Full parallax three-dimensional computer generated hologram with occlusion effect using ray casting technique

To cite this article: Hao Zhang et al 2013 J. Phys.: Conf. Ser. 415 012048

View the article online for updates and enhancements.

Related content
- A rendering method of background reflections on a specular surface for CGH
  T Ichikawa and Y Sakamoto
- Occlusion processing for computer generated hologram by conversion between the wavefront and light-ray information
  Koki Wakunami and Masahiro Yamaguchi
- Computational hologram synthesis and representation on spatial light modulators for real-time 3D holographic imaging
  Stephan Reichelt and Norbert Leister
Full parallax three-dimensional computer generated hologram with occlusion effect using ray casting technique

Hao Zhang, Qiaofeng Tan, Guofan Jin
State Key Laboratory of Precision Measurement Technology and Instruments, Tsinghua University, Beijing, 100084, China
E-mail: zhanghaocgh@hotmail.com

Abstract. Holographic display is capable of reconstructing the whole optical wave field of a three-dimensional (3D) scene. It is the only one among all the 3D display techniques that can produce all the depth cues. With the development of computing technology and spatial light modulators, computer generated holograms (CGHs) can now be used to produce dynamic 3D images of synthetic objects. Computation holography becomes highly complicated and demanding when it is employed to produce real 3D images. Here we present a novel algorithm for generating a full parallax 3D CGH with occlusion effect, which is an important property of 3D perception, but has often been neglected in fully computed hologram synthesis. The ray casting technique, which is widely used in computer graphics, is introduced to handle the occlusion issue of CGH computation. Horizontally and vertically distributed rays are projected from each hologram sample to the 3D objects to obtain the complex amplitude distribution. The occlusion issue is handled by performing ray casting calculations to all the hologram samples. The proposed algorithm has no restriction on or approximation to the 3D objects, and hence it can produce reconstructed images with correct shading effect and no visible artifacts. Programmable graphics processing unit (GPU) is used to perform parallel calculation. This is made possible because each hologram sample belongs to an independent operation. To demonstrate the performance of our proposed algorithm, an optical experiment is performed to reconstruct the 3D scene by using a phase-only spatial light modulator. We can easily perceive the accommodation cue by focusing our eyes on different depths of the scene and the motion parallax cue with occlusion effect by moving our eyes around. The experiment result confirms that the CGHs produced by our algorithm can successfully reconstruct 3D images with all the depth cues.

1. Introduction
Holographic display is the most promising one among all the 3D display techniques, since it can reconstruct the whole optical wave field of a three-dimensional (3D) scene and produce all the depth cues. The hologram plane forms a window, through which viewers can observe the 3D scene as if the object is floating behind. To generate a hologram for 3D display by calculation is not an easy task. With the development of computing technology and spatial light modulators (SLMs), computer generated holograms (CGHs) can now be used to produce dynamic 3D images of synthetic objects.

At present, there are two main challenges in computation based holographic 3D display: the limited space bandwidth product (SBWP) of the SLM and the intensive computations required to produce the CGHs, especially those with full parallax effect. The restricted SBWP of the SLM will limit the viewing angle and the image size of a 3D scene, while the computational challenge will affect both the
computational speed and the visual quality. Therefore, there is a need to investigate how best to calculate the CGH for 3D images in practice.

Furthermore, occlusion cue is an important property when viewing 3D scenes, but such an effect is not considered in most computational methods for CGHs. An occlusion effect is coupled with motion parallax, it should be distinguished from hidden surface removal, where the visible surfaces do not change when the viewpoint moves. In other words, for hidden surface removal, the front object does not block the back object when the viewer moves, while the occlusion effect implies that the visible surfaces change according to the movement of the viewpoint, which produces a correct depth cue for the viewers.

