Vis Comput (2010) 26: 83–96
DOI 10.1007/s00371-009-0378-5

ORIGINAL ARTICLE

Detail-driven digital hologram generation

Ivo Hanák · Martin Janda · Václav Skala

Published online: 15 July 2009

Abstract Digital holography is a technology with a potential to provide realistic 3D images. However, generation of digital holograms is a computationally demanding task. Thus, the performance is a major concern. We propose a new method that reduces spatial resolution in order to accelerate hologram generation. It employs the propagation between parallel planes for efficient optical field values evaluation and a computer graphics approach for approximating visibility. Our results show that the proposed reduction has only a minimal impact on the visual quality, while the formal computational complexity confirms performance improvement.

Keywords Digital holography · Hologram generation

1 Introduction

Holography [8] is a technique that allows one to store information about light propagating from a scene into a hologram. From the hologram it is possible to reconstruct the original light and thus to create an image of the scene. It is a principle similar to photography. Photography, however, only captures the intensity of an image created on a film by a lens. As a result, the image contains only one view direction and one depth of field. A hologram, on the other hand, uses a special optical setup to capture not only an intensity but light propagation directions as well. Therefore the reconstructed image provides a range of view directions and depths of field to a viewer. As a consequence, the image contains all depth cues available in the real world.

The features stated above render holography into a technology interesting for three-dimensional (3D) display. However, for eligible application, the holography needs to be digitized first. The digitization of holography brings new problems that are dealt with by the digital holography discipline. One particular problem is the generation of digital holograms of virtual scenes. In this paper we propose a solution of this problem.

In holography, light propagating from a scene is considered as an optical field. A monochromatic optical field in vacuum is at each point of space determined by an amplitude and a phase. When a hologram is taken, each point of its surface records information about the amplitude and phase of the optical field at that location. In digital holography, a hologram is usually discretized into a planar regular grid of points. The first step of the digital hologram generation is calculation of optical field values at the discrete hologram points. The second step is encoding the values into the digital hologram. This step is trivial compared to the first step and out of the scope of this paper. More information about the second step can be found in [9].

One of the reasons that make the digital holography so problematic is the large number of points a discrete hologram is composed of. To properly sample the high frequencies contained in an optical field, the pitch between points needs to be in an order of micrometers or less. Throughout this paper we use a quite large pitch 7 μm because it is supported by a device that allows printing a calculated hologram

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) and it has been supported by the EU project within FP6 under Grant 511568 with the acronym 3DTV. The access to the METACentrum computing facilities provided under the research intent MSM6383917201 is acknowledged.

I. Hanák (✉) · M. Janda · V. Skala
Department of Computer Science and Engineering, University of West Bohemia, Univerzitní 22, Plzeň, Czech Republic
e-mail: hanak@kiv.zcu.cz
url: http://holo.zcu.cz

and that we had a limited access to. If we apply this pitch to a 17-inch "holographic" LCD, we would end up calculating almost 2×10^{10} hologram points. This is almost 1,000× more points than current 17-inch LCDs have.

Another complication is that an optical field contains disturbances that have a complicated structure. Fortunately, the corresponding mathematical model [9] allows us to express one complicated disturbance as a sum of simpler ones, each of them described using an analytic function. One such simple disturbance is generated by a point light source (PLS). As a result, a cloud of PLS can be substituted for a virtual scene geometry, and the resulting optical field is a sum of individual disturbances generated by PLS.

The disadvantage of calculating optical field values from a cloud of PLS is the need to use a large number of sources in order to represent the original continuous surface accurately. However, if there are planar surfaces in the scene, then values of their optical fields at hologram points can be calculated all at once using methods exploiting the Fourier transformation, i.e., propagation in angular spectrum or Fresnel's approximation [9]. Such methods are convenient because in the discrete case the application of the fast Fourier transform (FFT) algorithm greatly accelerates the generation process. The application of the FFT requires PLS to be organized in a regular planar grid.

The digital hologram generation methods based on the FFT perform best when objects in a scene consist of planar surfaces as large as possible. This requirement is contradicted by the shape richness of objects usual in realistic scenes. Further, the requirement is especially inconvenient for solving visibility, i.e., checking direct visibility between each PLS and each point of a hologram. In general, solving visibility for the whole grid of PLS is more complex [21, 22] than solving visibility for only one PLS [29].

In this paper we propose a method for digital hologram generation that combines the simplicity of the PLS visibility solution and the speed of the FFT based approach for evaluating optical field values from a regular planar grid of PLS. The method also exploits the limited resolution of a human visual system [17], and at several stages of the generation process the solution precision is intentionally reduced. We show that the proposed method provides a faster solution compared to the methods working with a cloud of PLS, and simultaneously the method avoids visibility solution pitfalls involved in methods based on the FFT. This combination of approaches outperforms both the PLS and FFT-based methods, and this is the main contribution of the paper.

The main purpose of this paper is to present a new approach for fast generation of digital holograms. Therefore, the paper contains only a brief introduction into the computer-generated holography problematic. We designed the paper to focus on the algorithm rather than the physical model of both light behavior and interaction with a surface. Even though underlying physics is presented in one of sections, it is not essential for the description of the proposed method.

Section 2 gives a brief overview of holography mathematics used by the proposed method. In Sect. 3 we provide an overview of relevant previous work. The detailed description of the principle and practical implementation are elaborated in Sects. 4 and 5, respectively. A computational complexity of the method including a formal comparison to a method working with a cloud of PLS and a method working with FFT is presented in Sect. 6. The paper is concluded by presenting reconstruction of results in Sect. 7 and giving conclusions in Sect. 8.

2 Brief introduction to holography

In this section we present a brief introduction to a part of the holography mathematics that is exploited by our approach. The information provided constitute the most basic principles used in computer-generated holography. Readers experienced in this field may skip this section, and for others, we recommend [9, 10] for more thorough reading.

Let us first introduce a physical background of a point light source. In our case we use a monochromatic PLS emitting a spherical wave that is superimposed on the optical field [9], and we examine PLS at the plane $\kappa : z = 0$. With en exception of a close neighborhood of PLS, the emitted wave that is added to the optical field at the plane $\kappa : z = 0$ is

$$u(x, y) = \frac{A}{r} \exp\bigl[i(kr + \varphi)\bigr], \quad k = \frac{2\pi}{\lambda}, \tag{1}$$

where r is the distance between a point at $(x, y, 0)$ and PLS, $A = I^{1/2}$, I is an intensity of PLS detected at a surface of a unit sphere centered around PLS, φ is the phase of the wave emitted from PLS, λ is a wavelength of the radiation emitted from PLS, and i is the imaginary unit.

An optical field generated by a cloud of PLS can be approximated as a superposition of optical fields. Every of those fields is computed according to (1). Let us illustrate it with a cloud of two PLS. We are able to express an optical field value $u_z(x, y)$ generated on the plane $\kappa : z = 0$ by PLS located at $(0, 0, z)$, where $z \in \mathbb{R}$. When the first PLS is located at (α, β, γ) and the second PLS is located at (a, b, c), the resulting optical field value at the plane κ is $u(x, y) = u_\gamma(x - \alpha, y - \beta) + u_c(x - a, y - b)$.

Let us now consider a special case that is significant for the presented method. Let all calculated samples u_{mn} be located at points $\mathbf{u}_{mn}$ of a regular grid on the plane $\kappa : z = 0$, and let all PLS in the cloud be located at points $\mathbf{v}_{mn}$ of a rectangular grid on the plane $\rho_\zeta : z = \zeta$. Let us assume that there are $M \times N$ points, where both M and

N are even integers, and thus $m \in [-\frac{1}{2}M, \frac{1}{2}M - 1]$ and $n \in [-\frac{1}{2}N, \frac{1}{2}N - 1]$. Even though this case can be solved using the superposition mentioned above, there is a more efficient method. If the grid points on the plane ρ overlap with the grid points on the plane κ when orthogonally projected, it is possible to use a method known as propagation of the angular spectrum [9]. The description follows.

Let V be the values of a discrete optical field at points (PLS) $\mathbf{v}_{mn}$, and let U be the unknown values of the same optical field at samples u_{mn}. First, we express the values V in the frequency domain using the Fourier transform, i.e., $\mathcal{V} = \text{FFT}\{V\}$. A single frequency of the spectrum can be interpreted as a planar wave that deviates from the normal of the plane ρ_ζ by a given angle. The frequency spectrum of the optical field values is referred to as the angular spectrum. Second, we apply a propagation operator $\mathcal{H}(\zeta)$ such that $\mathcal{U} = \mathcal{V} \star \mathcal{H}(\zeta)$, where $\star$ denotes element-wise multiplication, the propagation operator is a matrix $\mathcal{H}(\zeta) = [h_{mn}]$, and

$$h_{mn} = \exp\left\{-i2\pi\zeta\left[\frac{1}{\lambda^2} - \left(\frac{m}{X}\right)^2 - \left(\frac{n}{Y}\right)^2\right]^{\frac{1}{2}}\right\}, \qquad (2)$$

where $X = 2D_x M$ and $Y = 2D_y N$. We obtain the optical field values at points $\mathbf{u}_{mn}$ as $U = \text{FFT}^{-1}\{\mathcal{U}\}$. This speeds up significantly the calculation of the optical field in a discrete environment because it uses FFT. However, it is valid only for the two-plane case that is described above.

