r(x, y) \otimes h(x, y, z_0)$ instead.

It is most common to demand for one sample of the function $u(x, y, 0)$ to influence its close neighbourhood only. For example, the kernel $r(x, y) = \mathrm{rect}(x/\Delta)\,\mathrm{rect}(y/\Delta)$, where $\mathrm{rect}(x)$ is a rectangular pulse of unity width and height centered in the origin, implies the function $u(x, y, 0)$ to be approximated with a piecewise constant function. We can construct other kernels as well that provide piecewise bilinear approximation, piecewise bicubic approximation and so on. In either case, the kernel $r(x, y)$ acts as a low-pass filter. If we use the filtered function $h(x, y, z_0) \otimes r(x, y)$ for construction of the function $h_p(x, y, z_0)$ in the equation (4), we get the result of the propagation calculation as precise as the function $u_r(x, y, 0)$ resembles the function $u(x, y, 0)$. Practical implementation of the procedure is described in [5].

We can repeat the analysis for the angular spectrum decomposition as well. We should not use the function $H(f_x, f_y, z_0)$ for construction of the function $H_p(f_x, f_y, z_0)$; we should use the Fourier transform of the filtered propagation kernel, the function $\mathcal{F}\{h(x, y, z_0\} \otimes r(x, y)\} = H(f_x, f_y, z_0)\,\mathcal{F}\{r(x, y)\}$. If we choose convenient kernel $r(x, y)$, we will know analytic form of its Fourier transform and calculation of the product will be simple. This step limits high frequencies that were caused by the discretization process.

It remains to solve the last, but important detail. In the beginning of the section we have stated that the calculations in frequency domain are done inside the area A. It is however possible that the support of the function $H_p(f_x, f_y, z_0)$ will not fit into the area A even if it was filtered with both filters $l(f_x, f_y)$ and $r(x, y)$. If we reject the values of the function $H_p(f_x, f_y, z_0)$ outside of A despite that fact, it means that the final function $u(x, y, z_0)$ was filtered by a third, still unjustified low-pass filter.

We can explain this final low-pass filter. It would be appropriate if the kernel $r(x, y)$ represents a perfect sinc low-pass filter limiting the frequency content of the function $u(x, y, z_0)$ to a range described by the area A. Then the kernel $r(x, y)$ has to be be spatially unlimited. If we do not care, the analysis is finished.

If we would rather keep the kernel $r(x, y)$ spatially limited (which is physically more natural), we have two choices to choose from. The first one is simple – we can use such a sampling period Δ/s, $s \in \mathbf{N}$ for the calculation so that the area A covers the support of $H_p(f_x, f_y, z_0)$ now. This leads to increase of time and memory demands of the calculation, of course. Moreover, if we demand sampling of the *target* to be Δ, we have to downsample the result; we have to admit that a lot of values were calculated needlessly.

The second one is a bit strange at first sight. Signal downsampling in spatial domain can be described easily in frequency domain – the frequencies f and $f + n/\Delta$ merge due to downsampling, where f is frequency f_x or f_y from the range $(-1/(2\Delta), 1/(2\Delta))$ and $n \in \mathbf{Z}$. This effect is called aliasing. We want to avoid it usually; however, if we are decided to sample the *target* with an insufficient sampling period, we have to accept aliasing. It is worth to note that aliasing needs not be harmful. If we are interested in intensities $|u(x, y, z_0)|^2$ only, aliasing in the real or imaginary part of the function $u(x, y, z_0)$ may be harmless. For example, if $u(x, y, z_0) = \cos x + \mathrm{j} \sin x$, then the intensity is always 1 regardless sampling period used, while intensity of "correctly sampled" (i. e. low-pass filtered) version can be either 1 or 0 which is not correct.

The procedure in the second case is obvious: we will perform the downsampling process in frequency domain. Let us calculate the function $H_p(f_x, f_y, z_0)$ in the area $(-s/(2\Delta), s/(2\Delta)) \times (-s/(2\Delta), s/(2\Delta))$, assume it is zero outside this area, and calculate

$$H_a(f_x, f_y, z_0) = \sum_{n_x, n_y} H_p(f_x + \frac{n_x}{\Delta}, f_y + \frac{n_y}{\Delta}, z_0)$$

for $[f_x, f_y] \in A$ and all suitable n_x, n_y. To calculate the propagation, we have to use the function $H_a(f_x, f_y, z_0)$.

6. Conclusion

The analysis shows how to numerically calculate the propagation of light using the angular spectrum decomposition method. Unlike the procedures described in the literature, it defines low-pass filters $l(f_x, f_y)$ and $r(x, y)$ that are needed to introduce for correct discretization. Then we can choose either more precise, slower calculation or faster, less precise by choosing their parameters. The analysis also shows that it is worth introducing aliasing of the transfer functions in certain situations. The results are shown in figures 2 and 3. The images may display wrong due to resampling; the reader is encouraged to look at the images in the multimedia attachment.

References

[1] Goodman J W 2004 *Introduction to Fourier Optics* 3rd ed (Roberts & Company Publishers) ISBN 0974707724
[2] Lalor E 1968 *J. Opt. Soc. Am.* **58** 1235–1237
[3] Hanák I, Janda M and Skala V 2010 *Visual Computer* **26**(2) 83–96
[4] Lobaz P 2011 *Opt. Express* **19** 32–39
[5] Lobaz P 2012 Calculation of a coherent light propagation in a free space using filtered convolution Tech. rep. University of West Bohemia URL `http://www.kiv.zcu.cz/en/research/publications/`
[6] Onural L 2007 *J. Opt. Soc. Am.* A **24** 359–367
[7] Matsushima K 2010 *Opt. Express* **18** 18453–18463

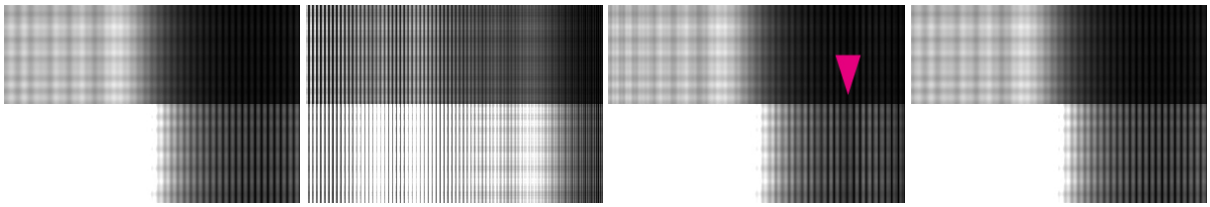


Figure 2. Light ($\lambda = 650$ nm) diffracted by a grating (3×3 mm^2) composed of vertical slits (slits distance 40 μm). Off-axis propagation to a distance of 300 mm, size of each image 3×3 mm^2. Lower half of each image is overexposed to show the details. *a)* Reference Rayleigh-Sommerfeld calculation. *b)* Angular spectrum based calculation without any modification. Notice that aliasing errors destroy the image completely. *c)* Angular spectrum based calculation with Matsushima's kernel filtering. Notice different brightness of fine stripes compared to the reference image. *d)* Angular spectrum based calculation with $l(f_x, f_y)$ sinc kernel of length 50.

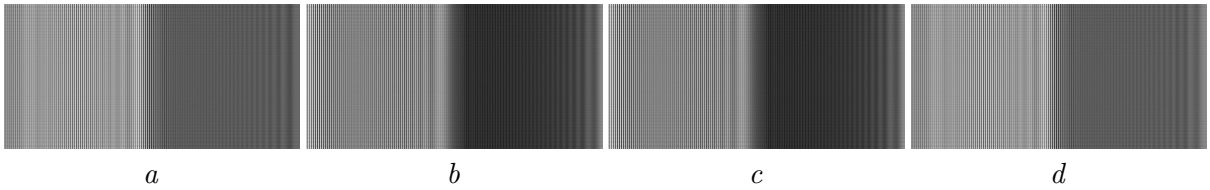


Figure 3. Light ($\lambda = 650$ nm) diffracted by a grating (3×3 mm^2) composed of vertical slits (slits distance 20 μm). Off-axis propagation to a distance of 50 mm, size of each image 3×1.5 mm^2. *a)* Reference Rayleigh-Sommerfeld calculation. *b)* Angular spectrum based calculation without any modification. Notice that the right half is much darker than in the reference image. *c)* Angular spectrum based calculation with Matsushima's kernel filtering. Notice that the image is the same as without any modification because aliasing did not occur. *d)* Angular spectrum based calculation with introduced aliasing. Notice that the image is almost the same as the reference one.

Appendix B

Double lookup table method for fast light propagation calculations

Double lookup table method for fast light propagation calculations

Petr Lobaz

Dept. of Computer Science and Engineering, University of West Bohemia, Czech Republic

E-mail: lobaz@kiv.zcu.cz

Abstract. A method of rapid and robust calculation of radially symmetric functions is presented. It is based on observation that $f(x, y, z) = f'(\rho, z)$, where $\rho = (x^2 + y^2)^{1/2}$. The method stores values of f' and ρ in look-up tables. It can be used e.g for fast computer generated hologram calculation. It is suitable for CPU, GPU or hardware implementation.

1. Introduction

In computer generated holography and digital holography, it is often necessary to evaluate radially symmetric functions of two variables such as convolution kernels for free space light propagation calculation, e.g. Rayleigh-Sommerfeld or Fresnel kernels [1]. There are several approaches how to accelerate their evaluation. First of all, a moderately complicated formula, such as the Rayleigh-Sommerfeld convolution kernel

$$K_{\mathrm{RS}}(x, y; z_0) = -\frac{1}{2\pi}\left(\mathrm{j}k - \frac{1}{r}\right)\frac{\exp(\mathrm{j}kr)}{r}\frac{z_0}{r}, \qquad \text{where } r = \sqrt{x^2 + y^2 + z_0^2} \tag{1}$$

can be approximated by a simpler formula, such as the Fresnel approximation. Such kernels are used to calculate light propagation from a plane $z = 0$ to a plane $z = z_0$; λ stands for a wavelength, $k = 2\pi/\lambda$ is a wave number, x and y are transverse spatial coordinates and j is the imaginary constant. Among acceleration, a simpler formula can have better numerical properties [2]. On the other hand, the approximation error must be taken into account.

Other acceleration method pre-calculates the function for every necessary x, y and z_0 and stores the values in a 3-D look-up table (LUT) [3, 4]. Some researchers do not calculate LUT at points where the function is not properly sampled, thus reduce its size [5]. Certain functions, e.g. the Fresnel convolution kernel, are separable, i.e. it holds $K(x, y; z_0) = K_x(x; z_0)K_y(y; z_0)$. In this case, it is sufficient to create one 2-D look-up table for each of the two factors K_x, K_y [6, 7, 8]. Indeed, this approach is not applicable for non-separable functions.

Some researchers try to accelerate function evaluation by using recurrence formulas, e.g. [9, 10]. It should be noted that influence of computer arithmetic rounding errors is usually poorly analysed and that recurrence formulas tend to produce a sequential computer code rather than a parallel one. An original approach to evaluation of radially symmetric function K is based on computer graphics algorithm for circle rasterisation [11]. Its biggest drawback is its complicated memory access, thus memory caching cannot be used efficiently. Recently, authors proposed a method that overcomes this difficulty using recurrence formulas [12].

Parts of the method we are going to analyse in following sections were independently described by other authors [12, 13, 14]. We will discuss these references after we describe the basic idea of the proposed method in Sec. 2. Detailed analysis is given in Sec. 3 and 4, results in Sec. 5.

2. Principle of the method

For light propagation calculation, $K_{\mathrm{RS}}(x, y; z_0)$ must be evaluated for a specific z_0 in a finite area of the xy plane. This area should be sampled using sampling distance Δ_{xy}. It is easy to see that $K_{\mathrm{RS}}(x, y; z_0)$ depends in fact on r and z_0, where z_0 is constant. Moreover, we can define $\rho(x, y) = (x^2 + y^2)^{1/2}$ and write $r = (\rho^2 + z_0^2)^{1/2}$ and $K_{\mathrm{RS}}(x, y; z_0) \equiv K_{\mathrm{RS}}(\rho; z_0)$.

To calculate $K_{\mathrm{RS}}(m\Delta_{xy}, n\Delta_{xy}; z_0)$, where m, n are integer indices, it is necessary to evaluate $\rho = \Delta_{xy}(m^2 + n^2)^{1/2}$ and $K_{\mathrm{RS}}(\rho; z_0)$. It is possible to calculate ρ directly or using a 2-D look-up table; we call this look-up table "rhoLUT". We can also build a look-up table with values $K_{\mathrm{RS}}(q\Delta_w; z_0)$ where Δ_w is sufficiently small and integer index q spans all possible values of ρ. We denote this look-up table "waveLUT" because it actually captures wave structure of light.

The waveLUT improves the calculation speed as the evaluation of K_{RS} or more complicated functions is usually slow. As any error in calculation of ρ can significantly affect the precision of the result, the pre-calculated rhoLUT improves numerical behaviour of the calculation.

The authors of [14] used a very similar method. Instead of rhoLUT, they calculated a look-up table directly for $r = (x^2 + y^2 + z_0^2)$, so they could not reuse it for other z_0 coordinate as in the case of rhoLUT. Moreover, sampling distance of this table was set to Δ_{xy}, and thus keeping this table for every z_0 would be memory inefficient. The authors of [13] use the table we call waveLUT here and although they in fact use rhoLUT as well (due to Matlab style of matrix manipulation), they do not discuss it explicitly, nor do they investigate its influence. The authors of [12] use waveLUT as well, but instead of rhoLUT, they use recurrence formulas to find values of ρ. They also do not discuss the influence of interpolation on calculation precision.

3. The rhoLUT analysis

The aim of the rhoLUT is to replace the calculation of $\rho(x, y) = (x^2 + y^2)^{1/2}$, $x \in [0, M\Delta_{xy}]$, $y \in [0, M\Delta_{xy}]$, by a look-up operation, optionally followed by an interpolation. Let us define it as a 2-D array $\mathrm{rhoLUT}[m, n] = \Delta_\rho\sqrt{m^2 + n^2}$ for integer indices $0 \le m, n < \lceil M\Delta_{xy}/\Delta_\rho \rceil$.

No interpolation is necessary for $\Delta_\rho = \Delta_{xy}$. For $\Delta_\rho \neq \Delta_{xy}$, either piecewise constant or piecewise bilinear interpolation can be used. It can be shown that the approximation error caused by the piecewise constant interpolation is not generally acceptable, as the approximate value of ρ is used for the K_{RS} calculation, where any error is greatly amplified. Piecewise bilinear interpolation gives much better results. If we denote $\rho_B(x, y)$ as a result of the rhoLUT look-up followed by the piecewise bilinear interpolation, it can be found that the approximation error $\rho_B(x, y) - \rho(x, y)$ vanishes for $x \to \infty$, $y \to \infty$, and its maximum is $\Delta_\rho(2 - \sqrt{2})/4 \approx 0.146\Delta_\rho$ located at $(x, y) = (\Delta_\rho/2, \Delta_\rho/2)$.

To select a small enough sampling distance Δ_ρ, it is not sufficient to examine error $\rho_B(x, y) - \rho(x, y)$ alone, as we need to take into account that approximate value ρ_B is used in subsequent calculation of K_{RS}. The most sensitive part of K_{RS} is $\exp(\mathrm{j}2\pi r/\lambda)$, because even a slight error in r is greatly amplified. It is thus necessary to analyse error $r(\rho_B, z_0) - r(\rho, z_0)$. Analysis reveals that this error is the biggest for ρ close to 0. As we know that $\rho_B(x, y) - \rho(x, y)$ is the biggest at $(x, y) = (\Delta_\rho/2, \Delta_\rho/2)$, we can guess that $r(\rho_B, z_0) - r(\rho, z_0)$ at this point estimates maximum error of r. We can thus define estimate of the maximum error of r as

$$\mathrm{maxRErr}(\Delta_\rho, z_0) = 1.2 \left\{ r\left[\rho_B\left(\frac{\Delta_\rho}{2}, \frac{\Delta_\rho}{2}\right), z_0\right] - r\left[\rho\left(\frac{\Delta_\rho}{2}, \frac{\Delta_\rho}{2}\right), z_0\right] \right\}, \tag{2}$$

where $r(\rho, z_0) = (\rho^2 + z_0^2)^{1/2}$ and factor 1.2 reflects the observation that the error of r for $(x, y) = (\Delta_\rho/2, \Delta_\rho/2)$ is approximately 85% of the maximum.

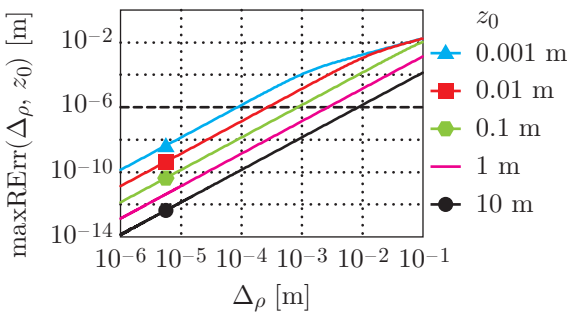


Figure 1. Approximate value of maximum error in calculation of r.

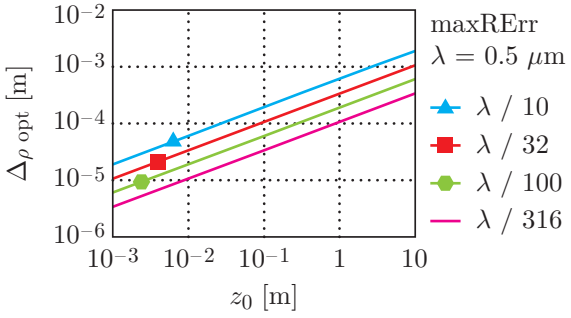


Figure 2. Optimal sampling distance $\Delta_{\rho\ \mathrm{opt}}$ calculated using Eq. (3).

It can be seen in Fig. 1 that in the log-log graph, maxRErr can be approximated by a line for a wide range of useful errors and Δ_ρ. Recall that for visible light, error of r must be well below 1 μm that is depicted by a thick dashed horizontal line.

A set of lines in a log-log graph is defined as $\log(\mathrm{maxRErr}) = \kappa \log(\Delta_\rho) + \xi_0 + \xi_1 \log(z_0)$, where κ is the slope and ξ_0 and ξ_1 define the intercept. We can measure in the graph that for a wide range of common z_0 and acceptable maxRErr, $\kappa \approx 2$, $\xi_0 \approx -2$ and $\xi_1 \approx -1$.

The linear approximation allows us to find optimal $\Delta_{\rho\ \mathrm{opt}}$ for a chosen maxRErr:

$$\Delta_{\rho\ \mathrm{opt}}(\mathrm{maxRErr}, z_0) = \exp\left(\frac{1}{\kappa} \log\left[\frac{\mathrm{maxRErr}}{(z_0)^{\xi_1} \exp(\xi_0)}\right]\right) \approx 2.72\sqrt{z_0\,\mathrm{maxRErr}}. \tag{3}$$

Example values of $\Delta_{\rho\ \mathrm{opt}}$ for $\lambda = 500$ nm can be found in Fig. 2. Note they are much larger than λ, thus rhoLUT size is usually small.

4. The waveLUT analysis

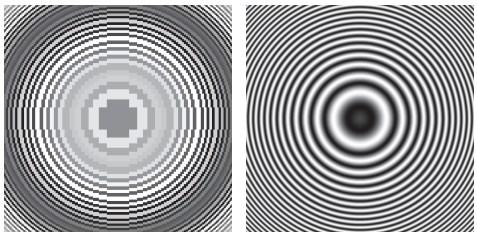
Once we have the approximate ρ value, we can calculate $K_{\mathrm{RS}}(\rho; z_0)$ using the waveLUT. We should set small enough Δ_w and define $\mathrm{waveLUT}[q; z_0] = K_{\mathrm{RS}}(q\Delta_w; z_0)$, where q is an integer index. Again, for a particular value ρ_0 we can estimate the value of $K_{\mathrm{RS}}(\rho_0; z_0)$ using piecewise constant approximation or piecewise linear approximation.

The extent of the waveLUT in the ρ direction is given by the rhoLUT. For a given maximum value $\rho_{\max}$, we can calculate the local frequency $\mathrm{lf}_{\mathrm{RS}}(\rho_{\max}; z_0)$ and set $\Delta_w < 1/[2\,\mathrm{lf}_{\mathrm{RS}}(\rho_{\max}; z_0)]$ to have at least two samples per cycle of $K_{\mathrm{RS}}(\rho; z_0)$. Our experiments show that good results are obtained with 8 samples per cycle and a piecewise linear interpolation.