Most of the current algorithms for synthesizing CGHs use techniques which represent the 3D scene geometry as a collection of primitives and calculate the superimposed complex amplitude in the hologram plane. The 3D objects are commonly divided into point sources or planar segments. The point source method is more flexible in describing many kinds of 3D objects, while the planar segments method can perform comparatively fast calculations. These physically based algorithms simulate the optical transmission process from the target 3D objects to a hologram plane, which records a precise description for these 3D objects. However, it is difficult to express the occlusion effect using such techniques, since the optical transmission processes of different primitives are independent. Although the silhouette method was introduced to these algorithms for full parallax CGH with occlusion, this method is specially designed with a fixed condition, namely, a 3D object with a 2D background, so it cannot produce the occlusion effect for an arbitrary 3D scene. Recently, Chen et al. proposed a method to handle the occlusion issue for a physically based algorithm, which runs visibility tests for each hologram sample using the occlude list. However, this method interpolates the sampling points through a 2D interpolation kernel, which is not a rigorous description of 3D objects in general. Different from the physically based methods, the diffraction-specific approach developed by Lucente considers the reconstruction step in holography rather than the simulation of the optical transmission process from the objects. It uses functional holographic elements (hogels) to reduce the CGH computation load. Moreover, this algorithm is specially designed for horizontal parallax only (HPO) holograms and display systems with high horizontal resolution, such as the Holovideo display system of MIT Media Laboratory. Stereogram modeling method and the methods derived from it can produce HPO CGH with occlusion effect easily because the hologram is generated through appropriately captured 2D parallax views. These kinds of methods are fast and they are well matched to hardware implementation. However, since the calculation of the stereogram is based on a set of 2D views, it is hard to produce all the depth cues, especially the accommodation cue.

Recently, some researchers attempted to introduce the ray casting technique from computer graphics to the calculation of CGHs. Janda et al. proposed a ray casting method to handle occlusion issue, but they restricted their 3D model to a collection of planar segments parallel to the hologram plane. This approximation could cause artifacts when reconstructing 3D images experimentally. Hanak et al. made use of the graphics pipeline to generate a full parallax hologram with occlusion effect. More recently, Chen et al. developed a similar algorithm to generate an HPO CGH with occlusion effect by using an OpenGL depth buffer. However, because such graphics pipeline based methods produce shear transformations to the 3D objects concerned, the shading effect of the 3D scenes will be affected.

Although the occlusion effect has been addressed by some researchers, none of them has produced an optical demonstration for full parallax 3D CGH with occlusion effect. In this paper we present a ray casting based algorithm for the calculation of full parallax 3D CGH with occlusion effect and demonstrate the result in an optical experiment. In contrast to the previously mentioned methods, this algorithm has no parallel planar segments restriction or shear transformations to the 3D objects. It can, therefore, produce precise description to the 3D scene. Programmable graphics processing unit (GPU) is used to perform parallel calculation. Experiment is performed to optically reconstruct the 3D scene by using a phase-only SLM. The theoretical framework of such CGH generation is described in
Section 2. Optical reconstruction results are shown in Section 3. Finally, brief discussion and conclusion are provided in Section 4.

2. Method

2.1. Ray casting algorithm for CGH generation
Ray casting is a widely used technique in computer graphics for hidden surface removal. The casting ray is projected from the eye position through each pixel of the image to find the nearest intersection object point, as shown in Figure 1(a). One can easily add light and shadow effect to a 3D scene during rendering. However, there are fundamental differences between this kind of image-order rendering and the CGH generation. The parts of a 3D scene which make a contribution to each specific hologram sample must be evaluated. This can be done by projecting a bundle of rays from each hologram sample to the 3D scene to find the nearest intersection object point for each ray, as shown in Figure 1(b). Only the nearest point provides a contribution to that hologram sample along the ray projecting direction. The occlusion issue is handled by performing ray casting calculations to all the hologram samples.

![Figure 1](image1.png)

Figure 1. (a) Ray casting technique using in computer graphics, each pixel decides one casting ray. (b) Ray casting technique using in CGH generation, each pixel corresponds many casting rays.