3 Previous work

The area of digital hologram generation is essential to digital holography, and therefore there is a wide range of solutions in the literature. For the purposes of this paper, only solutions working with a cloud of PLS and solutions exploiting FFT are relevant.

The principle of calculating optical field values from a cloud of PLS is simple. The optical field of one PLS is described analytically, and the final optical field is obtained as superposition [9] of the individual PLS optical fields, see Sect. 2. The very problem of the principle is a high number of sources required for representing the originally continuous surface. Various methods based on this principle therefore focus on acceleration that is usually linear.

The most frequently used acceleration approach is exploitation of the Fresnel approximation [11, 24, 27, 31], which employs the Taylor series for simplifying the calculation of the optical field values of a PLS. The Fresnel approximation is applicable only for smaller scenes rather distant from the hologram plane [9]. There is also a possibility to approximate the optical field function by a piecewise linear function [13] or by an iterative scheme [11, 23]. Another option is moving the calculation to the distributed environment [12, 24] and to the graphics processing unit (GPU) [1, 12, 14, 19, 25, 27]. The GPU-aided calculation provides results in a real time for small holograms (1,024 × 1,024) and a small number of PLS (thousands). Similar performance could be also achieved by employing a specialized hardware (HW) [11].

More significant acceleration could be achieved only by reducing the information stored in a hologram, e.g., by sacrificing the vertical parallax [17, 29]. By doing so, the computational complexity of the generation process is reduced by one order of magnitude, and when special output device [18] is employed, it is possible to achieve interactive generation and displaying. The obvious disadvantage of this approach is the vertically fixed viewing angle.

The listed methods use geometric optics for evaluating visibility [12, 29, 32]. If a scene is sufficiently small, it is possible to evaluate the visibility only once from some fixed viewpoint [13] without degrading the quality of the reconstructed image significantly. If the scene is larger, it is necessary to solve the visibility from more than one viewpoint [12, 29, 32].

In contrast to the methods that operate with a cloud of PLS, the solutions exploiting FFT allow calculating optical field values on a hologram plane from PLS on a source plane with the computational complexity of the FFT. The basic FFT applications assume that both the hologram plane and the source plane are parallel [9]. More advanced applications allow mutual tilting [5, 20–22, 28], which, however, introduces information loss due to angular spectrum deformation. The loss can be reduced by using a analytic expression for an angular spectrum of a triangle [2, 15]. This, however, leads to a loss of diffusivity of the surface and thus complicates observation of the hologram by a human viewer [16]. The serious problem of the FFT-based methods remains the visibility solution [21] that is sometimes almost ignored [2]. While the optical field values resulting from a planar shape are calculated in the frequency domain, the visibility of the same shape needs to be evaluated in the spatial domain. This leads to a multiple application of FFT and incorporation of the painter's algorithm [30].

To avoid any confusion, we would like to explicitly state how the new proposed method is related to our previous method [12]. While both methods follow the same goal and physical principle, they differ in the approach to a surface division and the approach to light propagation. In our previous method, the surface division is created during execution of the algorithm and directly relies on a pure ray-casting as well as the visibility does. The method is easily distributed and allows calculating nonplanar holograms. The method resembles a PLS-based renderer that needs parallel environment to run efficiently. On the other hand, the newly proposed method incorporates FFT-based approach, and it allows us to choose the level of detail and thus is faster on a single machine. We consider the possibility to choose the level of detail as the most important feature of this new method.

4 Principle description

In this section we present the basic principles of our method. We define the digital hologram generation task. Then the description of the significant stages of the method follows. For the sake of simplicity, we restrict the description to bare principles. The details of execution are provided in the next section.

The goal of the digital hologram generation task is to calculate values of the optical field generated by a virtual scene at all digital hologram points. The virtual scene consists of nonintersecting objects that are defined by their surface. The surfaces are represented as a triangular mesh. For our purposes, a triangular mesh consists of a set of vertices and a set of triangles. Each triangle is defined by three vertices and is oriented, hence the surface normal can be uniquely determined at every point of the mesh. This representation is the most usual one in computer graphics, and we comply to this standard.

The first step of our method is to express the complicated optical field generated by a triangular mesh as a sum of simpler ones, i.e., to decompose the mesh into a set of simple building elements. Some methods choose the PLS to be the element because the optical field of PLS is described by a simple function (see Sect. 2), and the visibility of PLS is easily solved by applying ray-casting: a ray is cast from each hologram point towards each PLS, and if the ray intersects the original mesh, the corresponding hologram point is kept unaffected. Notice the difference when compared to the process of classical 2D image generation, where one point of a surface (PLS) is projected into one point of the generated image, i.e., only one visibility test is required. When holography is concerned, one PLS affects all hologram points, which inherently increases the processing complexity by two orders of magnitude.

We also follow the decomposition principle, but instead of PLS we choose a planar patch as the building element. This results in less building elements (one patch replaces many PLS) and inherently less visibility tests providing each patch is still small enough to be considered as a single PLS when solving its visibility. We further reduce the number of visibility tests by grouping the hologram points into cells of a coarser grid that we refer as a visibility grid. All points in one cell of the visibility grid share the result of a visibility test after the test is evaluated. Nevertheless, the disadvantage of the planar patch is a complex calculation of its optical field. We overcome this problem by discretizing each patch into a regular planar grid of points and calculating the optical field values using the fast approach of FFT-based methods.

Let us elaborate on the principle described above, but first we formalize the parameters of the digital hologram generation method. We assume the left-handed coordinate system where Y-axis points up, X-axis points right, and Z-axis points from a digital hologram towards the scene. Note that whenever a viewer-dependent orientation term is used, the viewer is assumed to be located at $(0, 0, -\infty)$ and is looking in the Z-axis direction.

The digital hologram is a regular rectangular planar grid of points at the plane $\kappa : z = 0$. The resolution of the grid is $M \times N$, where both M and N are even integers. The location $\mathbf{u}_{mn}$ of one point is $(mD_x, nD_y, 0)$, where D_x is a pitch between the points in the X direction, D_y is a pitch in the Y direction, and $m \in [-\frac{1}{2}M, \frac{1}{2}M - 1]$, $n \in [-\frac{1}{2}N, \frac{1}{2}N - 1]$, $m, n \in \mathbb{Z}$.

The visibility grid is a structure constructed of hologram points. Each cell of the visibility grid consists of $E \times E$ hologram points. In the rest of the paper we will denote the visibility grid as G_κ and the grid cells as g_{lo} where the variable subscripts l and o are the column index and the row index to G_κ, respectively. The center $\mathbf{g}_{lo}$ of the cell g_{lo} is located at the plane κ, i.e., $\mathbf{g}_{lo} = (lED_x, oED_y, 0)$. The notation is illustrated in Fig. 1.

As we already mentioned, in the initial stage of the method we approximate objects using a cloud of simpler building elements, i.e., planar patches in our case. Such an approximation can be done in various ways; we, however, have decided to perform it as follows. All patches are oriented to be parallel to the plane κ. This simplifies the calculation of optical field values that are generated by the patch. Besides that, all planar patches used as building elements have the same size: $ED_x \times ED_y$. It is not a coincidence that this size matches the size of the visibility grid cell because the size $ED_x \times ED_y$ constitutes the smallest spatial detail and it is a parameter for our method. The fact that the size of a patch matches the size of the smallest detail allows us to treat each patch as PLS when solving visibility and illumination (the illumination is discussed in Sect. 5).