5. Results

As shown in [2], direct calculation of highly oscillatory functions such as K_{RS} is prone to numerical error in single precision calculations. No problems appeared in single precision environment when using a rhoLUT and a waveLUT pre-calculated in double precision. See Fig. 3 for an example of both correct and incorrect K_{RS} calculation. Tests also show that selection of Δ_ρ and Δ_w according to Sections 3 and 4 works as expected.

We have also measured actual error of calculation of K_{RS}. Choosing 8 samples per fringe in selection of Δ_w and linear interpolation in the waveLUT leads to maximum error 7.0% (1.1% on average). While this number may seem high, it should be noted that the maximum error appears in the finest fringes and has a negligible impact on the optical field properties. Combination of the rhoLUT and the waveLUT further increases the error; typical value of $\mathrm{maxRhoErr} = \lambda/100$ leads to the maximum error 8.8% (2.8% on average), which is still acceptable for our purposes.



single precision *correct calculation*

Figure 3. Real part of K_{RS} calculated in IEEE 754 single precision arithmetic and in double precision arithmetic ("correct calculation"). Calculation using look-up tables leads to the same correct result.

Calculation time was tested in realistic geometric scenarios. We have prepared two test cases – one for complicated filtered propagation kernels (see [15]), and one for the Rayleigh-Sommerfeld kernel. Complicated filtered propagation kernel calculation is accelerated mainly by utilizing waveLUT; 10× or 100× faster calculation is easily achieved, depending on the complexity of the kernel. Using waveLUT in simple kernel calculation leads to about 1.7× faster calculation. These numbers include the waveLUT calculation, which takes about 1% of the overall time.

Introducing rhoLUT enhances numerical behaviour in single precision environment; in double precision environment, this advantage is not important, as ρ can be easily evaluated directly. Unoptimized implementation of the rhoLUT can actually double calculation time compared to direct ρ calculation and the waveLUT. On the other hand, careful rhoLUT implementation that uses integer arithmetic whenever possible leads to further 20% to 40% speed-up compared to direct ρ calculation and the waveLUT.

6. Conclusion

We have introduced a method of calculation of arbitrary radially symmetric functions using a pair of look-up tables, a rhoLUT and a waveLUT. While using a waveLUT is always advantageous, using a rhoLUT has its pros and cons. The rhoLUT enhances numerical behaviour in a limited precision environment, but it must be carefully implemented to improve the speed of calculation.

Acknowledgments

This work was supported by the European Regional Development Fund (ERDF), project "NTIS – New Technologies for the Information Society", European Centre of Excellence, CZ.1.05/1.1.00/02.0090, and by Ministry of Education, Youth, and Sport of Czech Republic – University spec. research – 1311. I would like to thank P. Vaněček for collaboration on implementation and testing and L. Váša for valuable comments.

References

[1] Goodman J W 2004 *Introduction to Fourier Optics* 3rd ed (Roberts & Company Publishers) ISBN 0974707724
[2] Lobaz P and Vaněček P 2015 *Opt. Express* **23** 3260–3269
[3] Kim S C and Kim E S 2008 *Appl. Opt.* **47** D55–D62
[4] Cho S, Ju B K, Kim N Y and Park M C 2014 *Optical Engineering* **53** 054108
[5] Shimobaba T, Nakayama H, Masuda N and Ito T 2010 *Opt. Express* **18** 19504–19509
[6] Pan Y, Xu X, Solanki S, Liang X, Tanjung R B A, Tan C and Chong T C 2009 *Opt. Express* **17** 18543–18555
[7] Esmer G B 2013 *Appl. Opt.* **52** A18–A25
[8] Kim S C, Kim J M and Kim E S 2012 *Opt. Express* **20** 12021–12034
[9] Yoshikawa H, Iwase S and Oneda T 2000 *Proc. SPIE* **3956** 48–55
[10] Matsushima K and Takai M 2000 *Appl. Opt.* **39** 6587–6594
[11] Nishitsuji T, Shimobaba T, Kakue T, Masuda N and Ito T 2012 *Opt. Express* **20** 27496–27502
[12] Nishitsuji T, Shimobaba T, Kakue T and Ito T 2015 *Opt. Express* **23** 9852–9857
[13] Huang Y, Zhao K, Yan X, Pei C and Jiang X 2014 *Journal of Electronic Imaging* **23** 043013
[14] Lee S, Wey H C, Nam D K, Park D S and Kim C Y 2012 *Proc. SPIE* **8498** 84980O
[15] Lobaz P 2013 *Opt. Express* **21** 2795–2806

Appendix C

Memory-efficient reference calculation of light propagation using the convolution method

Memory-efficient reference calculation of light propagation using the convolution method

Petr Lobaz
Department of Computer Science and Engineering, University of West Bohemia, Univerzitni 8, 306 14 Plzen, Czech Republic
lobaz@kiv.zcu.cz

Abstract: In computational Fourier optics, computer generated holography, etc., coherent light propagation calculation between parallel planes is the essential task. A proper calculation discretization in the off-axis case leads to big memory demands in order to avoid aliasing errors. The proposed method typically cuts down the memory demands one hundred times. The principle of the method is based on the observation that there is a close correspondence between the reconstruction process (opposite of the sampling process) and prefiltering of the convolution kernel.

OCIS codes: (070.2025) Discrete optical signal processing; (070.7345) Wave propagation.

References and links

1. J. W. Goodman, *Introduction to Fourier Optics* (Roberts & Company Publishers, 2004), 3rd ed.
2. D. G. Voelz, *Computational Fourier Optics: A Matlab Tutorial*, Tutorial texts in optical engineering (SPIE Press, 2011).
3. U. Schnars and W. Jueptner, *Digital Holography: Digital Hologram Recording, Numerical Reconstruction, and Related Techniques* (Springer, 2005).
4. K. Matsushima, "Computer-generated holograms for three-dimensional surface objects with shade and texture," Applied Optics **44**, 4607–4614 (2005).
5. I. Hanák, M. Janda, and V. Skala, "Detail-driven digital hologram generation," Visual Computer **26**, 83–96 (2010).
6. Y. Sakamoto, M. Takase, and Y. Aoki, "Hidden surface removal using z-buffer for computer-generated hologram," Practical Holography XVII and Holographic Materials IX **5005**, 276–283 (2003).
7. M. Yamaguchi, "Ray-based and wavefront-based holographic displays for high-density light-field reproduction," Three-Dimensional Imaging, Visualization, and Display 2011 **8043**, 804306 (2011).
8. P. W. M. Tsang, J. P. Liu, K. W. K. Cheung, and T. C. Poon, "Modern methods for fast generation of digital holograms," 3D Research **1**, 11–18–18 (2010).
9. N. Delen and B. Hooker, "Free-space beam propagation between arbitrarily oriented planes based on full diffraction theory: a fast fourier transform approach," J. Opt. Soc. Am. A **15**, 857–867 (1998).
10. K. Matsushima, H. Schimmel, and F. Wyrowski, "Fast calculation method for optical diffraction on tilted planes by use of the angular spectrum of plane waves," J. Opt. Soc. Am. A **20**, 1755–1762 (2003).
11. L. Onural, "Exact solution for scalar diffraction between tilted and translated planes using impulse functions over a surface," J. Opt. Soc. Am. A **28**, 290–295 (2011).
12. E. Lalor, "Conditions for the validity of the angular spectrum of plane waves," J. Opt. Soc. Am. **58**, 1235–1237 (1968).
13. H. M. Ozaktas, S. O. Arik, and T. Coşkun, "Fundamental structure of fresnel diffraction: natural sampling grid and the fractional fourier transform," Opt. Lett. **36**, 2524–2526 (2011).
14. K. Matsushima, "Shifted angular spectrum method for off-axis numerical propagation," Opt. Express **18**, 18453–18463 (2010).

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

15. L. Onural, "Exact analysis of the effects of sampling of the scalar diffraction field," J. Opt. Soc. Am. A **24**, 359–367 (2007).
16. L. Onural, A. Gotchev, H. Ozaktas, and E. Stoykova, "A survey of signal processing problems and tools in holographic three-dimensional television," Circuits and Systems for Video Technology, IEEE Transactions on **17**, 1631 –1646 (2007).
17. P. Lobaz, "Reference calculation of light propagation between parallel planes of different sizes and sampling rates," Opt. Express **19**, 32–39 (2011).
18. V. Katkovnik, J. Astola, and K. Egiazarian, "Discrete diffraction transform for propagation, reconstruction, and design of wavefield distributions," Appl. Opt. **47**, 3481–3493 (2008).
19. E. Steward, *Fourier Optics: An Introduction*, Ellis Horwood Series in Physics (Dover Publications, 2004), 2nd ed.
20. K. Turkowski, "Filters for common resampling tasks," in "Graphics gems," , A. S. Glassner, ed. (Academic Press Professional, Inc., San Diego, CA, USA, 1990).
21. D. P. Mitchell and A. N. Netravali, "Reconstruction filters in computer-graphics," SIGGRAPH Comput. Graph. **22**, 221–228 (1988).

1. Introduction

Calculation of coherent light propagation in a free space is a fundamental tool in Fourier optics [1, 2], digital holography [3], computer generated holography (e.g. [4–8]) and other areas of optics. A very important and common task is the calculation of light propagation between two parallel planes; however, the general case of light propagation between tilted planes is also important in applications [9–11].

The problem is often given in this way: there is an area Σ in a plane $z = 0$ containing an image called the *source* that is lit by a coherent light, mostly by a plane wave. The task is to calculate the light field in a plane $z = z_0$ in an area called the *target*.

In this task, we usually assume validity of the scalar approximation of the light [1]. A good approximation of the correct solution is then given by, e.g., a Rayleigh-Sommerfeld integral of the first kind. In this article, let us assume this approximation as the reference one.

The Rayleigh-Sommerfeld solution cannot usually be used in an analytic calculation due to its complexity. Sometimes it is possible to get some results by using its mathematically equivalent form, the angular spectrum decomposition [12]. It is, however, most usual to restrict the calculation to the paraxial approximation in either the near (Fresnel) or far (Fraunhofer) region.

It is possible to evaluate the "reference" Rayleigh-Sommerfeld integral numerically using computers; thanks to its form of convolution, the calculation leads to the use of three fast Fourier transforms (FFT). The reason for the use of the aforementioned forms or approximations lies in the number of FFT's: the angular spectrum decomposition leads to two FFT's; the Fresnel or Fraunhofer approximation lead to just one FFT or fast fractional Fourier transform [13].

The implementations of these faster algorithms are unfortunately not straightforward, as the discretization of their equations leads to various problems. Correct implementation of the angular spectrum decomposition is especially tricky [14]; even Fresnel and Fraunhofer approximations have to be discretized carefully [2, 15, 16].

It is therefore wise to verify fast algorithms by comparing them with a reference method based on the carefully discretized Rayleigh-Sommerfeld integral [17, 18]. The discretization process must consider both correct sampling of the *source* and sampling of the illumination light field and the Rayleigh-Sommerfeld convolution kernel. It is also necessary to consider the inverse operation to the sampling, i.e. the reconstruction. All of these considerations often lead to sampling distances smaller than the wavelength of light, and therefore to huge memory demands. This article studies the discretization process and suggests a method to avoid huge memory demands and consequent time demands of large arrays FFT's.

The structure of the article is as follows. Section 2 shows a naive method of discretization; the

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

example given will show an illuminated amplitude diffraction grating with a sine transmittance profile. We will show that the naive discretization leads to "wrong" results; we will explain the "wrong" result in a physical way and show that a fine sampling leads to the correct result (and to huge memory demands). In Section 3, we will show how to lower memory demands by a simple 1D example. In Section 4 we will remove certain simplifications introduced in Section 3, and in Section 5 we will discuss the general 2D algorithm. Finally, in Section 6 we will present the time and memory demands of the algorithm and in Section 7 we will give conclusions.

In the rest of the article we will assume SI units. Absolute value of a complex amplitude is electric field amplitude, unit volt/m. For conversion of a complex amplitude to an intensity value, see e.g. [1, 2]. Please also note that 1D examples should not be interpreted physically as propagation integrals were derived for 3D space; they just demonstrate main ideas of the final algorithm.

2. Effect of naive discretization

Both theoretical analysis and experiments show that an amplitude diffraction grating with a sine transmittance profile illuminated by a plane wave (let us call it the *source*) creates just three diffraction maxima in the far field – the directly transmitted wave and the plus-minus first diffraction order [1]. Sampling of the *source* is easy in this case; it is necessary to use a sampling frequency at least $2\times$ higher than the frequency of the pattern. Let us choose the perpendicular illumination and set its complex amplitude at $z = 0$ to be 1. Let us choose a sampling whose samples coincide with maxima and minima of the transmittance of the grating, i.e. the samples will be progressively $\ldots, 0, 1, 0, 1, 0, 1, \ldots$ (this is exactly at the Nyquist limit, see also the end of the section), and let us calculate the diffraction pattern using the Rayleigh-Sommerfeld integral. It is given as

$$U(x,y,z_0) = \frac{-1}{2\pi} \iint_{\Sigma} U(\xi,\eta,0) \frac{\partial}{\partial z} \frac{\exp(\mathrm{j}kr)}{r} \mathrm{d}\xi \mathrm{d}\eta \tag{1}$$

where $U(x,y,z_0)$ is the calculated complex amplitude at a point $[x,y,z_0]$ of the *target*, $U(\xi,\eta,0)$ is the complex amplitude at a point $[\xi,\eta,0]$ of the *source* (i.e. the product of the complex amplitude of incoming light and the transmittance of the grating), Σ is the extent of the *source*, $\mathrm{j}^2 = -1$, $k = 2\pi/\lambda$ is the wave number and $r = ((x-\xi)^2 + (y-\eta)^2 + z_0^2)^{-1/2}$ is the distance between points $[x,y,z_0]$ and $[\xi,\eta,0]$.

Let us discretize the calculation by replacing the double integral with a double sum. The result shown in Fig. 1(a) differs a lot from the theoretical result. Where is the problem? (Note that Fig. 1 displays diffraction at sine grating of finite rectangular area in a finite distance, therefore the diffraction maxima have rectangular shape.)

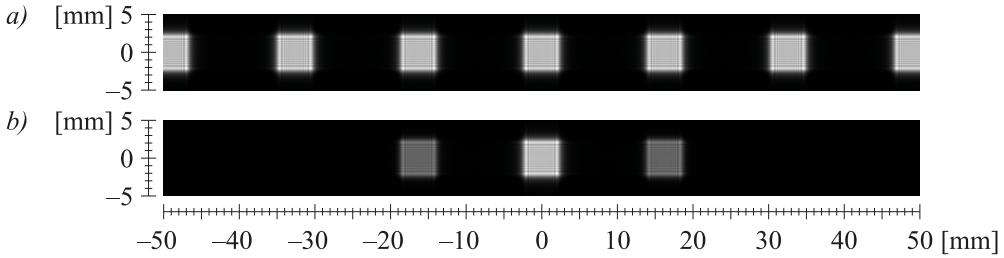


Fig. 1. Light diffraction at sine grating 5×5 mm^2 of period 50 cycles/mm illuminated perpendicularly by a plane wave with $\lambda = 650$ nm at a distance $z_0 = 0.5$ m. a) Discretization using $\Delta = 10$ μm. b) Discretization using $\Delta = 10/12$ μm $= 0.83$ μm.

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

When talking about discretization, let us discuss the direction of varying transmittance only (i.e. perpendicular to the grating stripes). The other direction is not important in this case.

The replacement of the integral by the sum means, in fact, that the original continuous function $U(\xi,\eta,0)$ is replaced by an array of Dirac pulses; in other words, by ideal point light sources arranged in a periodic lattice with spacing Δ. Light originating from a lattice of sources of equal complex amplitude interferes and creates m-th diffraction maximum in angle $\theta_m = \arcsin m\lambda/\Delta$ (see e.g. [19]), where m is an integer. In our case, every second sample is zero; that is, we are working in fact with a lattice with spacing 2Δ. This lattice creates diffraction maxima of equal intensities for all m, and the first diffraction maximum of this lattice coincides with the first diffraction maximum of the original sine grating, because its period is, thanks to sampling distance, equal to 2Δ. The result presented in Fig. 1a is therefore physically correct – although not for a sine grating, but for another experiment. The problem is that the discretization process did not take into account how to reconstruct the original continuous function $U(\xi,\eta,0)$ from the samples of the *source*, i.e. if there is zero transmittance between samples, if the samples represent a sine profile, a rectangular profile, etc. If we took this into account, the diffraction maxima created by the lattice would be attenuated somehow and the result would correspond with the original continuous situation.

A simple solution is easy. In the continuous situation, two close enough point light sources of the same complex amplitude do not create any interference pattern, i.e. at least one destructive interference. In the discretized situation, two point light sources of the same complex amplitude in adjacent samples can interfere destructively if the sampling distance is too big. Therefore, the sampling of the *source* has to represent the *source* correctly, and, moreover, the effect of the discretization has to be hidden – it must be such that the first-order destructive interference of adjacent point light sources created by the discretization (assume they have the same complex amplitude for now) has to lie out of the *target* area; note that in numerical calculations we are dealing with finite areas only. Mathematically, for any point T of the *target* and any adjacent samples S_1, S_2 of the *source*, the inequality $|T-S_1|-|T-S_2|<\lambda/2$ must hold as the density of the samples has to resemble continuous nature of the *source*. The same idea could be expressed in terms of correct sampling of the propagation integral kernel (e.g. [2, 14, 18]), but this explanation based on point light sources is perhaps more intuitive. It is interesting to note that this simple explanation based on point light source model has not been explicitly published yet (as far as I know). Also note that advanced solutions exist that do not need finer sampling, e.g. [13, 15]; the purpose of this paragraph is to provide simple insight and a starting point for the following sections.

The result of the simple solution is presented in Fig. 1b. It was necessary to work with $12\times$ finer sampling, i.e. with an array $12^2 = 144\times$ bigger. This results in noticeably higher memory and time demands. In the following text we will present a way to discretize correctly without increasing memory demands.

At the end of the section it is worth noting that the sampling of the sine grating in the example above was not done "correctly", as the Nyquist limit (in its simplest form) requires a sampling frequency higher than double the maximum frequency contained in a signal. We have used exactly double the maximum frequency, and moreover, we did not consider that the *source* is spatially limited. However, had we used the mathematically precise method, the result would be the same and the discussion would not be as clear.

3. Simplified solution

In search of a memory-efficient algorithm, let us start with the solution presented in the last section, i.e. the sampling finer than requested by the Nyquist limit. For the sake of clarity, let us introduce two simplifications: let the *source* and the *target* be one-dimensional objects in the

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

xz plane (this will be relaxed in Section 5) and let us suppose they are unbounded (this will be relaxed in Section 4).

Let the object *source* represented by samples *source*$[i]$, $i \in \mathbf{Z}$ (let the sample *source*$[0]$ be located at the point $[0,0]$) be perpendicularly illuminated by a plane wave of wavelength λ (see Fig. 2). We will consider transmittance of the *source* to be complex, i.e. any *source* illuminated by any light can be converted to this scenario.

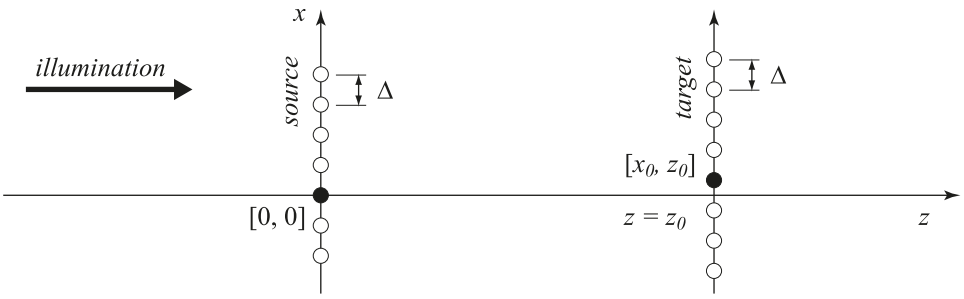


Fig. 2. Geometry of the simplified 1D case. Samples with index 0 are depicted as full circles, the others as empty circles.