Point source method is used in our ray casting algorithm to calculate the complex amplitude distribution of each intersection object point. The optical distribution of the spherical wave which is produced from one point source can be computed as:

$$H(x, y) = \frac{A}{r} \exp[i(kr + \phi)],$$

where $A$ is the wave amplitude of the point, $k = \frac{2\pi}{\lambda}$ is the wave number in free space, $\lambda$ is the wavelength, $\phi$ is the initial phase (often randomized as a uniform distribution between 0 and $2\pi$), and $r$ is the distance between the specific point source $(x_j, y_j, z_j)$ and the hologram sample $(x, y, 0)$:

$$r = \left[ (x - x_j)^2 + (y - y_j)^2 + z_j^2 \right]^{\frac{1}{2}}.$$
Figure 2. (a) Top view of the ray casting technique using in CGH generation, horizontal distributed casting rays are shot from each pixel. (b) Side view of the ray casting technique using in CGH generation, vertically distributed casting rays are shot from each pixel.

Lighting is an important feature when describing 3D scenes. When it comes to CGH computing, lighting affects the wave amplitude of the intersection object point $A$ in Eq. (1). Deduced from bidirectional reflectance distribution function (BRDF), which is the function that defines how light is reflected at the opaque surface of a 3D object, the intensity of each surface point $p$ can be computed as:

$$I_p = k_a I_a + k_d I_d \max (\hat{L} \cdot \hat{n}, 0),$$

(3)

where $k_a$ is the ambient reflection constant and $I_a$ is the intensity of the ambient light, $k_d$ is the diffuse reflection constant, $I_d$ is the intensity of the incoming light, $\hat{L}$ is the unit vector pointing from the light source to the surface, $\hat{n}$ is the unit normal vector at point $p$, and $\cdot$ is the dot product operator. So the light intensity of our model is constituted by ambient and Lambertian reflection. During computing, $k_a$ and $k_d$ can be customized to adjust the overall shading effect.

As the calculations performed on each of the hologram samples are independent of each other, we can explore the powerful parallel computing ability of GPU for calculation. The Computer Unified Device Architecture (CUDA) is used to perform the calculation. It is a parallel computing architecture developed by NVIDIA, which gives developers access to the virtual instruction set and memory of the parallel computational elements in CUDA GPUs. Each hologram sample is managed as a separate thread. A kernel function is executed on all the threads in order to perform the ray casting calculation, which loops over all the casting rays to compute the complex amplitudes for the hologram samples. The contribution of each casting ray can be calculated through Eq. (1). The kernel function evaluates the nearest intersection object point for each ray. Hence the distance between hologram sample and the intersection point can be calculated through Eq. (2). The wave amplitude of the intersection object point $A$ is computed through Eq. (3). The initial phase $\phi$ can be mapped to the pre-calculated lookup table restored in the 3D texture memory.

2.2. Distribution of casting rays

The distribution of casting rays has a close relationship with the discrete SLM device to be used to display the CGH. When a collimated beam illuminates on the SLM, the diffraction beam will separate in many orders due to the pixelated structure of the modulator. The maximum diffraction angle of the SLM is given by:

$$\sin \theta = f \frac{\lambda}{2d},$$

(4)

where $f$ is the highest spatial frequency that the SLM can display, and $d$ is the pixel pitch of the SLM, as shown in Figure 3. The angle of the casting rays is limited by this maximum diffraction angle.
This restriction automatically solves the aliasing problem from using a physically based algorithm to generate the CGHs in the situation when 3D objects are in a short distance to the hologram plane. Only the object points which are in the diffraction region of a specific hologram sample will provide contributions to that sample. From Eq. (4), we can deduce that the diffraction angle of a given SLM is directly dependent on its spatial resolution. This angle produces the motion parallax cue during the optical reconstruction. It is obvious that an SLM of higher spatial resolution would lead to a larger viewing angle.