The relation between patches and the visibility grid is tightened even more by restricting the position of each patch

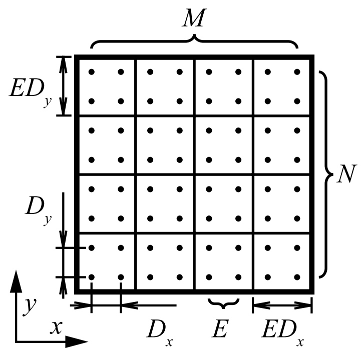

Fig. 1 Organization and notation of the hologram points as used in the paper, i.e., M denotes the number of columns of points, N denotes the number of rows of points, E denotes the number of rows and columns of points in one visibility grid cell, D_x denotes the pitch between columns of points, D_y denotes the pitch between rows of points, and ED_x and ED_x are the dimensions of one visibility grid cell

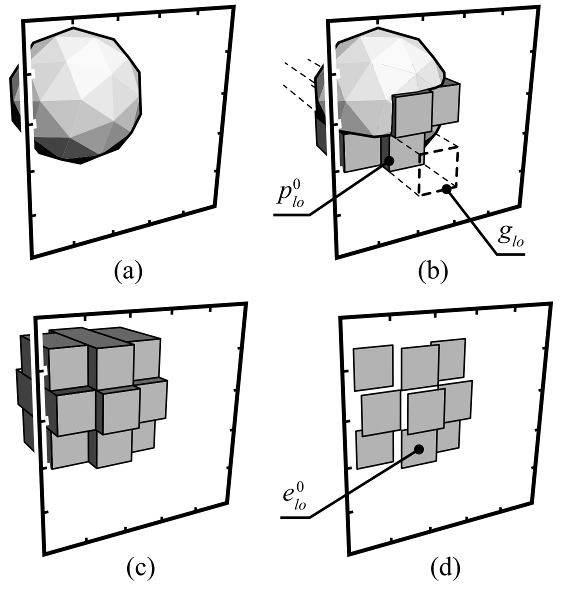

Fig. 2 Process of approximating objects by a cloud of patches. (**a**) A scene with a single convex object that is to be approximated. (**b**) A pillar p^0_{lo} is a cuboid segment of the cell g_{lo} extruded in the direction of the Z-axis. (**c**) The object is completely approximated by pillars and (**d**) for each pillar a patch is placed in the location of front-facing cap

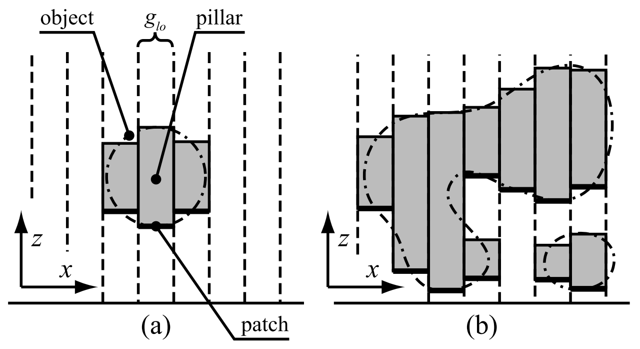

Fig. 3 (**a**) A top-down view of the situation from Fig. 2 for a fixed visibility grid index o showing a single pillar per cell, and (**b**) more complicated scene showing multiple pillars per cell. Dash-dotted line corresponds with the cross-section of the scene geometry. Notice that a center of a cell g_{lo} along the X-axis is located at $x = (l + \frac{1}{2})ED_x$

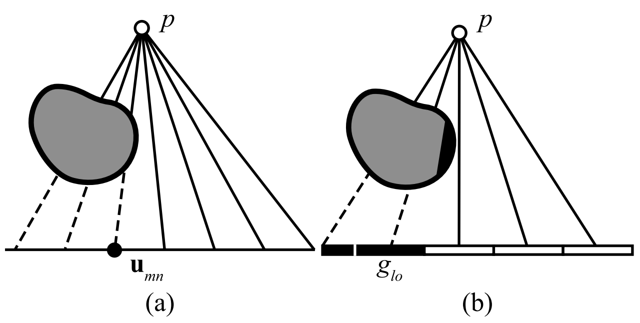

Fig. 4 (**a**) Ray-casting based visibility solution for a single PLS p, and (**b**) our approximation using the visibility grid. (**a**) While PLS p is not visible from the point $\mathbf{u}_{mn}$ using ray casting based solution, (**b**) our approximation leads to a different result for the point $\mathbf{u}_{mn}$ because the point belongs to the cell g_{lo}. This error can be interpreted as a deformation of an obstacle, i.e., a black filled portion of the obstacle is not considered. Notice that the point u_{mn} along the X-axis is located at $x = mD_x$ and that the center of a cell g_{lo} along the X-axis is located at $x = (l + \frac{1}{2})ED_x$

in X and Y directions so that each patch matches one of the visibility grid cells when projected to κ. Since more than one patch may match with the same cell, we denote the dth patch matching with the cell g_{lo} as the patch e^d_{lo}. The tight relation of the patches and the visibility grid allows us to solve the visibility more easily as we show later in this section.

For the purposes of the visibility solution and the means of creating the patches, we establish the concept of a pillar. A pillar is a cuboid segment of a cell g_{lo} extruded in the direction of the Z-axis. The pillar sides parallel with κ are the important ones. Providing all pillars are located completely behind κ with respect to the viewer, we denote the cap closer to κ as a front-facing cap and the cap farther from κ as a back-facing cap. The positions of the caps along the Z-axis are determined from the intersection of the extruded cell and the scene geometry. The process of a pillar creation is illustrated in Fig. 2(a–c). The exact calculation of the caps positions is not vital for the method principle, so we discuss it later in Sect. 5.

The number of pillars generated per cell depends on the geometry as depicted in Fig. 3, and therefore we denote the dth pillar corresponding to the cell g_{lo} as p^d_{lo}, see Fig. 2(b). The pillars and patches are directly related. The front-facing cap position of each pillar p^d_{lo} specifies the location of a patch e^d_{lo} as depicted in Fig. 2(d). The geometry approximation error we cause by using the patches depends on the obliqueness of the mesh faces with respect to the plane κ. As a consequence, the faces perpendicular to the plane κ are effectively neglected. Since this issue can be handled without altering the very principle of our method, we have decided not to discuss it in this paper for the sake of presentation clarity.

The close relation between the visibility grid, patches, and pillars proves advantageous when the visibility is solved. As noted at the beginning of this section, our visibility solution is an analogy to the solution done in the cloud of PLS operating methods [29], where from each PLS a ray is casted towards each hologram point, and if the ray intersects the scene geometry, the corresponding hologram point is kept unaffected as illustrated in Fig. 4(a).

The difference in our solution is that we have planar patches instead of PLS. Since we defined the size of the patches to be the smallest spatial detail of the scene, we can process them as if they were PLS, i.e., when the visibility is solved, a patch is either completely visible or completely invisible. As a result, we cast all rays only from the patch center.

The second difference is a reduction of the number of the cast rays. The reduction is achieved by testing only the centers of the visibility grid cells. All hologram points then adopt the result of all visibility tests from the corresponding cell. This approximation can be interpreted as an obstacle deformation observed from a hologram point of one partially occluded cell, see Fig. 4(b). The maximal amount of the deformation cannot exceed $\frac{1}{2}E \max\{D_x, D_y\}$, which is less then the chosen minimal detail size. The error caused by the approximation is therefore acceptable.

The third difference is the exploitation of the pillars. Instead of finding intersections between the rays and the original scene geometry, we are finding intersections between the rays and the pillars. This is advantageous due to the regular organization of the pillars that can be used for accelerating the intersections evaluation. The description of this acceleration is included in Sect. 5.

The whole process that solves visibility of one particular dth patch e^d_{qt} on coordinates (q, t) can be summarized as follows. First, we test the patch visibility from all cells of the grid G_κ. The test iterates through all cells, and for each cell g_{lo}, it examines all pillars whether they obstruct the patch e^d_{qt}. If the patch is obstructed, the next cell is examined. Otherwise, we compute optical field values U^d_{qt} of the patch e^d_{qt} at the hologram points, and the values corresponding to the cell g_{lo} are added to the resulting optical field values U. Then, the algorithm continues with examining the next cell. Note that the optical field values are computed only once for each patch at the moment of the first successful visibility test and are reused for other successful visibility tests. The complete algorithm is listed in Algorithm 1. The first step of the algorithm dealing with the objects decomposition is described in Sect. 5.

The computation of patch optical field values at hologram points corresponds with a propagation of an optical field between two parallel planes in a free space. We know the optical field values at the patch e^d_{qt} discrete points (it is an input, see Sect. 5), and we are interested in optical field values at hologram points on the plane κ. Both e^d_{qt} and κ are planar and parallel, and the distance between them constitutes the distance of propagation.

One way to handle such propagation is via propagation of the angular spectrum that is described in Sect. 2. The optical field at the source plane is transformed into frequency domain using the Fourier transform. The resulting frequencies constitute the angular spectrum of the optical field. Propagation of the angular spectrum is done by one elementwise matrix multiplication, and finally the desired optical field values are obtained through an inverse Fourier transform. The reason for using this approach is the fact that in a discrete environment the Fourier transform is done through FFT and this makes the whole approach very fast.