Let us calculate complex amplitudes of propagated light in the "plane" $z = z_0 > 0$ in the unbounded area *target* represented by samples *target*$[j]$, $j \in \mathbf{Z}$ (let the sample *target*$[0]$ be located at the point $[x_0, z_0]$). The samples *source*$[i]$ are samples of the complex function $U(x,0)$; the samples *target*$[j]$ represent the function $U(x,z_0)$ (see Eq. (1)). If the sampling distance is equal in both the *source* and the *target*, let us call it Δ, the following holds:

$$\begin{aligned} \mathit{target}[j] = U(j\Delta + x_0, z_0) &= \Delta \sum_{i=-\infty}^{\infty} U(i\Delta,0)\, h\big((j-i)\Delta + x_0, 0, z_0\big) = \\ &= \Delta \sum_{i=-\infty}^{\infty} \mathit{source}[i]\, h_{x_0,z_0,1}[j-i] = \\ &= \Delta\big(\mathit{source}[\,] \otimes h_{x_0,z_0,1}[\,]\big)[j] \end{aligned} \tag{2}$$

where $\otimes$ is the discrete convolution and the array $h_{x_0,z_0,1}[\,]$ represents the Rayleigh-Sommerfeld convolution kernel (impulse response) defined as

$$h_{x_0,z_0,ups}[j] = h(j\Delta/ups + x_0, 0, z_0) \tag{3}$$

and according to (1)

$$h(x,y,z) = -\frac{1}{2\pi}\frac{\partial}{\partial z}\frac{\exp(\mathrm{j}kr)}{r} = \frac{-z}{2\pi}\left(\mathrm{j}k - \frac{1}{r}\right)\frac{\exp(\mathrm{j}kr)}{r^2}, \quad r = \sqrt{x^2+y^2+z^2} \tag{4}$$

Equation (2) is a discretization of the integral (1), i.e. we have changed the integral to the sum and the differential to the difference Δ. As we will be interested just in the structure of the *target*, we will omit the constant term Δ.

Let us suppose that the *source* is sampled correctly, such that the structure of $U(x,0)$ is fully acquired, but not fine enough for the propagation calculation. Please note that this "correct sampling" is not the same as sampling that obeys the sampling theorem; for example, piecewise constant signal cannot be sampled correctly according to the basic formulation of the sampling theorem as it has infinite frequency extent. However, if we know that the signal is piecewise constant with "steps" at known locations, then we can express the signal precisely using just one sample per constant part of the signal.

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

For the propagation calculation, we have to use a *ups*-times finer sampling, $ups \in \mathbf{Z}$, $ups \geq 1$. Let us call the preliminary upsampled array $source_{ups}[\,]$. To obtain this array, let us put $(ups-1)$ zero samples between every two original samples of the *source* (see Fig. 3 top):

$$source_{ups}[j] = \begin{cases} source[j/ups] & \text{if } (j/ups) \in \mathbf{Z} \\ 0 & \text{otherwise} \end{cases}$$

The final upsampled array $source^{fin}_{ups}[\,]$ is calculated using convolution with a kernel $filter[\,]$, i.e. $source^{fin}_{ups}[\,] = source_{ups}[\,] \otimes filter[\,]$. The convolution kernel $filter[\,]$ has to be chosen according to the nature of the function $U(x,0)$: a rectangular kernel provides a piecewise constant interpolation (which is suitable if $U(x,0)$ represents a pixelated spatial light modulator), a windowed sinc kernel provides a good interpolation in terms of frequency content (which is suitable if $U(x,0)$ is a general continuous function), a triangular kernel provides a piecewise linear interpolation (which is faster than a windowed sinc kernel and can provide acceptable results), etc. (see Fig. 3). Examples of kernel implementations will be shown at the end of Section 4.

The main idea of the method to be derived exploits the fact that the length of the $filter[\,]$ array is much smaller than the length of the array $source[\,]$ and $target[\,]$. For simplicity, let us assume the length of the $filter[\,]$ array to be odd. Let us write it as $2 \times fwh + 1$, where $fwh \in \mathbf{Z}$, $fwh \geq 0$.

Thanks to associativity of the convolution, the following holds:

$$\begin{aligned} target_{ups}[\,] &= \big(source_{ups}[\,] \otimes filter[\,]\big) \otimes h_{x_0,z_0,ups}[\,] = source_{ups}[\,] \otimes \big(filter[\,] \otimes h_{x_0,z_0,ups}[\,]\big) = \\ &= source_{ups}[\,] \otimes h^{fin}_{x_0,z_0,ups}[\,] \end{aligned}$$

where $target_{ups}[\,]$ represents the *target* sampled with a period Δ/ups and $h^{fin}_{x_0,z_0,ups}[\,]$ is the propagation kernel convolved by the array $filter[\,]$. Specifically,

$$\begin{aligned} target_{ups}[j] &= \sum_{i=-\infty}^{\infty} source_{ups}[j-i]\; h^{fin}_{x_0,z_0,ups}[i] = \\ &= \sum_{i=-\infty}^{\infty} source_{ups}[j-i] \sum_{k=-fwh}^{fwh} filter[k]\; h_{x_0,z_0,ups}[i-k] \end{aligned}$$

However, we need the final result *ups*-times downsampled, i.e. $target[j] = target_{ups}[ups \times j]$. Moreover, the sample $source_{ups}[i]$ is zero for $i/ups \notin \mathbf{Z}$. We can therefore omit these samples from the sum. It follows that

$$\begin{aligned} target[j] = target_{ups}[ups \times j] &= \sum_{i=-\infty}^{\infty} source_{ups}[ups \times (j-i)]\; h^{fin}_{x_0,z_0,ups}[ups \times i] = \\ &= \sum_{i=-\infty}^{\infty} source[j-i]\; h^{fin}_{x_0,z_0,1}[i] \end{aligned} \tag{5}$$

where $h^{fin}_{x_0,z_0,1}[\,]$ is a filtered propagation kernel:

$$h^{fin}_{x_0,z_0,1}[i] = \sum_{k=-fwh}^{fwh} filter[k]\; h_{x_0,z_0,ups}[ups \times i - k] \tag{6}$$

The propagation calculation leads to two discrete convolutions: in the first one (5), we work with sampling period Δ, in the second one (6), with sampling period Δ/ups.

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

The aforementioned equations worked with an infinite extent of the indices in order to avoid array boundary effects. In the following section, we will adjust the indices extent but the structure of the result will remain the same. The section will also explain the advantage of the presented method, which can be briefly described as follows.

The discrete convolution can be calculated indirectly using FFT or directly using the definition Eq. (2). The first way is advantageous if the convolution kernel is large; we will use it, therefore, for Eq. (5). The second way is better in the opposite case; we will therefore use it for Eq. (6). Here we will also use a nice property of direct calculation: the samples $h^{fin}_{x_0,z_0,1}[i]$ can be calculated with minimum memory demands.

4. Practical 1D solution

Let us assume that the *source* is sampled using M samples and the *target* is sampled using N samples. If we want to use FFT for the *target*[] calculation, we have to work with cyclic convolution. This means that the arrays *source*[] and *target*[] have to be zero-padded to C samples, $C \geq M+N-1$ (see [17]). Then:

$$target[j] = \sum_{i=0}^{C-1} source[i \bmod C]\; h^{fin}_{x_0,z_0,1}[(j-i) \bmod C]$$

The array *source*[] contains correct values for sample indices $0,1,\ldots,M-1$, the array *target*[] for indices $0,1,\ldots,C-M$. The array $h^{fin}_{x_0,z_0,1}[]$ has to be then calculated as:

$$h^{fin}_{x_0,z_0,1}[i] = \begin{cases} \sum_{k=-fwh}^{fwh} filter[k]\; h_{x_0,z_0,ups}[ups \times i - k] & \text{if } 0 \leq i < N \\ \sum_{k=-fwh}^{fwh} filter[k]\; h_{x_0,z_0,ups}[ups \times (i-C) - k] & \text{if } N \leq i < C \end{cases}$$

It is worth noting that in the calculation of the sample $h^{fin}_{x_0,z_0,1}[i]$, it is possible to use the samples $h_{x_0,z_0,ups}[]$ calculated before, specifically for $h^{fin}_{x_0,z_0,1}[i-1]$. It is therefore convenient to save the samples $h_{x_0,z_0,ups}[]$ in a temporary buffer of size $2 \times fwh + 1$ samples, and to replace part of them in the calculation of $h^{fin}_{x_0,z_0,1}[i]$ with new values. The number of these new samples depends on the *filter* type.

To calculate the *filter*[] array, we have to choose the number of samples of the *source*[] array that contribute to the calculation of the interpolated sample in the $source^{fin}$[] array, or in other words, in the $h^{fin}_{x_0,z_0,1}[i]$ array. This user-defined number specifies all the parameters needed: the size of the *filter*[] array, the interpolation type, and the number of the samples shared in the calculation of the neighbouring samples of the $h^{fin}_{x_0,z_0,1}[]$ array.

It is practical to use separable kernels when working with 2D arrays. We can discuss them right now, when working with 1D arrays. For clarity, some examples are given in Fig. 3. To make things simpler, we will show pure interpolation kernels in both the Fig. 3 and the following text, i.e. they do not preserve signal energy. For propagation calculation it is, however, appropriate to normalize the filter, i.e. the sum of its coefficients equals 1.

Piecewise constant interpolation. It is suitable if the *source* is split to rectangular pixels of non-zero area. In this case it is appropriate for *ups* to be odd, due to symmetry. Then the interpolation process adds an even number of samples between every two samples of the *source*[] array. Therefore, $fwh = (ups-1)/2$ and the kernel is given as: $filter[i] = 1$ for $-fwh \leq i \leq fwh$.

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

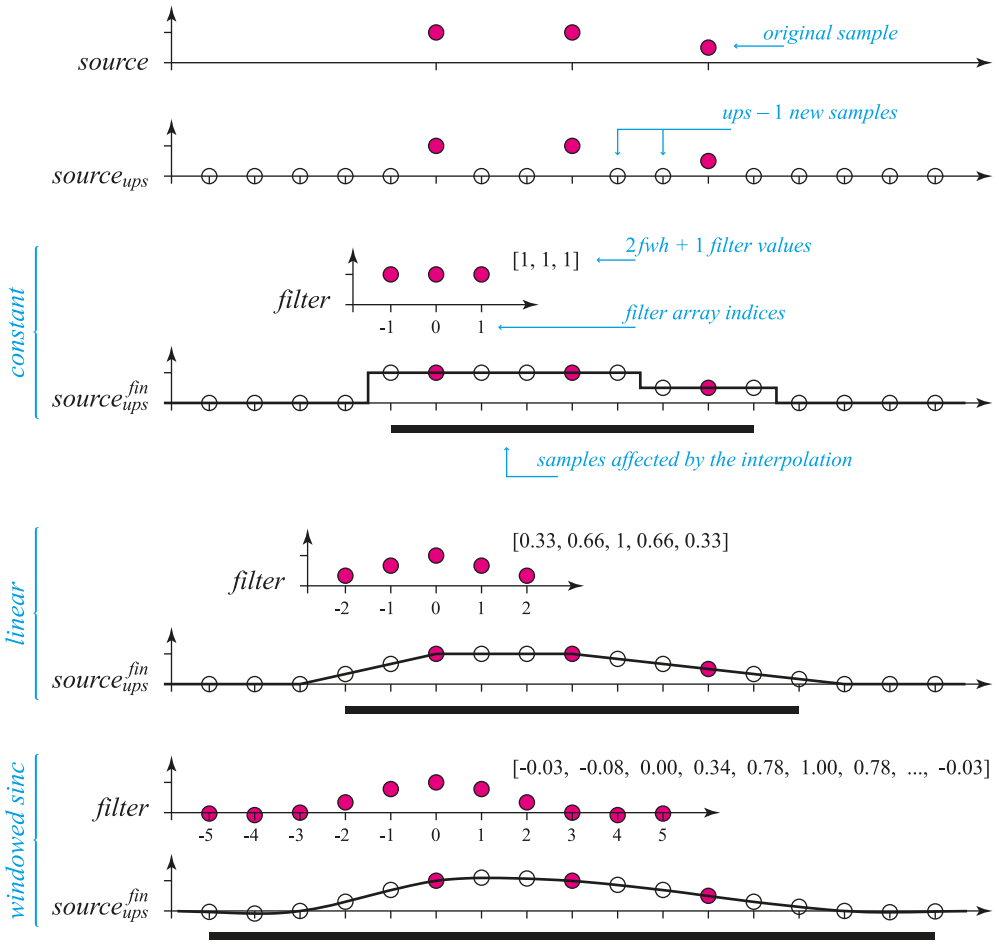


Fig. 3. Examples of interpolation convolution kernels (filters) for $ups = 3$. The windowed sinc filter shown is the normalized Lanczos filter for $a = 2$.

Piecewise linear interpolation. It is suitable if the *source* represents a continuous function and it does not contain fine details. We need two neighbouring original samples for the calculation of the interpolated one, i.e. $fwh = ups - 1$, $filter[i] = 1 - |i|/(fwh + 1)$ for $-fwh \le i \le fwh$.

Windowed sinc interpolation. It is suitable if the *source* represents a continuous function and we care about good replication of its frequency content. Choice of the number of the original $source[]$ samples has to be a compromise. The more samples are included, the better is the frequency content replication; on the other hand, too large kernels perform badly in the spatial domain. A Lanczos filter is considered a reasonable compromise. It considers $2a$ neighbouring samples, where usually $a = 2$ or $a = 3$ [20]. Then $fwh = a \times ups - 1$ and a preliminary kernel is defined as $filter_{prel}[i] = \text{lanczos}(a \times i/(fwh+1), a)$, where $\text{lanczos}(x,a) = a\sin(\pi x)\sin(\pi x/a)/(\pi^2 x^2)$ for $-fwh \le i \le fwh$. The final kernel has to be adjusted before use: the coefficients contributing to the same sample calculation, i.e. the coefficients ups samples apart, have to sum to 1 [21]. Mathematically, $filter[i] = filter_{prel}[i] / \sum_k filter_{prel}[i + k \times ups]$ for all allowed values of k.

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

5. Final 2D solution

Generalization of the aforementioned ideas is straightforward; instead of 1D cyclic convolutions (or FFT's), we have to use 2D versions. For simplicity, let us assume the sampling periods in both x and y directions to be the same, let us call them Δ.

To propagate the *source* sampled by $M_x \times M_y$ samples to the *target* sampled by $N_x \times N_y$ samples, let us create the arrays $source[,]$ and $target[,]$ of size $C_x \times C_y$, $C_x \geq M_x + N_x - 1$, $C_y \geq M_y + N_y - 1$. It is convenient to choose the numbers C_x and C_y so that the FFT of these arrays runs fast, e.g. powers of 2. The samples of the *source* must be stored in the array $source[,]$ at indices from $[0,0]$ to $[M_x - 1, M_y - 1]$. After the calculation, the correct samples are located in the array $target[,]$ at indices from $[0,0]$ to $[N_x - 1, N_y - 1]$.

As the next step, we have to choose the parameter *ups*. The way to do it is as follows. Let us assume for a while that we work with the original lattice, i.e. $ups = 1$. In the propagation calculation, we have to calculate (4) for every vector $T - S$, where S is a 3D position of a sample in the *source* and T is a 3D position of a sample in the *target*. In (4) we have to use $r = |T - S|$ (see its application in (2)). Let S_1 and S_2 be positions of adjacent samples in the *source* and these samples have the same value; they represent two point light sources of the same complex amplitude. Let us calculate $r_1 = |T - S_1|$ and $r_2 = |T - S_2|$. If $|r_1 - r_2| = \lambda/2$, then the contributions from S_1 and S_2 cancel each other at the point T, i.e. in this direction there is the first diffraction minimum. As we have shown in Section 2, we need to exclude the first diffraction minimum from the *target*. To make this happen, we have to refine the sampling, i.e. increase the parameter *ups* until $|r_1 - r_2| < \lambda/2$. If a more precise result is needed, we can refine further. Numerical experiments have shown that higher *ups* than those leading to $|r_1 - r_2| < \lambda/5$ did not have any significant impact. The adjacent points S_1, S_2 and the point T have to be chosen as "the worst case", i.e. the angle between $T - S_1$ (or $T - S_2$) and the z axis has to be as big as possible.

As the next step, we have to calculate the array $h^{fin}_{x_0,y_0,z_0,1}[,]$, where $[x_0, y_0, z_0]$ is the position of the sample $target[0,0]$. We assume the position of the sample $source[0,0]$ to be $[0,0,0]$. For the calculation, we need the numbers $h_{x_0,y_0,z_0,ups}[i,j] = h(i\Delta/ups + x_0, j\Delta/ups + y_0, z_0)$ (see (3) for comparison). Thanks to separability of the filters, we can calculate them for one upsampled row only, convolve them with $filter[]$ and downsample, i.e. to use the procedure described in Section 4. We need to calculate $2 \times fwh + 1$ of such rows, convolve them by columns and downsample; this procedure leads to one row of the array $h^{fin}_{x_0,y_0,z_0,1}[,]$. The other rows are calculated in the same way.

Finally, we can calculate the propagation itself:

$$target[,] = \mathbf{IFFT}\big(\mathbf{FFT}(source[,]) \odot \mathbf{FFT}(h^{fin}_{x_0,y_0,z_0,1}[,])\big)$$

where **FFT** and **IFFT** are fast Fourier transform and inverse fast Fourier transform, respectively, and $\odot$ is the Hadamard product (element-wise product). The propagation calculation is complete now. Examples of the propagations are shown in Fig. 4.

It is worth adding three remarks:

- The same result can be obtained using the basic method, i.e. using upsampling, interpolation of the *source*, and convolution with a common Rayleigh-Sommerfeld propagation kernel. The upsampled array $source_{ups}[,]$ would have $ups - 1$ zero samples between original samples in every row (column), and additionally fwh zero samples to the sides in order to correctly calculate the convolution with an interpolation kernel $filter[,]$ of size $(2 \times fwh + 1) \times (2 \times fwh + 1)$ samples. The size of the array $source_{ups}$ would then be $\big(2 \times fwh + 1 + ups \times (M_x - 1)\big) \times \big(2 \times fwh + 1 + ups \times (M_y - 1)\big)$ samples, sampling period Δ/ups. This means that the physical size of the *source* increases a bit for $fwh \geq 1$;

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

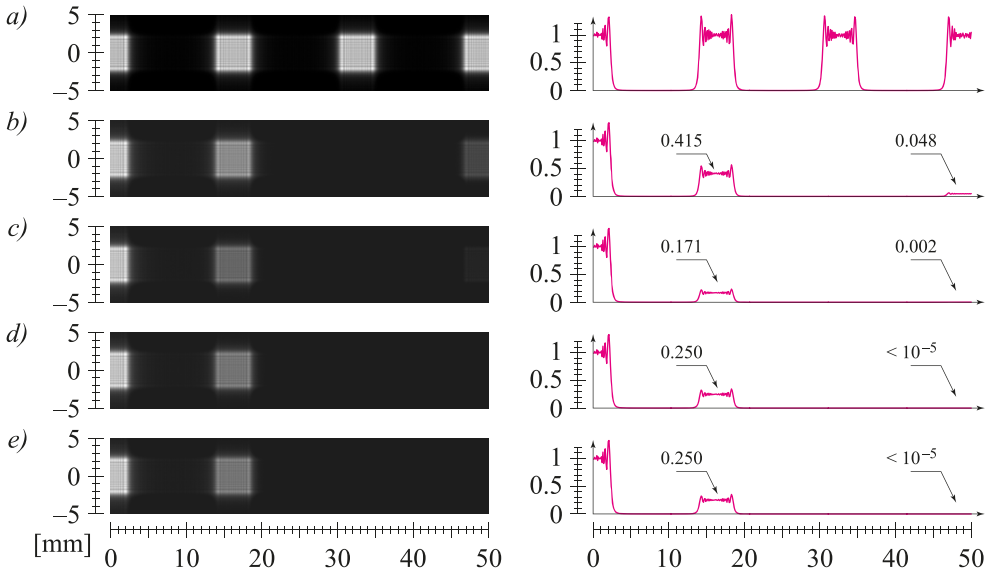


Fig. 4. Examples of diffraction by grating with vertical strips; grating size 5×5 mm^2, sampling period 10 μm, samples in each row 1, 0, 1, 0, ... (i.e. 50 slits/mm). Propagation distance 500 mm, illumination at normal incidence, $\lambda = 650$ nm. The left images show right halves of the diffraction patterns (compare with Fig. 1); the graphs to the right show the intensity relative to the central intensity. The interpolation used is a) none, b) rectangular filter, c) triangular filter, d) Lanczos filter, $a = 2$, e) Lanczos filter, $a = 3$.

this is the reason why the article never mentions the exact physical sizes of the *source* and the *target*. The additional borders are an interpolation artifact. However, their effect is negligible for big arrays.

- The proposed method with propagation kernel filtering is nothing else than a calculation rearrangement. The article [17] that describes the propagation calculation with large *source* and *target* or with different sampling periods of *source* and *target* is therefore fully compatible with the proposed method; is is sufficient to replace the calculation of propagation kernels.

- The calculation rearrangement leads to different rounding errors in numerical calculation and thus to differences between the basic and the proposed method. The differences are negligible, though. As it is hard to tell which method gives a more precise result, we will not discuss the numerical aspects of the proposed method.