Another parameter which needs to be considered is the angular sampling distance ($\beta$) of adjacent casting rays. This sampling distance is related to the density of the intersection object points. Dense intersection object points can give a solid surface effect to viewers. Normally we can choose $\beta = 1/60$, the same as the angular resolution of human eye. One may also adjust the sampling distance according to the requirements. Given the maximum diffraction angle and the angular sampling distance, the total sampling number for each hologram sample can be calculate as:

$$N = \left(\frac{2\theta}{\beta}\right)^2,$$

where the square is introduced because of the full parallax consideration.

3. Experimental results

To demonstrate the effect of our proposed method, an optical experiment is performed, using the HEO 1080P Phase Modulator (HOLOEYE Photonics AG). This is a liquid crystal on silicon (LCoS) analogue phase-only SLM with 1920 x 1080 pixels. The pixel pitch is $8 \mu m$, and the phase magnitude is addressed with 8 bit gray-scale levels. The wavelength of the laser used in our experiment is 635nm.

According to the discussion in Section 2, it can be deduced that the angular diffraction region of our SLM is within $\pm 2.27$. To get better solid surface effect for optical reconstruction, angular sampling distance is set to 0.01 in the calculation, which means the total sampling number for each hologram plane is 454$^2$. The 3D scene we use to generate the CGH contains two pyramids, together with a light source at infinity. The pyramids are positioned at distances of 125mm and 175mm from the hologram plane, respectively. The calculation is performed on an NVIDIA Quadro FX 2800M graphics card, and the total computation time for this example is 80 minutes. In the reconstruction stage, we use only the phase information of the CGH displayed on the SLM, as in a kinoform. This method can maintain good quality in reconstruction for 3D diffusive objects.
Figure 4. Optical reconstruction results when camera is focusing on (a) the front object and (b) the back object.

The reconstructed full parallax 3D scene with occlusion effect is confirmed by image recording. During which, a single lens reflex camera (Canon 500D) with a zoom lens (Tamron AF 19-200mm) is used to capture the 3D scene directly. Figure 4(a) and (b) show the reconstruction results where the camera is manually focused on the front and back objects separately (focal length: 200mm, f number: 6.3, object distance: 450mm). The front object shown in Figure 4(a) can be seen completely, while the back pyramid shown in Figure 4(b) is partially blocked by the front one. This shows that our method has successfully produced the accommodation cue with hidden surface removal effect. To demonstrate the occlusion effect of our method, the camera is placed at different viewing positions to capture the 3D objects (focal length: 200mm, f number: 20, object distance: 600mm). The results are shown in Figure 5, demonstrating that full parallax with occlusion effect can be perceived in the optical reconstruction. Long depth of field is needed when recording the images in Figure 5 because the two pyramids are displayed at different distances. This can be achieved by using a small relative aperture during image recording, but such a measure will decrease the quality of the captured images. We can also use our eyes to view the reconstructed 3D scene instead of the camera and perceive the depth cues of the scene.

Figure 5. Optical reconstruction results when viewing from (a) top, (b) bottom, (c) left and (d) right, showing occlusion effect and full parallax.
4. Discussion and conclusion
In summary, we propose an algorithm to generate full parallax CGH with occlusion effect, and demonstrate its effectiveness by optical reconstruction. The ray casting technique used in computer graphics is introduced in this algorithm, using casting rays projected from each hologram sample to find the nearest intersection point with the objects. Our approach does not require the target objects be transformed or restricted, and it allows a rigorous description to the 3D scene with correct shading effect when it is optically reconstructed. The computational speed is highly dependent on the number of casting rays sampled. Faster calculation can be achieved through the sacrifice of visual effect by limiting the angular diffraction region or by increasing the step of sampling angle. The calculation is implemented on GPU platform for parallel computation. The optical experiment is performed using a phase-only SLM. By loading the calculated phase CGH on the SLM, we successfully reconstruct the 3D scene with full parallax and occlusion effect. Finally it should be pointed out that our method can also be used to produce large holograms with complicated 3D scenes.