Algorithm 1 The core algorithm of the method

```
 1: Decompose a scene into pillars                         ▷ See Algorithm 2
 2: Let the final optical field $U$ be zero
 3: for all cells $g_{qt} \in G_\kappa$ do
 4:     for all $d$ such that the pillar $p^d_{qt}$ exists do
 5:         Create a patch $e^d_{qt}$ in the center of the front-facing cap of the pillar $p^d_{qt}$
 6:         for all cells $g_{vw} \in G_\kappa$ do
 7:             if the center of the front-facing cap $p^d_{qt}$ sees the center $\mathbf{g}_{vw}$ of the cell $g_{vw}$ then
 8:                 if optical field $U^d_{qt}$ is not calculated then
 9:                     Calculate optical field $U^d_{qt}$ from the patch $e^d_{qt}$
10:                 end if
11:                 Add a part of $U^d_{qt}$ corresponding to the cell $g_{vw}$ to $U$
12:             end if
13:         end for
14:     end for
15: end for
```

In the previous paragraphs we described the principle of our method. What is left is to describe the details of individual stages execution. The stages are objects decomposition into pillars, calculation of optical field values from a patch, and the specific approach to visibility solution. The details are presented in the following section.

5 Execution details

In the previous section we presented a principle of our method consisting of three fundamental stages: decomposition of objects into a set of pillars and consequently into a cloud of patches, calculation of optical field values due to patches, and, finally, the visibility solution. In this section we describe the important execution details of these stages.

The first stage of our method is the decomposition of the scene objects into a set of pillars that further determines the locations of patches. We make this decomposition via a ray-casting [30]. From the center $\mathbf{g}_{lo}$ of each cell g_{lo} we cast a ray into the scene. All rays are parallel to the Z-axis, and each intersection of the ray and the scene determines a location of one pillar cap. We denote each cap either as front-facing and back-facing by using a normal $\mathbf{n} = (x_\mathbf{n}, y_\mathbf{n}, z_\mathbf{n})$ of the surface at the intersection. Since the normal $\mathbf{n}$ points outwards and the Z-axis points from the hologram to the scene, we base our decision on the sign of $z_\mathbf{n}$. If $z_\mathbf{n} > 0$, the corresponding cap is back-facing, otherwise the cap is front-facing.

Next, we build a set S_{lo} that consists of all caps based on intersections of the given ray and the scene. Since each cap has a known location, we sort the set S_{lo} ascendantly by the Z coordinate of the caps location. Usually, each front-facing cap in the sorted set S_{lo} is followed by a back-facing cap, and these two caps define a single pillar. In a general case, a singularity in the sequence may occur as a result of a special triangular mesh and a ray configuration. The least bothersome singularities occur in scenes that contain intersecting triangles. We consider such scenes as invalid since such cases can be avoided easily. We consider and discuss further only the singularities caused by a ray hitting an edge or a vertex of a triangle and by having an unclosed mesh in a scene.

The addressed singularity occurs when the scene contains an unclosed mesh or when a ray intersects an edge of a triangle. In this case a front-facing cap is generated without a corresponding back-facing one or vice-versa. In order to rectify the singular pillars, we add a missing cap. The missing cap is shifted from the existing one along the Z-axis. We choose the shift distance to be equal to the chosen detail size mentioned in Sect. 4. If the shift causes the rectified pillar to intersect with another pillar, we merge both pillars together. By repeating the process for all cells g_{lo}, we generate all the pillars that are further needed for solving the visibility and creating the patches. The algorithm is listed in Algorithm 2, and it corresponds to the step 1 in Algorithm 1.

In the next stage, our method generates patches and calculates their optical field values. The center of each front-facing cap is located at $\mathbf{x}_{lo}^d = (x_{lo}^d, y_{lo}^d, z_{lo}^d)$, and we center the patch e_{lo}^d at $\mathbf{x}_{lo}^d$. Now, we need to choose the optical field on the surface of each patch. In the real world, the optical field would be determined by the light interaction with the scene content. However, the interaction is very complex, and thus we do not try to simulate it. Instead of that we create the optical field for each patch locally without considering the global interaction. Also, we do not distinguish between light reflected and emitted. So, we consider all patches as light sources with the given optical field characteristics.

Algorithm 2 A decomposition of a scene into pillars

```
1:  Let $\mathbf{S}$ be the scene
2:  for all cells $g_{qt} \in G_\kappa$ do
3:      Let $\mathbf{R}$ be a ray that is parallel with the Z-axis
4:      Send $\mathbf{R}$ from the center of the cell $g_{qt}$ towards the scene
5:      Find intersections $S_{qt} = \mathbf{R} \cap \mathbf{S}$, each intersection is a cap
6:      Mark the caps in $S_{qt}$ either as front-facing or back-facing
7:      Sort caps in $S_{qt}$ in ascending order of their Z coordinate
8:      if $S_{qt}$ contains singularities then
9:          Adds caps to $S_{qt}$ such that every front-facing cap is followed by a back-facing one
10:     end if
11:     for all front-facing caps in $S_{qt}$ do
12:         Build a pillar $p_{qt}^d$ using the front-facing cap and the following back-facing cap
13:     end for
14: end for
```

Since we operate in a discrete environment, the optical field is a grid of complex values where each value is defined by its amplitude and phase. The phase distribution over the optical field of a patch has a direct impact on the directional distribution of light propagating from the patch. Although the relation can be used for creating more complex reflective/emissive surface behavior, we decided to simulate only diffuse surfaces, i.e., uniform radiation/reflection into all directions. This behavior can be achieved by setting the phase of each optical field value of a patch randomly. The amplitude, on the other hand, determines only an amount of emitted/reflected light. Since a patch is too small for an amplitude variation to have any noticeable effect, we keep the amplitude constant for all optical field values of the given patch. The amplitude is quantified as a square root of intensity [9], and therefore we exploit the computer graphics illumination models for calculating the amplitude from the intensity reflected/emitted from the original surface at a location $\mathbf{x}_{lo}^d$ of the patch. The used illumination model is briefly described in Appendix A.1.

Since we now know the optical field values at all patches, we can proceed with the calculation of the optical field values at the hologram points. We defined the patch e_{lo}^d as a square section of a plane parallel to the plane κ. This allows us to calculate values of the optical field U_{lo}^d through the angular spectrum propagation that was introduced in Sect. 4. The distance of propagation is z_{lo}^d. We decided to use propagation of the angular spectrum because it allows solving the propagation efficiently. The efficiency is actually a consequence of a discrete environment in which we can employ FFT instead of the Fourier transform yielding a computational complexity $O(N^2 \log N)$ instead of $O(N^4)$, which is the computational complexity of the Fourier transform.

Alternatively, our method allows using the Fresnel approximation for calculating the optical field values generated by a patch at the hologram points. Unlike propagation of the angular spectrum, the Fresnel approximation requires only a single FFT for operation as shown in Appendix A.2. However, the Fresnel approximation can only be applied if the patches are sufficiently distant [9] from the hologram plane. In our case, this can be easily managed by placing the whole scene sufficiently far away. This means that we are able to calculate patches with completely random phase distribution for a cost of a single FFT. This is in contrast to the methods described in [21, 22], where optical field propagation is solved also between scene elements. In such a case the sufficient distance cannot be usually maintained, and the Fresnel approximation cannot be used, i.e., every scene element requires at least two executions of FFT.

The last stage of our method solves the visibility. As we already noted before, one PLS or one patch in our case influences the optical field values at all hologram points. That is true providing no obstacle is placed between PLS and the hologram plane. If there is such an obstacle, only those hologram points that are located outside of the shadow casted by the obstacle are influenced. Note that even though this omits diffraction on the obstacle, it produces acceptable results [29]. In our method we have to test the mutual visibility of each patch against each visibility grid cell. We do it by casting a ray from a patch to a cell and test the ray against all pillars for an intersection.

For a better performance, we can exploit the regular organization of the pillars and their close relation to the visibility grid G_κ as we described in Sect. 4. As a consequence, we are able to exclude from the visibility test those pillars that can never be intersected by a ray connecting a cell and a currently tested patch. The exclusion is based on observation that the cell to which the currently tested patch belongs and the cell whose visibility we are trying to determine constitute two endpoints of a line on 2D raster, i.e., the visibility grid. Only the pillars that belong to the grid cells intersected by that line can be intersected by the corresponding visibility test ray. Finding the intersected cells is a problem closely resembling the problem of a line rasterization that is a well-known problem solved by the computer graphics. One of existing algorithms known as DDA [7] is used with a slight modification to calculate a depth along the ray.