6. Time and memory requirements

The motivation to create the proposed method was to calculate reference propagation using a small amount of memory. Let us compare its memory and time demands with the basic method. In this section, we will assume square arrays for simplicity, i.e. $M_x = M_y = M$, $N_x = N_y = N$.

The basic method, as we have shown in the last section, upsamples the array $source[,]$ to the array $source_{ups}$ with $\left(2 \times fwh + 1 + ups \times (M-1)\right)^2$ samples and calculates propagation to the array $target_{ups}$. There is no need to introduce additional zero borders due to interpolation, i.e. the *target* will have $\left(1 + ups \times (N-1)\right)^2$ samples. The propagation will be calculated in the

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

arrays with $\left(2\times fwh+1+ups\times(M+N-2)\right)^2$ samples, which gives the memory requirements of the basic method.

The proposed method uses the original arrays $source[,]$ and $target[,]$, so the propagation will be calculated in arrays with $(M+N-1)^2$ samples. The memory demands are, however, a bit higher, as we have to take account of temporary memory used in the calculation of the filtered propagation kernel $h^{fin}_{x_0,y_0,z_0,1}[,]$. It is $2\times fwh+1$ samples for "row convolution" and $2\times fwh+1$ rows with $M+N-1$ samples for "column convolution"; together $(2\times fwh+1)(M+N)$ samples. Typically, $fwh=a\times ups$, where a is small (in our examples at most 3), and ups is much smaller than $M+N$. Therefore, it is possible to ignore this amount in further discussion.

By comparing the memory demands, we conclude that the propagation calculation using the proposed method takes approximately $\left(2\times fwh+1+ups\times(M_x+N_x-2)\right)^2/(M+N-1)^2\approx ups^2$-times less memory. Common experiments in computer generated holography with centimetre-sized fields using a sampling period of about 10 μm off-axis propagated to a distance in the order of tens of centimetres require the ups parameter up to 20. We can therefore say that the proposed method has about 100-times less memory demands than the basic method.

On the other hand, the time of the computation does not fall so quickly. This is because the calculation consists not just of a convolution calculation using the FFT, which is very fast in the proposed method, but of a convolution kernel calculation as well that is approximately as slow as in the basic method. More precisely, the FFT works with arrays approximately ups^2-times smaller than in the basic method, which means it is approximately ups^2-times faster. The convolution kernel calculation requires calculating the array $h_{x_0,y_0,z_0,ups}[,]$ (the same as in the basic method), filtering it by rows and by columns using a filter of width $2\times fwh+1$, where again $fwh=a\times ups$, and downsampling the result. The analysis shows that the convolution kernel calculation in the proposed method is approximately $4a^2$-times slower than in the basic method, where a is again a small number. It is not worth making a precise analysis, as the speed of the FFT, the Rayleigh-Sommerfeld convolution kernel and its filtering are hard to compare theoretically. It is more useful to measure the calculation times (see Fig. 5). The graph to the right shows that the speedup of the proposed method is not as big as its memory savings; this is mainly because the calculation of the convolution kernel dominates in the total time of the calculation.

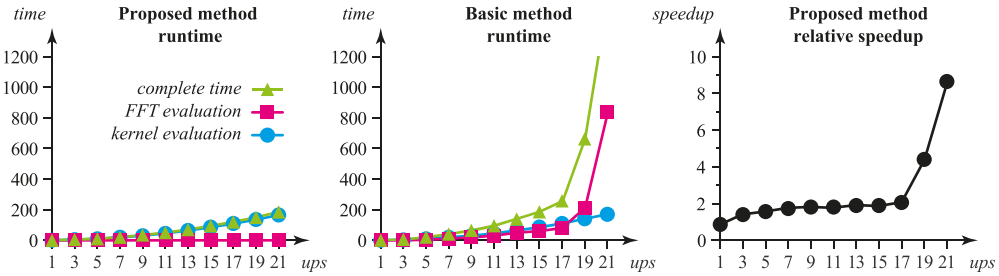


Fig. 5. Time of the calculation comparison. The graphs to the left and in the middle show the dependency of the time of propagation calculation for $N=M=500$ on the upsampling factor ups; the vertical scale used defines the time of the basic method for $ups=1$ to be 1. Besides the complete time of the calculation, the times of the FFTs and the propagation kernel calculation times are shown. The rightmost graph shows the ratio of the proposed and the basic method calculation times; e.g. a value of 4 means that the proposed method is 4× faster for a given ups.

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

7. Conclusions

In Section 2 we have shown and explained in terms of physics that in the discretization of the light propagation between parallel planes it is necessary to take into account both the correct sampling of the *source* and the opposite procedure, the reconstruction. We have shown that the correct result can be obtained using upsampling; we have shown in Section 5 how fine this upsampling should be. We have demonstrated in Section 6 that in typical tasks in computer generated holography the upsampling can be approximately $10\times$, which leads to $100\times$ slower calculation and $100\times$ bigger memory demands.

In Sections 3 to 5 we have derived a procedure based on filtration of the propagation kernel by the "interpolation" kernel that is used for interpolated upsampling of the *source*. Thanks to the properties of the interpolation kernel (small support, separability), the proposed method is faster and cuts the memory demands to approximately those values which would be needed if no upsampling was used. It can thus be said that the proposed method has approximately $100\times$ smaller memory demands than the basic method of the same precision. As the method just rearranges the calculation, the result is mathematically the same.

Acknowledgments

This work has been supported by the Ministry of Education, Youth and Sports of the Czech Republic projects LC06008 and LH12181. The author would like to thank Václav Skala for discussions concerning the subject.

#180735 - $15.00 USD Received 29 Nov 2012; revised 11 Jan 2013; accepted 11 Jan 2013; published 29 Jan 2013

Appendix D

Safe range of free space light propagation calculation in single precision

Safe range of free space light propagation calculation in single precision

Petr Lobaz[1,2,*] **and Petr Vaněček**[1]

[1]*Department of Computer Science and Engineering, Faculty of Applied Sciences, University of West Bohemia, Univerzitni 8, 306 14 Plzen, Czech Republic*
[2]*NTIS – New Technologies for the Information Society, European Centre of Excellence, Faculty of Applied Sciences, University of West Bohemia, Univerzitni 8, 306 14 Plzen, Czech Republic*

[*]*lobaz@kiv.zcu.cz*

Abstract: Calculations of free space light propagation, such as those used in digital holography, deal with distances much longer than a wavelength. Computer representation of real numbers must therefore provide enough precision to handle this situation. We show that single precision must be used with the utmost care, which is especially important in GPU calculations. We also show that Fresnel approximation significantly improves single precision calculations for distances bigger than about one metre.

OCIS codes: (000.4430) Numerical approximation and analysis; (070.2025) Discrete optical signal processing; (090.1995) Digital holography; (070.7345) Wave propagation.

References and links

1. D. Goldberg, "What every computer scientist should know about floating point arithmetic," ACM Computing Surveys **23**, 5–48 (1991).
2. Wikipedia, "List of Nvidia graphics processing units — Wikipedia, The Free Encyclopedia," (2014). [Online; accessed 25-July-2014]. http://en.wikipedia.org/w/index.php?title=List_of_Nvidia_graphics_processing_units
3. J. Song, J. Park, H. Park, and J.-I. Park, "Real-time generation of high-definition resolution digital holograms by using multiple graphic processing units," Opt. Eng. **52**, 015803 (2013).
4. A.-H. Phan, M. lan Piao, S.-K. Gil, and N. Kim, "Generation speed and reconstructed image quality enhancement of a long-depth object using double wavefront recording planes and a GPU," Appl. Opt. **53**, 4817–4824 (2014).
5. J.-M. Muller, N. Brisebarre, F. de Dinechin, C.-P. Jeannerod, V. Lefèvre, G. Melquiond, N. Revol, D. Stehlé, and S. Torres, *Handbook of Floating-Point Arithmetic* (Birkhäuser Boston, 2010).
6. J. W. Goodman, *Introduction to Fourier Optics* (Roberts & Company Publishers, 2004), 3rd ed.
7. I. Hanák, P. Zemčík, M. Žádník, and A. Herout, "Hologram synthesis accelerated in field programmable gate array by partial quadratic interpolation," Optical Engineering **8**, 1–7 (2009).
8. D. G. Voelz, *Computational Fourier Optics: A Matlab Tutorial*, Tutorial texts in optical engineering (SPIE Press, 2011).
9. P. Lobaz, "Memory-efficient reference calculation of light propagation using the convolution method," Opt. Express **21**, 2795–2806 (2013).
10. W. J. Dallas, "Phase quantization in holograms—a few illustrations," Appl. Opt. **10**, 674–676 (1971).

1. Introduction

It is a well known fact that double precision computer calculations usually provide more precise results than single precision ones [1]. It is also "conventional wisdom" that many calculations in wave optics (such as free space light propagation) are more reliable using double precision.

#220529 - $15.00 USD Received 6 Aug 2014; revised 22 Jan 2015; accepted 26 Jan 2015; published 4 Feb 2015

The question whether to use single or double precision numbers is quite unimportant in CPU calculations as they are often internally performed in higher than single precision and the final result is rounded to the requested one.

The question, however, is rather important in GPU calculations as current GPUs favour single precision calculations over double precision ones (see [2] or other list of GPU parameters). It is of utmost importance today as many researchers, especially in computer generated (digital) holography, utilize GPU calculations whenever possible (see e.g. [3, 4]). We will focus our attention to one of the most fundamental task in digital holography, Fourier optics and wave optics – calculation of light propagation in free space to a certain distance. As far as we know, no one analysed range of distances where it is safe to use single precision, where it is better to switch to double precision and what to do if we want to avoid double precision calculations.

In Sec. 2 we will show that single precision calculations can lead to significant problems even in realistic scenarios. In Sec. 3 we will point out the problem origin; as most readers are likely not computer scientists, an overview of important facts about floating-point representation is given there as well. Sec. 4 gives basic analysis of the problem and tells when it is safe to use single precision in on-axis calculations; Sec. 5 adds a few notes to an off-axis generalization. Sec. 6 shows that parabolic (Fresnel) approximations can save single-precision calculations in most cases. Finally, Sec. 7 supplements the discussion with reference to aliasing and phase quantization; Sec. 8 concludes the article.

2. The problem demonstration

Let us perform the most basic calculation in scalar wave optics: let us calculate an interference pattern on the plane $z = 0$ of two mutually coherent, monochromatic, equally bright point light sources. Simply written:

$$I(x,y,0) = \left| \frac{\exp(\mathrm{j}2\pi r_0/\lambda)}{r_0} + \frac{\exp(\mathrm{j}2\pi r_1/\lambda)}{r_1} \right|^2, \tag{1}$$

where $I(x,y,0)$ is the light intensity at a point $[x,y,0]$, r_0 and r_1 are the distances between the point $[x,y,0]$ and the particular point light source, $\mathrm{j}^2 = -1$ is the imaginary constant, and λ is the wavelength.

Let us place the light sources (for instance) to the points $[\pm 0.001 z_0, 0, z_0]$, set $\lambda = 500$ nm and calculate the pattern for various z_0 in both single and double precision. Figure 1 shows the result. The patterns for $z_0 = 10$ mm are basically the same regardless precision used. It is clear that for $z_0 = 100$ mm, the pattern calculated in single precision contains noise introduced by limited precision, but its quality is generally acceptable. The pattern for $z_0 = 1000$ mm in single precision loosely reminds the correct one, but its quality is generally not acceptable. Finally, for $z_0 = 10000$ mm, the single precision result is completely useless.

This demonstration naturally leads to several questions. What is the source of such behaviour? Where is the boundary between acceptable and unacceptable calculations? What do we mean by "acceptable"? Is it possible to improve the calculation without switching to higher precision? Let us try to answer them all.

3. Origin of the problem

Let us begin by reviewing the basic facts about floating-point numbers. We decided to include this short section written in a form of tutorial as we do not know any suitable, short enough reference that would explain all the necessary concepts. Readers familiar with theory of floating point calculations and their limitations can skip this section.

A floating-point number x is usually (according to the IEEE 754 standard for floating-point arithmetic) internally expressed as $x = s \times m \times 2^{e-p-1}$, where 1-bit number s (*sign*) is equal

#220529 - $15.00 USD Received 6 Aug 2014; revised 22 Jan 2015; accepted 26 Jan 2015; published 4 Feb 2015

9 Feb 2015 | Vol. 23, No. 3 | DOI:10.1364/OE.23.003260 | OPTICS EXPRESS 3261

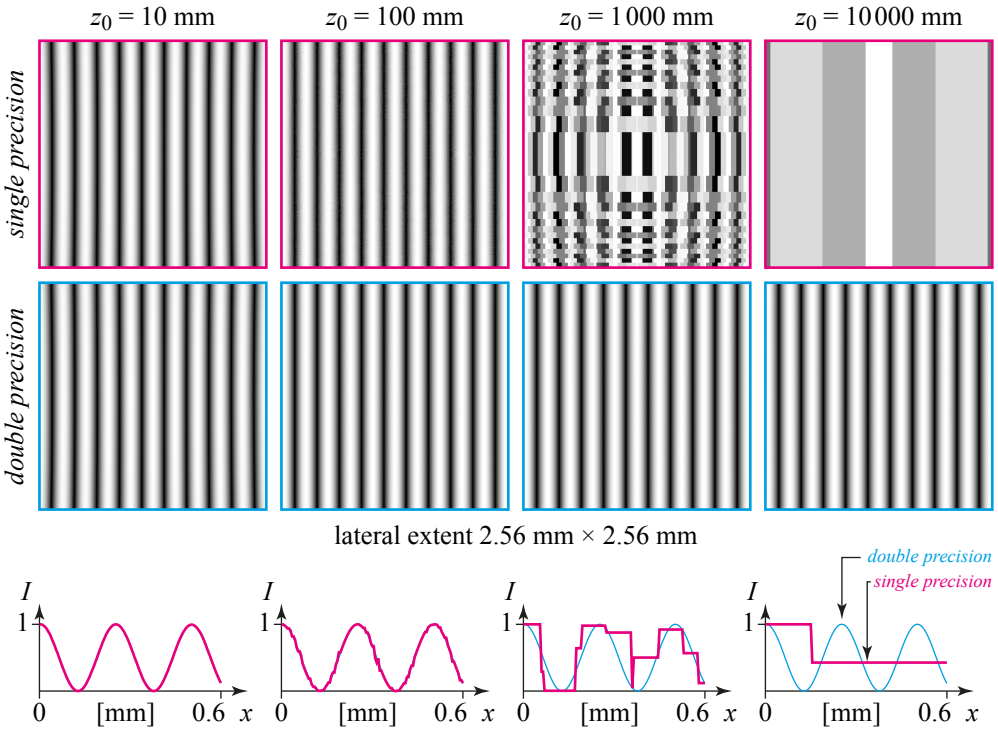


Fig. 1. Interference pattern (normalized intensity) of two point light sources located at $[\pm 0.001 z_0, 0, z_0]$ in the plane $z = 0$. The first row shows calculation in single precision, the second row in double precision. Each column stands for a particular z_0. The bottom row of graphs shows the normalized intensity values; for better clarity, only a segment of each image is shown.

to $+1$ or -1, m (*mantissa* or *significand*) is a p-bit unsigned integer and e (*exponent*) is a q-bit signed integer. Most often, significand $m \geq 2^{p-1}$ (number x is then called a *normalized number*), or in other words, the most significant bit of m is usually 1. The IEEE 754 standard defines $p = 24$ for single precision, $p = 53$ for double precision. As $\log_{10} 2^{24-1} \approx 6.9$, we can say single precision corresponds to about 7 significant decimal digits; double precision to about 16 decimal digits.

If the basic arithmetic operations ($+$, $-$, $\times$, $/$, mod, $\sqrt{}$) are performed according to the IEEE 754 standard, their results are the same as if they were calculated exactly and correctly rounded to the requested precision; the numerical error is therefore bounded [1]. As an example (we will recall it later in Sec. 4), let us see what happens in the addition operation.

When calculating (for instance) $25165824 + 1$ in single precision, we need to understand how the operands are internally expressed. Because $25165824 = 2^{24} + 2^{23}$, it cannot be directly represented by a 24-bit unsigned integer; instead, it is expressed as $m_1 \times 2^1$, where $m_1 = 2^{23} + 2^{22}$. The second operand can be also expressed in the normalized form as $1 = m_2 \times 2^{-23}$, where $m_2 = 2^{23}$. The exact sum is equal to $25165825 = 2^{24} + 2^{23} + 1$. Unfortunately, this number cannot be expressed by a 24-bit unsigned integer as it contains 25 significant bits. So the least significant bits (in this case one bit) must be cut off and the result must be altered in such a way the "rounding error" is the smallest. The altered 24-bit value is then used as a significand in the approximate floating-point value of the result, i.e. $(2^{23} + 2^{22}) \times 2^1$. It is clear that size of the rounding error is comparable with the unit at the least significant binary digit of the result.

#220529 - $15.00 USD Received 6 Aug 2014; revised 22 Jan 2015; accepted 26 Jan 2015; published 4 Feb 2015

9 Feb 2015 | Vol. 23, No. 3 | DOI:10.1364/OE.23.003260 | OPTICS EXPRESS 3262

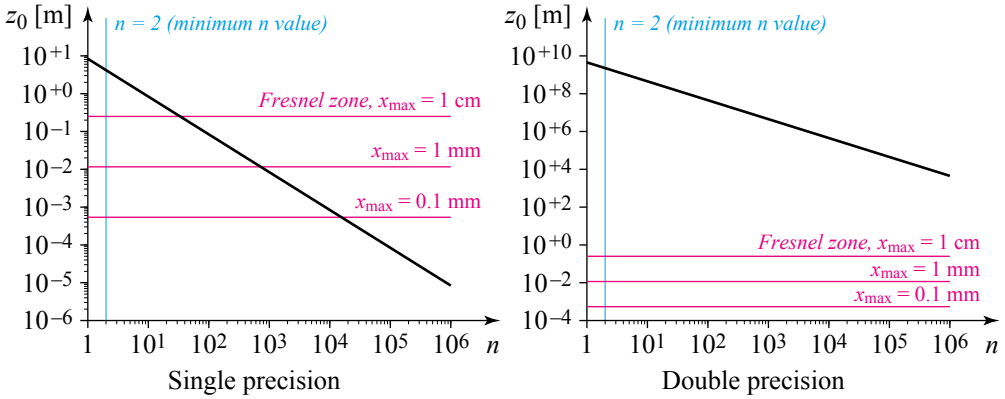


Fig. 2. Maximum z_0 according to inequality (2) as a function of n for $\lambda = 500$ nm, i.e. the maximum distance for which $(z_0^2 + (\lambda/n)^2)^{1/2} \neq z_0$ holds. For example, if we want to calculate in single precision (left graph) and to resolve about 100 values in the first fringe, z_0 should be at most about 0.1 m. "Safe region" is then *below* the black line. For convenience, Fresnel zones for different x_{max} are shown as magenta horizontal lines, see Sec. 6 for details. The Fresnel approximation is valid near the z axis ($|x| < x_{max}$, $|y| < x_{max}$) for z_0 *bigger* than the value depicted.

The real numerical problem appears when operands of floating-point operations are results of other floating-point operations, i.e. they are rounded versions of their "true values". Operation by operation, the numerical error accumulates without any bound.

The most numerically sensitive operations are subtraction and modulo. It is easy to see why. For example, both $t_1 = 8192 \times 8192 = 2^{13} \times 2^{13} = 2^{26}$ and $t_2 = 8191 \times 8193 = (2^{13} - 1) \times (2^{13} + 1) = 2^{26} - 1$ are evaluated as $2^{26} = 67108864$ in single precision, because significand is only 24 bits long; hence difference of their floating-point values is 0. The true difference value is of course $t_1 - t_2 = 67108864 - 67108863 = 1$. Generally, if the "true value" t_1 is approximated by a floating-point number $x_1 = t_1 + \varepsilon_1$ (ε_1 is the rounding error), other "true value" t_2 is approximated by $x_2 = t_2 + \varepsilon_2$, then subtraction of floating-point approximations is $x_1 - x_2 = t_1 - t_2 + \varepsilon_1 - \varepsilon_2 + \varepsilon_3$, where ε_3 is the rounding error of the subtraction. Naturally, if $t_1 \approx t_2$, then the subtraction result is strongly influenced by rounding errors, i.e. the relative error of the result is big.

The same problem (often called *cancellation* [1]) appears in the modulo operation, as $a \bmod b = a - \lfloor a/b \rfloor b$. The "hidden subtraction" inside the modulo operation damages the result, especially if a is just a floating-point approximation of a certain true value and $b \ll a$; in this case, $\lfloor a/b \rfloor \approx a/b$, therefore $\lfloor a/b \rfloor b \approx a$ and cancellation appears.