References
[1] S. A. Benton and V. M. Bove, Holographic Imaging, Wiley, Hoboken (2007).
[2] M. Lucente, “Interactive computation of holograms using a look-up table,” J. Electron. Imaging 2, 28-34 (1993).
[3] F. Yara, H. Kang, and Levent Onural, “State of the Art in Holographic Display: A Survey,” J. Display Technol. 6, 443-454 (2010).
[4] C. W. Slinger, C. D. Cameron, S. D. Coomer, R. J. Miller, D. A. Payne, A. P. Smith, M. G. Smith, M. Stanley, and P. J. Watson, “Recent developments in computer-generated holography: toward a practical electroholography system for interactive 3D visualization,” Proc. SPIE 5290, 27-41 (2004).
[5] K. Matsushima, “Computer-generated holograms for three-dimensional surface objects with shade and texture,” Appl. Opt. 44, 4607-4614 (2005).
[6] Y. Ichihashi, H. Nakayama, T. Ito, N. Masuda, T. Shimobaba, A. Shiraki, and T. Sugie, “HORN-6 special-purpose clustered computing system for electroholography,” Opt. Express 17, 13895-13903 (2009)
[7] H. Zhang, J. Xie, J. Liu, and Y. Wang, “Optical reconstruction of 3D images by use of pure-phase computer-generated holograms,” Chin. Opt. Lett. 7, 1101-1103 (2009)
[8] L. Ahrenberg, P. Benzie, M. Magnor, and J. Watson, “Computer generated holograms from three dimensional meshes using an analytic light transport model,” Appl. Opt. 47, 1567-1574 (2008)
[9] K. Matsushima and S. Nakahara, “Extremely high-definition full-parallax computer-generated hologram created by the polygon-based method,” Appl. Opt. 48, H54-H63 (2009)
[10] R. H.-Y. Chen and T. D. Wilkinson, “Computer generated hologram from point cloud using graphics processor,” Appl. Opt. 48, 6841-6850 (2009).
[11] M. Lucente, “Diffraction-specific fringe computation for electro-holography,” Ph. D. dissertation, Massachusetts Institute of Technology, (1994).
[12] M. Lucente, “Computational holographic bandwidth compression,” IBM Syst. J. 35(3&4), 349-365 (1996).
[13] M. Lucente and T. A. Galyean, “Rendering interactive holographic images,” in Proceedings of SIGGRAPH ’95, 387-394 (1995).
[14] Q. Y. J. Smithwick, J. Barabas, D. E. Smalley, and V. M. Bove, Jr., “Real-time shader rendering of holographic stereograms,” Proc. SPIE 7233, 723302 (2009).
[15] M. Janda, I. Hanak, and L. Onural, “Hologram synthesis for photorealistic reconstruction,” J. Opt. Soc. Am. A 25, 3083-3096 (2008)
[16] I. Hanak, M. Janda, V. Skala, “Full-Parallax Hologram Synthesis of Triangular Meshes using a Graphical Processing Unit,” In Proc. Of 3DTV-CON, (2007)
[17] R. H.-Y. Chen and T. D. Wilkinson, “Computer generated hologram with geometric occlusion
using GPU-accelerated depth buffer rasterization for three-dimensional display,” *Appl. Opt.* **48,** 4246-4255 (2009)

[18] K. Zhou, Q. Hou, R. Wang and B. Guo, “Real-Time KD-Tree Construction on Graphics Hardware,” *ACM Trans. on Graphics (Siggraph Asia)*, **27**(5), 126 (2008).

[19] X. Xu, S. Solanki, X. Liang, S. Xu, R. B. A. Tanjung, Y. Pan, F. Farbiz, B. Xu, and T.-C. Chong, “Computer-Generated Holography for Dynamic Display of 3D Objects with Full Parallax,” *The International Journal of Virtual Reality*, **8**(2), 33-38 (2009).

[20] F. E. Nicodemus, “Directional reflectance and emissivity of an opaque surface,” *Appl. Opt.* **4**, 767–773 (1965).

[21] H. Zhang, J. Xie, J. Liu, and Y. Wang, “Elimination of a zero-order beam induced by a pixelated spatial light modulator for holographic projection,” *Appl. Opt.* **48**, 5834-5841 (2009).