In order to explain the process, let g_{vw} and e^d_{st} be the cell and the patch, respectively, that we want to test. Using the DDA algorithm, we find all possible index pairs (a, b) that identify the intersected cells as illustrated in Fig. 5. Since each pillar corresponds to a cell, the DDA algorithm identifies potentially intersected pillars as well. During examination of an intersected cell g_{ab}, we test all pillars p^d_{ab}. The test for intersection of the ray and a pillar is done by comparing the depth intervals corresponding to all pillars p^d_{ab} with

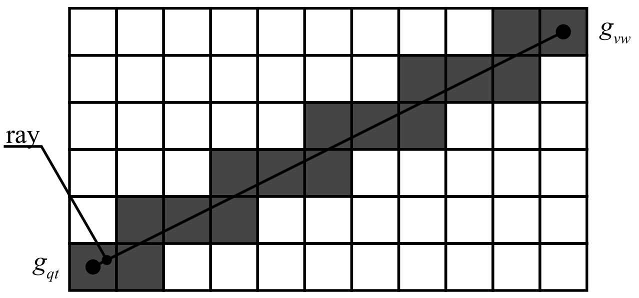

Fig. 5 Traversing cells in the visibility grid by the 2D-DDA algorithm. All pillars belonging to the grayed cells has to be tested

a corresponding depth interval of the ray. The corresponding depth interval of the ray is the interval between the point where the ray enters to the cell g_{ab} and the point where the ray leaves the cell g_{ab}. If the depth intervals are not mutually disjunctive, the cell g_{vw} and the patch corresponding to the front facing cap of the pillar p^d_{qt} are mutually invisible.

After solving the visibility the method proceeds as described in Sect. 4. The visibility solution is our method's last stage that we described in detail. In the following section we estimate a computational complexity and compare it with a computational complexity of PLS-based and FFT-based methods.

6 Computational complexity

We presented a method for digital hologram generation. The method intentionally reduces a level of detail to decrease the computational time. The digital hologram generation field lacks a methodology for comparing the output and the performance in general. Hence, we decided to compare the computational complexity. In this section we present an estimation of a computational complexity of the method and its comparison to different principles used by other methods.

Let us express the computational complexity formally. For simplicity, let us assume that the optical field values are calculated at $N \times N$ hologram points. The number of visibility grid cells along a single axis is $C = \lfloor \frac{N}{E} \rfloor$, see Fig. 1. For a single patch, the estimated number of operations is a sum of a number of pillars that are examined during the visibility test and the number of steps required for FFT. The computational complexity of 2D FFT is $O(N^2 \log N)$. The number of pillars examined during DDA traversal can be expressed as follows. If the cell g_{00} is the starting one and the cell g_{vw} is the ending one, the number of examined cells is $v + w$ at maximum. An exception is the case where $v = w$. In such a case we assume that the ray traverses from one cell to the next one through the common corner and that the number of examined cells is v. We can, therefore, use an arithmetic series to express the total number of examined cells when the visibility of the patch e^d_{00} corresponding with the cell g_{00} is solved. Computing the sum of the arithmetic series and assuming that each cell contains K pillars in average, the total computational complexity of processing one patch is

$$O\left[N^2 \log_2 N + K\left(C^3 - \frac{3C^2 - C}{2}\right)\right]. \tag{3}$$

The expression in (3) is valid for other cells as well, so we can now express the total complexity for the whole hologram. As $K \ll C$, we consider $KC^n \approx C^n$, $n \in \mathbb{N}$, in (3). Since $C = \lfloor \frac{N}{E} \rfloor$, (3) becomes

$$O\left(N^2 \log_2 N + \frac{N^3}{E^3} - \frac{3}{2}\frac{N^2}{E^2} + \frac{1}{2}\frac{N}{E}\right)$$

$$\approx O\left(N^2 \log_2 N + \frac{N^3}{E^3}\right). \tag{4}$$

There are $KC^2 \approx C^2$ patches that have to be processed. Therefore, for a larger N, the computational complexity of the complete generation process is approximately

$$O\left[N^4\left(\frac{\log_2 N}{E^2} + \frac{N}{E^5}\right)\right]. \tag{5}$$

Let us now compare our method with different methods based on different principles.

Methods which operate with a cloud of PLS yield a computational complexity of $O(PN^2)$, where P denotes the total number of PLS. This complexity does not include visibility solution. If the cloud of PLS represents a solid surface, $P \approx N^2$ for common scenes, and therefore the total complexity becomes $O(N^4)$. The expression in (5) shows that our method has a lower complexity if $(\frac{1}{E^2}\log_2 N + \frac{N}{E^5}) < 1$ and, for larger N, if $N < E^5$.

Methods using FFT yield a computational complexity of approximately $O(TN^2 \log_2 N)$, where T denotes a total number of planar surfaces, i.e., triangles, in a scene. This complexity includes visibility solution. According to the expression in (5), our method has lower complexity if $T \geq \frac{N^2}{E^5}(E^3 + \frac{N}{\log_2 N})$. Verification of assumed relations is presented in Sect. 7.

7 Results

We implemented our method and tested it on various scenes. Optical fields calculated by our method were evaluated through numerical simulations and optical experiments. This section presents and discuss the obtained results.

The simplest scene we used for testing our method is a scene containing a single plane parallel to the plane κ. In such a case our method is reduced to a proven Babinet's principle. Babinet's principle describes a relation between disturbances caused by two different screens put between the source and the observing plane. Since Babinet's principle is not crucial for presenting of results, we describe the principle and its relation to our method in Appendix A.3.

Furthermore, we tested our method on three scenes having different characteristics. The first scene "Convex" consists of one convex object. This is the simplest scene because for each cell g^{κ}_{lo}, there is at most one pillar p^{d}_{lo}. The second scene "Primitives" consists of several simple objects spatially distributed at significantly different depths. This scene demonstrates ability of our method to handle visibility and objects at significantly different depths. The third scene "Chessboard" is spatially more complex and contains small details. By this scene we demonstrate an influence of the decomposition on the visual quality of results.

For generating the results, we used the same parameters as those ones in Sect. 6, i.e., the pitch between samples is 7.0 µm, the patch consists of 32×32 samples, and the resolution of the calculated optical field is $6{,}144 \times 6{,}144$ samples. Among others, this is almost the maximum size of the hologram that we are able to handle in optical experiments. Notice that the size of the patch corresponds to a pixel size of a contemporary LCD, i.e., 0.22 mm.

Presented figures show an intensity of calculated optical field values because intensity is the only attribute that can be detected by the viewer. The intensity is either calculated from a numerical reconstruction or captured by a camera during an optical reconstruction. Both procedures are described in the following paragraph. In either case, the resulting images are not enhanced digitally except for scaling down and moving the white and the black point. Nonlinear deformation of intensities or filtering that might fake the visual quality is not applied.

In the case of numerical reconstruction we project the calculated optical field on a planar screen and calculate intensity as $I_{mn} = |u_{mn}|^2$, where u_{mn} is the value of the optical field at the point $\mathbf{u}_{mn}$. We project the optical field by propagating the angular spectrum. To improve accuracy of results, we pad the optical field with zeros to a resolution of $18{,}432 \times 18{,}432$ values $(18{,}432 = 3 \times 6{,}144)$ before the propagation. Calculated images show the scene that is sharp at the propagation distance, while the rest of the scene is blurred. Since the numerical reconstruction does not use a lens and a pinhole, calculated images lack perspective and have no depth of field.

In a case of the optical reconstruction we convert the optical field into a hologram by adding a reference wave and calculating intensity of the result [10]. We print the hologram and illuminate it with a quasi-coherent light source. To simulate the viewer, we use a regular camera with lenses to capture the intensity.

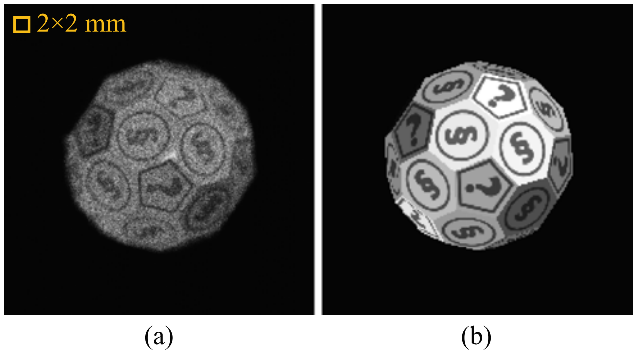

Fig. 6 (**a**) A numerical reconstruction of the scene "Convex," and (**b**) a rendering of the scene "Convex" using a standard means of computer graphics. The rendering uses an orthogonal projection of the scene, and it has a lower spatial resolution that is equal to the resolution of the visibility grid