A short look at Eq. (1) reveals origin of strange behaviour in single precision calculations. The complex exponential can be written as $\exp(\mathrm{j}2\pi r/\lambda) = \cos(2\pi r/\lambda) + \mathrm{j}\sin(2\pi r/\lambda)$, so it is necessary to evaluate trigonometric functions of a very large argument, because usually $r \gg \lambda$. The trigonometric function evaluation begins with *range reduction* [5] of its argument, which is basically $(\bmod 2\pi)$ operation; therefore it is very sensitive to cancellation due to rounding errors of the argument. Let us analyse when to expect strong cancellation problems.

4. On-axis analysis

The function $\exp(\mathrm{j}2\pi r/\lambda)$ is a periodic function with respect to r. We know that a periodic function evaluation starts with range reduction operation, in this case by $(2\pi r/\lambda \bmod 2\pi)$. It follows that when $2\pi r/\lambda \gg 2\pi$, this operation evaluated in floating-point arithmetic cuts many

#220529 - $15.00 USD Received 6 Aug 2014; revised 22 Jan 2015; accepted 26 Jan 2015; published 4 Feb 2015

9 Feb 2015 | Vol. 23, No. 3 | DOI:10.1364/OE.23.003260 | OPTICS EXPRESS 3263

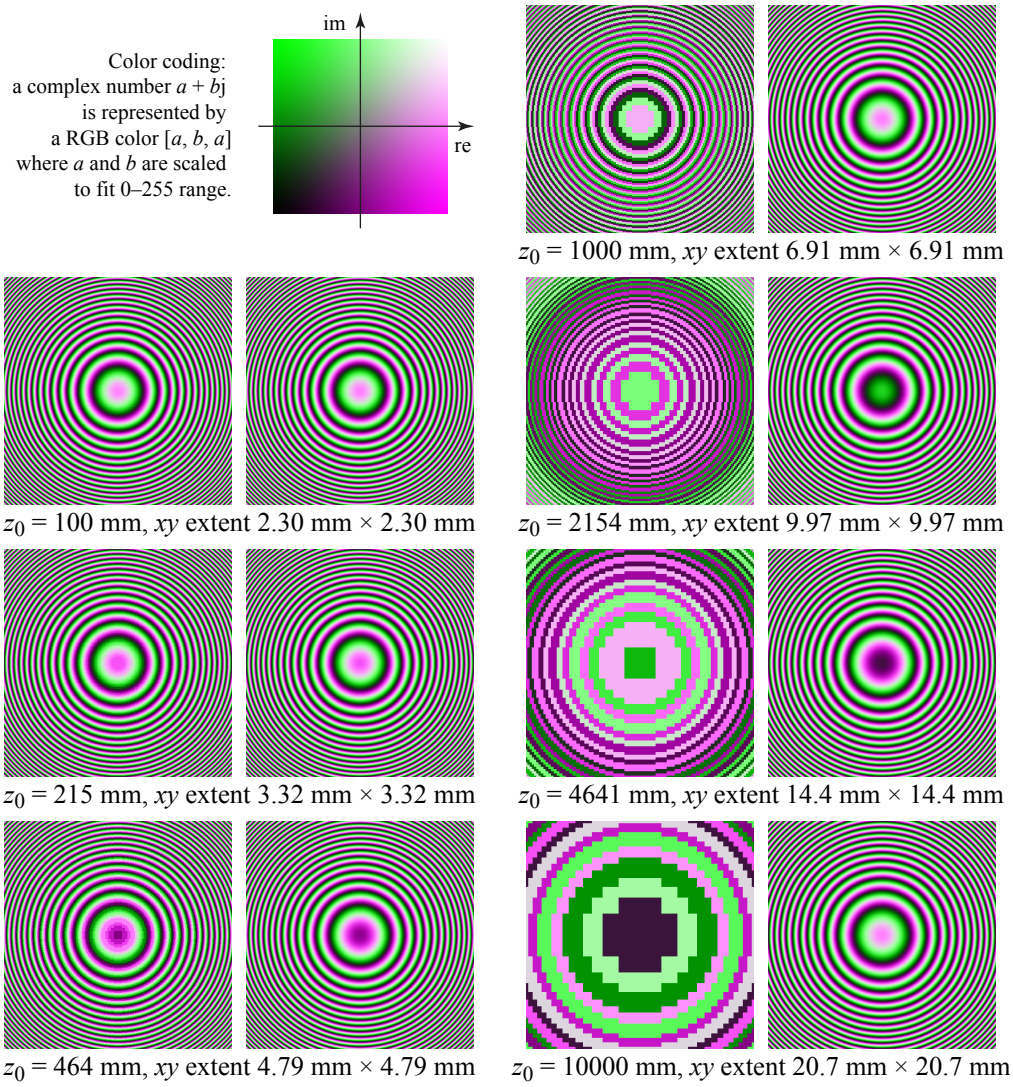


Fig. 3. Visualization of the phasor $\exp(\mathrm{j}2\pi r/\lambda)/r$ for various distances z_0 calculated in single and double precision. The lateral extent of each pair (single/double precision) is the same; it was chosen so that all images contain approximately the same number of fringes. It is not a fault that all double precision images look similar to each other; they really do.

significant digits of r. An extreme cancellation appears when $2\pi r/\lambda$ and e.g. $2\pi r/\lambda + \pi$ are expressed by the same floating point number; in this case, sine or cosine of both numbers is the same.

In order to capture fine details of $\exp(\mathrm{j}2\pi r/\lambda)$, the argument must be evaluated in floating-point arithmetic in such a way that its value differs for r_0 and $r_1 = r_0 + \lambda/n$ where $n \geq 2$. In other words, we need at least two different results per period in the same way as alias-free sampling requires at least two samples per period.

#220529 - $15.00 USD Received 6 Aug 2014; revised 22 Jan 2015; accepted 26 Jan 2015; published 4 Feb 2015

9 Feb 2015 | Vol. 23, No. 3 | DOI:10.1364/OE.23.003260 | OPTICS EXPRESS 3264

For simplicity, let us place the first point of evaluation to $[0,0,z_0]$, $z_0 > 0$, therefore $r_0 = z_0$. Let us place the second point of evaluation to $[x_1,0,z_0]$ in such a way that $r_1 = (x_1^2 + z_0^2)^{1/2} = r_0 + \lambda/n = z_0 + \lambda/n$. It is easy to see that $x_1^2 = 2z_0\lambda/n + (\lambda/n)^2$.

Let us continue. We need to evaluate $r_1 = (x_1^2 + z_0^2)^{1/2}$. It is absolutely necessary to hold inequality $x_1^2 + z_0^2 \neq z_0^2$ in floating-point evaluation; we have seen in Sec. 3 that floating-point summation may violate it under certain circumstances. If we assume $x_1 < z_0$, then the inequality will hold if

$$x_1^2 > 2^{-p+1} z_0^2,$$

where p is again number of significand (mantissa) bits, in case of single precision $p = 24$. After substitution,

$$2z_0\lambda/n + (\lambda/n)^2 > 2^{-p+1} z_0^2.$$

It is easy to solve the quadratic inequality for z_0; the square root in the result can be then approximated by a Taylor series (assuming $2^{-p} \ll 1$) and we get final simple result:

$$z_0 < \lambda 2^p / n. \tag{2}$$

For $\lambda = 500$ nm, $p = 24$ (single precision) and $n = 2$ we get approximately $z_0 < 4.19$ m. For $p = 53$ (double precision) we get approximately $z_0 < 2.25 \times 10^9$ m.

It is worth explaining meaning of this value. First of all, by setting $n = 2$ we calculated upper limit of z_0; beyond it, the evaluated value of $\exp(\mathrm{j}2\pi r/\lambda)$ has no significant digits. We also explored just one source of rounding error, the range reduction operation; other operations also contribute to the overall error, however their contribution is very small. We have found that the upper limit is slightly less than the calculated one, about $0.95 \times \lambda 2^p/n$. See Sec. 7 for further details.

Second, we are usually interested in evaluation of $\exp(\mathrm{j}2\pi r/\lambda)$ in a non-zero area, but this analysis is valid for points only on z axis. We should therefore generalize the analysis to an off-axis case; see Sec. 5. It would be definitely possible to show just off-axis analysis as it is more practical; however, its derivation is more complicated and the idea can be seen in on-axis analysis more clearly.

Finally, although evaluation near the upper limit theoretically captures the correct structure of the function $\exp(\mathrm{j}2\pi r/\lambda)$, the error is usually unacceptably big as the result has about one significant digit. For practical calculations, we should set $n \gg 2$, see Fig. 2 for a dependency of z_0 on n and Fig. 3 for actual evaluation of the function $\exp(\mathrm{j}2\pi r/\lambda)$. See also Sec. 7 for further details.

5. Off-axis analysis

We are usually interested in evaluation of the function $\exp(\mathrm{j}2\pi r/\lambda)$ in a non-zero area. Without loss of generality, we can analyse the function evaluation on the xz plane only.

Let us again place the first point of evaluation to $[x_0,0,z_0]$, i.e. $r_0 = (x_0^2 + z_0^2)^{1/2}$, and the second point of evaluation to $[x_1,0,z_0]$ such that $r_1 = (x_1^2 + z_0^2)^{1/2} = r_0 + \lambda/n$. By setting $a = x_0/z_0$ and applying the same procedure as in Sec. 4, we get simple rule:

$$z_0 < \lambda \frac{2^p}{n\sqrt{1+a^2}}. \tag{3}$$

The estimation of the upper z_0 seems to be reliable, see Fig. 4 for the visualization of the case $a = 1$ (i.e. 45° inclination), $\lambda = 500$ nm; inequality (3) predicts the calculation to be right for approximately $z_0 < 2.1$ m.

#220529 - $15.00 USD Received 6 Aug 2014; revised 22 Jan 2015; accepted 26 Jan 2015; published 4 Feb 2015

9 Feb 2015 | Vol. 23, No. 3 | DOI:10.1364/OE.23.003260 | OPTICS EXPRESS 3265

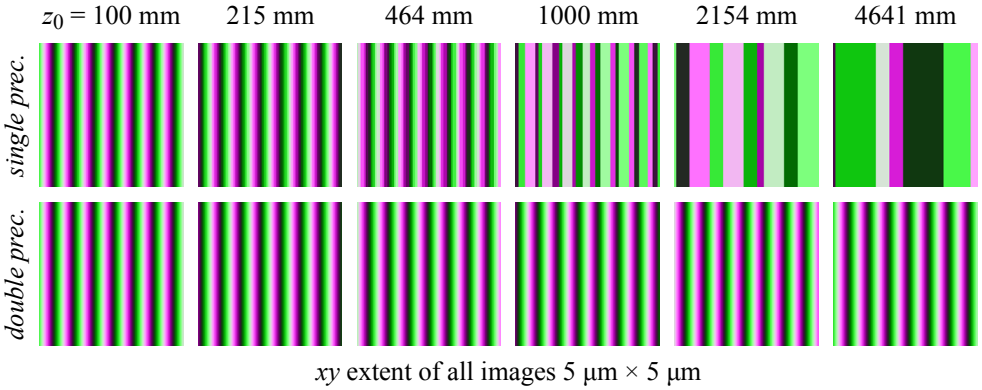


Fig. 4. Visualization of the phasor of off-axis calculation $\exp(\mathrm{j}2\pi r/\lambda)/r$ for various distances z_0 in single and double precision. The center of each image is set to $[z_0, 0, z_0]$, i.e. $a = 1$ (see inequality (3)). The color coding is the same as in Fig. 3.

For paraxial calculations where $x_0/z_0 \ll 1$, the result of inequality (2) is virtually the same as of inequality (3). In practical estimations, we can use inequality (2) even for off-axis calculations provided that we ask for "high enough" n; one should, however, ask if the scalar approximation of light is reliable in highly off-axis cases.

6. Fresnel approximation

The analysis and examples given clearly show that single precision calculations should be used with the utmost care; on the other hand, double precision calculations are good enough for most practical cases. Still, single precision calculation are attractive in massively parallel systems such as GPUs as single precision circuits naturally occupy less space on a chip, consume less power etc. It would be then desirable to avoid single precision problems without going to double precision.

A simplistic approach would be to use double precision for the calculation of r and the range reduction. Unfortunately, switching between single and double precision tends to be slow as GPU architectures are not optimized for such a weird task. Moreover, we still need double precision for a substantial part of the calculation.

We have seen that the precision problem arises when z_0 is big. On the other hand, in a big distance, we can use paraxial parabolic (Fresnel) approximation

$$\exp(\mathrm{j}2\pi\sqrt{x^2+y^2+z_0^2}/\lambda) \approx \exp(\mathrm{j}2\pi z_0/\lambda)\exp\left(\mathrm{j}\pi\frac{x^2+y^2}{z_0\lambda}\right) \tag{4}$$

provided that $x, y \ll z_0$, or more precisely (according to [6]) $z_0 \gg [\pi(x^2+y^2)^2_{\max}/4\lambda]^{1/3}$.

It is worth noting that there are no cancellation problems in Eq. (4). First of all, there is no summation of a large z_0 and small x, y in the argument of exp() function; recall that the most serious error appears when floating-point value of $x^2+y^2+z_0^2$ is almost the same as floating-point value of z_0^2. In Eq. (4), arguments dependent on z_0 and x, y are separated to separate exponential functions, i.e. their arguments are range reduced independently. There is no other modulo or subtraction operation prone to cancellation in Eq. (4). Visualization of the calculation in Fig. 5 therefore shows no significant sign of numerical problems except, of course, slightly different pattern due to z_0 quantization in single precision (i.e. z_0 in single precision $\neq z_0$ in

#220529 - $15.00 USD Received 6 Aug 2014; revised 22 Jan 2015; accepted 26 Jan 2015; published 4 Feb 2015

9 Feb 2015 | Vol. 23, No. 3 | DOI:10.1364/OE.23.003260 | OPTICS EXPRESS 3266

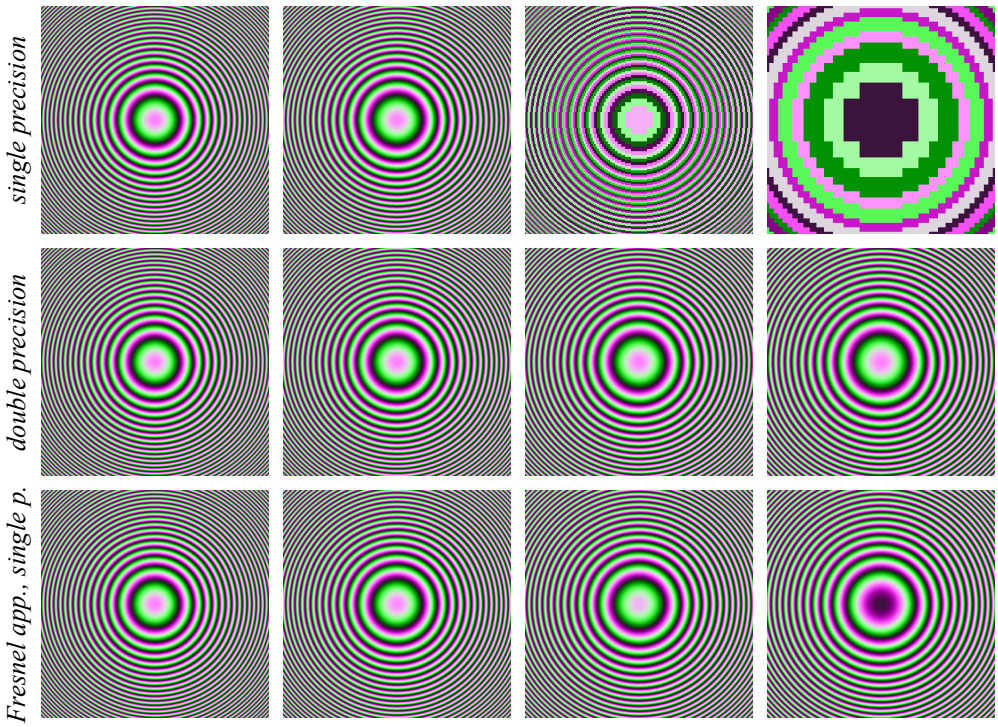


Fig. 5. Comparison of the phasor $\exp(\mathrm{j}2\pi r/\lambda)/r$ for various distances z_0 in single and double precision and in the Fresnel approximation in single precision. The lateral extent was chosen so that the images contain approximately the same number of fringes. The color coding is the same as in Fig. 3. The phase in the center of the image for Fresnel approximation, $z_0 = 10000$mm is different from double precision calculation because such a big $2\pi z_0$ cannot be represented in single precision well enough.

double precision). For convenience, Fig. 2 also shows various "Fresnel approximation safe" zones.

In an off-axis case, where the area of interest is located around a point $[x_0, y_0, z_0]$, it is possible to use parabasal parabolic approximation that approximates

$$\sqrt{(x+x_0)^2+(y+y_0)^2+z_0^2} \approx r_0 + \frac{xx_0+yy_0}{r_0} + \frac{x^2+y^2}{2r_0},$$

where $r_0 = (x_0^2+y_0^2+z_0^2)^{1/2}$, provided that $x, y \ll r_0$.

The only problem remains if it is necessary to evaluate $\exp(\mathrm{j}2\pi r/\lambda)$ over a large area, i.e. we cannot use parabolic approximation. It should be possible to use partial quadratic approximation of the square root [7], but we did not verify it.

7. Additional notes

We have mentioned in Sec. 4 that setting $n = 2$ in inequality (2) leads to an approximate value of the upper limit of z_0; beyond that limit, information about structure of the function $\exp(\mathrm{j}2\pi r/\lambda)$

#220529 - $15.00 USD Received 6 Aug 2014; revised 22 Jan 2015; accepted 26 Jan 2015; published 4 Feb 2015

9 Feb 2015 | Vol. 23, No. 3 | DOI:10.1364/OE.23.003260 | OPTICS EXPRESS 3267

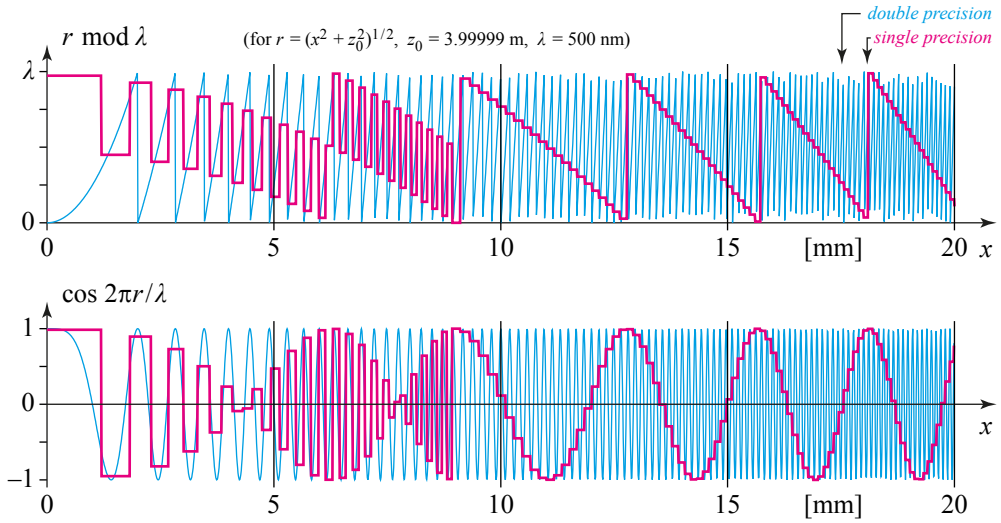


Fig. 6. A closer look at calculation $\exp(\mathrm{j}2\pi r/\lambda)/r$ for $z_0 = 3.99999$ in single and double precision. The upper graph shows "range reduction" of the argument, i.e. $(2\pi r/\lambda) \bmod 2\pi$, which is basically the same as $2\pi(r \bmod \lambda)$. The lower graph shows the real part of $\exp(\mathrm{j}2\pi r/\lambda)$. The distance z_0 was chosen so that the transition between "correct" and "wrong" calculations can be easily seen; in this case, the transition appears near $x = 9$ mm.

is lost. Let us try to find experimentally an example of precise z_0 where the effect of "information loss" can be seen.

Figure 6 shows evaluation of the function for $z_0 = 3.99999$ m. The upper part shows the result of range reduction in both single and double precision. It is clear that for $x < 9$ mm the evaluation performs "well" – at least in a way that single precision calculation returns two distinct values in a single cycle. In other words, the value of r is quantized to just two levels in each cycle.