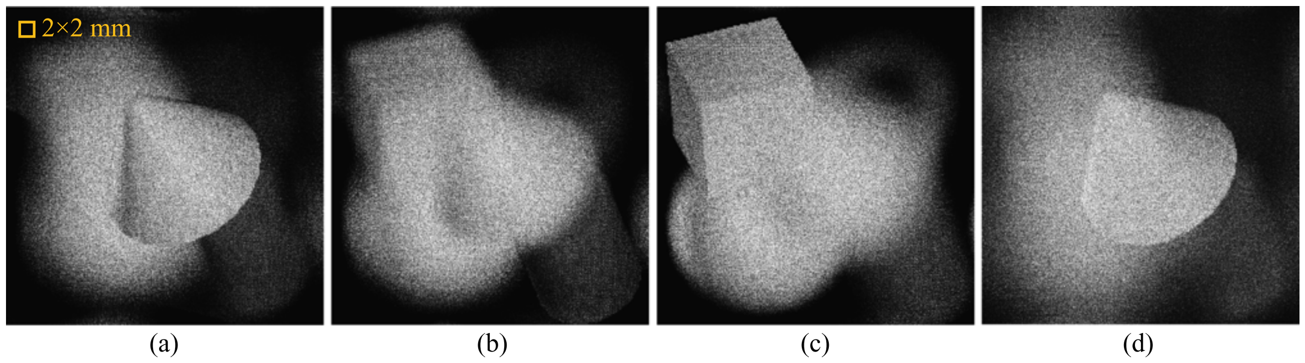

Fig. 7 The numerical reconstructions of the scene "Primitives" at various depths. The scene contains six objects, and we focused on three of them: (**a**) a cone at a distance of 0.40 m, (**b**) a cylinder at 0.45 m, and (**c**) a cube at 0.50 m. Furthermore, (**d**) a cone at 0.40 m is presented without properly solved visibility. When compared to the cone (**a**), the left side of the cone (**d**) is disturbed by other objects that are blurred

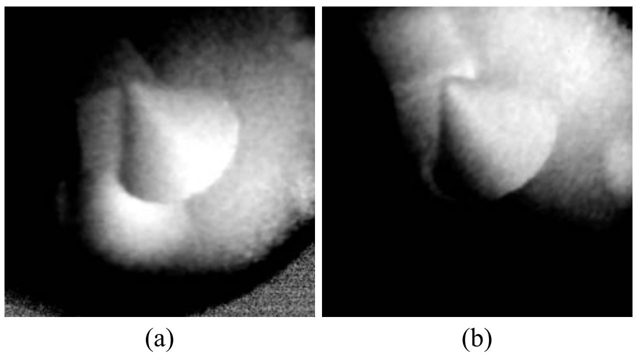

Fig. 8 The optical reconstructions of the scene "Primitives" captured by a camera. (**a**) A tip of the cone occludes an edge of a cube, while (**b**) the tip is below the edge of the cube when viewed from a different location. Although the relative location of the cone and the cube changes due to a perspective distortion, the visibility is correct in both cases

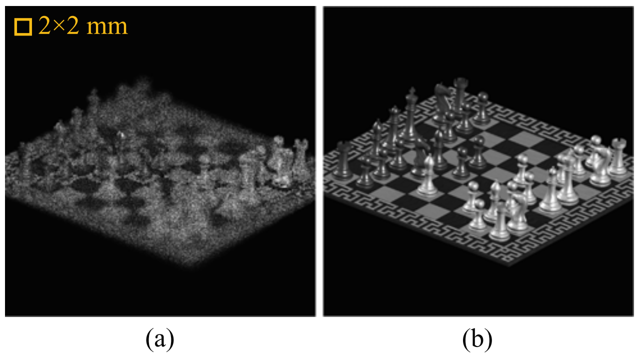

Fig. 9 (**a**) A numerical reconstruction of the scene "Chessboard" compared with (**b**) a computer graphics rendering done at full resolution

The numerical reconstruction in Fig. 6(a) shows that our method works for a single convex shape. The only difference between a classical computer graphics rendering of the scene at lower resolution depicted in Fig. 6(b) and the reconstruction is a blur and a grainy texture. Patches in the middle of the shape are almost in focus, while patches closer to the edge of the shape are out of focus and therefore blurred. The grainy texture on the surface is caused by a random phase distribution of an optical field of the patch. Since the size of the grain is similar to the pitch between points $\mathbf{u}_{mn}$, it is too small to disturb the viewer significantly.

How well our method handles visibility is illustrated in Fig. 7(a, d). A result in Fig. 7(d) shows a cone disturbed by blurred objects behind the cone because it was reconstructed from an optical field that was calculated without applying results of visibility tests. Especially the left side of the cone is affected. Unlike that, a result in Fig. 7(a) shows an undisturbed cone thanks to our solution of the visibility. Results in Fig. 7(a–c) show that objects at various depths were encoded into the optical field successfully. Objects in the focus are sharp with well-defined edges, while the rest of the objects is blurred. This shows an ability of our method to handle scenes containing objects at significantly different depths. Furthermore, the ability to handle visibility is verified through the optical reconstruction. As illustrated in Fig. 8, objects change their relative positions due to the perspective distortion when viewed from different angles. A top of a cone in Fig. 8(a) occludes an edge of a cube, while in Fig. 8(b) the top of the cone is below the edge of the cube. Since objects do not overlap in images, the visibility is solved correctly.

The effect of a detail reduction on a scene containing a small detail is presented in Fig. 9. When compared to a full resolution rendering in Fig. 9(b), the reconstruction depicted in Fig. 9(a) shows an expected reduction of scene detail. Besides that, the reconstruction contains already the mentioned blur and the grainy texture.

As the next step, we tested validity of computational complexity derived in Sect. 6. For that purpose, we measured computation times for the scene "Primitives" using various resolutions of the optical field and various resolutions of the patch. The method was implemented using the C++ language and the FFTW library [6]. All times were

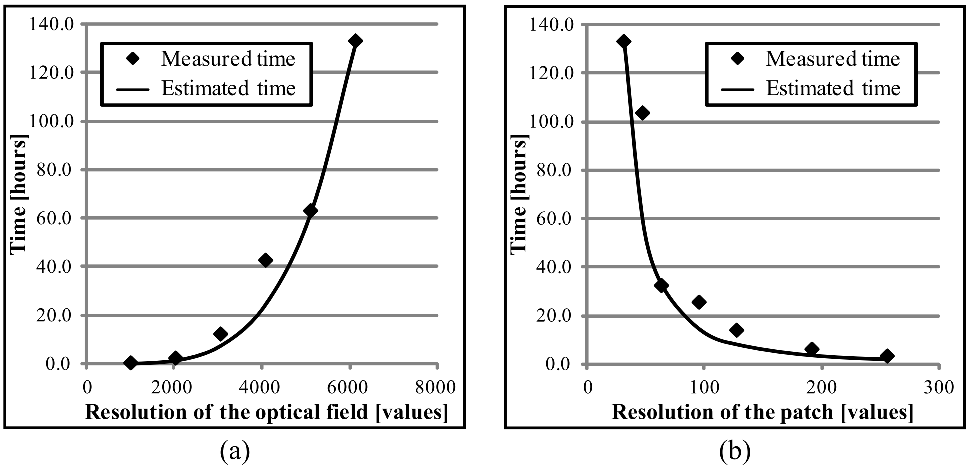

Fig. 10 Time measurements of the implemented method with various parameters: (**a**) a constant resolution of the patch and (**b**) a constant resolution of the optical field. Graphs also show an estimated computation time that was calculated using (5). Measurements illustrate a similarity between measured times and times estimated from the computational complexity

measured on a PC with Intel Xeon 3.2 GHz. The measured times were compared to times predicted by computational complexity.

In the first set of measurements we kept the resolution of the patch equal to 32 × 32 values, and we calculated optical fields at various resolutions. The pitch between the values was scaled proportionally to the resolution of the field so that its real size is always 43 × 43 mm. This preserved the ratio of a number of pillars to a number of cells almost constant, and therefore we were able to compare the measured times. Results of the measurements are provided in Fig. 10(a). In the second set of measurements, we kept the resolution of the optical field equal to 6,144 × 6,144 values and used different resolutions of the patch. Unlike the first set, we kept the pitch between points $\mathbf{u}_{mn}$ constant. This again preserved the ratio of a number of pillars to a number of cells. Results of the measurements are provided in Fig. 10(b).

The predicted times correspond to a result of (5) multiplied by a constant σ. This is because the expression in (5) is a computational complexity where multiplicative constants are neglected. The constant σ modifies the result of the expression in (5) so that the predicted time p_c becomes equal to a calibration measurement t_c. The calibration measurement t_c is measured using an optical field of 6,144 × 6,144 values with a patch of 32 × 32 values. Therefore, the predicted time p_p corresponding to a measured time t_p is $p_p = \sigma s_p$, where $\sigma = \frac{t_c}{s_c}$, and s_c and s_p are results of the expression in (5) calculated with parameters corresponding to the measurements t_c and t_p, respectively. The estimated times presented in Fig. 10 correspond with measured times, and we can consider the expression in (5) as a valid estimation of the computation complexity of our method.

Next, we verified our assumptions about other methods. For purpose of time estimation of other methods, we implemented a PLS method and an FFT method and measured time per a single PLS and per a single triangle, respectively. The implementation of PLS-based method omits visibility, and the implementation of FFT-based method is a partial implementation. The original FFT-based method as described in [21] consists of five steps that are executed in a sequence: tilting of the angular spectrum, conversion of the spectrum to the spatial domain, rasterization of a triangle, calculation of the spectrum, and application of the propagation operator similar to (2). We implemented only the second, the fourth, and the fifth steps. Therefore, in both cases we obtain a lower estimation, and a full implementation will be either the same or even slower.

We used similar parameters as in the previous measurements, i.e., a sampling step of 7.0 μm, a resolution 6,144 × 6,144 samples, and a patch resolution 32 × 32 samples. The measurement was executed on a PC with Intel Xeon 3.2 GHz. Considering the methods depending on a cloud of PLS, we estimated that our method is faster if the scene consist of more than 470,844 PLS. However, the measurement showed that the minimum is 33,845 PLS. Considering the FFT-based methods, we estimated that the minimum number of triangles is 37,413 triangles, but the measurement showed that the minimum number of triangles is 10,144 triangles. This proves that the comparison of methods through the expression in (5) is valid, and our method has the potential to be faster than the pure FFT-based methods and the pure PLS-based methods.

Furthermore, we tested an effect of using different sizes of patches on the overall visual quality. For the testing, we used the scene "Primitives" because it contains objects at various depths. We reconstructed the fields at 0.5 m (the cube) using the same parameters as in Fig. 7. The numerical reconstructions that are presented in Fig. 11 show expected decrease in the spatial resolution. The only disturbing artifact is the overlapping blur at edges of the patches. This is caused by the visibility approximation that cannot handle such a small detail. Nevertheless, we assume that larger patch size will be used only for preview purposes, and therefore the degradation visual quality with increasing patch size

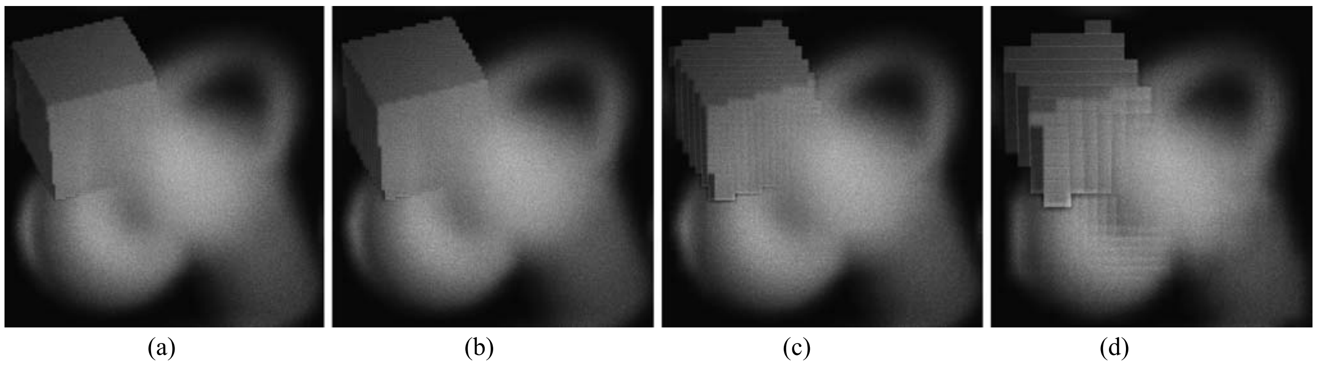

Fig. 11 The numerical reconstructions of the scene "Primitives" at 0.50 m (a cube). Used optical fields were calculated considering a different resolutions of the patch, i.e., (**a**) 48 × 48 samples, (**b**) 64 × 64 samples, (**c**) 128 × 128 samples, and (**d**) 256 × 256 samples. For a patch resolution of 32 × 32 samples, refer to Fig. 7(c)

can be neglected. A smaller patch size of 32 × 32 samples, which will be used for the final holograms, did not cause any visible and disturbing artifacts in optical reconstruction depicted in Fig. 8.

In this section we presented numerical and optical reconstructions. These reconstructions turned out to correspond with the input scenes quite well. We also verified the estimated computational complexity by comparing it with the measured times. To complete the evaluation, we compared our method to a PLS-based method and an FFT-based method. For that purpose, we used partial implementations that gave as lower estimations of time, and we showed that our estimations can be considered valid. However, we are not able to provide a comparison with a digital hologram of a real-world scene due to lacking a proper equipment.

8 Conclusion and future work

Our main goal when we were developing the proposed method was to achieve the ability to control the level of detail recorded in a hologram in order to reduce the calculation time. We fulfilled the goal by decomposing objects to patches. The size of patches controls the level of detail. In order to maintain a good visual quality, the size should be chosen as close to the human observer resolution as possible. However, if speed of evaluation is more important, the level of detail can be lowered even further in order to reduce the calculation time. This ability is useful for generating fast previews or for generating large holograms.

In the presented method we have employed two quite severe approximations. The first approximation replaces the surfaces in the virtual scene with patches having lower spatial resolution. Despite this approximation, we are able to handle a majority of the surface such that they appear solid in reconstructions. The second approximation is an estimation of the visibility using geometric optics at the reduced resolution. Despite this approximation, the viewer or a camera sees the scene without any artifacts caused by unexpected overlapping or disappearing of objects. Therefore our method can be applied to generate holograms of visually attractive virtual scenes.

To evaluate our method even further, we estimated computation complexity and verified it experimentally. The estimated complexity shows a speedup when compared to the hologram generation from a large cloud of PLS. Also, with increasing number of triangles in the scene, our method can calculate the optical field faster than methods that use strictly a propagation of the angular spectrum.

The presented results were created by an implementation intended as a proof of concept, and this was our goal. Thus, there is still a space for improving the performance of the method and optimizing the implementation. One of areas for further improvement is a process of objects decomposition to patches. The process is not well suited for large planar surfaces that are almost perpendicular with the recording plane. The worst case scenario is an axis-aligned cube. In such a case, only the front face of the cube will be recorded. In the future work we will deal mostly with developing of more efficient decomposition process that shall render the limitation obsolete.

Acknowledgements The authors would like to thank Levent Onural and Metodii Kovachev for invaluable discussions, Libor Váša for making photos of optical reconstructions, and Ivana Kolingerová for a great help with refining the paper.

Appendix

A.1 Phong's illumination model

The original Phong's illumination model was published in [26]. We use the slightly modified version as presented in [3, 30]. To estimate the intensity of light reflected from a surface point towards the viewer position, we need to know the normal vector $\mathbf{n}$ at the examined surface point, the light vector $\mathbf{l}_i$ that represents direction to the ith light source, the incident intensity I_i of the ith light source, the view vector $\mathbf{v}$ that represents the examined outgoing direction, the vector $\mathbf{h}_i$ bisecting the angle subtended by $\mathbf{l}_i$ and $\mathbf{v}$, and material properties, i.e., the diffuse reflectivity r_d, specular reflectivity r_s, and the coefficient n indicating the smoothness of the surface. If all vectors $\mathbf{n}$, $\mathbf{l}_i$, and $\mathbf{h}_i$ are normalized, then the outgoing intensity O is obtained as

$$O = \sum_i I_i \left[r_D (\mathbf{n} \cdot \mathbf{l}_i) + r_s (\mathbf{n} \cdot \mathbf{h}_i)^n \right]. \tag{6}$$

A.2 Fresnel approximation

The theory can be found in [9]. The setup is similar to a setup in Sect. 2. Let there be two planes. The first plane is populated with a regular grid of points $\mathbf{u}_{mn}$, and the second one with a regular grid of points $\mathbf{v}_{mn}$. The planes are parallel, and there is no obstacle in-between. The orthogonal distance between the planes is z and the orthogonal projection of grids overlaps. If we know the optical field values at the points $\mathbf{u}_{mn}$, we can calculate optical field values at the points $\mathbf{v}_{mn}$ as

$$v_{mn} = \sum_{m'} \sum_{n'} u_{m'n'} \frac{\exp(ikr)}{r} \frac{z}{r}, \tag{7}$$

where $r = [D_x^2(m-m')^2 + D_y^2(n-n')^2 + z^2]^{1/2}$ is the distance between the point $\mathbf{v}_{mn}$ and the point $\mathbf{u}_{m'n'}$, D_x is the distance between two neighboring points along the X-axis, D_y is the distance between two neighboring points along the Y-axis, and i is the imaginary unit.