The lower part of the figure shows the result of $\cos(2\pi r/\lambda)$. It is no surprise that for $x <$ 9 mm, the single precision result contains the same frequency as the double precision one. The effect for $x > 9$ mm would be normally called *aliasing* [8] as a high frequency content is misinterpreted as a low frequency content. In this case, *aliasing is a direct consequence of quantization of floating point numbers*.

The sampling theorem [6] states that the signal can be recovered if we have more than two samples per cycle. It is then worth asking if e.g. calculation for $z = 2.2$ m (see Fig. 7) is good enough. Let us then try to calculate (in single precision) a complex optical field on the plane $z = z_0$ of a single point source located at the origin and calculate the back-propagation (in double precision) to the plane $z = 0$. Figure 7 shows the results for various z_0; we have used the filtered Rayleigh-Sommerfeld convolution [9] for propagation calculation. It is clearly seen that even optical fields heavily damaged by errors of evaluation in single precision, but in the "safe zone of single precision", perform very well. Naturally, optical fields for z_0 beyond its upper limit are irreparably damaged.

This result is by no means surprising. Floating-point calculations lead to quantization of the argument of $\exp(\mathrm{j}2\pi r/\lambda)$, and therefore to phase quantization of the optical field. The effects of phase quantization are shown in e.g. [10]; it is known that even coarsely quantized phase does not completely damage information "stored" in an optical field.

#220529 - $15.00 USD Received 6 Aug 2014; revised 22 Jan 2015; accepted 26 Jan 2015; published 4 Feb 2015

9 Feb 2015 | Vol. 23, No. 3 | DOI:10.1364/OE.23.003260 | OPTICS EXPRESS 3268

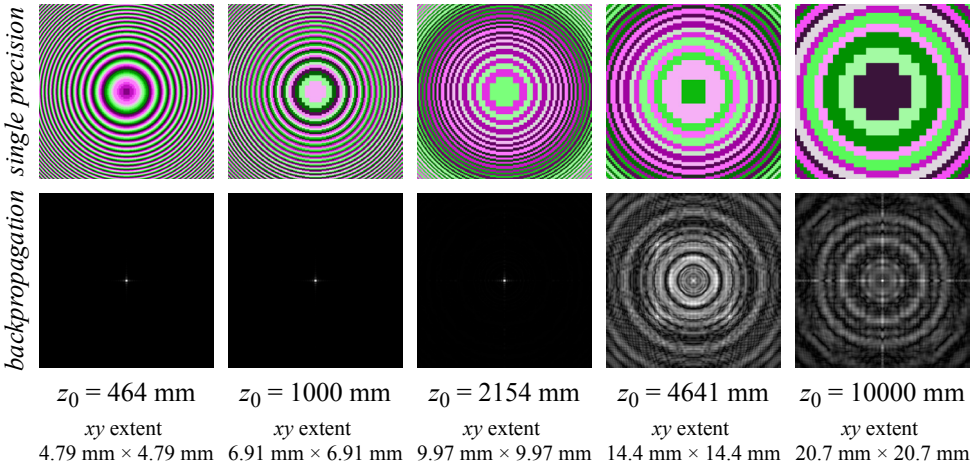


Fig. 7. Upper row: phasor of the optical field in the plane $z = z_0$ of a single point light source located at the origin. The lateral extent was chosen so that the images contain approximately the same number of fringes. Please note that fringes are damaged for $z_0 = 4641$ mm and $z_0 = 10000$ mm due to single precision calculation, not due to incorrect sampling rate. Lower row: normalized intensity of the light field backpropagated to the plane $z_0 = 0$. The images should contain just a single bright dot. Naturally, the last two images are damaged as the optical field was calculated incorrectly.

8. Conclusions

We have shown that unwary use of single precision floating-point numbers in calculations of free space light propagation can lead to significant errors or even completely pointless results, such as those presented in Sec. 2. The errors stem from the calculation of $(r \bmod \lambda)$, where r is a distance; serious precision problems can appear for approximately $r > 10^6\lambda$, the results are pointless for approximately $r > 10^7\lambda$. We have also shown that in certain cases, Fresnel approximation or similar one can significantly improve range where single precision calculations remain valid. For example, wrong results shown in Sec. 2 can be fixed using Fresnel approximation or at least predicted using analysis in Sec. 5. We have also shown that double precision calculations are safe in most practical situations.

While the analysis of computer arithmetic may seem to be rather technical and unrelated to optics, we believe the opposite is true. There are many approximations that are used in optics, e.g. scalar approximation of light, Fresnel approximation of light propagation, etc. Applicability of such "classical" approximations are widely studied using "classical" tools such as mathematical analysis; results are obvious for practical optics calculations. As most calculations today are performed with computer approximation of arithmetic, we believe that analysis of computer arithmetic issues should belong to standard mathematical toolbox of an optician. This is especially true because GPU calculations are so popular today and limited precision numbers are usually necessary in massively parallel calculations.

Acknowledgements

This work was supported by the European Regional Development Fund (ERDF), project "NTIS – New Technologies for the Information Society", European Centre of Excellence, CZ.1.05/1.1.00/02.0090, and by Ministry of Education, Youth, and Sport of Czech Republic – University spec. research – 1311.

#220529 - $15.00 USD Received 6 Aug 2014; revised 22 Jan 2015; accepted 26 Jan 2015; published 4 Feb 2015

Appendix E

Reference calculation of light propagation between parallel planes of different sizes and sampling rates

Reference calculation of light propagation between parallel planes of different sizes and sampling rates

Petr Lobaz*

Department of Computer Science and Engineering, University of West Bohemia, Univerzitní 8, 306 14 Plzeň, Czech Republic
**lobaz@kiv.zcu.cz*

Abstract: The article deals with a method of calculation of off-axis light propagation between parallel planes using discretization of the Rayleigh-Sommerfeld integral and its implementation by fast convolution. It analyses zero-padding in case of different plane sizes. In case of memory restrictions, it suggests splitting the calculation into tiles and shows that splitting leads to a faster calculation when plane sizes are a lot different. Next, it suggests how to calculate propagation in case of different sampling rates by splitting planes into interleaved tiles and shows this to be faster than zero-padding and direct calculation. Neither the speedup nor memory-saving method decreases accuracy; the aim of the proposed method is to provide reference data that can be compared to the results of faster and less precise methods.

OCIS codes: (070.2025) Discrete optical signal processing; (070.7345) Wave propagation.

References and links

1. J. W. Goodman, *Introduction to Fourier Optics* (Roberts & Company Publishers, 2004), 3rd ed.
2. E. Lalor, "Conditions for the validity of the angular spectrum of plane waves," J. Opt. Soc. Am. **58**(9), 1235–1237 (1968).
3. L. Onural, "Exact analysis of the effects of sampling of the scalar diffraction field," J. Opt. Soc. Am. A **24**(2), 359–367 (2007).
4. L. Onural, A. Gotchev, H. Ozaktas, and E. Stoykova, "A survey of signal processing problems and tools in holographic three-dimensional television," IEEE Trans. Circ. Syst. Video Tech. **17**(11), 1631–1646 (2007).
5. V. Katkovnik, J. Astola, and K. Egiazarian, "Discrete diffraction transform for propagation, reconstruction, and design of wavefield distributions," Appl. Opt. **47**(19), 3481–3493 (2008).
6. N. Delen, and B. Hooker, "Free-space beam propagation between arbitrarily oriented planes based on full diffraction theory: a fast Fourier transform approach," J. Opt. Soc. Am. A **15**(4), 857–867 (1998).
7. J.-L. Kaiser, E. Quertemont, and R. Chevallier, "Light propagation in the pseudo-paraxial fresnel approximation," Opt. Commun. **233**(4-6), 261–269 (2004).
8. E. Sziklas, and A. Siegman, "Diffraction calculations using fast Fourier transform methods," Proc. IEEE **62**(3), 410–412 (1974).
9. R. P. Muffoletto, J. M. Tyler, and J. E. Tohline, "Shifted Fresnel diffraction for computational holography," Opt. Express **15**(9), 5631–5640 (2007).
10. F. Zhang, G. Pedrini, and W. Osten, "Reconstruction algorithm for high-numerical-aperture holograms with diffraction-limited resolution," Opt. Lett. **31**(11), 1633–1635 (2006).
11. K. Matsushima, and S. Nakahara, "Extremely high-definition full-parallax computer-generated hologram created by the polygon-based method," Appl. Opt. **48**(34), H54–H63 (2009).
12. M. Frigo, and S. G. Johnson, "The design and implementation of FFTW3," Proc. IEEE **93**(2), 216–231 (2005) (Special issue on "Program Generation, Optimization, and Platform Adaptation").

1. Introduction

A fundamental tool of digital holography, or computer generated holography, is a numerical simulation of coherent light propagating in free space. We will use, as usual, scalar approximation of the vectorial nature of light, and will not consider time-dependent behavior of the light [1]. One of the most usual tasks is to calculate light propagation between two

#137599 - $15.00 USD Received 2 Nov 2010; revised 10 Dec 2010; accepted 12 Dec 2010; published 20 Dec 2010

parallel planes; the Rayleigh-Sommerfeld integral [1], its mathematical equivalent angular spectrum [1,2] or their various approximations are used most often.

Those equations need to be discretized for numerical calculation, i.e. one has to both sample and spatially restrict optical fields involved. The discretization itself leads to various errors well described in literature [3,4]. The calculation tries to avoid those errors while trying to work as fast as possible.

It is, however, often hard to decide what is an error of discretization itself, what is an inherent error of a given fast method and what is just an error of a particular implementation. The goal of this article is to describe the reference numerical calculation of light propagation between parallel planes that has just one "error", the discretization. Any other method can be then compared to this one. As we will assume fine sampling that does not lead to aliasing errors, we have to deal with a large amount of data. We will try to handle it as fast as possible while retaining the accuracy of the calculation.

The proposed method focuses on off-axis light propagation between two rectangular areas that share neither size nor sampling, or, between a spatial light modulator (SLM) and a camera sensor. Reference calculation is described, e.g., in [5], off-axis propagation, e.g., in [6,7]; different samplings are treated in [8] by coordinate system change, in [9] by shifted convolution kernel, in [10] by scaled Fourier transform; different sizes of SLM and sensor is solved in [10,11] using tiling while decreasing memory demands. This article solves all the requirements in a unified way by using the convolution approach and tiling; in contrast to references given, it deals with optimization too.

2. One-dimensional case

We are going to show all the principles in a one-dimensional case before showing the full 2D version. This means that we will calculate light propagation between two line segments instead of two rectangular areas. We will call them *Source* and *Target*.

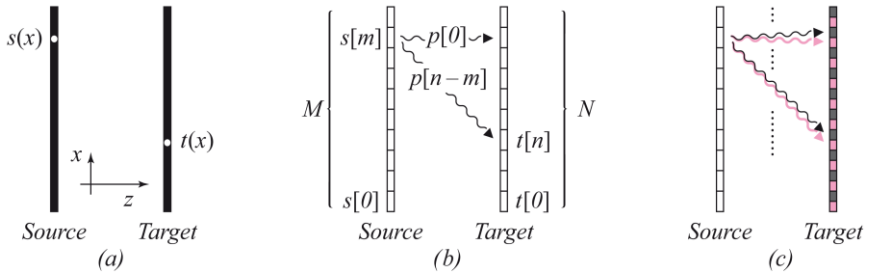


Fig. 1. One-dimensional light propagation. Description of (a), (b), (c) is given in the text below.

In linear optics it is assumed that light sources do not influence each other and that every single point $t(x)$ of *Target* (e.g. a camera sensor) is affected by the shining of all points $s(x)$ of *Source* (e.g. a SLM, see Fig. 1a). As mentioned in the introduction, we will assume scalar approximation of coherent light, i.e. the light source can be completely described by amplitude A and phase ϕ, or complex amplitude $A\exp(\mathrm{j}\phi)$, where $\mathrm{j}^2 = -1$.

Light changes both amplitude and phase by propagation. This change can by described by multiplication with some complex number p. Light propagation is space invariant; therefore, it is just the mutual position of points on *Source* and *Target* that matters, not their absolute positions. It follows that the calculation will have a convolution form:

$$t(x) = \int_{Source} s(\xi)\, p(x-\xi)\,\mathrm{d}\xi \qquad (1)$$

In the equation there is no distance along z axis between *Source* and *Target* because it is a constant and can be a part of the function p.

#137599 - $15.00 USD Received 2 Nov 2010; revised 10 Dec 2010; accepted 12 Dec 2010; published 20 Dec 2010

We can discretize the equation by equidistant splitting of *Source* and *Target* into M and N basic elements $s[m]$ $(0 \le m \le M-1)$ and $t[n]$ $(0 \le n \le N-1)$, i.e. by uniform sampling. Therefore, the element $t[n]$ receives from the element $s[m]$ light with complex amplitude $s[m]p[n-m]$, see Fig. 1b. Equation (1) can be written in discrete form:

$$t[n] = \sum_{m=0}^{M-1} s[m]p[n-m] \qquad \text{for } 0 \le m \le M-1, \quad 0 \le n \le N-1. \tag{2}$$

To compute all elements $t[n]$, we have to know numbers $p[n-m]$ for $-(M-1) \le n-m \le N-1$, i.e. $M+N-1$ different values. We can obtain them in different ways, some of which will be mentioned in section 3. For now, let us assume we know them.

The calculation of all elements $t[n]$ is most often done by rewriting Eq. (2) as a cyclic convolution and subsequent use of the discrete Fourier transform. The cyclic convolution has a form:

$$t[n] = \sum_{m=0}^{C-1} s[m]p[(n-m) \bmod C] \qquad \text{for } 0 \le m \le C-1, \quad 0 \le n \le C-1, \tag{3}$$

where $C \ge M+N-1$ and $s[m] = 0$ for $M \le m \le C-1$ (i.e., the s is zero-padded to the size C). Notice that the important values of $t[n]$ are those for $0 \le n \le C-M$; the others are damaged by the cyclic behavior of indices in arithmetic $(\bmod C)$. It is also worth mentioning that in cyclic convolution it is usually assumed that $M = N$. The proof of validity for the case $M \ne N$ consists just in the expansion of equations for $t[n]$. Finally, C is an arbitrary number bigger than or equal to $M+N-1$. By choosing a suitable C, we can speed up the computation significantly, because

$$t = \mathbf{IDFT}(\mathbf{DFT}(s) \times \mathbf{DFT}(p)), \tag{4}$$

where t, s and p are C-dimensional vectors (arrays) of complex numbers, **DFT** is a discrete Fourier transform of a vector, **IDFT** is an inverse discrete Fourier transform of a vector and $\times$ is the Hadamard product (element by element product). The speedup is expected due to the fact that the calculation of **DFT**, or **IDFT**, can be done in time $O(C \log C)$ [12]. Choosing a suitable C is important because the actual calculation time is highly sensitive to its character.

Let us assume that the sampling rate of *Target* is twice as fine as the sampling rate of *Source*. Then the subset of even samples from *Target* has the same sampling rate as *Source*. Consequently, we can easily calculate the propagation of *Source* to even samples of *Target*. Obviously, we can do the same with odd samples. It means we can split the calculation of light propagation into two calculations; they are denoted in Fig. 1c by black and magenta arrows. It follows that the same principle can be applied when the sampling rate of *Target* is τ-times finer than the sampling rate of *Source*, where τ is a natural number; we have to split *Target* into τ "*interleaved tiles*", i.e. the calculation has to be split into τ calculations.

The same situation appears when *Source* has σ-times finer sampling than *Target*. The idea can be generalized: if the sampling rates of *Source* and *Target* are in a ratio $\sigma : \tau$, where σ and τ are coprime natural numbers (i.e. σ elements of *Source* have the same size as τ elements of *Target*), we can split the calculation into $\sigma \times \tau$ independent calculations. More precisely, we have to σ-times calculate the propagation for the sampling rate ratio $1 : \tau$ and sum the results. Usually we do not care about the exact value of the sampling rate; therefore, we can choose such σ and τ that approximate the desired sampling fairly well.

#137599 - $15.00 USD Received 2 Nov 2010; revised 10 Dec 2010; accepted 12 Dec 2010; published 20 Dec 2010

It follows from Eq. (3) that the vectors s and t have to be padded to size C. This means that for $M=N$, approximately 50% of elements are held in memory uselessly; in 2D convolution, it is as much as 75%. Therefore, for big M and N we can expect a lack of memory very soon, especially when using special hardware for **DFT** calculation, e.g. GPU. For example a naive approach to propagation of a microscopic *Source* to an extended detector *Target* could require hundreds of gigabytes.

We need two memory spaces of size $C \geq M+N-1$ for the calculation of Eq. (4); in practice a restriction may appear: that just two spaces of size $P<C$ are available. Let us assume, for example, that $M=512$, $N=1024$ and $P=1024$. We cannot make the calculation directly because $C \geq 1535$; but we can split *Target* in the middle into two parts (let us call them "*common tiles*") with $N'=512$ elements and make two calculations. For them, we need only two spaces of size at least $M+N'-1=1023$, and therefore we are not limited by $P=1024$. The same idea would apply if $M=1024$, $N=512$ and $P=1024$: we would calculate the light propagation of both small parts of *Source* to *Target* and sum the results. As in the "different sampling" case, even this idea can be generalized: *Source* can be split into S parts, *Target* into T parts and then we have to calculate $S \times T$ propagations.

3. Two-dimensional case

The extension of equations from section 2 to the 2D case is straightforward: instead of sums we just put double sums there. For calculation of light propagation of a part of *Source* to a part of *Target*, it is necessary to carefully calculate the convolution kernel, i.e. 2D array p. A practical aid is the fact that its element $p[0,0]$ describes light propagation from element $s[0,0]$ of a particular part of *Source* to element $t[0,0]$ of a particular part of *Target*. The equations for sampling rates, samples counts and offsets are simple but technically demanding, so we will not show them here.

We should, however, mention the calculation of convolution kernel values. The Rayleigh-Sommerfeld equation [2] for the light propagation from the plane $z=0$ to point $[x,y,z]$ is:

$$U(x,y,z)=\frac{-1}{2\pi}\iint_{-\infty}^{\infty}U(\xi,\eta,0)\frac{\partial}{\partial z}\frac{\exp(\mathrm{j}kr)}{r}\mathrm{d}\xi\,\mathrm{d}\eta$$

where $U(x,y,z)$ is a complex amplitude of light in a point $[x,y,z]$, $k=2\pi/\lambda$ is a wavenumber (λ is a wavelength), and r is the distance between points $[x,y,z]$ and $[\xi,\eta,0]$. This equation can be written in a convolution form:

$$U(x,y,z)=U(x,y,0)\otimes\left[\frac{-z}{2\pi}\left(\mathrm{j}k-\frac{1}{\sqrt{x^2+y^2+z^2}}\right)\frac{\exp(\mathrm{j}k\sqrt{x^2+y^2+z^2})}{x^2+y^2+z^2}\right] \quad (5)$$

where $\otimes$ is the 2D convolution operator, $(f\otimes g)(x,y)=\iint_{-\infty}^{\infty}f(\xi,\eta)g(x-\xi,y-\eta)\mathrm{d}\xi\,\mathrm{d}\eta$.

The simplest method of convolution kernel discretization (the expression on the right side of the convolution operator in Eq. (5)) is a plain sampling, i.e. its calculation for a particular x, y (z is a constant). Alternatively, we can assume that a sample of *Source* in fact expresses – using some pixel spread function – the shining of a particular non-zero-area element of *Source* and alter the kernel accordingly. If we take non-zero area of *Target* (sensor) elements into account, we can pre-filter the kernel. If *Source* is a mathematical model of a real display, we can even measure the kernel. The proposed method therefore does not depend on particular features of the kernel; its only assumption is the description of light

#137599 - $15.00 USD Received 2 Nov 2010; revised 10 Dec 2010; accepted 12 Dec 2010; published 20 Dec 2010

propagation as a convolution. In our implementation, we did not deal with advanced methods of kernel calculation, however, and we have chosen plain sampling.