Next, we approximate the distance r by the first two terms of Maclaurin expansion, i.e.,

$$r \approx z + \frac{D_x^2(m-m')^2 + D_y^2(n-n')^2}{2z}. \tag{8}$$

The approximation is valid only for a minimal orthogonal distance between the planes. When the approximation is applied, the optical field at points $\mathbf{v}_{mn}$ becomes

$$\begin{aligned} v_{mn} = {} & \frac{\exp(ikz)}{z} \exp\left(ik \frac{x^2+y^2}{2z} \right) \\ & \times \sum_{m'} \sum_{n'} u_{m'n'} \exp\left(ik \frac{x'^2 + y'^2}{2z} \right) \\ & \times \exp\left(-i2\pi \frac{xx' + yy'}{\lambda z} \right), \end{aligned} \tag{9}$$

where $x = mD_x$, $x' = m'D_x$, $y = mD_y$, and $y' = m'D_y$. The resulting expression is known as the Fresnel approximation, and it resembles the Fourier transform of the sample modulated by the function $\exp(ik\frac{x'^2+y'^2}{2z})$. Therefore, the Fresnel approximation allows us to calculate optical field values for a cost of a single FFT.

A.3 Babinet's principle and its application

The Babinet's principle [4] describes a behavior of an optical field disturbed by two planar screens. Both screens contain openings that when added up fill the whole plane. Let us have a source that emits waves and thus forms an optical field U on the plane κ. If the first screen is placed between the source and the plane κ, the optical field U is disturbed, and an optical field U_1 is detected on the plane κ instead. Similarly, when the second screen is used, an optical field U_2 is detected on the plane κ. The relation between the fields is $U = U_1 + U_2$.

Let us now consider a planar screen that is parallel to the plane κ. An original source of waves is behind the screen, and thus we can consider the screen to be a source of waves. The waves emitted by the screen forms an optical field U on the plane κ. Now, we make a half of the screen black, i.e., opaque. As a consequence, we detect a disturbed optical field U_1 on the plane κ. When the second half is made black instead of the first one, we detect an optical field U_2. According to the Babinet's principle, $U_1 + U_2$ gives us an undisturbed optical field U. Furthermore, we apply the principle on the field U_1 to decompose it into two optical fields. Such a recursive application of the principle can be repeated until used screen contains only a single opening of a size equal to the patch. Even in that case we are able to reconstruct the original field U from fields generated due to screens. This shows that our method works for the scene consisting of a plane parallel to the plane κ.

References

1. Ahrenberg, L., Benzie, P., Magnor, M., Watson, J.: Computer generated holography using parallel commodity graphics hardware. Opt. Express **14**(17), 7636–7641 (2006)
2. Ahrenberg, L., Benzie, P., Magnor, M., Watson, J.: Computer generated holograms from three dimensional meshes using an analytic light transport model. Appl. Opt. **47**(10), 1567–1574 (2008)
3. Blinn, J.: Models of light reflection for computer synthesized pictures. SIGGRAPH Comput. Graph. **11**(2), 192–198 (1977)
4. Born, M., Wolf, E.: Principles of Optics, 7th edn. Cambridge University Press, Cambridge (2005)
5. Esmer, G., Onural, L.: Computation of holographic patterns between tilted planes. In: Holography 2005, vol. 6252, p. 62521. SPIE, Bellingham (2006)
6. Frigo, M., Johnson, S.G.: Fftw. http://www.fftw.org/
7. Fujimoto, A., Tanaka, T., Iwata, K.: Arts: Accelerated ray-tracing system. IEEE Comput. Graph. Appl. **6**(4), 16–26 (1986)
8. Gabor, D.: Microscopy by reconstructed wavefronts. R. Soc. Lond. Proc. Ser. A **197**, 454–487 (1949)
9. Goodman, J.: Introduction to Fourier Optics, 3rd edn. Roberts & Company Publishers (2005)
10. Hariharan, P.: Optical Holography: Principles, Techniques and Applications, 2nd edn. Cambridge University Press, Cambridge (1996)

11. Ito, T., Masuda, N., Yoshimura, K., Shiraki, A., Shimobaba, T., Sugie, T.: Special-purpose computer horn-5 for a real-time electroholography. Opt. Express **13**(6), 1923–1932 (2005)
12. Janda, M., Hanák, I., Onural, L.: Hologram synthesis from photorealistic reconstruction. J. Opt. Soc. Am. A **25**(12), 3038–3096 (2008)
13. Kang, H., Yamaguchi, T., Yoshikawa, H.: Accurate phase-added stereogram to improve the coherent stereogram. Appl. Opt. **47**(19), D44–D54 (2008)
14. Kang, H., Yamaguchi, T., Yoshikawa, H.: Gpu-based acceleration method for coherent holographic stereogram calculation. In: Proc. of Biomedical Optics (2008)
15. Kim, H., Hahn, J., Lee, B.: Mathematical modeling of triangle-mesh-modeled three-dimensional surface objects for digital holography. Appl. Opt. **47**(19), D117–D127 (2008)
16. Lesem, L., Hirsch, P., Jordan, J.: Computer synthesis of holograms for 3-d display. Commun. ACM **11**(10), 661–673 (1968)
17. Lucente, M.: Diffraction-specific fringe computation for electro-holography. Ph.D. thesis, MIT (1994)
18. Lucente, M., Galyean, T.A.: Rendering interactive holographic images. In: SIGGRAPH'95, pp. 387–394 (1995)
19. Masuda, N., Ito, T., Tanaka, T., Shiraki, A., Sugie, T.: Computer generated holography using parallel commodity graphics hardware. Opt. Express **14**(2), 603–608 (2006)
20. Matsushima, K.: Computer-generated holograms for three-dimensional surface objects with shade and texture. Appl. Opt. **44**(22), 4607–4614 (2005)
21. Matsushima, K.: Exact hidden-surface removal in digitally synthetic full-parallax hologram. In: Practical Holography XIX: Materials and Applications, vol. 5742, pp. 25–32. SPIE, Bellingham (2005)
22. Matsushima, K., Kondoh, A.: A wave optical algorithm for hidden-surface removal in digitally synthetic full-parallax holograms for three-dimensional objects. In: Practical Holography XVIII: Materials and Applications, vol. 5290, pp. 90–97. SPIE, Bellingham (2004)
23. Matsushima, K., Takai, M.: Recurrence formulas for fast creation of synthetic three-dimensional holograms. Appl. Opt. **39**(35), 6587–6594 (2000)
24. Nishi, S., Shiba, K., Mori, K., Nakayama, S., Murashima, S.: Fast calculation of computer-generated Fresnel holograms utilizing distributed parallel processing and array operation. Opt. Rev. **12**(4), 287–292 (2005)
25. Petz, C., Magnor, M.: Fast hologram synthesis for 3d geometry models using graphics hardware. In: Practical Holography XVII and Holographic Materials IX, vol. 5005, pp. 266–275. SPIE, Bellingham (2003)
26. Phong, B.: Illumination for computer generated pictures. Commun. ACM **18**(6), 311–317 (1975)
27. Ritter, A., Böttger, J., Deussen, O., König, M., Strothotte, T.: Hardware-based rendering of full-parallax synthetic holograms. Appl. Opt. **38**(11), 1364–1369 (1999)
28. Tommasi, T., Bianco, B.: Computer-generated holograms of tilted planes by a spatial frequency approach. J. Opt. Soc. Am. A **10**, 299–305 (1993)
29. Underkoffler, J.: Occlusion processing and smooth surface shading for fully computed synthetic holography. Pract. Hologr. XI Hologr. Mater. III **3011**, 19–30 (1997)
30. Watt, A.: 3D Computer Graphics, 3rd edn. Addison–Wesley, Reading (2000)
31. Yoshikawa, H., Iwase, S., Oneda, T.: Fast computation of Fresnel holograms employing difference. In: Practical Holography XIV and Holographic Materials VI, vol. 3956, pp. 48–55. SPIE, Bellingham (2000)
32. Ziegler, R., Croci, S., Gross, M.: Lighting and occlusion in a wave-based framework. Comput. Graph. Forum **27**(2), 211–220 (2008)

Ivo Hanák is a PhD student at University of West Bohemia. His interests are digital holography and GPU.

Martin Janda is a PhD student at University of West Bohemia. His interest is digital holography.

Václav Skala is a full Professor of Computer Science at the University of West Bohemia in Plzen and the head of the Center of Computer Graphics and Visualization (http://herakles.zcu.cz). He is a member of Eurographics (since 1985). He is a member of the editorial boards of Computers & Graphics (Pergamon Press) and The Visual Computer (Springer Verlag) and IPC member of several international conferences and workshops. He is a founder of the WSCG International Conferences in Central Europe on Computer Graphics and Visualization (http://wscg.zcu.cz) and NET Technologies conferences (http://dotnet.zcu.cz) held every year at the University of West Bohemia, Plzen, Czech Republic.