4. Theoretical time of computation

The calculation works with arrays of size $C = C_x \times C_y$ samples. It consists of three **DFT** 's (more precisely, two forward and one backward), calculation of the convolution kernel p with complexity $O(C)$ and the Hadamard product of the same complexity. For a large C, only times spent on **DFT** 's matter. The time of calculation using the fast Fourier transform is therefore proportional [12] to

$$(\textbf{DFT}\text{'s count})C\log C \tag{6}$$

In the following analysis, we will assume light propagation from a square area of size $M \times M$ samples to a square area $N \times N$ samples. For the convolution calculation, we will assume memory space $C = (M+N)\times(M+N)$. Time of calculation is then given by:

$$t_{\text{basic}}(M,N) = 3(M+N)^2 \log(M+N)^2 \equiv 6(M+N)^2 \log(M+N)$$

Let us begin with the case that *Source* and *Target* share the sampling. Then we can also split *Target* into $T \times T$ common tiles of size $(N/T)\times(N/T)$. It follows that we calculate **DFT**(*Source*) once and calculate **DFT**(p) and **IDFT**(**DFT**(*Source*)$\otimes$**DFT**(p)) $T \times T$ -times, while we assume the array sizes for **DFT** as $C = (M+N/T)\times(M+N/T)$. Using Eq. (6), we get time of calculation

$$t_{\text{common tiles}}(M,N,T) = \left(1+2T^2\right)2\left(M+\frac{N}{T}\right)^2 \log\left(M+\frac{N}{T}\right)$$

Let us assume that $M = \omega N$ and watch the speedup

$$s_1(\omega,N,T) = \frac{t_{\text{basic}}(\omega N,N)}{t_{\text{common tiles}}(\omega N,N,T)}$$

of tiled calculation versus direct calculation. The graph of $s_1(\omega,N,T)$ shows that tiled calculation starts to be faster for $T = 2$ and $\omega < 1/4$, while for bigger T it is faster for even smaller ω. Figure 2a shows the graph for $N = 1024$; for different N it does not change very much.

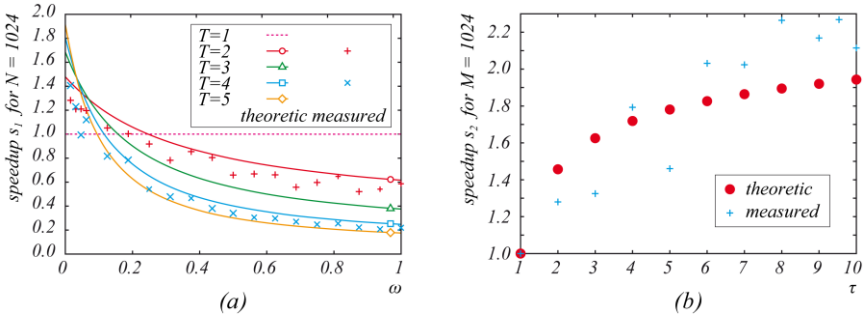


Fig. 2. The ratio of direct calculation time to (a) tiled calculation for given ratio ω of *Source* to *Target* side sizes (for *Target* side size N = 1024), and (b) tiled calculation for given ratio τ of *Target* to *Source* and sampling rates (for *Source* side size M = 1024). FFTW library was used for time measurement.

#137599 - $15.00 USD Received 2 Nov 2010; revised 10 Dec 2010; accepted 12 Dec 2010; published 20 Dec 2010

A more interesting result arises when *Source* and *Target* have different sampling rates. Let us assume that *Target* has τ -times finer sampling than *Source*, i.e. $N = \tau M$. For direct calculation, *Source* must have the same sampling rate as *Target*, i.e. it must have $\tau M \times \tau M$ samples too. We can increase number of samples by interpolation or just by interleaving current ones by a suitable amount of zeros. By substituting $C = (\tau M + \tau M) \times (\tau M + \tau M)$ into Eq. (6), we get time of calculation

$$t_{\text{upscaled}}(M, \tau) = 6(\tau M + \tau M)^2 \log(\tau M + \tau M) \equiv 24\tau^2 M^2 \log(2\tau M)$$

If we split *Target* into $\tau \times \tau$ interleaved tiles, we have to calculate $\tau \times \tau$ light propagations from *Source* to a part of *Target* of size $M \times M$ samples. By substituting $C = (M + M) \times (M + M)$ into Eq. (6), we get time of calculation

$$t_{\text{interleaved tiles}}(M, \tau) = \left(1 + 2\tau^2\right) 2\left(M + M\right)^2 \log\left(M + M\right) \equiv \left(1 + 2\tau^2\right) 8M^2 \log\left(2M\right)$$

The speedup

$$s_2(M, \tau) = \frac{t_{\text{upscaled}}(M, \tau)}{t_{\text{interleaved tiles}}(M, \tau)} \equiv \frac{3\tau^2 \log(2\tau M)}{\left(1 + 2\tau^2\right) \log\left(2M\right)}$$

is bigger than 1 for $\tau > 1$, i. e., tiled calculation is always faster; a graph of $s_2(M, \tau)$ dependent on τ for $M = 1024$ is in Fig. 2b. Again, a different M leads to a similar graph.

One can see in Fig. 2 that measured speedups are scattered around theoretical curves. A discussion of this fact is contained in the following section.

Until now we have assumed that *Target* has more samples than *Source*. That led to saving nearly one-third of all **DFT**'s. In the opposite situation this does not hold any more; however, the results of the analysis are similar, although not so outstanding. The analysis of general situations, where the ratio of sampling rates or sizes is a general fraction, leads to the following results: speedup is greater when the nominator and denominator of the sampling rate ratio are big too; splitting of *Source* and *Target* of the same sampling rate into common tiles in a complicated ratio leads rather to slowdown if we do not mind memory restrictions.

5. Implementation notes

The implementation of the proposed method is simple, although due to a number of variables quite demanding. The first step is to estimate the ratio of *Source* and *Target* sampling rates, and to split them into interleaved tiles of common sampling. In case we have enough memory to calculate light propagations between these tiles, we can calculate them (but see later). In another case, we are facing a still unresolved problem.

Let us assume that two memory spaces of size C are available. The interleaved tiles created by splitting *Source* and *Target* due to the requirement of common sampling have to be split into $S_x \times S_y$ common tiles of size $M_x \times M_y$, and $T_x \times T_y$ common tiles of size $N_x \times N_y$ (due to integer division one row and one column may be smaller) so that:

$$(M_x + N_x - 1)(M_y + N_y - 1) \le C$$

while the time of the calculation has to be as short as possible. We cannot trust the theoretical time complexity of **DFT** $O(C \log C)$ when searching for optimum; algorithms of **DFT** for special array sizes (e.g. power of 2) are usually faster than for comparable (e.g. prime) array sizes. For example, FFTW library used in our implementation calculates **DFT** of special array sizes up to 6 × faster than for comparable prime sizes. This is the first expected reason for scattered look of the measured data in Fig. 2. One can see in Fig. 2b that measured speedups

#137599 - $15.00 USD Received 2 Nov 2010; revised 10 Dec 2010; accepted 12 Dec 2010; published 20 Dec 2010

are bigger than theoretical ones. This behavior starts to appear for $M > 512$ in our implementation. We expect this behavior appears due to a following fact. When calculating a propagation without any tiling, we have to increase *Source* side size τ-times, so **DFT** has to work with a big array. Using interleaved tiling, **DFT** works with smaller arrays that fit into cache memory more easily, and therefore bigger speedup can be expected.

Implemented heuristic suggests splitting in this way. First, we have measured calculation times $time[i]$ of one-dimensional **DFT** 's of array sizes i. From these times we have picked "friendly-size" ones that satisfy condition $time[friendly\text{-}size] < time[i]$ for all measured $i > friendly\text{-}size$. Two-dimensional **DFT** is separable, so it is expected that a two-dimensional array of *friendly-size* side sizes will be "friendly" too. Next, we have measured calculation times for those arrays. The propagation itself is then calculated in 2D "friendly-size" arrays. In case we need to split into common tiles due to memory restrictions, we choose tiles of maximum width (tiles are then in fact horizontal stripes); height of *Source* and *Target* tiles is chosen to be approximately equal and as big as possible. We have compared this heuristic to the optimal solution found using brute force; it suggests at most approximately $1.7\times$ worse tiling.

A topic that has not been discussed yet is the precision of the proposed algorithm compared to the precision of direct calculation without any tiling. We have tried to propagate the *Source* to the *Target* of the same size (for several different sizes) using different tiling schemes. We have found that the relative difference of the calculated complex amplitudes is in the order of 10^{-10} or smaller; the difference was calculated as the absolute value of complex amplitudes differences. This difference is so small that we have not tried to find where it comes from.

6. Conclusion

A method for reference calculation of light propagation between two rectangular areas of parallel planes has been presented. These areas do not have to have either the same sampling rate or the same size, see Fig. 3. The calculation can be split into tiles to meet memory restrictions given; on the other hand, saving memory often leads to worse calculation times. We have shown that in the case of a simple sampling rates ratio, the proposed method actually speeds up the calculation up to $2\times$ by splitting the areas into interleaved tiles compared to zero-padding and direct calculation. We have shown as well that if one area is much bigger than the other, it is faster to split the bigger one into common tiles. A still unresolved problem is how to split the calculation into common tiles (due to memory restrictions) to get the fastest calculation. We have, however, proposed suboptimal heuristic to address this problem.

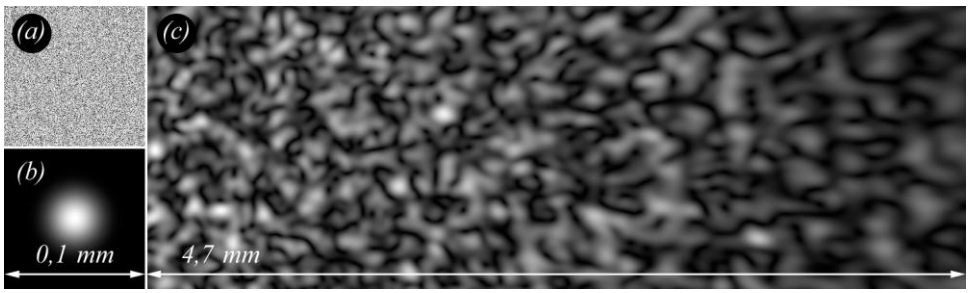


Fig. 3. Speckle simulation: an example of algorithm output. A patch of size 0.1 × 0.1 mm^2 with a random phase (a) and a Gaussian intensity (b) was sampled to a 1024 × 1024 array of complex amplitudes. Intensity of the off-axis propagation to the distance 5 mm is shown in the subimage (c), size of the subimage is 4.7 × 1.6 mm^2. A naive approach to the convolution would require approx. 28 GiB of memory while proposed algorithm can work on a common PC.

#137599 - $15.00 USD Received 2 Nov 2010; revised 10 Dec 2010; accepted 12 Dec 2010; published 20 Dec 2010

Acknowledgments

This work has been supported by the Ministry of Education, Youth and Sports of the Czech Republic under the research program LC-06008 (Center for Computer Graphics). The author would like to thank Ivo Hanák, Václav Skala and the reviewers for their valuable comments.

#137599 - $15.00 USD Received 2 Nov 2010; revised 10 Dec 2010; accepted 12 Dec 2010; published 20 Dec 2010

Appendix F

Binarizace počítačem generovaného hologramu pomocí ditheringu

[Binarization of computer generated holograms using dithering noise]

Zasláno k publikaci v časopisu Jemná mechanika a optika
ISSN 0447-6441

Publikováno v Roč. 56, č. 10 (2011), s. 290, pokračování č. 11-12 (2011), s. 303–305

název článku: Binarizace počítačem generovaného hologramu pomocí ditheringu
název článku anglicky: Computer generated hologram binarization using dithering noise
klíčová slova: difrakce, difrakční mřížka, počítačem generovaný hologram, binarizace, dithering
klíčová slova anglicky: diffraction, diffraction grating, computer generated hologram, binarization, dithering
abstrakt:
Článek se zabývá zobrazením počítačem generovaného hologramu 3D objektu na binárním amplitudovém prostorovém modulátoru světla, respektive binarizací difraktivních struktur. Rozvíjí myšlenku binarizovat difraktivní strukturu pomocí techniky ditheringu. Dospívá k závěru, že technika je vhodná pro jednoduché struktury typu difrakční mřížka, ale není vhodná pro počítačem generované hologramy 3D objektu.
abstrakt anglicky:
The article discusses display of a computer generated hologram of a 3D object using a binary amplitude spatial light modulator. More precisely, it deals with binarization of a diffractive structure using dithering noise. It concludes that the dithering method is suitable for simple diffractive structures such as diffractive gratings, but fails to provide a good binarization of a computer generated hologram of a 3D object.
autoři:

- Ing. Petr Lobaz
  Západočeská univerzita v Plzni
  Fakulta aplikovaných věd
  katedra informatiky a výpočetní techniky
  Univerzitní 8, 306 14 Plzeň
  lobaz@kiv.zcu.cz
- RNDr. Libor Kovář, CSc.
  Vysoké učení technické v Brně
  Fakulta informačních technologií
  Ústav počítačové grafiky a multimédií
  Božetěchova 2, 612 66 Brno
  kovarlib@fit.vutbr.cz

Binarizace počítačem generovaného hologramu pomocí ditheringu

1. Úvod

Displeje poskytující třírozměrný vjem jsou založeny na různých principech [1]. Nejjednodušší z nich jsou založeny na binokulárním vidění člověka, tedy poskytují pravému a levému oku diváka „správný“ obraz 3D scény; některé takové typy displejů jsou dnes běžně komerčně dostupné (vyžadující po divákovi aktivní nebo pasivní brýle), jiné (autostereoskopické) existují jako komerční prototypy a prakticky nic nebrání jejich masové výrobě. Velkou nevýhodou všech takových displejů je zejména konflikt mezi oční vergencí a akomodací [1]; oči se při sledování takového displeje chovají jinak než při sledování reálné scény. Dalšími potížemi (více či méně řešitelnými) jsou omezení počtu diváků, restrikce na pozorovací prostředí apod.

Technologie, která potenciálně řeší veškeré potíže, je elektronický holografický displej. Není totiž závislý na binokulárním vidění člověka, ale pokouší se napodobit přímo světelné pole. Elektronické holografické displeje dostatečné kvality však zatím neexistují; pro testování výsledků teoretického výzkumu počítačového generování hologramů (computer generated holography, CGH) [2, 3, 4] se proto zatím používá elektronová nebo laserová litografie (ty však neumožňují zobrazení videa) [3], rychlé fotorefraktivní materiály [5], akusticko-optické modulátory nebo prostorové modulátory světla (spatial light modulator, SLM) [6].

Používání amplitudových prostorových modulátorů světla založených na mikrozrcátkové technologii (digital micromirror device, DMD) [6] má kromě nesporných výhod (rychlá odezva, vynikající kontrast) také jednu podstatnou nevýhodu – jde o binární modulátor světla s pevně danou rozlišovací schopností. I tak jednoduchá úloha, jakou je zobrazení amplitudové mřížky se sinovým průběhem reflektance, je na DMD těžko řešitelná, neboť vlivem binarizace se průběh reflektance mění na obdélníkový (navíc s hranami v pevně daném rastru) a mřížka začne generovat nežádoucí difrakční řády. Jejich potlačení, resp. potlačení vlivu binarizace, se dá řešit iterativními algoritmy [7], které jsou ovšem časově náročné; půltónováním [8], které se dá úspěšně aplikovat jen při submikrometrovém rozlišení difraktivní struktury; posuny jednotlivých vrypů [9], které lze zajistit u výroby fyzické mřížky, nikoliv ovšem při simulaci na displeji; nebo zašuměním signálu [10], tzv. ditheringem.

Článek [10] ukazuje, jak zašuměním signálu, tj. difraktivní struktury, potlačit nežádoucí difrakční řády. Kromě počítačem generovaných hologramů se zabývá i jednoduchými mřížkami se sinovým profilem; uvažuje však jednak fázové modulátory světla, jednak předpokládá možnost generování šumu s jemnou (8bitovou) kvantizací. V tomto příspěvku ukážeme, jak se klasický dithering [11] projevuje na binárním amplitudovém prostorovém modulátoru světla, a to jak při zobrazování jednoduchých mřížek, tak při zobrazení počítačem generovaných hologramů.

Struktura článku je následující. V kapitole 2 připomeneme chování amplitudové difrakční mřížky se sinovým a obdélníkovým profilem, v kapitole 3 shrneme princip ditheringu.

V kapitole 4 ukážeme vliv DMD na difrakční mřížku i na počítačem generovaný hologram a ukážeme, že zašumění má dobrý vliv na mřížku, ale nikoliv na hologram. V kapitole 5 se pokusíme postup zašumění modifikovat a ukážeme, že ani modifikace nemá na rekonstrukci hologramu dobrý vliv. V závěru výsledky experimentů shrneme.

V následujícím textu budeme pozorovat difrakci na různých strukturách. Budeme-li pozorovat výpočetní simulaci, budeme předpokládat, že struktura mění procházející světlo lokální změnou transmitance. V případě experimentu na DMD čipu budeme měnit odražené světlo lokální změnou reflektance. Experiment ale připravíme tak, aby byl ekvivalentní transmitančnímu případu.

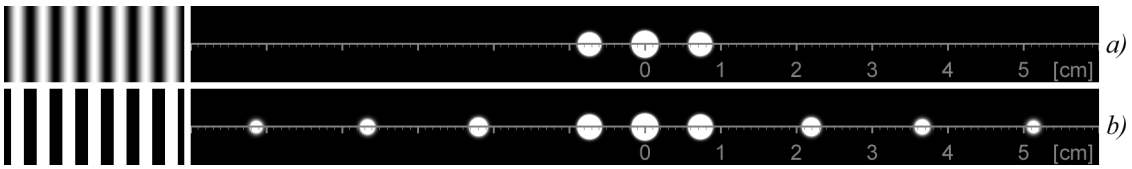
Jemná mechanika a optika ISSN 0447-6441, Roč. 56, č. 10 (2011), s. 290

2. Amplitudová difrakční mřížka

Popis chování amplitudové transmitanční difrakční mřížky patří k základním znalostem v optice [12]. Nekonečně velká mřížka s periodou struktury Δ (tj. prostorovou frekvencí $f = 1/\Delta$) kolmo nasvícená koherentní rovinnou vlnou vlnové délky λ vytváří ve vzdálené oblasti pod úhlem θ_m (úhel mezi nedifraktovaným a difraktovaným směrem šíření vlny) m-té difrakční maximum, kde m je celé číslo; platí *mřížková rovnice*

$$\sin\theta_m = m\frac{\lambda}{\Delta}.$$

Intenzita difrakčních maxim je pak dána transmitančním profilem. Je-li průběh sinový, jsou významná pouze maxima 0., +1. a –1. řádu, viz obr. 1. Je-li průběh obdélníkový, je třeba uvažovat všechny řády. Některé však mohou vlivem poměru šířky průhledné štěrbiny a periody mřížky (faktor plnění, *fill-factor*) zmizet, např. pro faktor plnění 0,5 mizí maxima sudých řádů (kromě 0. řádu).



Obr. 1: Difrakce na mřížce 11,574 čar/mm (perioda 8× rozteč mikrozrcátek) osvětlené paprskem He-Ne laseru (632,8 nm) ve vzdálenosti 1000 mm, výpočetní simulace. a) Transmitance ideální mřížky (předpokládáme rozsah 0 až 1), resp. mřížky s kvantizací na 256 stupňů šedi, a simulace difrakčního vzoru. b) Struktura binarizovaná prahem 0,5 a její difrakční vzor.

3. Dithering

Libovolný reálný signál $s(\boldsymbol{x})$ je zapotřebí před číslicovým zpracováním kvantizovat, tj. vyjádřit jeho obor hodnot konečným počtem diskrétních úrovní; říkáme jim kvantizační úrovně. Jejich počet závisí jak na použité technologii zpracování signálu, tak na požadavcích na odchylku původního a kvantizovaného signálu. Této odchylce říkáme kvantizační šum $n_K(\boldsymbol{x})$. Podrobné informace o kvantizaci lze najít v libovolné monografii o číslicovém zpracování signálu, např. [13].

Nejjednodušší kvantizací je binarizace, tj. máme k dispozici dvě kvantizační úrovně s_0 a s_1. Rozhodnutí, zda signál v daném bodu reprezentovat první či druhou kvantizační úrovní, se nejčastěji dělá porovnáním s prahovou hodnotou t. Výsledkem kvantizace je nový signál s_K:

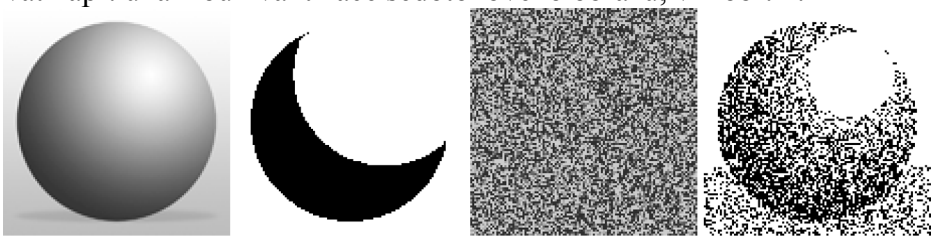
$$s_\mathrm{K}(\boldsymbol{x}) = \begin{cases} s_0 & \text{je-li } s_\mathrm{K}(\boldsymbol{x}) \le t \\ s_1 & \text{je-li } s_\mathrm{K}(\boldsymbol{x}) > t \end{cases}.$$

Prahovou hodnotu t volíme tak, aby byl kvantizační šum $n_\mathrm{K}(\boldsymbol{x}) = s(\boldsymbol{x}) - s_K(\boldsymbol{x})$ minimální ve smyslu zvoleného kritéria, např. minimální amplitudy.

Vliv kvantizace se dá někdy potlačovat tzv. ditheringem; používá se například ve zpracování obrazu nebo zvuku [11]. Jeho princip je jednoduchý – před binarizací se k signálu přičte náhodný šum $n_D(\boldsymbol{x})$ s amplitudou d:

$$s_\mathrm{KD}(\boldsymbol{x}) = \begin{cases} s_0 & \text{je-li } s_\mathrm{K}(\boldsymbol{x}) + n_\mathrm{D}(\boldsymbol{x}) \le t \\ s_1 & \text{je-li } s_\mathrm{K}(\boldsymbol{x}) + n_\mathrm{D}(\boldsymbol{x}) > t \end{cases}.$$

Je-li práh t aritmetickým průměrem kvantizačních úrovní, $t = (s_1 + s_0)/2$, volí se často amplituda ditheringového šumu $d = (s_1 - s_0)/2$. Kvalita kvantizovaného signálu se objektivně samozřejmě zhorší, ale statisticky bude kvantizovaný signál původnímu signálu bližší. To lze dokumentovat např. ukázkou kvantizace šedotónového obrazu, viz obr. 2.



a) *b)* *c)* *d)*

Obr. 2: Vliv ditheringového šumu na binarizaci. a) Šedotónový obrázek. b) Prostá binarizace. c) Náhodný šum použitý pro dithering. d) Binarizace s použitím ditheringového šumu.

4. Difraktivní struktura na DMD

Při zobrazení difraktivní struktury na DMD musíme brát v úvahu tři jevy:

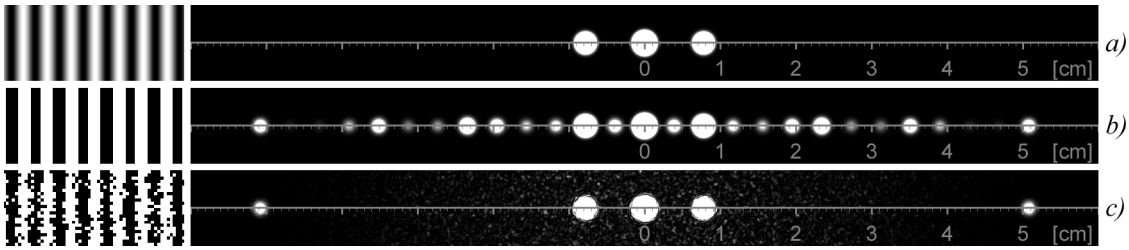
1. DMD čip světlo buď odrazí, nebo neodrazí. Jde tedy o binární modulátor amplitudy. Používá-li se DMD v režimu zobrazení stupňů šedi (typicky s 256 kvantizačními úrovněmi), jde o oklamání oka pulsně-šířkovou modulací, tj. dané mikrozrcátko periodicky přepíná mezi stavy odráží světlo – neodráží světlo.
2. DMD čip je rozdělen na obrazové elementy (*pixely*). Hranice mezi odrazivou a neodrazivou oblastí tedy nemůže ležet kdekoliv, ale jen v předem daném rastru.
3. Mikrozrcátko nepokrývá celou plochu pixelu. Odrazivá plocha je tedy de facto překryta neodrazivou čtvercovou mřížkou s roztečí stejnou, jako je rozteč pixelů.

V experimentech jsme použili čip Texas Instruments 0.95" 1080p (1920×1080) 2×LVDS ze sady DLP® Discovery™ 4100:

- rozlišení čipu 1920 sloupců × 1080 řádek,
- rozteč zrcátek 10,8 μm,
- mikrozrcátko pokrývá 91 % plochy pixelu.

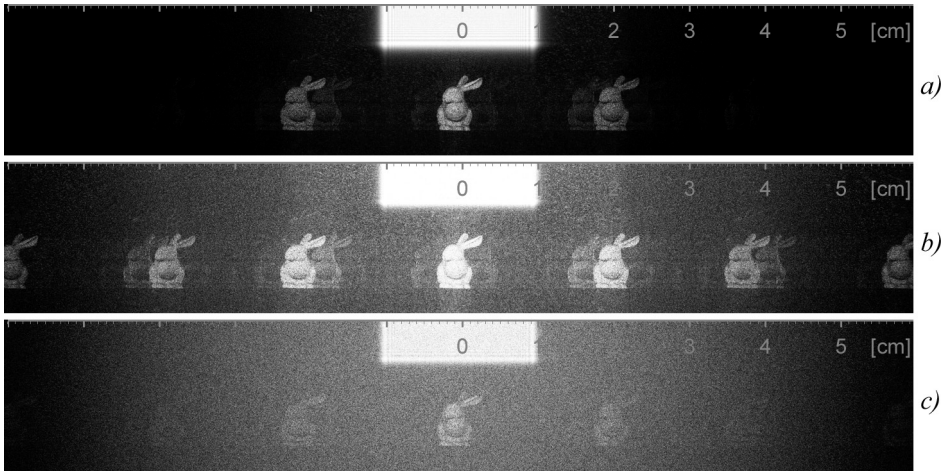
V numerických simulacích jsme uvažovali stejné parametry; výpočty probíhaly referenčním způsobem podle [14] s použitím plného (nezjednodušeného) Rayleigh-Sommerfeldova integrálu I. druhu.

Zabývejme se nejprve zobrazením jednoduché mřížky, viz obr. 3. Druhý bod z výše uvedeného výčtu jevů říká, že binarizaci lze ideálně provést pouze tehdy, je-li perioda mřížky sudým násobkem rozteče mikrozrcátek. V opačném případě je výsledná binarizovaná struktura součtem několika obdélníkových průběhů s různými faktory plnění. To pochopitelně ovlivní difrakční vzor; taková binarizovaná mřížka je velmi špatnou aproximací mřížky se sinovým průběhem; pro srovnání viz obr. 1. Ditheringem se na jednu stranu zmenší kontrast difrakčního vzoru (původně černé pozadí je poškozeno speckle šumem), na druhou stranu mřížka přestane tvořit nežádoucí difrakční řády. Jakkoliv binarizovaná mřížka ovšem nepřestane tvořit difrakční maxima pod úhlem daným roztečí mikrozrcátek, protože ostré změny reflektance na hranicích pixelů jsou neodstranitelné.



Obr. 3: Difrakce na mřížce 12,346 čar/mm (perioda 7,5× rozteč mikrozrcátek) osvětlené paprskem He-Ne laseru (632,8 nm) ve vzdálenosti 1000 mm, výpočetní simulace. a) Transmitance ideální mřížky (předpokládáme rozsah 0 až 1), resp. mřížky s kvantizací na 256 stupňů šedi, a simulace difrakčního vzoru. b) Struktura binarizovaná prahem 0,5 a její difrakční vzor. Nestejně široké čáry struktury jsou důsledkem neceločíselného násobku rozteče mikrozrcátek. c) Struktura binarizovaná prahem 0,5 a ditheringovým šumem amplitudy 0,5 a její difrakční vzor. Speckle šum uprostřed snižuje kontrast obrazu. Silné difrakční řády po stranách odpovídají difrakci na rastru mikrozrcátek.

Jiná situace nastává, pokoušíme-li se zobrazovat komplikovanější difraktivní strukturu, jmenovitě počítačem generovaný Fresnelův hologram 3D objektu. Z obrázku 4 je zřejmé, že prostá binarizace prahováním také vytváří vyšší difrakční řády, prahování s ditheringem ale nyní tvoří tolik šumu, že se značně poškozuje i minus první difrakční řád, reálný obraz tvořený hologramem.



Obr. 4: Rekonstrukce počítačem generovaného amplitudového Fresnelova hologramu 3D objektu ve vzdálenosti 1 m (úhel objektové a referenční vlny 0,75° ve svislé rovině), výpočetní simulace. Světlý obdélník uprostřed každého obrázku je nedifraktované světlo. a) Hologram s kvantizací na 256 úrovní šedi. b) Binarizovaný hologram. c) Hologram binarizovaný s použitím ditheringového šumu.

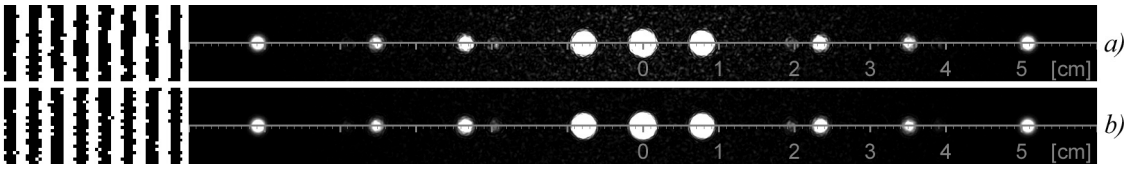
5. Modifikace ditheringového šumu

Ditheringový šum, který jsme k difraktivní struktuře přičítali, jsme tvořili generátorem náhodných čísel s rovnoměrným rozložením pravděpodobnosti. Takový šum má ploché spektrum [13]. Difrakční vzor ve vzdálené oblasti je proto také rovnoměrný a svým svitem všude snižuje kontrast. Nabízí se proto myšlenka, zda by ditheringu neprospělo užívání šumu s jiným než rovnoměrným rozložením pravděpodobnosti. Taková myšlenka není nová;

například při kvantizaci audiosignálu se používá ditheringový šum, jehož frekvenční charakteristika koreluje s frekvenční citlivostí lidského ucha [11].
Pokud bychom ze spektra ditheringového šumu vynechali například nízké frekvence, měl by se šum v difrakci projevovat především pod vyššími difrakčními úhly. Víme-li, pod jakým úhlem se vytváří reálný obraz hologramu, měli bychom být schopni vytvořit takový ditheringový šum, který jej bude poškozovat minimálně.
Nejjednodušší metodou generování šumu $n_{\mathrm{DS}}(\boldsymbol{x})$ s daným spektrem je digitální filtrace rovnoměrného šumu $n_{\mathrm{D}}(\boldsymbol{x})$, resp. jeho konvoluce s jistým konvolučním jádrem K.
V jednorozměrném případu:

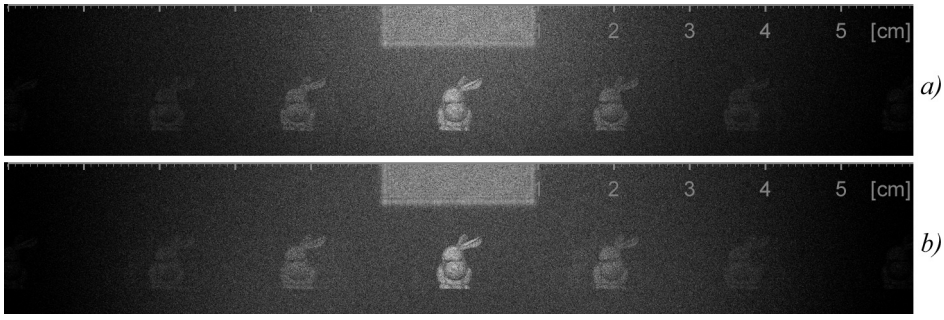
$$n_{\mathrm{DS}}(x) = \sum_j K(j) n_{\mathrm{D}}(x - j)$$

Pro ověření principu jsme zkusili konvoluční jádra $K_{\mathrm{L}} = [1\ 1]$ (low-pass) a $K_{\mathrm{H}} = [1\ -1]$ (high-pass), viz obr. 5. Je zřejmé, že základní idea funguje: v případě nízkofrekvenčního šumu je speckle koncentrován poblíž nedifraktovaného paprsku, v případě vysokofrekvenčního je naopak střed difrakčního vzoru relativně čistý. Na druhou stranu už dithering nestačí odstraňovat všechny nežádoucí difrakční řády.



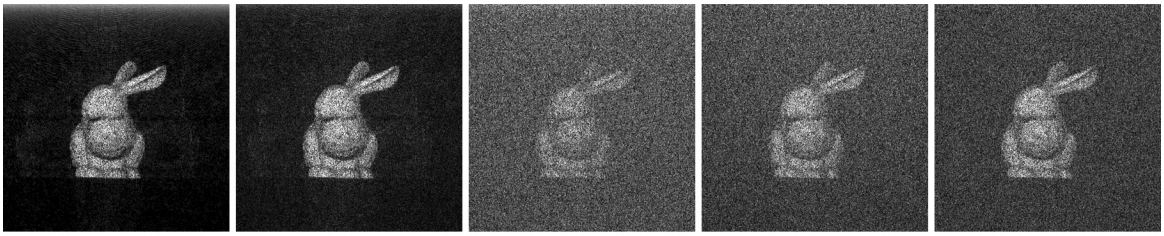
Obr. 5: Difrakce na mřížce 12,346 čar/mm (perioda 7,5× rozteč mikrozrcátek) osvětlené paprskem He-Ne laseru (632,8 nm) ve vzdálenosti 1000 mm, výpočetní simulace. Viz též obr. 3. a) Struktura binarizovaná prahem 0,5 a ditheringovým šumem amplitudy 0,5 filtrovaným jádrem K_L a její difrakční vzor. b) Struktura binarizovaná prahem 0,5 a ditheringovým šumem amplitudy 0,5 filtrovaným jádrem K_H a její difrakční vzor.

Stejné chování můžeme pozorovat i na Fresnelově hologramu, viz obr. 6. I tentokrát se speckle kumuluje v předpokládané oblasti a schopnost rušit nežádoucí difrakční řády klesá.



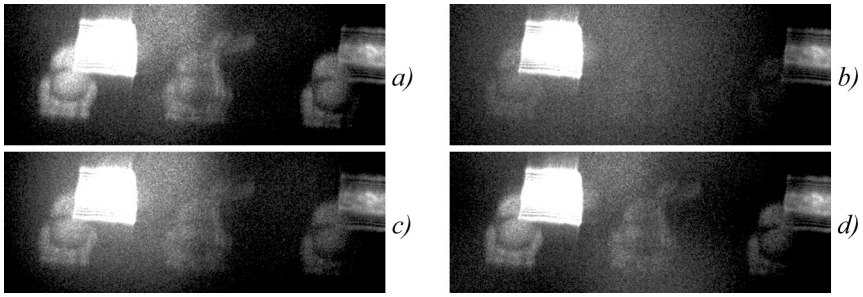
Obr. 6: Rekonstrukce počítačem generovaného amplitudového Fresnelova hologramu 3D objektu ve vzdálenosti 1 m (úhel objektové a referenční vlny 0,75° ve svislé rovině), výpočetní simulace. a) Hologram binarizovaný s použitím nízkofrekvenčního ditheringového šumu. b) Hologram binarizovaný s použitím vysokofrekvenčního ditheringového šumu. Povšimněte si rozdílného rozložení speckle šumu.

Podívejme se však na dosažené výsledky z pohledu kontrastu. Na obrázku 7 jsou jasy jednotlivých difrakčních vzorů (rekonstrukcí hologramu) upraveny tak, aby obraz figurky byl stále stejně jasný. Na první je pohled vidět, že jakékoliv experimenty s ditheringem vedou ke zhoršení odstupu jasu obrazu a jasu pozadí. Dokonce se ukazuje, že původní binarizace pouhým prahováním dává téměř stejné výsledky jako rekonstrukce původního (nebinarizovaného) hologramu. Srovnatelné výsledky je možné pozorovat na fotografiích skutečného experimentu, viz obr. 8.



a) *b)* *c)* *d)* *e)*

Obr. 7: Detail rekonstrukce počítačem generovaného hologramu, výpočetní simulace. a) Původní hologram. b) Prostá binarizace. c) Binarizace s obyčejným ditheringem. d) Binarizace s nízkofrekvenčním ditheringovým šumem. e) Binarizace s vysokofrekvenčním ditheringovým šumem. Poměr jasů obrazu a pozadí je postupně 32,7; 18,9; 2,3; 3,8 a 5,0.



a) *b)* *c)* *d)*

Obr. 8: Detail rekonstrukce počítačem generovaného hologramu, reálný experiment. a) Prostá binarizace hologramu. b) Binarizace s obyčejným ditheringem. c) Binarizace s nízkofrekvenčním ditheringovým šumem. d) Binarizace s vysokofrekvenčním ditheringovým šumem. Povšimněte si rozdílného rozložení speckle šumu.

6. Závěr

Ověřili jsme, že binarizace difraktivní struktury vede k vytváření nežádoucích difrakčních řádů. Rovněž jsme ověřili, že tyto řády lze do jisté míry eliminovat technikou ditheringu. Ukázali jsme, že pro zobrazování počítačem generovaných Fresnelových hologramů 3D objektů je tato technika nevhodná jak v klasické, tak v nově navržené podobě, neboť silně degraduje kontrast pozorovaného obrazu. Jak klasická, tak nově navržená technika ditheringu je nicméně pravděpodobně použitelná u jednoduchých difraktivních struktur, které zaměřují difraktovaný paprsek do jistého směru, například u optických pinzet (optical tweezers). To může být předmětem dalšího výzkumu.

Poděkování

Práce vznikla za podpory projektu LC-06008 (Centra počítačové grafiky) Ministerstva školství, mládeže a tělovýchovy České republiky.

Literatura

[1] Benzie, P.; Watson, J.; Surman, P.; Hopf, K.; Urey, H.; Sainov, V.; von Kopylow, C. A Survey of 3DTV Displays: Techniques and Technologies. In: *IEEE Transactions on Circuits and Systems for Video Technology.* Volume 17, Issue11 (2007), pp. 1647–1658. DOI: 10.1109/TCSVT.2007.905377

[2] Hanák, I.; Janda, M.; Skala, V. Detail-driven digital hologram generation. In: *The Visual Computer.* Vol. 26, no. 2 (2010), pp. 83–96. ISSN 0178-2789.

[3] Matsushima, K.; Nakahara, S. High-definition full-parallax CGHs created by using the polygon-based method and the shifted angular spectrum method. In: *Practical Holography XXIV: Materials and Applications (2010).* DOI: 10.1117/12.844606.

[4] Tsang, P. W. M.; Liu, J. P.; Cheung, K. W. K.; Poon, T.-C. Modern Methods for fast generation of digital holograms. In: *3D Research.* Volume 1, Number 2, 3. DOI: 10.1007/3DRes.02(2010)03.

[5] Tay, S.; Blanche, P.-A.; Voorakaranam, R.; Tunc, A. V.; Lin, W.; Rokutanda, S.; Gu, T.; Flores, D.; Wang, P.; Li, G.; St Hilaire, P.; Thomas, J.; Norwood, R. A.; Yamamoto, M.; Peyghambarian, N. An updatable holographic three-dimensional display. In: *Nature* 451, pp. 694–698 (2008). DOI:10.1038/nature06596.

[6] Yaraş, F.; Hoonjong K.; Onural, L. State of the Art in Holographic Displays: A Survey. In: *Journal of Display Technology.* Volume 6, Issue 10 (2010), pp. 443–454. DOI: 10.1109/JDT.2010.2045734.

[7] Wyrowski, F. Digital holography: diffractive optics on the basis of scalar diffraction theory. In: *Workshop on Digital Holography, Proceedings.* Vol. 1718 (1993). DOI: 10.1117/12.138567.

[8] Kitamura, M. *Computer Hologram and Creation Method Thereof.* US Patent 7573622. August 11, 2009.

[9] Gao, N.; Xie C. High-order diffraction suppression using modulated groove position gratings. In: *Optics Letters.* Doc. ID 152983. (posted 09/27/2011, in press).

[10] Maurer, C.; Schwaighofer, A.; Jesacher, A.; Bernet, S.; Ritsch-Marte, M. Suppression of undesired diffraction orders of binary phase holograms. In: *Applied Optics.* Vol. 47 Issue 22, pp. 3994–3998 (2008). DOI:10.1364/AO.47.003994.

[11] Watkinson, J. *Art of Digital Audio*. Third Edition. Oxford: Focal Press, 2001. ISBN 0240515870.

[12] Malý, P. *Optika*. Vyd. 1. Praha: Karolinum, 2008. ISBN 978-80-246-1342-0.

[13] Smith, S. W. *Digital signal processing: a practical guide for engineers and scientists*. Amsterdam: Newnes, 2003. ISBN 0-7506-7444-X.

[14] Lobaz, P. Reference calculation of light propagation between parallel planes of different sizes and sampling rates. In: *Optics Express.* Vol. 19, no. 1 (2011), pp. 32-39. ISSN 1094-4087.